This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decollection of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, the following notice is applicable:

U.S. GOVERNMENT END USERS. Oracle programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, delivered to U.S. Government end users are “commercial computer software” pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc., AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information on content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services.

Ce logiciel et la documentation qui l’accompagnent sont protégés par les lois sur la propriété intellectuelle. Ils sont concédés sous licence et soumis à des restrictions d’utilisation et de divulgation. Sauf disposition de votre contrat de licence ou de la loi, vous ne pouvez pas copier, reproduire, traduire, diffuser, modifier, breveter, transmettre, distribuer, exposer, exécuter, publier ou afficher le logiciel, même partiellement, sous quelque forme et par quelque procédé que ce soit. Par ailleurs, il est interdit de procéder à toute ingénierie inverse du logiciel, de le désassembler ou de le décompiler, excepté à des fins d’interopérabilité avec des logiciels tiers ou tel que prescrit par la loi.

Les informations fournies dans ce document sont susceptibles de modification sans préavis. Par ailleurs, Oracle Corporation ne garantit pas qu’elles soient exemptes d’erreurs et vous invite, le cas échéant, à lui en faire part par écrit.

Si ce logiciel, ou la documentation qui l’accompagne, est concédé sous licence au Gouvernement des États-Unis, ou à toute entité qui délivre la licence de ce logiciel ou l’utilise pour le compte du Gouvernement des États-Unis, la notice suivante s’applique:

U.S. GOVERNMENT END USERS. Oracle programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, delivered to U.S. Government end users are “commercial computer software” pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. No other rights are granted to the U.S. Government.

Ce logiciel ou matériel a été développé pour un usage général dans le cadre d’applications de gestion des informations. Ce logiciel ou matériel n’est pas conçu ni n’est destiné à être utilisé dans des applications à risque, notamment dans des applications pouvant causer des dommages corporels. Si vous utilisez ce logiciel ou matériel dans le cadre d’applications dangereuses, il est de votre responsabilité de prendre toutes les mesures de secours, de sauvegarde, de redondance et autres mesures nécessaires à son utilisation dans des conditions optimales de sécurité. Oracle Corporation et ses affiliés déclinent toute responsabilité quant aux dommages causés par l’utilisation de ce logiciel ou matériel pour ce type d’applications.

Oracle et Java sont des marques déposées d’Oracle Corporation et/ou de ses affiliés. Tout autre nom mentionné peut correspondre à des marques appartenant à d’autres propriétaires qu’Oracle.

Intel et Intel Xeon sont des marques ou des marques déposées d’Intel Corporation. Toutes les marques SPARC sont utilisées sous licence et sont des marques ou des marques déposées d’SPARC International, Inc. AMD, Opteron, le logo AMD et le logo AMD Opteron sont des marques ou des marques déposées d’Advanced Micro Devices. UNIX est une marque déposée d’The Open Group.

Ce logiciel ou matériel et la documentation qui l’accompagne peuvent fournir des informations ou des liens donnant accès à des contenus, des produits et des services émanant de tiers. Oracle Corporation et ses affiliés ne sauraient être tenus pour responsables des pertes subies, des coûts occasionnés ou des dommages causés par l’accès à des contenus, produits ou services tiers, ou à leur utilisation.
Contents

Preface ................................................................................................................................................... 23

Part I Security Overview .................................................................................................................... 25

1 Security Services (Overview) ........................................................................................................... 27
   System Security .................................................................................................................................... 27
   Cryptographic Services ....................................................................................................................... 28
   Authentication Services .................................................................................................................... 29
   Authentication With Encryption ...................................................................................................... 30
   Auditing ................................................................................................................................................ 30
   Security Policy .................................................................................................................................... 30

Part II System, File, and Device Security ............................................................................................ 33

2 Managing Machine Security (Overview) ........................................................................................... 35
   Controlling Access to a Computer System .......................................................................................... 35
   Maintaining Physical Security ......................................................................................................... 35
   Maintaining Login Control ............................................................................................................... 36
   Controlling Access to Devices .......................................................................................................... 40
   Device Policy (Overview) .................................................................................................................. 41
   Device Allocation (Overview) .......................................................................................................... 42
   Controlling Access to Machine Resources ....................................................................................... 42
   Address Space Layout Randomization ............................................................................................ 42
   Limiting and Monitoring Superuser Access ..................................................................................... 43
   Configuring Role-Based Access Control to Replace Superuser ....................................................... 43
   Preventing Unintentional Misuse of System Resources ................................................................... 44
   Restricting setuid Executable Files .................................................................................................. 45
Contents

Using the Secure by Default Configuration ................................................................. 45
Using Resource Management Features ........................................................................ 45
Using Oracle Solaris Zones ........................................................................................ 46
Monitoring Use of Machine Resources ..................................................................... 46
Monitoring File Integrity ............................................................................................ 46
Controlling Access to Files ......................................................................................... 47
  Protecting Files With Encryption .............................................................................. 47
  Using Access Control Lists ....................................................................................... 47
  Sharing Files Across Machines ................................................................................ 48
  Restricting root Access to Shared Files ..................................................................... 48
Controlling Network Access ........................................................................................ 48
  Network Security Mechanisms .................................................................................. 49
  Authentication and Authorization for Remote Access ............................................... 49
Firewall Systems .......................................................................................................... 51
Encryption and Firewall Systems .................................................................................. 51
Reporting Security Problems ...................................................................................... 52

3 Controlling Access to Systems (Tasks) ................................................................... 53
Securing Logins and Passwords (Tasks) ..................................................................... 53
  Securing Logins and Passwords (Task Map) .............................................................. 53
  ▼ How to Change the root Password ........................................................................ 54
  ▼ How to Display a User’s Login Status ................................................................. 55
  ▼ How to Display Users Without Passwords ........................................................... 56
  ▼ How to Temporarily Disable User Logins ............................................................ 56
  About Failed Logins .................................................................................................. 57
Changing the Default Algorithm for Password Encryption (Tasks) .............................. 57
  ▼ How to Specify an Algorithm for Password Encryption ....................................... 57
  ▼ How to Specify a New Password Algorithm for an NIS Domain ....................... 58
  ▼ How to Specify a New Password Algorithm for an LDAP Domain .................... 59
Monitoring and Restricting root Access (Tasks) ......................................................... 60
  ▼ How to Monitor Who Is Using the su Command ................................................ 60
  ▼ How to Restrict and Monitor root Logins ............................................................. 61
Controlling Access to System Hardware (Tasks) ......................................................... 62
  ▼ How to Require a Password for SPARC Hardware Access .................................. 62
  ▼ How to Disable a System’s Abort Sequence ....................................................... 63
4 Virus Scanning Service (Tasks) .......................................................................................... 65
  About Virus Scanning ........................................................................................................ 65
  About the Vscan Service ..................................................................................................... 66
  Using the Vscan Service (Tasks) ....................................................................................... 66
    ▼ How to Enable Virus Scanning on a File System ....................................................... 67
    ▼ How to Enable the Vscan Service ............................................................................. 68
    ▼ How to Add a Scan Engine ....................................................................................... 68
    ▼ How to View Vscan Properties ............................................................................... 68
    ▼ How to Change Vscan Properties ............................................................................ 69
    ▼ How to Exclude Files From Virus Scans ................................................................. 69

5 Controlling Access to Devices (Tasks) ............................................................................... 71
  Configuring Device Policy (Tasks) .................................................................................... 71
    Configuring Device Policy (Task Map) ......................................................................... 71
    ▼ How to View Device Policy .................................................................................... 72
    ▼ How to Audit Changes in Device Policy .................................................................. 72
    ▼ How to Retrieve IP MIB-II Information From a /dev/*/Device ................................ 73
  Managing Device Allocation (Tasks) ................................................................................ 73
    Managing Device Allocation (Task Map) .................................................................... 73
    ▼ How to Enable Device Allocation .......................................................................... 74
    ▼ How to Authorize Users to Allocate a Device ....................................................... 74
    ▼ How to View Allocation Information About a Device ............................................ 75
    ▼ How to Forcibly Allocate a Device ......................................................................... 76
    ▼ How to Forcibly Deallocate a Device ..................................................................... 76
    ▼ How to Change Which Devices Can Be Allocated ................................................. 77
    ▼ How to Audit Device Allocation ............................................................................ 78
  Allocating Devices (Tasks) ............................................................................................... 78
    ▼ How to Allocate a Device ....................................................................................... 78
    ▼ How to Mount an Allocated Device ....................................................................... 79
    ▼ How to Deallocate a Device ................................................................................. 81
  Device Protection (Reference) ......................................................................................... 82
    Device Policy Commands ............................................................................................ 82
    Device Allocation ....................................................................................................... 83
Contents

- How to Change Special File Permissions in Absolute Mode ................................................. 120
- Protecting Against Programs With Security Risk (Task Map) ............................................. 121
- How to Find Files With Special File Permissions ............................................................... 121
- How to Disable Programs From Using Executable Stacks .................................................... 122

Part III Roles, Rights Profiles, and Privileges ................................................................................. 125

8 Using Roles and Privileges (Overview) ............................................................................................... 127
Role-Based Access Control (Overview) ............................................................................................... 127
  - RBAC: An Alternative to the Superuser Model ....................................................................... 127
  - RBAC Elements and Basic Concepts ....................................................................................... 130
  - Privilege Escalation .................................................................................................................. 133
  - RBAC Authorizations ................................................................................................................. 133
  - Authorizations and Privileges ................................................................................................. 134
  - Privileged Applications and RBAC .......................................................................................... 134
  - RBAC Rights Profiles .............................................................................................................. 135
  - RBAC Roles .............................................................................................................................. 136
  - Profile Shells and RBAC .......................................................................................................... 137
  - Name Service Scope and RBAC .............................................................................................. 137
  - Security Considerations When Directly Assigning Security Attributes .................................. 137
  - Usability Considerations When Directly Assigning Security Attributes ............................ 138

Privileges (Overview) ........................................................................................................................ 138
  - Privileges Protect Kernel Processes ....................................................................................... 138
  - Privilege Descriptions ............................................................................................................. 139
  - Administrative Differences on a System With Privileges ..................................................... 140
  - Privileges and System Resources ......................................................................................... 141
  - How Privileges Are Implemented ......................................................................................... 142
  - How Processes Get Privileges ............................................................................................... 143
  - Assigning Privileges .............................................................................................................. 144
  - Privileges and Devices .......................................................................................................... 145
  - Privileges and Debugging ...................................................................................................... 146
About RBAC in This Release .............................................................................................................. 146
### 9 Using Role-Based Access Control (Tasks) .......................................................... 149

- Using RBAC (Tasks) ........................................................................................................ 149
- Viewing and Using RBAC Defaults (Tasks) ....................................................................... 150
  - Viewing and Using RBAC Defaults (Task Map) .............................................................. 150
  - ▼ How to View All Defined Security Attributes .............................................................. 150
  - ▼ How to View Your Assigned Rights .............................................................................. 151
  - ▼ How to Assume a Role .................................................................................................. 154
  - ▼ How to Change the Security Attributes of a User ......................................................... 155
  - ▼ How to Use Your Assigned Administrative Rights ....................................................... 157
- Customizing RBAC for Your Site (Tasks) ........................................................................ 160
  - Initially Configuring RBAC (Task Map) ......................................................................... 160
  - ▼ How to Plan Your RBAC Implementation .................................................................. 161
  - ▼ How to Create a Role ..................................................................................................... 163
  - ▼ How to Assign a Role .................................................................................................... 165
  - ▼ How to Audit Roles ....................................................................................................... 166
  - ▼ How to Create a Rights Profile .................................................................................... 167
  - ▼ How to Clone and Modify a System Rights Profile ..................................................... 169
  - ▼ How to Create an Authorization .................................................................................. 171
  - ▼ How to Add RBAC Properties to Legacy Applications ............................................... 172
  - ▼ How to Troubleshoot RBAC and Privilege Assignment .............................................. 174
- Managing RBAC (Tasks) .................................................................................................. 177
  - Managing RBAC (Task Map) .......................................................................................... 177
  - ▼ How to Change the Password of a Role ...................................................................... 178
  - ▼ How to Change the Security Attributes of a Role ....................................................... 179
  - ▼ How to Reorder Assigned Security Attributes ............................................................. 180
  - ▼ How to Restrict an Administrator to Explicitly Assigned Rights ................................ 181
  - ▼ How to Enable a User to Use Own Password to Assume a Role .................................. 182
  - ▼ How to Change the root Role Into a User ..................................................................... 183
- Using Privileges (Tasks) ..................................................................................................... 185
  - ▼ How to List the Privileges on the System .................................................................... 186
  - ▼ How to Determine the Privileges That You Have Been Directly Assigned ................ 187
  - ▼ How to Determine the Privileged Commands That You Can Run ............................... 188
  - ▼ How to Determine the Privileges on a Process ............................................................ 189
  - ▼ How to Determine Which Privileges a Program Requires ......................................... 191
  - ▼ How to Apply Extended Privilege Policy to a Port ..................................................... 192
  - ▼ How to Run a Shell Script With Privileged Commands ............................................... 193
<table>
<thead>
<tr>
<th>Chapter</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 Security Attributes in Oracle Solaris (Reference)</td>
<td>195</td>
</tr>
<tr>
<td>Rights Profiles</td>
<td>195</td>
</tr>
<tr>
<td>Viewing the Contents of Rights Profiles</td>
<td>197</td>
</tr>
<tr>
<td>Order of Search for Assigned Security Attributes</td>
<td>197</td>
</tr>
<tr>
<td>Authorizations</td>
<td>198</td>
</tr>
<tr>
<td>Authorization Naming Conventions</td>
<td>198</td>
</tr>
<tr>
<td>Delegation Authority in Authorizations</td>
<td>198</td>
</tr>
<tr>
<td>RBAC Databases</td>
<td>199</td>
</tr>
<tr>
<td>RBAC Databases and the Naming Services</td>
<td>199</td>
</tr>
<tr>
<td>user_attr Database</td>
<td>200</td>
</tr>
<tr>
<td>auth_attr Database</td>
<td>200</td>
</tr>
<tr>
<td>prof_attr Database</td>
<td>201</td>
</tr>
<tr>
<td>exec_attr Database</td>
<td>201</td>
</tr>
<tr>
<td>policy.conf File</td>
<td>201</td>
</tr>
<tr>
<td>RBAC Commands</td>
<td>202</td>
</tr>
<tr>
<td>Commands That Manage RBAC</td>
<td>202</td>
</tr>
<tr>
<td>Selected Commands That Require Authorizations</td>
<td>203</td>
</tr>
<tr>
<td>Privileges</td>
<td>204</td>
</tr>
<tr>
<td>Administrative Commands for Handling Privileges</td>
<td>204</td>
</tr>
<tr>
<td>Files With Privilege Information</td>
<td>205</td>
</tr>
<tr>
<td>Privileges and Auditing</td>
<td>205</td>
</tr>
<tr>
<td>Prevention of Privilege Escalation</td>
<td>206</td>
</tr>
<tr>
<td>Legacy Applications and the Privilege Model</td>
<td>207</td>
</tr>
<tr>
<td>Part IV Cryptographic Services</td>
<td>209</td>
</tr>
<tr>
<td>11 Cryptographic Framework (Overview)</td>
<td>211</td>
</tr>
<tr>
<td>Introduction to the Cryptographic Framework</td>
<td>211</td>
</tr>
<tr>
<td>Terminology in the Cryptographic Framework</td>
<td>213</td>
</tr>
<tr>
<td>Scope of the Cryptographic Framework</td>
<td>215</td>
</tr>
<tr>
<td>Administrative Commands in the Cryptographic Framework</td>
<td>215</td>
</tr>
<tr>
<td>User-Level Commands in the Cryptographic Framework</td>
<td>216</td>
</tr>
<tr>
<td>Plugins to the Cryptographic Framework</td>
<td>216</td>
</tr>
<tr>
<td>Cryptographic Services and Zones</td>
<td>217</td>
</tr>
<tr>
<td>Cryptographic Framework and FIPS-140</td>
<td>217</td>
</tr>
</tbody>
</table>
### Part V  Authentication Services and Secure Communication  ................................................................. 263

#### 14 Using Pluggable Authentication Modules  ................................................................................. 265

- **PAM (Overview)** ....................................................................................................................... 265
- Benefits of Using PAM .................................................................................................................. 265
- Introduction to the PAM Framework .............................................................................................. 266
- Changes to PAM for This Release .................................................................................................. 267
- **PAM (Tasks)** ............................................................................................................................. 267
  - PAM (Task Map) ......................................................................................................................... 268
  - Planning for Your PAM Implementation ..................................................................................... 268
  - ▼ How to Add a PAM Module ...................................................................................................... 269
  - ▼ How to Prevent Rhost-Style Access From Remote Systems With PAM .............................. 270
  - ▼ How to Log PAM Error Reports ............................................................................................... 270
  - ▼ How to Assign a Customized PAM Policy to a User .............................................................. 271
  - ▼ How to Assign a New Rights Policy to All Users .................................................................... 272
- **PAM Configuration (Reference)** .............................................................................................. 272
  - PAM Configuration Search Order ............................................................................................. 273
  - PAM Configuration File Syntax ................................................................................................. 273
  - Per User Authentication Policy .................................................................................................... 274
  - How PAM Stacking Works .......................................................................................................... 275
  - PAM Stacking Example ............................................................................................................... 278

#### 15 Using Secure Shell  .................................................................................................................... 281

- **Secure Shell (Overview)** ......................................................................................................... 281
- Secure Shell Authentication ............................................................................................................ 282
- Secure Shell in the Enterprise ........................................................................................................ 283
- Secure Shell and the OpenSSH Project ........................................................................................ 283
- Secure Shell and FIPS-140 ............................................................................................................ 285
- **Configuring Secure Shell (Tasks)** ............................................................................................ 285
  - Configuring Secure Shell (Task Map) ......................................................................................... 285
  - ▼ How to Set Up Host-Based Authentication for Secure Shell ................................................ 286
  - ▼ How to Configure Port Forwarding in Secure Shell ............................................................... 288
  - ▼ How to Create User and Host Exceptions to Secure Shell Defaults .................................. 289
  - ▼ How to Create an Isolated Directory for sftp Files ................................................................. 290
- **Using Secure Shell (Tasks)** ....................................................................................................... 292
## Contents

### Oracle Solaris 11.1 Administration: Security Services • February 2014

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Using Secure Shell (Task Map)</td>
<td>292</td>
</tr>
<tr>
<td>▼ How to Generate a Public/Private Key Pair for Use With Secure Shell</td>
<td>292</td>
</tr>
<tr>
<td>▼ How to Change the Passphrase for a Secure Shell Private Key</td>
<td>294</td>
</tr>
<tr>
<td>▼ How to Log In to a Remote Host With Secure Shell</td>
<td>295</td>
</tr>
<tr>
<td>▼ How to Reduce Password Prompts in Secure Shell</td>
<td>296</td>
</tr>
<tr>
<td>▼ How to Remotely Administer ZFS With Secure Shell</td>
<td>297</td>
</tr>
<tr>
<td>▼ How to Use Port Forwarding in Secure Shell</td>
<td>299</td>
</tr>
<tr>
<td>▼ How to Copy Files With Secure Shell</td>
<td>300</td>
</tr>
<tr>
<td>▼ How to Set Up Default Secure Shell Connections to Hosts Outside a Firewall</td>
<td>301</td>
</tr>
<tr>
<td><strong>16 Secure Shell (Reference)</strong></td>
<td>303</td>
</tr>
<tr>
<td>A Typical Secure Shell Session</td>
<td>303</td>
</tr>
<tr>
<td>Session Characteristics in Secure Shell</td>
<td>303</td>
</tr>
<tr>
<td>Authentication and Key Exchange in Secure Shell</td>
<td>304</td>
</tr>
<tr>
<td>Command Execution and Data Forwarding in Secure Shell</td>
<td>305</td>
</tr>
<tr>
<td>Client and Server Configuration in Secure Shell</td>
<td>305</td>
</tr>
<tr>
<td>Client Configuration in Secure Shell</td>
<td>305</td>
</tr>
<tr>
<td>Server Configuration in Secure Shell</td>
<td>306</td>
</tr>
<tr>
<td>Keywords in Secure Shell</td>
<td>306</td>
</tr>
<tr>
<td>Host-Specific Parameters in Secure Shell</td>
<td>310</td>
</tr>
<tr>
<td>Secure Shell and Login Environment Variables</td>
<td>310</td>
</tr>
<tr>
<td>Maintaining Known Hosts in Secure Shell</td>
<td>311</td>
</tr>
<tr>
<td>Secure Shell Files</td>
<td>312</td>
</tr>
<tr>
<td>Secure Shell Commands</td>
<td>314</td>
</tr>
<tr>
<td><strong>17 Using Simple Authentication and Security Layer</strong></td>
<td>317</td>
</tr>
<tr>
<td>SASL (Overview)</td>
<td>317</td>
</tr>
<tr>
<td>SASL (Reference)</td>
<td>318</td>
</tr>
<tr>
<td>SASL Plug-ins</td>
<td>318</td>
</tr>
<tr>
<td>SASL Environment Variable</td>
<td>318</td>
</tr>
<tr>
<td>SASL Options</td>
<td>319</td>
</tr>
<tr>
<td><strong>18 Network Services Authentication (Tasks)</strong></td>
<td>321</td>
</tr>
<tr>
<td>Overview of Secure RPC</td>
<td>321</td>
</tr>
</tbody>
</table>
NFS Services and Secure RPC ........................................................................ 321
Kerberos Authentication ........................................................................... 322
DES Encryption With Secure NFS ............................................................... 322
Diffie-Hellman Authentication and Secure RPC ........................................ 322
Administering Authentication With Secure RPC (Tasks) ......................... 326
Administering Secure RPC (Task Map) .................................................... 326
▼ How to Restart the Secure RPC Keyservers ....................................... 326
▼ How to Set Up a Diffie-Hellman Key for an NIS Host ......................... 326
▼ How to Set Up a Diffie-Hellman Key for an NIS User ......................... 327
▼ How to Share NFS Files With Diffie-Hellman Authentication ............ 328

Part VI Kerberos Service ......................................................................... 331

19 Introduction to the Kerberos Service .................................................. 333
What Is the Kerberos Service? ................................................................. 333
How the Kerberos Service Works ............................................................ 334
   Initial Authentication: the Ticket-Granting Ticket ............................... 335
   Subsequent Kerberos Authentications ................................................. 336
   Kerberos Remote Applications ......................................................... 338
   Kerberos Principals ........................................................................ 338
   Kerberos Realms ............................................................................ 339
   Kerberos Servers .......................................................................... 339
   Kerberos Security Services ............................................................ 340
Components of Various Kerberos Releases ........................................... 341
   Kerberos Components ................................................................ 341
   About Kerberos in this Release .................................................... 342

20 Planning for the Kerberos Service ....................................................... 345
Why Plan for Kerberos Deployments? .................................................... 345
Planning Kerberos Realms ................................................................... 346
   Realm Names ............................................................................... 346
   Number of Realms ..................................................................... 346
   Realm Hierarchy ...................................................................... 347
Mapping Host Names Onto Realms ......................................................... 347
## Contents

- Client and Service Principal Names ................................................................. 347
- Ports for the KDC and Admin Services .............................................................. 348
- The Number of Slave KDCs ............................................................................. 348
- Mapping GSS Credentials to UNIX Credentials ............................................... 349
- Automatic User Migration to a Kerberos Realm .............................................. 349
- Which Database Propagation System to Use ..................................................... 350
- Clock Synchronization Within a Realm .............................................................. 350
- Client Configuration Options ........................................................................... 350
- Improving Client Login Security ....................................................................... 351
- KDC Configuration Options ............................................................................ 351
- Trusts of Services for Delegation ...................................................................... 352
- Kerberos Encryption Types ............................................................................... 352
- Online Help URL in the Graphical Kerberos Administration Tool ...................... 353

## 21 Configuring the Kerberos Service (Tasks)

- Configuring the Kerberos Service (Task Map) .................................................. 355
- Configuring Additional Kerberos Services (Task Map) ....................................... 356
- Configuring KDC Servers ................................................................................. 356
  - ▼ How to Automatically Configure a Master KDC ......................................... 357
  - ▼ How to Interactively Configure a Master KDC .......................................... 358
  - ▼ How to Manually Configure a Master KDC ............................................. 359
  - ▼ How to Configure a KDC to Use an LDAP Data Server ............................. 363
  - ▼ How to Automatically Configure a Slave KDC ........................................ 369
  - ▼ How to Interactively Configure a Slave KDC .......................................... 370
  - ▼ How to Manually Configure a Slave KDC ............................................. 371
  - ▼ How to Refresh the Ticket-Granting Service Keys on a Master Server ...... 375
- Configuring Cross-Realm Authentication ......................................................... 376
  - ▼ How to Establish Hierarchical Cross-Realm Authentication ..................... 376
  - ▼ How to Establish Direct Cross-Realm Authentication .............................. 377
- Configuring Kerberos Network Application Servers ........................................ 378
  - ▼ How to Configure a Kerberos Network Application Server ..................... 378
  - ▼ How to Use the Generic Security Service With Kerberos When Running FTP 380
- Configuring Kerberos NFS Servers ................................................................. 381
  - ▼ How to Configure Kerberos NFS Servers ............................................. 382
  - ▼ How to Create a Credential Table ......................................................... 383
Contents

▼ How to Add a Single Entry to the Credential Table ................................................................. 384
▼ How to Provide Credential Mapping Between Realms ........................................................ 384
▼ How to Set Up a Secure NFS Environment With Multiple Kerberos Security Modes ..... 385
Configuring Kerberos Clients .......................................................................................................... 387
  Configuring Kerberos Clients (Task Map) ................................................................................ 387
▼ How to Create a Kerberos Client Installation Profile ............................................................. 388
▼ How to Automatically Configure a Kerberos Client .............................................................. 388
▼ How to Interactively Configure a Kerberos Client ................................................................. 390
▼ How to Configure a Kerberos Client for an Active Directory Server .................................. 393
▼ How to Manually Configure a Kerberos Client .................................................................. 394
▼ How to Disable Verification of the Ticket-Granting Ticket ................................................ 399
▼ How to Access a Kerberos Protected NFS File System as the root User ......................... 400
▼ How to Configure Automatic Migration of Users in a Kerberos Realm ............................... 401
▼ How to Configure Account Lockout ...................................................................................... 404
▼ How to Automatically Renew All Ticket-Granting Tickets (TGTs) ..................................... 404
Synchronizing Clocks Between KDCs and Kerberos Clients ...................................................... 405
  Swapping a Master KDC and a Slave KDC .......................................................................... 407
▼ How to Configure a Swappable Slave KDC ........................................................................ 407
▼ How to Swap a Master KDC and a Slave KDC .................................................................... 408
Administering the Kerberos Database ........................................................................................... 411
  Backing Up and Propagating the Kerberos Database .............................................................. 411
▼ How to Back Up the Kerberos Database ............................................................................. 413
▼ How to Restore the Kerberos Database ............................................................................... 413
▼ How to Convert a Kerberos Database After a Server Upgrade ............................................ 414
▼ How to Reconfigure a Master KDC to Use Incremental Propagation .................................. 415
▼ How to Reconfigure a Slave KDC to Use Incremental Propagation .................................... 417
▼ How to Configure a Slave KDC to Use Full Propagation ................................................... 418
▼ How to Verify That the KDC Servers Are Synchronized ..................................................... 421
▼ How to Manually Propagate the Kerberos Database to the Slave KDCs .............................. 423
  Setting Up Parallel Propagation ............................................................................................. 423
  Configuration Steps for Setting Up Parallel Propagation ..................................................... 424
  Administering the Stash File ................................................................................................... 425
▼ How to Remove a Stash File .................................................................................................. 425
▼ How to Employ a New Master Key ....................................................................................... 426
Managing a KDC on an LDAP Directory Server ......................................................................... 427
▼ How to Mix Kerberos Principal Attributes in a Non-Kerberos Object Class Type ............ 428
Contents

▼ How to Destroy a Realm on an LDAP Directory Server .......................................................... 429
Increasing Security on Kerberos Servers .................................................................................. 429
▼ How to Restrict Access to KDC Servers .................................................................................. 429
▼ How to Use a Dictionary File to Increase Password Security .................................................. 430

22 Kerberos Error Messages and Troubleshooting ..................................................................... 433
Kerberos Error Messages ............................................................................................................ 433
SEAM Tool Error Messages ...................................................................................................... 433
Common Kerberos Error Messages (A-M) ............................................................................. 434
Common Kerberos Error Messages (N-Z) ............................................................................. 443
Kerberos Troubleshooting ......................................................................................................... 447
▼ How to Identify Problems With Key Version Numbers ......................................................... 447
Problems With the Format of the krb5.conf File ..................................................................... 448
Problems Propagating the Kerberos Database ......................................................................... 448
Problems Mounting a Kerberized NFS File System ................................................................. 449
Problems Authenticating as the root User ............................................................................ 449
Observing Mapping From GSS Credentials to UNIX Credentials ............................................ 450
Using DTrace With the Kerberos Service .................................................................................. 450

23 Administering Kerberos Principals and Policies (Tasks) .......................................................... 453
Ways to Administer Kerberos Principals and Policies ............................................................. 453
SEAM Tool ................................................................................................................................ 454
Command-Line Equivalents of the SEAM Tool ..................................................................... 454
The Only File Modified by the SEAM Tool ............................................................................ 455
Print and Online Help Features of the SEAM Tool ................................................................. 455
Working With Large Lists in the SEAM Tool ......................................................................... 456
▼ How to Start the SEAM Tool ................................................................................................. 457
Administering Kerberos Principals .......................................................................................... 458
Administering Kerberos Principals (Task Map) ..................................................................... 458
Automating the Creation of New Kerberos Principals ............................................................ 459
▼ How to View the List of Kerberos Principals ....................................................................... 459
▼ How to View a Kerberos Principal’s Attributes .................................................................... 461
▼ How to Create a New Kerberos Principal ............................................................................. 463
▼ How to Duplicate a Kerberos Principal ................................................................................ 466
▼ How to Modify a Kerberos Principal ..................................................................................... 466
Contents

24 Using Kerberos Applications (Tasks) ................................................................. 491
Kerberos Ticket Management ................................................................. 491
  Do You Need to Worry About Tickets? ..................................................... 491
  Creating a Kerberos Ticket ................................................................. 492
  Viewing Kerberos Tickets ................................................................. 493
  Destroying Kerberos Tickets .............................................................. 494
Kerberos Password Management .......................................................... 495
  Advice on Choosing a Password ......................................................... 495
  Changing Your Password ................................................................. 496
  Granting Access to Your Account ........................................................ 498
Kerberos User Commands ................................................................. 500
  Overview of Kerberized Commands ................................................... 500
  Forwarding Kerberos Tickets .............................................................. 503
  Using Kerberized Commands (Examples) ............................................. 504
25 The Kerberos Service (Reference) ................................................................. 507
   Kerberos Files ........................................................................................................ 507
   Kerberos Commands ............................................................................................... 509
   Kerberos Daemons .................................................................................................. 510
   Kerberos Terminology ............................................................................................ 510
      Kerberos-Specific Terminology ........................................................................... 510
      Authentication-Specific Terminology ................................................................. 511
   Types of Tickets ...................................................................................................... 512
   How the Kerberos Authentication System Works .................................................. 516
   How the Kerberos Service Interacts With DNS and the nsswitch Service ............. 516
   Gaining Access to a Service Using Kerberos ........................................................ 516
      Obtaining a Credential for the Ticket-Granting Service ..................................... 516
      Obtaining a Credential for a Server ..................................................................... 517
      Obtaining Access to a Specific Service ............................................................... 518
   Using Kerberos Encryption Types .......................................................................... 519
   Using the gsscred Table .......................................................................................... 521
   Notable Differences Between Oracle Solaris Kerberos and MIT Kerberos .......... 521

Part VII Auditing in Oracle Solaris ............................................................................. 523

26 Auditing (Overview) ............................................................................................ 525
   What Is Auditing? .................................................................................................... 525
   Audit Terminology and Concepts .......................................................................... 526
      Audit Events ......................................................................................................... 528
      Audit Classes and Preselection .......................................................................... 529
      Audit Records and Audit Tokens ........................................................................ 530
      Audit Plugin Modules ........................................................................................... 531
      Audit Logs ............................................................................................................ 531
      Storing and Managing the Audit Trail ................................................................. 533
      Ensuring Reliable Time Stamps ......................................................................... 534
      Managing a Remote Repository ......................................................................... 534
   How Is Auditing Related to Security? .................................................................... 534
   How Does Auditing Work? .................................................................................... 535
   How Is Auditing Configured? ................................................................................ 536
   Auditing on a System With Oracle Solaris Zones ................................................ 537
Contents

- How to Configure Per-Zone Auditing ................................................................. 581
- Enabling and Disabling the Audit Service (Tasks) .............................................. 582
- How to Refresh the Audit Service ................................................................. 582
- How to Disable the Audit Service ................................................................. 584
- How to Enable the Audit Service ................................................................. 585

Managing Audit Records on Local Systems (Tasks) ........................................ 585
  - Managing Audit Records on Local Systems (Task Map) ................................ 585
- How to Display Audit Record Definitions .................................................... 586
- How to Merge Audit Files From the Audit Trail ........................................... 587
- How to Select Audit Events From the Audit Trail ......................................... 589
- How to View the Contents of Binary Audit Files .......................................... 591
- How to Clean Up a not_terminated Audit File .............................................. 593
- How to Prevent Audit Trail Overflow .......................................................... 594

Troubleshooting the Audit Service (Tasks) ..................................................... 595
  - Troubleshooting the Audit Service (Task Map) ........................................... 595
- How to Determine That Auditing Is Running ............................................... 596
- How to Lessen the Volume of Audit Records That Are Produced .................. 598
- How to Audit All Commands by Users ....................................................... 600
- How to Find Audit Records of Changes to Specific Files ......................... 602
- How to Update the Preselection Mask of Logged In Users ....................... 604
- How to Prevent the Auditing of Specific Events ......................................... 605
- How to Limit the Size of Binary Audit Files ............................................... 606
- How to Compress Audit Files on a Dedicated File System ......................... 606
- How to Audit Logins From Other Operating Systems ............................... 607
- How to Audit FTP and SFTP File Transfers ............................................... 608

29 Auditing (Reference) .................................................................................. 611
- Audit Service .................................................................................................. 611
- Audit Service Man Pages ................................................................................ 612
- Rights Profiles for Administering Auditing ................................................... 614
- Auditing and Oracle Solaris Zones ............................................................... 614
- Audit Configuration Files and Packaging .................................................... 615
- Audit Classes .................................................................................................. 615
  - Audit Class Syntax ...................................................................................... 615
- Audit Plugins .................................................................................................. 616
<table>
<thead>
<tr>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>Audit Remote Server</td>
</tr>
<tr>
<td>Audit Policy</td>
</tr>
<tr>
<td>Audit Policies for Asynchronous and Synchronous Events</td>
</tr>
<tr>
<td>Process Audit Characteristics</td>
</tr>
<tr>
<td>Audit Trail</td>
</tr>
<tr>
<td>Conventions for Binary Audit File Names</td>
</tr>
<tr>
<td>Audit Record Structure</td>
</tr>
<tr>
<td>Audit Record Analysis</td>
</tr>
<tr>
<td>Audit Token Formats</td>
</tr>
<tr>
<td>acl Token</td>
</tr>
<tr>
<td>argument Token</td>
</tr>
<tr>
<td>attribute Token</td>
</tr>
<tr>
<td>cmd Token</td>
</tr>
<tr>
<td>exec_args Token</td>
</tr>
<tr>
<td>exec_env Token</td>
</tr>
<tr>
<td>file Token</td>
</tr>
<tr>
<td>fmri Token</td>
</tr>
<tr>
<td>group Token</td>
</tr>
<tr>
<td>header Token</td>
</tr>
<tr>
<td>ip address Token</td>
</tr>
<tr>
<td>ip port Token</td>
</tr>
<tr>
<td>ipc Token</td>
</tr>
<tr>
<td>IPC_perm Token</td>
</tr>
<tr>
<td>path Token</td>
</tr>
<tr>
<td>path_attr Token</td>
</tr>
<tr>
<td>privilege Token</td>
</tr>
<tr>
<td>process Token</td>
</tr>
<tr>
<td>return Token</td>
</tr>
<tr>
<td>sequence Token</td>
</tr>
<tr>
<td>socket Token</td>
</tr>
<tr>
<td>subject Token</td>
</tr>
<tr>
<td>text Token</td>
</tr>
<tr>
<td>trailer Token</td>
</tr>
<tr>
<td>use of authorization Token</td>
</tr>
<tr>
<td>use of privilege Token</td>
</tr>
<tr>
<td>user Token</td>
</tr>
</tbody>
</table>
Preface

Oracle Solaris 11.1 Administration: Security Services explains how to administer security features on one or more Oracle Solaris systems.

Note – This Oracle Solaris release supports systems that use the SPARC and x86 families of processor architectures. The supported systems appear in the Oracle Solaris OS: Hardware Compatibility Lists. This document cites any implementation differences between the platform types.

Who Should Use This Book

This book is intended for administrators who are responsible for security on Oracle Solaris systems. To use this book, you should have more than two years of UNIX system administration experience. Attending training courses in UNIX system administration might be helpful.

Access to Oracle Support

Oracle customers have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

Typographic Conventions

The following table describes the typographic conventions that are used in this book.

<table>
<thead>
<tr>
<th>Typeface</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>AaBbCc123</td>
<td>The names of commands, files, and directories, and onscreen computer output</td>
<td>Edit your .login file. Use ls -a to list all files. machine_name% you have mail.</td>
</tr>
</tbody>
</table>
Shell Prompts in Command Examples

The following table shows UNIX system prompts and superuser prompts for shells that are included in the Oracle Solaris OS. In command examples, the shell prompt indicates whether the command should be executed by a regular user or a user with privileges.

**TABLE P-2  Shell Prompts**

<table>
<thead>
<tr>
<th>Shell</th>
<th>Prompt</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bash shell, Korn shell, and Bourne shell</td>
<td>$</td>
</tr>
<tr>
<td>Bash shell, Korn shell, and Bourne shell for superuser</td>
<td>#</td>
</tr>
<tr>
<td>C shell</td>
<td>machine_name%</td>
</tr>
<tr>
<td>C shell for superuser</td>
<td>machine_name#</td>
</tr>
</tbody>
</table>
PART I

Security Overview

This book focuses on the features that enhance security in the Oracle Solaris OS. This book is intended for system administrators and users of these security features. Chapter 1, "Security Services (Overview)," introduces the topics in the book.
To maintain the security of the Oracle Solaris OS, the software provides the following features:

- **“System Security” on page 27** – The ability to prevent intrusion, to protect machine resources and devices from misuse, and to protect files from malicious modification or unintentional modification by users or intruders.

- **“Cryptographic Services” on page 28** – The ability to scramble data so that only the sender and the designated receiver can read the contents, and to manage cryptographic providers and public key objects.

- **“Authentication Services” on page 29** – The ability to securely identify a user, which requires the user’s name and some form of proof, typically a password.

- **“Authentication With Encryption” on page 30** – The ability to ensure that authenticated parties can communicate without interception, modification, or spoofing.

- **“Auditing” on page 30** – The ability to identify the source of security changes to the system, including file access, security-related system calls, and authentication failures.

- **“Security Policy” on page 30** – The design and implementation of security guidelines for a system or network of systems.

### System Security

System security ensures that the system's resources are used properly. Access controls can restrict who is permitted access to resources on the system. Oracle Solaris features for system security and access control include the following:

- **Login administration tools** – Commands for monitoring and controlling a user’s ability to log in. See “Securing Logins and Passwords (Task Map)” on page 53.

- **Hardware access** – Commands for limiting access to the PROM, and for restricting who can boot the system. See “Controlling Access to System Hardware (Tasks)” on page 62.
Cryptographic Services

- **Resource access** – Tools and strategies for maximizing the appropriate use of machine resources while minimizing the misuse of those resources. See “Controlling Access to Machine Resources” on page 42.


- **Role-based access control (RBAC)** – An architecture for creating special, restricted user accounts that are permitted to perform specific administrative tasks. See “Role-Based Access Control (Overview)” on page 127.

- **Privileges** – Discrete rights on processes to perform operations. These process rights are enforced in the kernel. See “Privileges (Overview)” on page 138.

- **Device management** – Device policy additionally protects devices that are already protected by UNIX permissions. Device allocation controls access to peripheral devices, such as a microphone or CD-ROM drive. Upon deallocation, device-clean scripts can then erase any data from the device. See “Controlling Access to Devices” on page 40.

- **BART file verification** – A snapshot, called a manifest, of the file attributes of files on a system. By comparing the manifests across systems or on one system over time, changes to files can be monitored to reduce security risks. See Chapter 6, “Verifying File Integrity by Using BART (Tasks).”

- **File permissions** – Attributes of a file or directory. Permissions restrict the users and groups that are permitted to read, write, or execute a file, or search a directory. See Chapter 7, “Controlling Access to Files (Tasks).”

- **Antivirus software** – A vscan service checks files for viruses before an application uses the files. A file system can invoke this service to scan files in real time for the most recent virus definitions before the files are accessed by any clients of the file system.

  The real-time scan is performed by third-party applications. A file can be scanned when it is opened and after it is closed. See Chapter 4, “Virus Scanning Service (Tasks).”

**Cryptographic Services**

Cryptography is the science of encrypting and decrypting data. Cryptography is used to insure integrity, privacy, and authenticity. Integrity means that the data has not been altered. Privacy means that the data is not readable by others. Authenticity for data means that what was delivered is what was sent. User authentication means that the user has supplied one or more proofs of identity. Authentication mechanisms mathematically verify the source of the data or the proof of identity. Encryption mechanisms scramble data so that the data is not readable by a casual observer. Cryptographic services provide authentication and encryption mechanisms to applications and users.

- **Cryptographic Framework** – A central framework of cryptographic services for kernel-level and user-level consumers that is based on the following standard: RSA Security Inc. PKCS #11 Cryptographic Token Interface (Cryptoki). Uses include passwords, IPsec,
and third-party applications. The framework centralizes hardware and software sources for encryption. The PKCS #11 library provides an API for third-party developers to plug in the cryptographic requirements for their applications. See Chapter 11, “Cryptographic Framework (Overview).”

- **Encryption mechanisms per application** –
  - For the use of DES in Secure RPC, see “Overview of Secure RPC” on page 321.
  - For the use of DES, 3DES, AES, and ARCFOUR in the Kerberos service, see Chapter 19, “Introduction to the Kerberos Service.”
  - For the use of RSA, DSA, and ciphers such as AES in Secure Shell, see Chapter 15, “Using Secure Shell.”
  - For the use of cryptographic algorithms in passwords, see “Changing the Default Algorithm for Password Encryption (Tasks)” on page 57.

- The Key Management Framework (KMF) provides a central utility for managing public key objects, including policy, keys, and certificates. KMF manages these objects for OpenSSL, NSS, and PKCS #11 public key technologies. See Chapter 13, “Key Management Framework.”

### Authentication Services

Authentication is a mechanism that identifies a user or service based on predefined criteria. Authentication services range from simple name-password pairs to more elaborate challenge-response systems, such as token cards and biometrics. Strong authentication mechanisms rely on a user supplying information that only that person knows, and a personal item that can be verified. A user name is an example of information that the person knows. A smart card or a fingerprint, for example, can be verified. The Oracle Solaris features for authentication include the following:

- **Secure RPC** – An authentication mechanism that uses the Diffie-Hellman protocol to protect NFS mounts and a naming service, such as NIS. See “Overview of Secure RPC” on page 321.

- **Pluggable Authentication Module (PAM)** – A framework that enables various authentication technologies to be plugged into a system entry service without recompiling the service. Some of the system entry services include `login` and `ftp`. See Chapter 14, “Using Pluggable Authentication Modules.”

- **Simple Authentication and Security Layer (SASL)** – A framework that provides authentication and security services to network protocols. See Chapter 17, “Using Simple Authentication and Security Layer.”

- **Secure Shell** – A secure remote login and transfer protocol that encrypts communications over an insecure network. See Chapter 15, “Using Secure Shell.”

- **Kerberos service** – A client-server architecture that provides encryption with authentication over a secured network. See Part VI, “Kerberos Service.”
Authentication With Encryption

Authentication with encryption is the basis of secure communication. Authentication helps ensure that the source and the destination are the intended parties. Encryption codes the communication at the source, and decodes the communication at the destination. Encryption prevents intruders from reading any transmissions that the intruders might manage to intercept. The Oracle Solaris features for secure communication include the following:

- **Secure Shell** – A protocol for protecting data transfers and interactive user network sessions from eavesdropping, session hijacking, and “man-in-the-middle” attacks. Strong authentication is provided through public key cryptography. X windows services and other network services can be tunneled safely over Secure Shell connections for additional protection. See Chapter 15, “Using Secure Shell.”

- **Kerberos service** – A client-server architecture that provides authentication with encryption. See Part VI, “Kerberos Service.”

- **Internet Protocol Security Architecture (IPsec)** – An architecture that provides IP datagram protection. Protections include confidentiality, strong integrity of the data, data authentication, and partial sequence integrity. See Securing the Network in Oracle Solaris 11.1.

Auditing

Auditing is a fundamental concept of system security and maintainability. Auditing is the process of examining the history of actions and events on a system to determine what happened. The history is kept in a log of what was done, when it was done, by whom, and what was affected. See Part VII, “Auditing in Oracle Solaris.”

Security Policy

The phrase security policy, or policy, is used throughout this book to refer to an organization’s security guidelines. Your site’s security policy is the set of rules that define the sensitivity of the information that is being processed and the measures that are used to protect the information from unauthorized access. Security technologies such as Secure Shell, authentication, RBAC, authorization, privileges, and resource control provide measures to protect information.

Some security technologies also use the word policy when describing specific aspects of their implementation. For example, Oracle Solaris uses audit policy options to configure some aspects of audit policy. The following table points to glossary, man page, and information about features that use the word policy to describe specific aspects of their implementation.
### Use of the Word “Policy” in Oracle Solaris

<table>
<thead>
<tr>
<th>&quot;Policy&quot; Term</th>
<th>Selected Man Pages</th>
<th>Further Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>audit policy</td>
<td>auditconfig(1M)</td>
<td>Chapter 26, &quot;Auditing (Overview)&quot;</td>
</tr>
<tr>
<td>policy in the Cryptographic Framework</td>
<td>cryptoadm(1M)</td>
<td>Chapter 11, “Cryptographic Framework (Overview)”</td>
</tr>
<tr>
<td>device policy</td>
<td>getdevpolicy(1M)</td>
<td>“Controlling Access to Devices” on page 40</td>
</tr>
<tr>
<td>Kerberos policy</td>
<td>krb5.conf(4)</td>
<td>Chapter 23, “Administering Kerberos Principals and Policies (Tasks)”</td>
</tr>
<tr>
<td>network policies</td>
<td>ipfilter(5), ipadm(1M),</td>
<td>Securing the Network in Oracle Solaris 11.1</td>
</tr>
<tr>
<td></td>
<td>ike.config(4), ipsecconf(1M),</td>
<td></td>
</tr>
<tr>
<td></td>
<td>routeadm(1M)</td>
<td></td>
</tr>
<tr>
<td>password policy</td>
<td>passwd(1), crypt.conf(4),</td>
<td>“Maintaining Login Control” on page 36</td>
</tr>
<tr>
<td></td>
<td>pam.conf(4), policy.conf(4)</td>
<td>“How to Assign a Customized PAM Policy to a User” on page 271</td>
</tr>
<tr>
<td>policy for public key</td>
<td>kmfcfg(1)</td>
<td>Chapter 13, “Key Management Framework”</td>
</tr>
<tr>
<td>technologies</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RBAC policy</td>
<td>rbac(5), policy.conf(4)</td>
<td>“policy.conf File” on page 201</td>
</tr>
</tbody>
</table>
This section covers security that can be configured on a non-networked system. The chapters discuss planning, monitoring, and controlling access to the disk, to files, and to peripheral devices.

- Chapter 2, “Managing Machine Security (Overview)”
- Chapter 3, “Controlling Access to Systems (Tasks)”
- Chapter 4, “Virus Scanning Service (Tasks)”
- Chapter 5, “Controlling Access to Devices (Tasks)”
- Chapter 6, “Verifying File Integrity by Using BART (Tasks)”
- Chapter 7, “Controlling Access to Files (Tasks)”
Managing Machine Security (Overview)

Keeping a machine’s information secure is an important system administration responsibility. This chapter provides overview information about managing machine security.

- “Controlling Access to a Computer System” on page 35
- “Controlling Access to Devices” on page 40
- “Controlling Access to Machine Resources” on page 42
- “Controlling Access to Files” on page 47
- “Controlling Network Access” on page 48
- “Reporting Security Problems” on page 52

## Controlling Access to a Computer System

In the workplace, all computers that are connected to a server can be thought of as one large multifaceted system. You are responsible for the security of this larger system. You need to defend the network from outsiders who are trying to gain access. You also need to ensure the integrity of the data on the computers within the network.

At the file level, Oracle Solaris provides standard security features that you can use to protect files, directories, and devices. At the system and network levels, the security issues are mostly the same. The first line of security defense is to control access to your system, as described in the following sections.

## Maintaining Physical Security

To control access to your system, you must maintain the physical security of your computing environment. For instance, a system that is logged in and left unattended is vulnerable to unauthorized access. An intruder can gain access to the operating system and to the network. The computer’s surroundings and the computer hardware must be physically protected from unauthorized access.
You can protect a SPARC system from unauthorized access to the hardware settings. Use the `eprom` command to require a password to access the PROM. For more information, see “How to Require a Password for SPARC Hardware Access” on page 62. To protect x86 hardware, consult the vendor documentation.

### Maintaining Login Control

You also must prevent unauthorized logins to a system or the network, which you can do through password assignment and login control. All accounts on a system must have a password. A password is a simple authentication mechanism. An account without a password makes your entire network accessible to an intruder who guesses a user name. A strong password algorithm protects against brute force attacks.

When a user logs in to a system, the `login` command checks the appropriate naming service or directory service database according to the information in the name switch service, `svc:/system/name-service/switch`. The following databases can affect login:

- `files` – Designates the `/etc` files on the local system
- `ldap` – Designates the LDAP directory service on the LDAP server
- `nis` – Designates the NIS database on the NIS master server
- `dns` – Designates the domain name service on the network

For a description of the naming service, see the `nscd(1M)` man page. For information about naming services and directory services, see Oracle Solaris Administration: Naming and Directory Services.

The `login` command verifies the user name and password that were supplied by the user. If the user name is not in the password database, the `login` command denies access to the system. If the password is not correct for the user name that was specified, the `login` command denies access to the system. When the user supplies a valid user name and its corresponding password, the system grants the user access to the system.

PAM modules can streamline login to applications after a successful system login. For more information, see Chapter 14, "Using Pluggable Authentication Modules."

Sophisticated authentication and authorization mechanisms are available on Oracle Solaris systems. For a discussion of authentication and authorization mechanisms at the network level, see "Authentication and Authorization for Remote Access" on page 49.

### Managing Password Information

When users log in to a system, they must supply both a user name and a password. Although logins are publicly known, passwords must be kept secret. Passwords should be known only to each user. Users must choose their passwords carefully and change them often.
Passwords are initially created when you set up a user account. To maintain security on user accounts, you can set up password aging to force users to routinely change their passwords. You can also disable a user account by locking the password. For detailed information about administering passwords, see Chapter 1, “Managing User Accounts and User Environments (Overview),” in Managing User Accounts and User Environments in Oracle Solaris 11.1 and the passwd(1) man page.

Local Passwords

If your network uses local files to authenticate users, the password information is kept in the system's /etc/passwd and /etc/shadow files. The user name and other information are kept in the /etc/passwd file. The encrypted password itself is kept in a separate shadow file, /etc/shadow. This security measure prevents a user from gaining access to the encrypted passwords. While the /etc/passwd file is available to anyone who can log in to a system, only the root account can read the /etc/shadow file. You can use the passwd command to change a user's password on a local system.

NIS Passwords

If your network uses NIS to authenticate users, password information is kept in the NIS password map. NIS does not support password aging. You can use the command passwd -r nis to change a user's password that is stored in an NIS password map.

LDAP Passwords

The Oracle Solaris LDAP naming service stores password information and shadow information in the ou=people container of the LDAP directory tree. On the Oracle Solaris LDAP naming service client, you can use the passwd -r ldap command to change a user's password. The LDAP naming service stores the password in the LDAP repository.

Password policy is enforced on the Oracle Directory Server Enterprise Edition. Specifically, the client's pam_ldap module follows the password policy controls that are enforced on the Oracle Directory Server Enterprise Edition. For more information, see “LDAP Naming Services Security Model” in Working With Naming and Directory Services in Oracle Solaris 11.1.

Password Encryption

Strong password encryption provides an early barrier against attack. Oracle Solaris software provides six password encryption algorithms. The Blowfish, MD5, and SHA algorithms provide robust password encryption.

Password Algorithm Identifiers

You specify the algorithms configuration for your site in the /etc/security/policy.conf file. In the policy.conf file, the algorithms are named by their identifier, as shown in the following table. For the identifier-algorithm mapping, see the /etc/security/crypt.conf file.


### TABLE 2-1 Password Encryption Algorithms

<table>
<thead>
<tr>
<th>Identifier</th>
<th>Description</th>
<th>Algorithm Man Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>The MD5 algorithm that is compatible with MD5 algorithms on BSD and Linux systems.</td>
<td><code>crypt_bsdmd5(5)</code></td>
</tr>
<tr>
<td>2a</td>
<td>The Blowfish algorithm that is compatible with the Blowfish algorithm on BSD systems.</td>
<td><code>crypt_bsdbf(5)</code></td>
</tr>
<tr>
<td>md5</td>
<td>The Sun MD5 algorithm, which is considered stronger than the BSD and Linux version of MD5.</td>
<td><code>crypt_sunmd5(5)</code></td>
</tr>
<tr>
<td>5</td>
<td>The SHA256 algorithm. SHA stands for Secure Hash Algorithm. This algorithm is a member of the SHA-2 family. SHA256 supports 255-character passwords.</td>
<td><code>crypt_sha256(5)</code></td>
</tr>
<tr>
<td>6</td>
<td>The SHA512 algorithm.</td>
<td><code>crypt_sha512(5)</code></td>
</tr>
<tr>
<td><strong>unix</strong></td>
<td>Deprecated. The traditional UNIX encryption algorithm. This algorithm can be of use when connecting to old systems.</td>
<td><code>crypt_unix(5)</code></td>
</tr>
</tbody>
</table>

### Algorithms Configuration in the policy.conf File

The following shows the default algorithms configuration in the policy.conf file:

```
#...
# crypt(3c) Algorithms Configuration
#
# CRYPT_ALGORITHMS_ALLOW specifies the algorithms that are allowed to be used for new passwords. This is enforced only in crypt_gensalt(3c).
# CRYPT_ALGORITHMS_ALLOW=1,2a,md5,5,6
#
# To deprecate use of the traditional unix algorithm, uncomment below and change CRYPT_DEFAULT= to another algorithm. For example,
# CRYPT_DEFAULT=1 for BSD/Linux MD5.
#
# CRYPT_ALGORITHMS_DEPRECATE=__unix__
#
# The Oracle Solaris default is a SHA256 based algorithm. To revert to the policy present in Solaris releases set CRYPT_DEFAULT=__unix__, which is not listed in crypt.conf(4) since it is internal to libc.
# CRYPT_DEFAULT=5
#...
```

When you change the value for CRYPT_DEFAULT, the passwords of new users are encrypted with the algorithm that is associated with the new value.

When existing users change their passwords, how their old password was encrypted affects which algorithm is used to encrypt the new password. For example, assume that
CRYPT_ALGORITHMS_ALLOW=1, 2a, md5, 5, 6, and CRYPT_DEFAULT=1. The following table shows which algorithm would be used to generate the encrypted password.

<table>
<thead>
<tr>
<th>Identifier</th>
<th>Password Algorithm</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>crypt_bsdmd5</td>
<td>The 1 identifier is also the value of CRYPT_DEFAULT. The user’s password continues to be encrypted with the crypt_bsdmd5 algorithm.</td>
</tr>
<tr>
<td>2a</td>
<td>crypt_bsdbf</td>
<td>The 2a identifier is in the CRYPT_ALGORITHMS_ALLOW list. Therefore, the new password is encrypted with the crypt_bsdbf algorithm.</td>
</tr>
<tr>
<td>md5</td>
<td>crypt_md5</td>
<td>The md5 identifier is in the CRYPT_ALGORITHMS_ALLOW list. Therefore, the new password is encrypted with the crypt_md5 algorithm.</td>
</tr>
<tr>
<td>5</td>
<td>crypt_sha256</td>
<td>The 5 identifier is in the CRYPT_ALGORITHMS_ALLOW list. Therefore, the new password is encrypted with the crypt_sha256 algorithm.</td>
</tr>
<tr>
<td>6</td>
<td>crypt_sha512</td>
<td>The 6 identifier is in the CRYPT_ALGORITHMS_ALLOW list. Therefore, the new password is encrypted with the crypt_sha512 algorithm.</td>
</tr>
<tr>
<td><em>unix</em></td>
<td>crypt_unix</td>
<td>The <em>unix</em> identifier is not in the CRYPT_ALGORITHMS_ALLOW list. Therefore, the crypt_unix algorithm cannot be used. The new password is encrypted with the CRYPT_DEFAULT algorithm.</td>
</tr>
</tbody>
</table>

For more information about configuring the algorithm choices, see the policy.conf(4) man page. To specify password encryption algorithms, see “Changing the Default Algorithm for Password Encryption (Tasks)” on page 57.

Special System Accounts

The root account is one of several special system accounts. Of these accounts, only the root account is assigned a password and can log in. The nuucp account can log in for file transfers. The other system accounts either protect files or run administrative processes without using the full powers of root.

Caution – Never change the password setting of a system account. System accounts from Oracle Solaris are delivered in a safe and secure state.

The following table lists some system accounts and their uses. The system accounts perform special functions. Each account has a UID that is less than 100.
TABLE 2–2   Selected System Accounts and Their Uses

<table>
<thead>
<tr>
<th>System Account</th>
<th>UID</th>
<th>Use</th>
</tr>
</thead>
<tbody>
<tr>
<td>root</td>
<td>0</td>
<td>Has almost no restrictions. Can override other protections and permissions. The root account has access to the entire system. The password for the root account should be very carefully protected. The root account owns most of the Oracle Solaris commands.</td>
</tr>
<tr>
<td>daemon</td>
<td>1</td>
<td>Controls background processing.</td>
</tr>
<tr>
<td>bin</td>
<td>2</td>
<td>Owns some Oracle Solaris commands.</td>
</tr>
<tr>
<td>sys</td>
<td>3</td>
<td>Owns many system files.</td>
</tr>
<tr>
<td>adm</td>
<td>4</td>
<td>Owns some administrative files.</td>
</tr>
<tr>
<td>lp</td>
<td>71</td>
<td>Owns the object data files and spooled data files for the printer.</td>
</tr>
<tr>
<td>uucp</td>
<td>5</td>
<td>Owns the object data files and spooled data files for UUCP, the UNIX-to-UNIX copy program.</td>
</tr>
<tr>
<td>nuucp</td>
<td>9</td>
<td>Is used by remote systems to log in to the system and start file transfers.</td>
</tr>
</tbody>
</table>

Remote Logins

Remote logins offer a tempting avenue for intruders. Oracle Solaris provides several commands to monitor, limit, and disable remote logins. For procedures, see “Securing Logins and Passwords (Task Map)” on page 53.

By default, remote logins cannot gain control or read certain system devices, such as the system mouse, keyboard, frame buffer, or audio device. For more information, see the `logindevperm(4)` man page.

Controlling Access to Devices

Peripheral devices that are attached to a computer system pose a security risk. Microphones can pick up conversations and transmit them to remote systems. CD-ROMs can leave their information behind for reading by the next user of the CD-ROM device. Printers can be accessed remotely. Devices that are integral to the system can also present security issues. For example, network interfaces such as `bge0` are considered integral devices.
Oracle Solaris software provides several methods of controlling access to devices.

- **Set device policy** – You can require that the process that is accessing a particular device be running with a set of privileges. Processes without those privileges cannot use the device. At boot time, Oracle Solaris software configures device policy. Third-party drivers can be configured with device policy during installation. After installation, you, as the administrator, can add device policy to a device.

- **Make devices allocatable** – You can require that a user must allocate a device before use. Allocation restricts the use of a device to one user at a time. You can further require that the user be authorized to use the device.

- **Prevent devices from being used** – You can prevent the use of a device, such as a microphone, by any user on a computer system. A computer kiosk might be a good candidate for making certain devices unavailable for use.

- **Confine a device to a particular zone** – You can assign the use of a device to a non-global zone. For more information, see “Device Use in Non-Global Zones” in Oracle Solaris 11.1 Administration: Oracle Solaris Zones, Oracle Solaris 10 Zones, and Resource Management. For a more general discussion of devices and zones, see “/dev File System in Non-Global Zones” in Oracle Solaris 11.1 Administration: Oracle Solaris Zones, Oracle Solaris 10 Zones, and Resource Management.

---

**Device Policy (Overview)**

The device policy mechanism enables you to specify that processes that open a device require certain privileges. Devices that are protected by device policy can only be accessed by processes that are running with the privileges that the device policy specifies. Oracle Solaris provides default device policy. For example, network interfaces such as `bge0` require that the processes that access the interface be running with the `net_rawaccess` privilege. The requirement is enforced in the kernel. For more information about privileges, see “Privileges (Overview)” on page 138.

In Oracle Solaris, devices are protected with file permissions *and* with device policy. For example, the `/dev/1p` file has 666 permissions. However, the device can only be opened by a process with the appropriate privileges.

The configuration of device policy can be audited. The `AUE_MODDEVPLCY` audit event records changes in device policy.

For more information about device policy, see the following:

- “Configuring Device Policy (Task Map)” on page 71
- “Device Policy Commands” on page 82
- “Privileges and Devices” on page 145
Device Allocation (Overview)

The device allocation mechanism enables you to restrict access to a peripheral device, such as a CD-ROM. If device allocation is not enabled, peripheral devices are protected only by file permissions. For example, by default, peripheral devices are available for the following uses:

- Any user can read and write to a CD-ROM drive or diskette.
- Any user can attach a microphone.
- Any user can access an attached printer.

Device allocation can restrict a device to authorized users. Device allocation can also prevent a device from being accessed at all. A user who allocates a device has exclusive use of that device until the user deallocates the device. When a device is deallocated, device-cleanup scripts erase any leftover data. You can write a device-cleanup script to purge information from devices that do not have a script. For an example, see “Writing New Device-Cleanup Scripts” on page 88.

Attempts to allocate a device, deallocate a device, and list allocatable devices can be audited. The audit events are part of the other audit class.

For more information about device allocation, see the following:

- “Managing Device Allocation (Task Map)” on page 73
- “Device Allocation” on page 83
- “Device Allocation Commands” on page 84

Controlling Access to Machine Resources

Some system resources are protected by default. Additionally, as system administrator, you can control and monitor system activity. You can set limits on who can use what resources. You can log resource use, and you can monitor who is using the resources. You can also set up your systems to minimize improper use of resources.

Address Space Layout Randomization

Oracle Solaris tags many of its userland binaries to enable address space layout randomization (ASLR). ASLR randomizes the starting address of key parts of an address space. This security defense mechanism can cause Return Oriented Programming (ROP) attacks to fail when they try to exploit software vulnerabilities.

Zones inherit this randomized layout for their processes. Because the use of ASLR might not be optimal for all binaries, the use of ASLR is configurable at the zone level and at the binary level.
Three ASLR configurations are possible:

- **Disabled** – ASLR is disabled for all binaries.
- **Tagged binaries** – ASLR is controlled by the tag that is coded in the binaries. The default Oracle Solaris value for ASLR is `tagged-binaries`. Many binaries in the Oracle Solaris release are tagged to use ASLR.
- **Enabled** – ASLR is enabled for all binaries, except for those that are explicitly tagged to disable it.

The `sxadm` command is used to configure ASLR. You must assume the `root` role to run this command. For examples and information, see the `sxadm(1M)` man page. For developer assistance, see *Developer’s Guide to Oracle Solaris 11 Security*.

### Limiting and Monitoring Superuser Access

Your system requires a `root` password for superuser access. In the default configuration, a user cannot remotely log in to a system as `root`. When logging in remotely, a user must log in with the user’s user name and then use the `su` command to become `root`. You can monitor who has been using the `su` command, especially those users who are trying to gain superuser access. For procedures that monitor superuser and limit access to superuser, see “Monitoring and Restricting root Access (Tasks)” on page 60.

### Configuring Role-Based Access Control to Replace Superuser

Role-based access control (RBAC), a feature of Oracle Solaris, is designed to distribute the capabilities of superuser to administrative roles. Superuser, the `root` user, has access to every resource in the system. With RBAC, you can replace many of `root`’s responsibilities with a set of roles with discrete powers. For example, you can set up one role to handle user account creation and another role to handle system file modification. Although you might not modify the `root` account, you can leave the account as a role, then not assign the role. This strategy effectively removes `root` access to the system.

Each role requires that a known user log in with her or his user name and password. After logging in, the user then assumes the role with a specific role password. For more information about RBAC, see “Role-Based Access Control (Overview)” on page 127.
Preventing Unintentional Misuse of System Resources

You can prevent you and your users from making unintentional errors in the following ways:

- You can keep from running a Trojan horse by correctly setting the `PATH` variable.
- You can assign a restricted shell to users. A restricted shell prevents user error by steering users to those parts of the system that the users need for their jobs. In fact, through careful setup, you can ensure that users access only those parts of the system that help the users work efficiently.
- You can set restrictive permissions on files that users do not need to access.

Setting the PATH Variable

You should take care to correctly set the `PATH` variable. Otherwise, you can accidentally run a program that was introduced by someone else. The intruding program can corrupt your data or harm your system. This kind of program, which creates a security hazard, is referred to as a Trojan horse. For example, a substitute `su` program could be placed in a public directory where you, as system administrator, might run the substitute program. Such a script would look just like the regular `su` command. Because the script removes itself after execution, you would have little evidence to show that you have actually run a Trojan horse.

The `PATH` variable is automatically set at login time. The path is set through your initialization files, such as `.bashrc` and `/etc/profile`. When you set up the user search path so that the current directory (.) comes last, you are protected from running this type of Trojan horse. The `PATH` variable for the root account should not include the current directory at all.

Assigning a Restricted Shell to Users

The standard shell allows a user to open files, execute commands, and so on. The restricted shell limits the ability of a user to change directories and to execute commands. The restricted shell is invoked with the `/usr/lib/rsh` command. Note that the restricted shell is not the remote shell, which is `/usr/sbin/rsh`.

The restricted shell differs from a standard shell in the following ways:

- The user is limited to the user’s home directory, so the user cannot use the `cd` command to change directories. Therefore, the user cannot browse system files.
- The user cannot change the `PATH` variable, so the user can use only commands in the path that is set by the system administrator. The user also cannot execute commands or scripts by using a complete path name.
- The user cannot redirect output with `>` or `>>`.

The restricted shell enables you to limit a user’s ability to stray into system files. The shell creates a limited environment for a user who needs to perform specific tasks. The restricted shell is not completely secure, however, and is only intended to keep unskilled users from inadvertently doing damage.
For information about the restricted shell, use the `man -s1m rsh` command to see the `rsh(1M)` man page.

**Restricting Access to Data in Files**

Because Oracle Solaris is a multiuser environment, file system security is the most basic security risk on a system. You can use traditional UNIX file protections to protect your files. You can also use the more secure access control lists (ACLs).

You might want to allow some users to read some files, and give other users permission to change or delete some files. You might have some data that you do not want anyone else to see. Chapter 7, “Controlling Access to Files (Tasks),” discusses how to set file permissions.

**Restricting setuid Executable Files**

Executable files can be security risks. A few executable programs still have to be run as root to work properly. These `setuid` programs run with the user ID set to 0. Anyone who is running these programs runs the programs with the root ID. A program that runs with the root ID creates a potential security problem if the program was not written with security in mind.

Except for the executables that Oracle ships with the `setuid` bit set to root, you should disallow the use of `setuid` programs. If you cannot disallow the use of `setuid` programs, then you must restrict their use. Secure administration requires few `setuid` programs.

For more information, see “Protecting Executable Files From Compromising Security” on page 114. For procedures, see “Protecting Against Programs With Security Risk (Task Map)” on page 121.

**Using the Secure by Default Configuration**

By default, when Oracle Solaris is installed, a large set of network services are disabled. This configuration is called “Secure by Default” (SBD). With SBD, the only network service that accepts network requests is the `sshd` daemon. All other network services are disabled or handle local requests only. To enable individual network services, such as `ftp`, you use the Service Management Facility (SMF) feature of Oracle Solaris. For more information, see the `netservices(1M)` and `smf(5)` man pages.

**Using Resource Management Features**

Oracle Solaris software provides sophisticated resource management features. Using these features, you can allocate, schedule, monitor, and cap resource use by applications in a server consolidation environment. The resource controls framework enables you to set constraints on system resources that are consumed by processes. Such constraints help to prevent denial-of-service attacks by a script that attempts to flood a system’s resources.
With these resource management features, you can designate resources for particular projects. You can also dynamically adjust the resources that are available. For more information, see Part I, "Oracle Solaris Resource Management," in Oracle Solaris 11.1 Administration: Oracle Solaris Zones, Oracle Solaris 10 Zones, and Resource Management.

**Using Oracle Solaris Zones**

Oracle Solaris zones provide an application execution environment in which processes are isolated from the rest of the system within a single instance of the Oracle Solaris OS. This isolation prevents processes that are running in one zone from monitoring or affecting processes that are running in other zones. Even a process running with superuser capabilities cannot view or affect activity in other zones.

Oracle Solaris zones are ideal for environments that place several applications on a single server. For more information, see Part II, “Oracle Solaris Zones,” in Oracle Solaris 11.1 Administration: Oracle Solaris Zones, Oracle Solaris 10 Zones, and Resource Management.

**Monitoring Use of Machine Resources**

As a system administrator, you need to monitor system activity. You need to be aware of all aspects of your machines, including the following:

- What is the normal load?
- Who has access to the system?
- When do individuals access the system?
- What programs normally run on the system?

With this kind of knowledge, you can use the available tools to audit system use and monitor the activities of individual users. Monitoring is very useful when a breach in security is suspected. For more information about the audit service, see Chapter 26, “Auditing (Overview).”

**Monitoring File Integrity**

As a system administrator, you need assurance that the files that were installed on the systems that you administer have not changed in unexpected ways. In large installations, a comparison and reporting tool about the software stack on each of your systems enables you to track your systems. The Basic Audit Reporting Tool (BART) enables you to comprehensively validate systems by performing file-level checks of one or more systems over time. Changes in a BART manifest across systems, or for one system over time, can validate the integrity of your systems. BART provides manifest creation, manifest comparison, and rules for scripting reports. For more information, see Chapter 6, “Verifying File Integrity by Using BART (Tasks).”
Controlling Access to Files

Oracle Solaris is a multiuser environment. In a multiuser environment, all the users who are logged in to a system can read files that belong to other users. With the appropriate file permissions, users can also use files that belong to other users. For more discussion, see Chapter 7, “Controlling Access to Files (Tasks).” For step-by-step instructions on setting appropriate permissions on files, see “Protecting Files (Tasks)” on page 115.

Protecting Files With Encryption

You can keep a file secure by making the file inaccessible to other users. For example, a file with permissions of 600 cannot be read except by its owner and by the root account. A directory with permissions of 700 is similarly inaccessible. However, someone who guesses your password or who discovers the root password can access that file. Also, the otherwise inaccessible file is preserved on a backup tape every time that the system files are backed up to offline media.

ZFS file systems can be created with on-disk encryption. For more information, see “Encrypting ZFS File Systems” in Oracle Solaris 11.1 Administration: ZFS File Systems.

The Cryptographic Framework provides digest, mac, and encrypt commands. Regular users can use these commands to protect files and directories. For more information, see Chapter 11, “Cryptographic Framework (Overview).”

Using Access Control Lists

ACLs, pronounced “ackkls,” can provide greater control over file permissions. You add ACLs when traditional UNIX file protections are not sufficient. Traditional UNIX file protections provide read, write, and execute permissions for the three user classes: owner, group, and other. An ACL provides finer-grained file security.

ACLs enable you to define fine-grained file permissions, including the following:

- Owner file permissions
- File permissions for the owner’s group
- File permissions for other users who are outside the owner’s group
- File permissions for specific users
- File permissions for specific groups
- Default permissions for each of the previous categories

To protect ZFS files with access control lists (ACLs), see Chapter 7, “Using ACLs and Attributes to Protect Oracle Solaris ZFS Files,” in Oracle Solaris 11.1 Administration: ZFS File Systems. For information about using ACLs on legacy file systems, see “Using Access Control Lists to Protect UFS Files” on page 113.
Sharing Files Across Machines

A network file server can control which files are available for sharing. A network file server can also control which clients have access to the files, and what type of access is permitted for those clients. In general, the file server can grant read-write access or read-only access either to all clients or to specific clients. Access control is specified when resources are made available with the share command.

When you create an NFS share of a ZFS file system, the file system is permanently shared until you remove the share. SMF automatically manages the share when the system is rebooted. For more information, see “Oracle Solaris ZFS and Traditional File System Differences” in Oracle Solaris 11.1 Administration: ZFS File Systems.

Restricting root Access to Shared Files

In general, superuser is not allowed root access to file systems that are shared across the network. The NFS system prevents root access to mounted file systems by changing the user of the requester to the user nobody with the user ID 60001. The access rights of user nobody are the same as those access rights that are given to the public. The user nobody has the access rights of a user without credentials. For example, if the public has only execute permission for a file, then user nobody can only execute that file.

An NFS server can grant root access to a shared file system on a per-host basis. To grant these privileges, use the root=hostname option to the share command. You should use this option with care. For a discussion of security options with NFS, see Chapter 3, “Accessing Network File Systems (Reference),” in Managing Network File Systems in Oracle Solaris 11.1.

Controlling Network Access

Computers are often part of a network of computers. A network allows connected computers to exchange information. Networked computers can access data and other resources from other computers on the network. Computer networks create a powerful and sophisticated computing environment. However, networks complicate computer security.

For example, within a network of computers, individual systems allow the sharing of information. Unauthorized access is a security risk. Because many people have access to a network, unauthorized access is more likely, especially through user error. A poor use of passwords can also allow unauthorized access.
Network Security Mechanisms

Network security is usually based on limiting or blocking operations from remote systems. The following figure describes the security restrictions that you can impose on remote operations.

**FIGURE 2–1  Security Restrictions for Remote Operations**

The firewall restricts the types of remote operations that the systems at a particular site can perform with systems that are outside the firewall.

Remote systems use authentication to restrict access to specific users.

Remote systems use authorization to restrict authenticated users from performing operations on their file systems.

Authentication and Authorization for Remote Access

*Authentication* is a way to restrict access to specific users when these users access a remote system. Authentication can be set up at both the system level and the network level. After a user has gained access to a remote system, *authorization* is a way to restrict operations that the user can perform. The following table lists the services that provide authentication and authorization.
TABLE 2–3 Authentication Services for Remote Access

<table>
<thead>
<tr>
<th>Service</th>
<th>Description</th>
<th>For More Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPSec</td>
<td>IPSec provides host-based and certificate-based authentication and network  traffic encryption.</td>
<td>Chapter 6, “IP Security Architecture (Overview),” in Securing the Network in Oracle Solaris 11.1</td>
</tr>
<tr>
<td>Kerberos</td>
<td>Kerberos uses encryption to authenticate and authorize a user who is logging in to the system.</td>
<td>For an example, see “How the Kerberos Service Works” on page 334.</td>
</tr>
<tr>
<td>LDAP</td>
<td>The LDAP directory service can provide both authentication and authorization at the network level.</td>
<td>Oracle Solaris Administration: Naming and Directory Services</td>
</tr>
<tr>
<td>Remote login commands</td>
<td>The remote login commands enable users to log in to a remote system over the network and use its resources. Some of the remote login commands are rlogin, rcp, and ftp. If you are a “trusted host,” authentication is automatic. Otherwise, you are asked to authenticate yourself.</td>
<td>Chapter 3, “Accessing Remote Systems (Tasks),” in Managing Remote Systems in Oracle Solaris 11.1</td>
</tr>
<tr>
<td>SASL</td>
<td>The Simple Authentication and Security Layer (SASL) is a framework that provides authentication and optional security services to network protocols. Plugins enable you to choose an appropriate authentication protocol.</td>
<td>“SASL (Overview)” on page 317</td>
</tr>
<tr>
<td>Secure RPC</td>
<td>Secure RPC improves the security of network environments by authenticating users who make requests on remote machines. You can use either a UNIX, DES, or Kerberos authentication mechanism for Secure RPC.</td>
<td>“Overview of Secure RPC” on page 321</td>
</tr>
<tr>
<td></td>
<td>Secure RPC can also be used to provide additional security in an NFS environment. An NFS environment with secure RPC is called Secure NFS.</td>
<td>“NFS Services and Secure RPC” on page 321</td>
</tr>
<tr>
<td>Secure Shell</td>
<td>Secure Shell encrypts network traffic over an unsecured network. Secure Shell provides authentication by the use of passwords, public keys, or both.</td>
<td>“Secure Shell (Overview)” on page 281</td>
</tr>
</tbody>
</table>

A possible substitute for Secure RPC is the Oracle Solaris privileged port mechanism. A privileged port is assigned a port number less than 1024. After a client system has authenticated the client’s credential, the client builds a connection to the server by using the privileged port. The server then verifies the client credential by examining the connection’s port number.

Clients that are not running Oracle Solaris software might be unable to communicate by using the privileged port. If the clients cannot communicate over the port, you see an error message that appears similar to the following:

"Weak Authentication
NFS request from unprivileged port"
Firewall Systems

You can set up a firewall system to protect the resources in your network from outside access. A firewall system is a secure host that acts as a barrier between your internal network and outside networks. The internal network treats every other network as untrusted. You should consider this setup as mandatory between your internal network and any external networks, such as the Internet, with which you communicate.

A firewall acts as a gateway and as a barrier. As a gateway, it passes data between the networks. As a barrier, it blocks the free passage of data to and from the network. A user on the internal network must log in to the firewall system to access hosts on remote networks. Similarly, a user on an outside network must first log in to the firewall system before being granted access to a host on the internal network.

A firewall can also be useful between some internal networks. For example, you can set up a firewall or a secure gateway computer to restrict the transfer of packets by address or by protocol. For example, you can allow packets for transferring mail, but not allow packets for the `ftp` command.

In addition, all electronic mail that is sent from the internal network is first sent to the firewall system. The firewall then transfers the mail to a host on an external network. The firewall system also receives all incoming electronic mail, and distributes the mail to the hosts on the internal network.

**Caution** – A firewall prevents unauthorized users from accessing the hosts on your network. You should maintain strict and rigidly enforced security on the firewall, but security on other hosts on the network can be more relaxed. However, an intruder who can break into your firewall system can then gain access to all the other hosts on the internal network.

A firewall system should not have any trusted hosts. A trusted host is a host from which a user can log in without being required to supply a password. A firewall system should not share any of its file systems, or mount any file systems from other servers.

IPsec and the IP Filter feature of Oracle Solaris can provide firewall protection. For more information about protecting network traffic, see *Securing the Network in Oracle Solaris 11.1*.

Encryption and Firewall Systems

Most local area networks transmit data between computers in blocks that are called packets. Through a procedure that is called packet smashing, unauthorized users from outside the network can corrupt or destroy data.
Packetsmashinginvolvescapturingthepacketsbeforethepacketsreachtheirdestination. The intruder then injects arbitrary data into the contents, and sends the packets back on their original course. On a local area network, packet smashing is impossible because packets reach all systems, including the server, at the same time. Packet smashing is possible on a gateway, however, so make sure that all gateways on the network are protected.

The most dangerous attacks affect the integrity of the data. Such attacks involve changing the contents of the packets or impersonating a user. Attacks that involve eavesdropping do not compromise data integrity. An eavesdropper records conversations for later replay. An eavesdropper does not impersonate a user. Although eavesdropping attacks do not attack data integrity, the attacks do affect privacy. You can protect the privacy of sensitive information by encrypting data that goes over the network.

- To encrypt remote operations over an insecure network, see Chapter 15, “Using Secure Shell.”
- To encrypt and authenticate data across a network, see Chapter 19, “Introduction to the Kerberos Service.”
- To encrypt IP datagrams, see Chapter 6, “IP Security Architecture (Overview),” in Securing the Network in Oracle Solaris 11.1.

**Reporting Security Problems**

If you experience a suspected security breach, you can contact the Computer Emergency Response Team/Coordination Center (CERT/CC). CERT/CC is a Defense Advanced Research Projects Agency (DARPA) funded project that is located at the Software Engineering Institute at Carnegie Mellon University. This agency can assist you with any security problems you are having. This agency can also direct you to other Computer Emergency Response Teams that might be more appropriate for your particular needs. For current contact information, consult the CERT/CC [http://www.cert.org/contact_cert/](http://www.cert.org/contact_cert/) web site.
This chapter describes the procedures for controlling who can access Oracle Solaris systems.

The chapter covers the following topics:

- "Securing Logins and Passwords (Tasks)" on page 53
- "Changing the Default Algorithm for Password Encryption (Tasks)" on page 57
- "Monitoring and Restricting root Access (Tasks)" on page 60
- "Controlling Access to System Hardware (Tasks)" on page 62

For overview information about system security, see Chapter 2, "Managing Machine Security (Overview)."

### Securing Logins and Passwords (Tasks)

You can limit remote logins, require users to have passwords, and require the root account to have a complex password. You can also display a security message to users, monitor failed access attempts, and disable logins temporarily.

### Securing Logins and Passwords (Task Map)

The following task map points to procedures that monitor user logins and that disable user logins.
### Securing Logins and Passwords (Tasks)

<table>
<thead>
<tr>
<th>Task</th>
<th>Description</th>
<th>For Instructions</th>
</tr>
</thead>
</table>
| Inform users of site security at login. | Displays a text message on the login screen with site security information. | “How to Place a Security Message in Banner Files” in Oracle Solaris 11 Security Guidelines  
|                                   |                                                                             | “How to Place a Security Message on the Desktop Login Screen” in Oracle Solaris 11 Security Guidelines |
| Change the root password.         | Ensures that the root account complies with password requirements.           | “How to Change the root Password” on page 54                                     |
| Display a user’s login status.    | Lists extensive information about a user’s login account, such as full name and password aging information. | “How to Display a User’s Login Status” on page 55                                  |
| Find users who do not have passwords. | Finds only those users whose accounts do not require a password.             | “How to Display Users Without Passwords” on page 56                                |
| Disable logins temporarily.       | Denies user logins to a machine as part of system shutdown or routine maintenance. | “How to Temporarily Disable User Logins” on page 56                                |

#### ▼ How to Change the root Password

When you change the root password, you must comply with the password requirements that apply to all users of the system.

**Before You Begin**

You must assume the root role. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

- **Change your password.**

  ```
  # passwd root
  New Password:
  Re-enter new Password:
  passwd: password successfully changed for root
  ```

  A message prints to the screen if your password does not conform to requirements. The messages are informative. After three attempts, you must run the command again to change the password.

  ```
  passwd: Password too short - must be at least 6 characters.
  passwd: The password must contain at least 2 alphabetic character(s).
  passwd: The password must contain at least 1 numeric or special character(s).
  ```
How to Display a User's Login Status

Before You Begin

You must assume the root role. For more information, see "How to Use Your Assigned Administrative Rights" on page 157.

Display a user's login status by using the \texttt{logins} command.

\begin{verbatim}
# logins -x -l username

-x Displays an extended set of login status information.
-\texttt{l} \textit{username} Displays the login status for the specified user. The variable \textit{username} is a user's login name. Multiple login names are separated by commas.
\end{verbatim}

The \texttt{logins} command uses the appropriate password database to obtain a user's login status. The database can be the local \texttt{/etc/passwd} file, or a password database for the naming service. For more information, see the \texttt{logins(1M)} man page.

Example 3–1

Displaying a User's Login Status

In the following example, the login status for the user \texttt{jdoe} is displayed.

\begin{verbatim}
# logins -x -l jdoe
jdoe 500 staff 10 Jaylee Jaye Doe /home/jdoe /bin/bash PS 010103 10 7 -1

jdoe Identifies the user's login name.
500 Identifies the user ID (UID).
staff Identifies the user's primary group.
10 Identifies the group ID (GID).
Jaylee Jaye Doe Identifies the comment.
/home/jdoe Identifies the user's home directory.
/bin/bash Identifies the login shell.
PS 010170 10 7 -1

Specifies the password aging information:
\begin{itemize}
\item Last date that the password was changed
\item Number of days that are required between changes
\item Number of days before a change is required
\item Warning period
\end{itemize}
\end{verbatim}
How to Display Users Without Passwords

You must assume the root role. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

Display all users who have no passwords by using the `logins` command.

```
# logins -p
```

The `-p` option displays a list of users with no passwords. The `logins` command uses the `passwd` database from the local system unless a distributed naming service is specified in the `password` property of the `system/name-service/switch` service.

Example 3–2 Displaying Accounts Without Passwords

In the following example, the user `pmorph` and the role `roletop` do not have passwords.

```
# logins -p
pmorph 501 other 1 Polly Morph
roletop 211 admin 1 Role Top
```

How to Temporarily Disable User Logins

Temporarily disable user logins during system shutdown or routine maintenance. root logins are not affected. For more information, see the `nologin(4)` man page.

You must become an administrator who is assigned the `solaris.admin.edit/etc/nologin` authorization. By default, the root role has this authorization. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1 Create the `/etc/nologin` file in a text editor.

```
# pfedit /etc/nologin
```

For an example of using the `solaris.admin.edit/etc/nologin` authorization, see Example 3–3.

2 Include a message about system availability.

3 Close and save the file.

Example 3–3 Disabling User Logins

In this example, a user is authorized to write the notification of system unavailability.
% pfedit /etc/nologin
***No logins permitted.***
***The system will be unavailable until 12 noon.***

About Failed Logins

To monitor all failed login attempts to the system, refer to the audit trail that is generated by the auditing service. For more information, see Part VII, “Auditing in Oracle Solaris.”

Changing the Default Algorithm for Password Encryption (Tasks)

By default, user passwords are encrypted with the crypt_sha256 algorithm. You might want to change the default algorithm to interoperate in a heterogeneous networked environment, such as to log in to frequently used older systems on the network.

▼ How to Specify an Algorithm for Password Encryption

In this procedure, the BSD-Linux version of the MD5 algorithm is the default encryption algorithm that is used when users change their passwords. This algorithm is suitable for a mixed network of systems that run the Oracle Solaris, BSD, and Linux versions of UNIX. For a list of password encryption algorithms and algorithm identifiers, see Table 2–1.

Before You Begin

You must assume the root role. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

- Specify the identifier for your chosen encryption algorithm.
  Type the identifier as the value for the CRYPT_DEFAULT variable in the /etc/security/policy.conf file.

    You might want to comment the file to explain your choice.

    # cat /etc/security/policy.conf
    ...
    CRYPT_ALGORITHMS_ALLOW=1,2a,md5,5,6
    #
    # Use the version of MD5 (5) that works with Linux and BSD systems.
    # Passwords previously encrypted with SHA256 (1) will be encrypted
    # with MD5 when users change their passwords.
    #
#CRYPT_DEFAULT=5
CRYPT_DEFAULT=1

In this example, the algorithms configuration ensures that the sha256 algorithm is not used to encrypt a password. Users whose passwords were encrypted with the sha256 module get a crypt_bsdmd5-encrypted password when they change their passwords.

For more information about configuring the algorithm choices, see the policy.conf(4) man page.

Example 3–4  Constraining Password Encryption Algorithms in a Heterogeneous Environment

In this example, the administrator on a network that includes BSD and Linux systems configures passwords to be usable on all systems. Because some network applications cannot handle SHA512 encryption, the administrator does not include its identifier in the list of allowed algorithms. The administrator retains the SHA256 algorithm, 5, as the value for the CRYPT_DEFAULT variable. The CRYPT_ALGORITHMS_ALLOW variable contains the MD5 identifier, which is compatible with BSD and Linux systems, and the Blowfish identifier, which is compatible with BSD systems. Because 5 is the CRYPT_DEFAULT algorithm, it does not need to be listed in the CRYPT_ALGORITHMS_ALLOW list. However, for maintenance purposes, the administrator places 5 in the CRYPT_ALGORITHMS_ALLOW list and the unused identifiers in the CRYPT_ALGORITHMS_DEPRECATE list.

CRYPT_ALGORITHMS_ALLOW=1,2a,5
CRYPT_DEFAULT=5

How to Specify a New Password Algorithm for an NIS Domain

When users in an NIS domain change their passwords, the NIS client consults its local algorithms configuration in the /etc/security/policy.conf file. The NIS client system encrypts the password.

Before You Begin  You must assume the root role. For more information, see "How to Use Your Assigned Administrative Rights" on page 157.

1  Specify the password encryption algorithm in the /etc/security/policy.conf file on the NIS client.

2  Copy the modified /etc/security/policy.conf file to every client system in the NIS domain.
To minimize confusion, copy the modified /etc/security/policy.conf file to the NIS root server and to the slave servers.

▼ How to Specify a New Password Algorithm for an LDAP Domain

When the LDAP client is properly configured, the LDAP client can use the new password algorithms. The LDAP client behaves just as an NIS client behaves.

Before You Begin

You must assume the root role. For more information, see "How to Use Your Assigned Administrative Rights" on page 157.

1 Specify a password encryption algorithm in the /etc/security/policy.conf file on the LDAP client.

2 Copy the modified policy.conf file to every client system in the LDAP domain.

3 Ensure that the client's /etc/pam.conf file does not use a pam_ldap module.

Ensure that a comment sign (#) precedes entries that include pam_ldap.so.1. Also, do not use the server_policy option with the pam_authtok_store.so.1 module.

The PAM entries in the client's pam.conf file enable the password to be encrypted according to the local algorithms configuration. The PAM entries also enable the password to be authenticated.

When users in the LDAP domain change their passwords, the LDAP client consults its local algorithms configuration in the /etc/security/policy.conf file. The LDAP client system encrypts the password. Then, the client sends the encrypted password, with a {crypt} tag, to the server. The tag tells the server that the password is already encrypted. The password is then stored, as is, on the server. For authentication, the client retrieves the stored password from the server. The client then compares the stored password with the encrypted version that the client has just generated from the user's typed password.

Note – To take advantage of password policy controls on the LDAP server, use the server_policy option with the pam_authtok_store entries in the pam.conf file. Passwords are then encrypted on the LDAP server. For the procedure, see Chapter 11, "Setting Up Oracle Directory Server Enterprise Edition With LDAP Clients (Tasks)," in Working With Naming and Directory Services in Oracle Solaris 11.1.
Monitoring and Restricting root Access (Tasks)

By default, the root role is assigned to the initial user, and cannot directly log in to the local system or remotely log in to any Oracle Solaris system.

▼ How to Monitor Who Is Using the su Command

The sulog file lists every use of the switch user (su) command, not only the su attempts that are used to switch from user to root.

Before You Begin

You must assume the root role. For more information, see "How to Use Your Assigned Administrative Rights" on page 157.

- Monitor the contents of the /var/adm/sulog file on a regular basis.

```
# more /var/adm/sulog
SU 12/20 16:26 + pts/0 stacey-root
SU 12/21 10:59 + pts/0 stacey-root
SU 01/12 11:11 + pts/0 root-rimmer
SU 01/12 14:56 + pts/0 jdoe-root
SU 01/12 14:57 + pts/0 jdoe-root
```

The entries display the following information:

- The date and time that the command was entered.
- If the attempt was successful. A plus sign (+) indicates a successful attempt. A minus sign (-) indicates an unsuccessful attempt.
- The port from which the command was issued.
- The name of the user and the name of the switched identity.

The su logging in this file is enabled by default through the following entry in the /etc/default/su file:

```
SULOG=/var/adm/sulog
```

Troubleshooting

Entries that include ??? indicate that the controlling terminal for the su command cannot be identified. Typically, system invocations of the su command before the desktop appears include ???, as in SU 10/10 08:08 + ?? root-root. After the user starts a desktop session, the ttynam command returns the value of the controlling terminal to the sulog: SU 10/10 10:10 + pts/3 jdoe-root.

Entries similar to the following can indicate that the su command was not invoked on the command line: SU 10/10 10:20 + ?? root-oracle. A Trusted Extensions user might have switched to the oracle role by using a GUI.
\textbf{How to Restrict and Monitor root Logins}

This method immediately detects root attempts to access the local system.

**Before You Begin**
You must assume the root role. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1. **View the CONSOLE entry in the /etc/default/login file.**
   
   CONSOLE=/dev/console
   
   By default, the console device is set to /dev/console. With this setting, root can log in to the console. root cannot log in remotely.

2. **Verify that root cannot log in remotely.**
   
   From a remote system, try to log in as root.
   
   ```
   mach2 % ssh -l root mach1
   Password: <Type root password of mach1>
   Password: 
   Password: 
   Permission denied (gssapi-keyex,gssapi-with-mic,publickey,keyboard-interactive).
   ```
   In the default configuration, root is a role, and roles cannot log in. Also, in the default configuration the ssh protocol prevents root user login.

3. **Monitor attempts to become root.**
   
   By default, attempts to become root are printed to the console by the SYSLOG utility.

   a. **Open a terminal console on your desktop.**

   b. **In another window, use the su command to become root.**
      
      ```
      % su -
      Password: <Type root password>
      #
      ```
      A message is printed on the terminal console.
      
      ```
      Sep 7 13:22:57 mach1 su: 'su root' succeeded for jdoe on /dev/pts/6
      ```

**Example 3–5** Logging root Access Attempts

In this example, root attempts are not being logged by SYSLOG. Therefore, the administrator is logging those attempts by removing the comment from the #CONSOLE=/dev/console entry in the /etc/default/su file.
# CONSOLE determines whether attempts to su to root should be logged
# to the named device
CONSOLE=/dev/console

When a user attempts to become root, the attempt is printed on the terminal console.

SU 09/07 16:38 + pts/8 jdoe-root

Troubleshooting

To become root from a remote system when the /etc/default/login file contains the default
CONSOLE entry, users must first log in with their user name. After logging in with their user name,
users then can use the su command to become root.

If the console displays an entry similar to Last login: Wed Sep 7 15:13:11 2011 from
mach2, then the system is configured to permit remote root logins. To prevent remote root access,
change the #CONSOLE=/dev/console entry to CONSOLE=/dev/console in the
/etc/default/login file. To return the ssh protocol to the default, see the sshd_config(4)
man page.

Controlling Access to System Hardware (Tasks)

You can protect the physical system by requiring a password to gain access to the hardware
settings. You can also protect the system by preventing a user from using the abort sequence to
leave the windowing system.

To protect the BIOS, consult the vendor documentation.

How to Require a Password for SPARC Hardware Access

Before You Begin

You must become an administrator who is assigned the Device Security, Maintenance and
Repair, or System Administrator rights profile. For more information, see "How to Use Your
Assigned Administrative Rights" on page 157.

1 In a terminal window, type the PROM security mode.

    # eeprom security-mode=command

    Changing PROM password:
    New password: <Type password>
    Retype new password: <Retype password>

    Choose the value command or full. For more details, see the eeprom(1M) man page.
If, when you type the preceding command, you are not prompted for a PROM password, the system already has a PROM password.

2 **(Optional) To change the PROM password, type the following command:**

```
# eeprom security-password= Press Return
```

Changing PROM password:
New password:  <Type password>
Retype new password:  <Retype password>

The new PROM security mode and password are in effect immediately. However, they are most likely to be noticed at the next boot.

⚠️ **Caution** – Do not forget the PROM password. The hardware is unusable without this password.

▼ **How to Disable a System's Abort Sequence**

**Note** – Some server systems have a key switch. When the key switch is set in the secure position, the switch overrides the software keyboard abort settings. So, any changes that you make with the following procedure might not be implemented.

**Before You Begin**
You must become an administrator who is assigned the `solaris.admin.edit/etc/default/kbd` authorization. By default, the root role has this authorization. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1 **Change the value of KEYBOARD_ABORT to disable.**

Comment out the enable line in the `/etc/default/kbd` file. Then, add a disable line:

```
# cat /etc/default/kbd
...
# KEYBOARD_ABORT affects the default behavior of the keyboard abort
# sequence, see kbd(1) for details. The default value is "enable".
# The optional value is "disable". Any other value is ignored.
...
#KEYBOARD_ABORT=enable
KEYBOARD_ABORT=disable
```

2 **Update the keyboard defaults.**

```
# kbd -i
```
This chapter provides information about using antivirus software, and covers the following topics:

- “About Virus Scanning” on page 65
- “About the Vscan Service” on page 66
- “Using the Vscan Service (Tasks)” on page 66

### About Virus Scanning

Data is protected from viruses by a scanning service, vscan, that uses various scan engines. A scan engine is a third-party application, residing on an external host, that examines a file for known viruses. A file is a candidate for virus scanning if the file system supports the vscan service, the service has been enabled, and the type of file has not been exempted. The virus scan is then performed on a file during open and close operations if the file has not been scanned with the current virus definitions previously or if the file has been modified since it was last scanned.

The vscan service can be configured to use multiple scan engines. It is recommended that the vscan service use a minimum of two scan engines. The requests for virus scans are distributed among all available scan engines. Table 4–1 shows the scan engines that are supported when configured with their most recent patch.

<table>
<thead>
<tr>
<th>Antivirus Software</th>
<th>ICAP Support</th>
</tr>
</thead>
<tbody>
<tr>
<td>Symantec Antivirus Scan Engine 4.3</td>
<td>Is supported</td>
</tr>
<tr>
<td>Symantec Antivirus Scan Engine 5.1</td>
<td>Is supported</td>
</tr>
</tbody>
</table>
### About the Vscan Service

The benefit of the real-time scan method is that a file is scanned with the latest virus definitions before it is used. By using this approach, viruses can be detected before they compromise data.

The following describes the virus scanning process:

1. When a user opens a file from the client, the vscan service determines whether the file needs to be scanned, based on whether the file has been scanned with the current virus definitions previously and if the file has been modified since it was last scanned.
   - If the file needs to be scanned, the file is transferred to the scan engine. If a connection to a scan engine fails, the file is sent to another scan engine. If no scan engine is available, the virus scan fails and access to the file might be denied.
   - If the file does not need to be scanned, the client is permitted to access the file.

2. The scan engine scans the file using the current virus definitions.
   - If a virus is detected, the file is marked as quarantined. A quarantined file cannot be read, executed, or renamed but it can be deleted. The system log records the name of the quarantined file and the name of the virus and, if auditing has been enabled, an audit record with the same information is created.
   - If the file is not infected, the file is tagged with a scan stamp and the client is permitted to access the file.

### Using the Vscan Service (Tasks)

Scanning files for viruses is available when the following requirements are met:

- At least one scan engine is installed and configured.
- The files reside on a file system that supports virus scanning.
- Virus scanning is enabled on the file system.
- The vscan service is enabled.
- The vscan service is configured to scan files of the specified file type.
The following table points to the tasks you perform to set up the vscan service.

<table>
<thead>
<tr>
<th>Task</th>
<th>Description</th>
<th>For Instructions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Install a scan engine.</td>
<td>Installs and configures one or more of the supported third-party products listed in Table 4–1.</td>
<td>See the product documentation.</td>
</tr>
<tr>
<td>Enable the file system to allow virus scans.</td>
<td>Enables virus scans on a ZFS file system. By default, scans are disabled.</td>
<td>&quot;How to Enable Virus Scanning on a File System&quot; on page 67</td>
</tr>
<tr>
<td>Enable the vscan service.</td>
<td>Starts the scan service.</td>
<td>&quot;How to Enable the Vscan Service&quot; on page 68</td>
</tr>
<tr>
<td>Add a scan engine to the vscan service.</td>
<td>Includes specific scan engines in the vscan service.</td>
<td>&quot;How to Add a Scan Engine&quot; on page 68</td>
</tr>
<tr>
<td>Configure the vscan service.</td>
<td>Views and changes vscan properties.</td>
<td>&quot;How to View Vscan Properties&quot; on page 68</td>
</tr>
<tr>
<td>Configure the vscan service for specific file types.</td>
<td>Specifies the file types to include and exclude in a scan.</td>
<td>&quot;How to Change Vscan Properties&quot; on page 69</td>
</tr>
</tbody>
</table>

▼ How to Enable Virus Scanning on a File System

Use the file system command to allow virus scans of files. For example, to include a ZFS file system in a virus scan, use the zfs(1M) command.

Before You Begin

You must become an administrator who is assigned the ZFS File System Management or the ZFS Storage Management rights profile. For more information, see "How to Use Your Assigned Administrative Rights" on page 157.

Enable virus scanning on a ZFS file system, for example, pool/volumes/voll.

```
# zfs set vscan=on path/pool/volumes/voll
```
How to Enable the Vscan Service

**Before You Begin**
You must become an administrator who is assigned the VSCAN Management rights profile. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

- Enable the virus scanning service.
  ```
  # svcadm enable vscan
  ```

How to Add a Scan Engine

**Before You Begin**
You must become an administrator who is assigned the VSCAN Management rights profile. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

- To add a scan engine to the vscan service with default properties, type:
  ```
  # vscanadm add-engine engine_ID
  ```
  For more information, see the `vscanadm(1M)` man page.

How to View Vscan Properties

**Before You Begin**
You must become an administrator who is assigned the VSCAN Management rights profile. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

- View the properties of the vscan service, of all scan engines, or of a specific scan engine.
  - To view the properties of a particular scan engine, type:
    ```
    # vscanadm get-engine engineID
    ```
  - To view the properties of all scan engines, type:
    ```
    # vscanadm get-engine
    ```
  - To view one of the properties of the vscan service, type:
    ```
    # vscanadm get -p property
    ```
    where `property` is one of the parameters described in the man page for the `vscanadm(1M)` command.
    For example, if you want to see the maximum size of a file that can be scanned, type:
    ```
    # vscanadm get max-size
    ```
How to Change Vscan Properties

You can change the properties of a particular scan engine and the general properties of the vscan service. Many scan engines limit the size of the files they scan, so the vscan service’s max-size property must be set to a value less than or equal to the scan engine’s maximum allowed size. You then define whether files that are larger than the maximum size, and therefore not scanned, are accessible.

Before You Begin

You must become an administrator who is assigned the VSCAN Management rights profile. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1 View the current properties by using the vscanadm show command.

2 Set the maximum size for virus scans to, for example, 128 megabytes.
   # vscanadm set -p max-size=128M

3 Specify that access is denied to any file that is not scanned due to its size.
   # vscanadm set -p max-size-action=deny
   For more information, see the vscanadm(1M) man page.

How to Exclude Files From Virus Scans

When you enable antivirus protection, you can specify that all files of specific types are excluded from the virus scan. Because the vscan service affects the performance of the system, you can conserve system resources by targeting specific file types for virus scans.

Before You Begin

You must become an administrator who is assigned the VSCAN Management rights profile. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1 View the list of all file types that are included in the virus scan.
   # vscanadm get -p types

2 Specify the types of files to be scanned for virus:

   ▪ Exclude a specific file type, for example the JPEG type, from the virus scan.
     # vscanadm set -p types=-jpg,+

   ▪ Include a specific file type, for example executable files, in the virus scan.
     # vscanadm set -p types=+exe,-*
   For more information, see the vscanadm(1M) man page.
This chapter provides step-by-step instructions for protecting devices, in addition to a reference section. The chapter covers the following topics:

- “Configuring Device Policy (Tasks)” on page 71
- “Managing Device Allocation (Tasks)” on page 73
- “Allocating Devices (Tasks)” on page 78
- “Device Protection (Reference)” on page 82

For overview information about device protection, see “Controlling Access to Devices” on page 40.

Configuring Device Policy (Tasks)

Device policy restricts or prevents access to devices that are integral to the system. The policy is enforced in the kernel.

Configuring Device Policy (Task Map)

The following task map points to device configuration procedures that are related to device policy.

<table>
<thead>
<tr>
<th>Task</th>
<th>Description</th>
<th>For Instructions</th>
</tr>
</thead>
<tbody>
<tr>
<td>View the device policy for the devices on your system.</td>
<td>Lists the devices and their device policy.</td>
<td>“How to View Device Policy” on page 72</td>
</tr>
<tr>
<td>Audit changes in device policy.</td>
<td>Records changes in device policy in the audit trail.</td>
<td>“How to Audit Changes in Device Policy” on page 72</td>
</tr>
</tbody>
</table>
### How to View Device Policy

- Display the device policy for all devices on your system.

  ```
  % getdevpolicy | more
  DEFAULT
  read_priv_set=none
  write_priv_set=none
  ip:*  
  read_priv_set=net_rawaccess
  write_priv_set=net_rawaccess
  ...
  ```

**Example 5–1** Viewing the Device Policy for a Specific Device

In this example, the device policy for three devices is displayed.

```
% getdevpolicy /dev/allkmem /dev/ipsecesp /dev/bge
/dev/allkmem
read_priv_set=all
write_priv_set=all
/dev/ipsecesp
read_priv_set=sys_net_config
write_priv_set=sys_net_config
/dev/bge
read_priv_set=net_rawaccess
write_priv_set=net_rawaccess
```

### How to Audit Changes in Device Policy

By default, the `as` audit class includes the `AUE_MODDEVPLCY` audit event.

**Before You Begin**

You must become an administrator who is assigned the Audit Configuration rights profile. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

- Preselect the audit class that includes `AUE_MODDEVPLCY` audit event.

  ```
  # auditconfig -getflags
  current-flags
  # auditconfig -setflags current-flags,as
  ```

  For detailed instructions, see “How to Preselect Audit Classes” on page 554.
How to Retrieve IP MIB-II Information From a /dev/* Device

Applications that retrieve Oracle Solaris IP MIB-II information should open /dev/arp, not /dev/ip.

1 Determine the device policy on /dev/ip and /dev/arp.

   % getdevpolicy /dev/ip /dev/arp
   /dev/ip
   read_priv_set=net_rawaccess
   write_priv_set=net_rawaccess
   /dev/arp
   read_priv_set=none
   write_priv_set=none

   Note that the net_rawaccess privilege is required for reading and writing to /dev/ip. No privileges are required for /dev/arp.

2 Open /dev/arp and push the tcp and udp modules.

   No privileges are required. This method is equivalent to opening /dev/ip and pushing the arp, tcp and udp modules. Because opening /dev/ip now requires a privilege, the /dev/arp method is preferred.

Managing Device Allocation (Tasks)

Device allocation is commonly implemented at sites that require an additional layer of device security. Typically, users must have authorization to access allocatable devices.

Managing Device Allocation (Task Map)

The following task map points to procedures that enable, configure, and troubleshoot device allocation. Device allocation is not enabled by default. After device allocation is enabled, see “Allocating Devices (Tasks)” on page 78 for instructions on allocating devices.

<table>
<thead>
<tr>
<th>Task</th>
<th>Description</th>
<th>For Instructions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Make a device allocatable.</td>
<td>Enables a device to be allocated to one user at a time. Removes allocation restrictions from all devices.</td>
<td>“How to Enable Device Allocation” on page 74</td>
</tr>
<tr>
<td>Disable device allocation.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Authorize users to allocate a device.</td>
<td>Assigns device allocation authorizations to users.</td>
<td>“How to Authorize Users to Allocate a Device” on page 74</td>
</tr>
<tr>
<td>View the allocatable devices on your system.</td>
<td>Lists the devices that are allocatable, and the state of the device.</td>
<td>“How to View Allocation Information About a Device” on page 75</td>
</tr>
</tbody>
</table>
Managing Device Allocation (Tasks)

<table>
<thead>
<tr>
<th>Task</th>
<th>Description</th>
<th>For Instructions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Forcibly allocate a device.</td>
<td>Allocates a device to a user who has an immediate need.</td>
<td>&quot;How to Forcibly Allocate a Device&quot; on page 76</td>
</tr>
<tr>
<td>Forcibly deallocate a device.</td>
<td>Deallocates a device that is currently allocated to a user.</td>
<td>&quot;How to Forcibly Deallocate a Device&quot; on page 76</td>
</tr>
<tr>
<td>Change the allocation properties of a device.</td>
<td>Changes the requirements for allocating a device.</td>
<td>&quot;How to Change Which Devices Can Be Allocated&quot; on page 77</td>
</tr>
<tr>
<td>Audit device allocation.</td>
<td>Records device allocation in the audit trail</td>
<td>&quot;How to Audit Device Allocation&quot; on page 78</td>
</tr>
<tr>
<td>Create a device-cleanscript.</td>
<td>Purges data from a physical device.</td>
<td>&quot;Writing New Device-Clean Scripts&quot; on page 88</td>
</tr>
</tbody>
</table>

▼ How to Enable Device Allocation

Before You Begin
You must become an administrator who is assigned the Device Security rights profile. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1 Enable the device allocation service and verify that the service is enabled.

   # svcadm enable svc:/system/device/allocate
   # svc -x allocate
   svc:/system/device/allocate:default (device allocation)
     State: online since September 10, 2011 01:10:11 PM PDT
     See: allocate(1)
     See: deallocate(1)
     See: list_devices(1)
     See: device_allocate(1M)
     See: mkdevalloc(1M)
     See: mkdevmaps(1M)
     See: dminfo(1M)
     See: device_maps(4)
     See: /var/svc/log/system-device-allocate:default.log
     Impact: None.

2 To disable the device allocation service, use the disable subcommand.

   # svcadm disable device/allocate

▼ How to Authorize Users to Allocate a Device

Before You Begin
You must become an administrator who is assigned the User Security rights profile. Your rights profiles must include the solaris.auth.delegate authorization. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1 Create a rights profile that contains the appropriate authorization and commands.
Typically, you would create a rights profile that includes the `solaris.device.allocate` authorization. Follow the instructions in “How to Create a Rights Profile” on page 167. Give the rights profile appropriate properties, such as the following:

- Rights profile name: Device Allocation
- Granted authorizations: `solaris.device.allocate`
- Commands with privileges: `mount` with the `sys_mount` privilege, and `umount` with the `sys_mount` privilege

2 (Optional) Create a role for the rights profile.

Follow the instructions in “How to Create a Role” on page 163. Use the following role properties as a guide:

- Role name: devicealloc
- Role full name: Device Allocator
- Role description: Allocates and mounts allocated devices
- Rights profile: Device Allocation

This rights profile must be the first in the list of profiles that are included in the role.

3 Assign the rights profile to authorized users or authorized roles.

4 Teach the users how to use device allocation.

For examples of allocating removable media, see “How to Allocate a Device” on page 78.

▼ How to View Allocation Information About a Device

Before You Begin
You have completed “How to Enable Device Allocation” on page 74.

You must become an administrator who is assigned the Device Security rights profile. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

Display information about allocatable devices on your system.

```
# list_devices device-name
```

where `device-name` is one of the following:

- `audio[n]` – Is a microphone and speaker.
- `fd[n]` – Is a diskette drive.
- `rmdisk[n]` – Is a removable media device, such as a USB.
- `st[n]` – Is a tape drive.
Troubleshooting

If the list_devices command returns an error message similar to the following, then either device allocation is not enabled, or you do not have sufficient permissions to retrieve the information.

```
list_devices: No device maps file entry for specified device.
```

For the command to succeed, enable device allocation and assume a role with the solaris.device.revoke authorization.

▼ How to Forcibly Allocate a Device

Forcible allocation is used when someone has forgotten to deallocate a device. Forcible allocation can also be used when a user has an immediate need for a device.

Before You Begin

You must become an administrator who is assigned the solaris.device.revoke authorization. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1. Determine if you have the appropriate authorizations in your role.
   ```sh
   $ auths
   solaris.device.allocate solaris.device.revoke
   ```

2. Forcibly allocate the device to the user who needs the device.
   In this example, a USB drive is forcibly allocated to the user jdoe.
   ```sh
   $ allocate -U jdoe
   ```

▼ How to Forcibly Deallocate a Device

Devices that a user has allocated are not automatically deallocated when the process terminates or when the user logs out. Forcible deallocation is used when a user has forgotten to deallocate a device.

Before You Begin

You must become an administrator who is assigned the solaris.device.revoke authorization. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1. Determine if you have the appropriate authorizations in your role.
   ```sh
   $ auths
   solaris.device.allocate solaris.device.revoke
   ```
2 Forcibly deallocate the device.
   In this example, the printer is forcibly deallocated. The printer is now available for allocation by another user.
   $ deallocate -f /dev/lp/printer-1

▼ How to Change Which Devices Can Be Allocated

Before You Begin
   Device allocation must be enabled for this procedure to succeed. To enable device allocation, see “How to Enable Device Allocation” on page 74. You must assume the root role.

● Specify if authorization is required, or specify the solaris.device.allocate authorization.
   Change the fifth field in the device entry in the device_allocate file.
   audio;audio;reserved;reserved;solaris.device.allocate;/etc/security/lib/audio_clean
   fd0;fd;reserved;reserved;solaris.device.allocate;/etc/security/lib/fd_clean
   sr0;sr;reserved;reserved;solaris.device.allocate;/etc/security/lib/sr_clean
   where solaris.device.allocate indicates that a user must have the solaris.device.allocate authorization to use the device.

Example 5–2 Permitting Any User to Allocate a Device
   In the following example, any user on the system can allocate any device. The fifth field in every device entry in the device_allocate file has been changed to an at sign (@).
   
   # pfedit /etc/security/device_allocate
   audio;audio;reserved;reserved;@
   fd0;fd;reserved;reserved;@
   sr0;sr;reserved;reserved;@
   ...

Example 5–3 Preventing Some Peripheral Devices From Being Used
   In the following example, the audio device cannot be used. The fifth field in the audio device entry in the device_allocate file has been changed to an asterisk (*).
   
   # pfedit /etc/security/device_allocate
   audio;audio;reserved;reserved;*
   fd0;fd;reserved;reserved;solaris.device.allocate;/etc/security/lib/fd_clean
   sr0;sr;reserved;reserved;solaris.device.allocate;/etc/security/lib/sr_clean
   ...

Example 5–4 Preventing All Peripheral Devices From Being Used
   In the following example, no peripheral device can be used. The fifth field in every device entry in the device_allocate file has been changed to an asterisk (*).
Allocating Devices (Tasks)

By default, the device allocation commands are in the other audit class.

**Before You Begin**
You must become an administrator who is assigned the Audit Configuration rights profile. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

- Preselect the other audit class.

  ```
  $ auditconfig -getflags current-flags
  $ auditconfig -setflags current-flags,ot
  ```

  For detailed instructions, see “How to Preselect Audit Classes” on page 554.

**Allocating Devices (Tasks)**
Device allocation reserves the use of a device to one user at a time. Devices that require a mount point must be mounted. The following procedures show users how to allocate devices.

**How to Allocate a Device**

**Before You Begin**
Device allocation must be enabled, as described in “How to Enable Device Allocation” on page 74. If authorization is required, the user must have the authorization.

1. Allocate the device.
   Specify the device by device name.
   ```
   % allocate device-name
   ```

2. Verify that the device is allocated.
   Run the identical command.
   ```
   % allocate device-name
   allocate. Device already allocated.
   ```

**Example 5–5** Allocating a Microphone
In this example, the user jdoe allocates a microphone, audio0.
% whoami
jdoe
% allocate audio0

**Example 5–6  Allocating a Printer**

In this example, a user allocates a printer. No one else can print to printer-1 until the user deallocates it, or until the printer is forcibly allocated to another user.

% allocate /dev/lp/printer-1

For an example of forcible deallocation, see “How to Forcibly Deallocate a Device” on page 76.

**Example 5–7  Allocating a USB Drive**

In this example, a user allocates a USB drive, rmdisk1.

% allocate rmdisk1

**Troubleshooting**

If the `allocate` command cannot allocate the device, an error message is displayed in the console window. For a list of allocation error messages, see the `allocate(1)` man page.

**How to Mount an Allocated Device**

Devices mount automatically if you are granted the appropriate privileges. Follow this procedure if the device fails to mount.

**Before You Begin**

You have allocated the device. You are assigned the privileges that are required for mounting the device, as described in “How to Authorize Users to Allocate a Device” on page 74.

1. **Assume a role that can allocate and mount a device.**

   % su - role-name
   Password: <Type role-name password>
   $

2. **Create and protect a mount point in the role’s home directory.**

   You only need to do this step the first time that you need a mount point.

   $ mkdir mount-point ; chmod 700 mount-point

3. **List the allocatable devices.**

   $ list_devices -l
   List of allocatable devices
4 Allocate the device.
   Specify the device by device name.
   $ allocate device-name

5 Mount the device.
   $ mount -o ro -F filesystem-type device-path mount-point
   where
   -o ro Indicates that the device is to be mounted read-only. Use -o rw to make
   the device writable.
   -F filesystem-type Indicates the file system format of the device. Typically, a CD-ROM is
   formatted with an HSFS file system. A diskette is typically formatted with
   a PCFS file system.
   device-path Indicates the path to the device. The output of the list_devices -l
   command includes the device-path.
   mount-point Indicates the mount point that you created in Step 2.

Example 5–8 Allocating a CD-ROM Drive

In this example, a user assumes a role that can allocate and mount a CD-ROM drive, sr0. The
drive is formatted as an HSFS file system.

% roles
devicealloc
% su - devicealloc
Password: <Type devicealloc password>
$ mkdir /home/devicealloc/mymnt
$ chmod 700 /home/devicealloc/mymnt
$ list_devices -l
...
device: sr0 type: sr files: /dev/sr0 /dev/rsr0 /dev/dsk/c0t2d0s0 ...
...
$ allocate sr0
$ mount -o ro -F hsfs /dev/sr0 /home/devicealloc/mymnt
$ cd /home/devicealloc/mymnt ; ls
List of the contents of CD-ROM
Troubleshooting

If the `mount` command cannot mount the device, an error message is displayed:

```
mount: insufficient privileges
```

Check the following:

- Verify that you are executing the `mount` command in a profile shell. If you have assumed a role, the role has a profile shell. If you are a user who has been assigned a profile with the `mount` command, you must create a profile shell. For the list of available profile shells, see the `pfexec(1)`.
- Verify that you own the specified mount point. You must have read, write, and execute access to the mount point.

Contact your administrator if you still cannot mount the allocated device. “How to Troubleshoot RBAC and Privilege Assignment” on page 174 is a starting point.

▼ How to Deallocate a Device

Deallocation enables other users to allocate and use the device when you are finished.

Before You Begin

You must have allocated the device.

1. If the device is mounted, unmount the device.
   
   ```
   $ cd $HOME
   $ umount mount-point
   ```

2. Deallocate the device.
   
   ```
   $ deallocate device-name
   ```

Example 5–9 Deallocating a Microphone

In this example, the user `jdoe` deallocates the microphone, `audio`.

```
% whoami
jdoe
% deallocate audio0
```

Example 5–10 Deallocating a CD-ROM Drive

In this example, the Device Allocator role deallocates a CD-ROM drive. After the message is printed, the CD-ROM is ejected.

```
$ whoami
devicealloc
$ cd /home/devicealloc
$ umount /home/devicealloc/mymnt
$ ls /home/devicealloc/mymnt
```
$ deallocate sr0
/dev/sr0: 3260
/dev/rsr0: 3260
...

sr_clean: Media in sr0 is ready. Please, label and store safely.

Device Protection (Reference)

Devices in Oracle Solaris are protected by kernel device policy. Peripheral devices can be protected by device allocation. Device allocation is optionally enabled, and is enforced at the user level.

Device Policy Commands

Device management commands administer the device policy on local files. Device policy can include privilege requirements. Users who are assigned the Device Management and Device Security rights profiles can manage devices.

The following table lists the device management commands.

<table>
<thead>
<tr>
<th>TABLE 5-1 Device Management Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td>Man Page for Command</td>
</tr>
<tr>
<td><strong>devfsadm(1M)</strong></td>
</tr>
<tr>
<td><strong>getdevpolicy(1M)</strong></td>
</tr>
<tr>
<td><strong>add_drv(1M)</strong></td>
</tr>
<tr>
<td><strong>update_drv(1M)</strong></td>
</tr>
<tr>
<td><strong>rem_drv(1M)</strong></td>
</tr>
</tbody>
</table>
Device Allocation

Device allocation can protect your site from loss of data, computer viruses, and other security breaches. Unlike device policy, device allocation is optional. Device allocation uses authorizations to limit access to allocatable devices.

Components of Device Allocation

The components of the device allocation mechanism are as follows:

- The `svc:/system/device/allocate` service. For more information, see the `smf(5)` man page and the man pages for the device allocation commands.
- The `allocate`, `dallocate`, `dminfo`, and `list_devices` commands. For more information, see “Device Allocation Commands” on page 84.
- The Device Management and Device Security rights profiles. For more information, see “Device Allocation Rights Profiles” on page 83.
- Device-cleans scripts for each allocatable device.

These commands and scripts use the following local files to implement device allocation:

- The `/etc/security/device_allocate` file. For more information, see the `device_allocate(4)` man page.
- The `/etc/security/device_maps` file. For more information, see the `device_maps(4)` man page.
- A lock file, in the `/etc/security/dev` directory, for each allocatable device.
- The changed attributes of the lock files that are associated with each allocatable device.

*Note* – The `/etc/security/dev` directory might not be supported in future releases of Oracle Solaris.

Device Allocation Service

The `svc:/system/device/allocate` service controls device allocation. This service is off by default.

Device Allocation Rights Profiles

The Device Management and Device Security rights profiles are required to manage devices and device allocation.

These rights profiles include the following authorizations:

- `solaris.device.allocate` – Required to allocate a device
- `solaris.device.cdrw` – Required to read and write a CD-ROM
Device Protection (Reference)

- `solaris.device.config` – Required to configure the attributes of a device
- `solaris.device.mount.alloptions.fixed` – Required to specify mount options when mounting a fixed device
- `solaris.device.mount.alloptions.removable` – Required to specify mount options when mounting a removable device
- `solaris.device.mount.fixed` – Required to mount a fixed device
- `solaris.device.mount.removable` – Required to mount a removable device
- `solaris.device.revoke` – Required to revoke or reclaim a device

**Device Allocation Commands**

With uppercase options, the `allocate`, `deallocate`, and `list_devices` commands are administrative commands. Otherwise, these commands are user commands. The following table lists the device allocation commands.

**TABLE 5-2 Device Allocation Commands**

<table>
<thead>
<tr>
<th>Man Page for Command</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>dminfo(1M)</code></td>
<td>Searches for an allocatable device by device type, by device name, and by full path name.</td>
</tr>
<tr>
<td><code>list_devices(1)</code></td>
<td>Lists the status of allocatable devices. Lists all the device-special files that are associated with any device that is listed in the <code>device_maps</code> file.</td>
</tr>
<tr>
<td></td>
<td>With the <code>-U</code> option, lists the devices that are allocatable or allocated to the specified user ID. This option allows you to check which devices are allocatable or allocated to another user. You must have the <code>solaris.device.revoke</code> authorization.</td>
</tr>
<tr>
<td><code>allocate(1)</code></td>
<td>Reserves an allocatable device for use by one user. By default, a user must have the <code>solaris.device.allocate</code> authorization to allocate a device. You can modify the <code>device_allocate</code> file to not require user authorization. Then, any user on the system can request the device to be allocated for use.</td>
</tr>
<tr>
<td><code>deallocate(1)</code></td>
<td>Removes the allocation reservation from a device.</td>
</tr>
</tbody>
</table>

**Authorizations for the Allocation Commands**

By default, users must have the `solaris.device.allocate` authorization to reserve an allocatable device. To create a rights profile to include the `solaris.device.allocate` authorization, see “How to Authorize Users to Allocate a Device” on page 74.
Administrators must have the `solaris.device.revoke` authorization to change the allocation state of any device. For example, the `-U` option to the `allocate` and `list_devices` commands, and the `-F` option to the `deallocate` command require the `solaris.device.revoke` authorization.

For more information, see “Selected Commands That Require Authorizations” on page 203.

**Allocate Error State**

A device is put in an *allocate error state* when the `deallocate` command fails to deallocate, or when the `allocate` command fails to allocate. When an allocatable device is in an allocate error state, then the device must be forcibly deallocated. Only a user or role with the Device Management rights profile or the Device Security rights profile can handle an allocate error state.

The `deallocate` command with the `-F` option forces deallocation. Or, you can use `allocate -U` to assign the device to a user. Once the device is allocated, you can investigate any error messages that appear. After any problems with the device are corrected, you can forcibly deallocate it.

**device_maps File**

Device maps are created when you set up device allocation. The `/etc/security/device_maps` file includes the device names, device types, and device-special files that are associated with each allocatable device.

The `device_maps` file defines the device-special file mappings for each device, which in many cases is not intuitive. This file allows programs to discover which device-special files map to which devices. You can use the `dminfo` command, for example, to retrieve the device name, the device type, and the device-special files to specify when you set up an allocatable device. The `dminfo` command uses the `device_maps` file to report this information.

Each device is represented by a one-line entry of the form:

```
device-name:device-type:device-list
```

**EXAMPLE 5-11 Sample device_maps Entry**

The following is an example of an entry in a `device_maps` file for a diskette drive, `fd0`:

```
fd0:\
/dev/diskette /dev/rdiskette /dev/fd0a /dev/rfd0a \ 
/dev/fd0b /dev/rfd0b /dev/fd0c /dev/fd0 /dev/rfd0c /dev/rfd0:\
```

Lines in the `device_maps` file can end with a backslash (`\`) to continue an entry on the next line. Comments can also be included. A pound sign (`#`) comments all subsequent text until the next
newline that is not immediately preceded by a backslash. Leading and trailing blanks are
allowed in any field. The fields are defined as follows:

device-name  Specifies the name of the device. For a list of current device names, see “How to
View Allocation Information About a Device” on page 75.

device-type  Specifies the generic device type. The generic name is the name for the class of
devices, such as st, fd, rmdisk, or audio. The device-type field logically groups
related devices.

device-list  Lists the device-special files that are associated with the physical device. The
device-list must contain all of the special files that allow access to a particular
device. If the list is incomplete, a malevolent user can still obtain or modify
private information. Valid entries for the device-list field reflect the device files
that are located in the /dev directory.

device_allocate File
You can modify the /etc/security/device_allocate file to change devices from allocatable
to nonallocatable, or to add new devices. A sample device_allocate file follows.

st0;st;;;;/etc/security/lib/st_clean
fd0;fd;;;;/etc/security/lib/fd_clean
sr0;sr;;;;/etc/security/lib/sr_clean
audio;audio;;;*;/etc/security/lib/audio_clean

An entry in the device_allocate file does not mean that the device is allocatable, unless the
entry specifically states that the device is allocatable. In the sample device_allocate file, note
the asterisk (*) in the fifth field of the audio device entry. An asterisk in the fifth field indicates to
the system that the device is not allocatable. Therefore, the device cannot be used. Other values
or no value in this field indicates that the device can be used.

In the device_allocate file, each device is represented by a one-line entry of the form:

device-name;device-type;reserved;reserved;auths;device-exec

Lines in the device_allocate file can end with a backslash (\) to continue an entry on the next
line. Comments can also be included. A pound sign (#) comments all subsequent text until the
next newline that is not immediately preceded by a backslash. Leading and trailing blanks are
allowed in any field. The fields are defined as follows:

device-name  Specifies the name of the device. For a list of current device names, see “How to
View Allocation Information About a Device” on page 75.

device-type  Specifies the generic device type. The generic name is the name for the class of
devices, such as st, fd, and sr. The device-type field logically groups related
devices. When you make a device allocatable, retrieve the device name from
the device-type field in the device_maps file.
Oracle reserves the two fields that are marked reserved for future use.

Specifies whether the device is allocatable. An asterisk (*) in this field indicates that the device is not allocatable. An authorization string, or an empty field, indicates that the device is allocatable. For example, the string "solaris.device.allocate" in the auths field indicates that the solaris.device.allocate authorization is required to allocate the device. An at sign (@) in this file indicates that the device is allocatable by any user.

Supplies the path name of a script to be invoked for special handling, such as cleanup and object-reuse protection during the allocation process. The device-exec script is run any time that the device is acted on by the deallocate command.

For example, the following entry for the sr0 device indicates that the CD-ROM drive is allocatable by a user with the solaris.device.allocate authorization:

```
sr0;sr;reserved;reserved;solaris.device.allocate;/etc/security/lib/sr_clean
```

You can decide to accept the default devices and their defined characteristics. After you install a new device, you can modify the entries. Any device that needs to be allocated before use must be defined in the device_allocate and device_maps files for that device's system. Currently, cartridge tape drives, diskette drives, CD-ROM drives, removable media devices, and audio chips are considered allocatable. These device types have device-clean scripts.

Note – Xylogics and Archive tape drives also use the st_clean script that is supplied for SCSI devices. You need to create your own device-clean scripts for other devices, such as terminals, graphics tablets, and other allocatable devices. The script must fulfill object-reuse requirements for that type of device.

### Device-Clean Scripts

Device allocation satisfies part of what security auditors call the object reuse requirement. The device-clean scripts address the security requirement that all usable data be purged from a physical device before reuse. The data is cleared before the device is allocatable by another user. By default, cartridge tape drives, diskette drives, CD-ROM drives, and audio devices require device-clean scripts. Oracle Solaris provides the scripts. This section describes what device-clean scripts do.

### Device-Clean Script for Tapes

The st_clean device-clean script supports three tape devices:

- SCSI ¼-inch tape
- Archive ¼-inch tape
Open-reel ½-inch tape

The st_clean script uses the rewoffl option to the mt command to clean up the device. For more information, see the mt(1) man page. If the script runs during system boot, the script queries the device to determine if the device is online. If the device is online, the script determines if the device has media in it. The ½-inch tape devices that have media in them are placed in the allocate error state. The allocate error state forces the administrator to manually clean up the device.

During normal system operation, when the deallocate command is executed in interactive mode, the user is prompted to remove the media.Deallocation is delayed until the media is removed from the device.

Device-Clean Scripts for Diskettes and CD-ROM Drives

The following device-cleanscripts are provided for diskettes and CD-ROM drives:

- **fd_clean script** – Is a device-cleanscript for diskettes.
- **sr_clean script** – Is a device-cleanscript for CD-ROM drives.

The scripts use the eject command to remove the media from the drive. If the eject command fails, the device is placed in the allocate error state. For more information, see the eject(1) man page.

Device-Clean Script for Audio

Audio devices are cleaned up with an audio_clean script. The script performs an AUDIO_GETINFO ioctl system call to read the device. The script then performs an AUDIO_SETINFO ioctl system call to reset the device configuration to the default.

Writing New Device-Clean Scripts

If you add more allocatable devices to the system, you might need to create your own device-cleanscripts. The deallocate command passes a parameter to the device-cleanscripts. The parameter, which is shown here, is a string that contains the device name. For more information, see the device_allocate(4) man page.

```bash
clean-script -[I|i|f|S] device-name
```

Device-cleanscripts must return “0” for success and greater than “0” for failure. The options -I, -f, and -S determine the running mode of the script:

- **-I** Is needed during system boot only. All output must go to the system console. Failure or inability to forcibly eject the media must put the device in the allocate error state.
- **-i** Similar to the -I option, except that output is suppressed.
-f  Is for forced cleanup. The option is interactive and assumes that the user is available to respond to prompts. A script with this option must attempt to complete the cleanup if one part of the cleanup fails.

-S  Is for standard cleanup. The option is interactive and assumes that the user is available to respond to prompts.
This chapter describes the file integrity tool, BART. BART is a command-line tool that enables you to verify the integrity of files on a system over time. This chapter covers the following topics:

- “BART (Overview)” on page 91
- “Using BART (Tasks)” on page 93
- “BART Manifests, Rules Files, and Reports (Reference)” on page 103

**BART (Overview)**

BART is a file integrity scanning and reporting tool that uses cryptographic-strength checksums and file system metadata to determine changes. BART can help you detect security breaches or troubleshoot performance issues on a system by identifying corrupted or unusual files. Using BART can reduce the costs of administering a network of systems by easily and reliably reporting discrepancies in the files that are installed on deployed systems.

BART enables you to determine what file-level changes have occurred on a system, relative to a known baseline. You use BART to create a baseline or control manifest from a fully installed and configured system. You can then compare this baseline with a snapshot of the system at a later time, generating a report that lists file-level changes that have occurred on the system since it was installed.

**BART Features**

BART uses simple syntax that is both powerful and flexible. The tool enables you to track file changes on a given system over time. You can also track file differences between similar systems. Such comparisons can help you locate corrupted or unusual files, or systems whose software is out of date.
Additional benefits and uses of BART include the following:

- You can specify which files to monitor. For example, you can monitor local customizations, which can assist you in reconfiguring software easily and efficiently.
- You can troubleshoot system performance issues.

**BART Components**

BART creates two main files, a manifest and a comparison file, or report. An optional rules file enables you to customize the manifest and report.

**BART Manifest**

A manifest is a file-level snapshot of a system at a particular time. The manifest contains information about attributes of files, which can include some uniquely identifying information, such as a checksum. Options to the bart create command can target specific files and directories. A rules file can provide more fine-grained filtering, as described in “BART Rules File” on page 93.

**Note** – By default, BART catalogs all ZFS file systems under the root (/) directory. Other file system types, such as NFS or TMPFS file systems, and mounted CD-ROMs are cataloged.

You can create a manifest of a system immediately after an initial Oracle Solaris installation. You can also create a manifest after configuring a system to meet your site’s security policy. This type of control manifest provides you with a baseline for later comparisons.

A baseline manifest can be used to track file integrity on the same system over time. It can also be used as a basis for comparison with other systems. For example, you could take a snapshot of other systems on your network and then compare those manifests with the baseline manifest.

Reported file discrepancies indicate what you need to do to synchronize the other systems with the baseline system.

For the format of a manifest, see “BART Manifest File Format” on page 103. To create a manifest, use the bart create command, as described in “How to Create a Control Manifest” on page 94.

**BART Report**

A BART report lists per-file discrepancies between two manifests. A discrepancy is a change to any attribute for a given file that is cataloged for both manifests. Additions or deletions of file entries are also considered discrepancies.

For a useful comparison, the two manifests must target the same file systems. You must also create and compare the manifests with the same options and rules file.
For the format of a report, see “BART Reporting” on page 105. To create a report, use the bart compare command, as described in “How to Compare Manifests for the Same System Over Time” on page 97.

**BART Rules File**

A BART rules file is a file that you create to filter or target particular files and file attributes for inclusion or exclusion. You then use this file when creating BART manifests and reports. When you compare manifests, the rules file aids in flagging discrepancies between the manifests.

**Note** – When you create a manifest by using a rules file, you must use the same rules file to create the comparison manifest. You must also use the rules file when comparing the manifests. Otherwise, the report would list many invalid discrepancies.

Using a rules file to monitor specific files and file attributes on a system requires planning. Before you create a rules file, decide which files and file attributes on the system you want to monitor.

As a result of user error, a rules file can also contain syntax errors and other ambiguous information. If a rules file has errors, these errors are also reported.

For the format of a rules file, see “BART Rules File Format” on page 104 and the bart_rules(4) man page. To create a rules file, see “How to Customize a BART Report by Using a Rules File” on page 101.

**Using BART (Tasks)**

The bart command is used to create and compare manifests. Any user can run this command. However, users can only catalog and monitor files that they have permission to access. So, users and most roles can usefully catalog the files in their home directory, but the root account can catalog all files, including system files.

**BART Security Considerations**

BART manifests and reports are readable by anyone. If BART output might contain sensitive information, take appropriate measures to protect the output. For example, use options that generate output files with restrictive permissions or place output files in a protected directory.
Using BART (Task Map)

<table>
<thead>
<tr>
<th>Task</th>
<th>Description</th>
<th>For Instructions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create a BART manifest.</td>
<td>Generates a list of information about every file that is installed on a system.</td>
<td>“How to Create a Control Manifest” on page 94</td>
</tr>
<tr>
<td>Create a custom BART manifest.</td>
<td>Generates a list of information about specific files that are installed on a system.</td>
<td>“How to Customize a Manifest” on page 96</td>
</tr>
<tr>
<td>Compare BART manifests.</td>
<td>Generates a report that compares changes to a system over time. Or, generates a report that compares one or several systems to a control system.</td>
<td>“How to Compare Manifests for the Same System Over Time” on page 97 “How to Compare Manifests From Different Systems” on page 99</td>
</tr>
<tr>
<td>(Optional) Customize a BART report.</td>
<td>Generates a custom BART report in one of the following ways: ▪ By specifying attributes ▪ By using a rules file</td>
<td>“How to Customize a BART Report by Specifying File Attributes” on page 101 “How to Customize a BART Report by Using a Rules File” on page 101</td>
</tr>
</tbody>
</table>

▼ How to Create a Control Manifest

This procedure explains how to create a baseline manifest for your system. Use this type of manifest when you are installing many systems from a central image. Or, use this type of manifest to run comparisons when you want to verify that the installations are identical. For more information about baseline manifests, see “BART Manifest” on page 92. To understand the format conventions, see Example 6–1.

**Note** – Do not attempt to catalog networked file systems. Using BART to monitor networked file systems consumes large resources to generate manifests of little value.

Before You Begin

To create a control manifest of your system, you must assume the root role.

1. After customizing your Oracle Solaris system to your site’s security requirements, create a control manifest and redirect the output to a file.
   
   ```
   # bart create options > control-manifest
   ```
   
   **-R** Specifies the root directory for the manifest. All paths specified by the rules are interpreted relative to this directory. All paths reported in the manifest are relative to this directory.

   **-I** Accepts a list of individual files to be cataloged, either on the command line or read from standard input.
-r  Is the name of the rules file for this manifest. A - argument reads the rules file from standard input.

-n  Turns off content signatures for all regular files in the file list. This option can be used to improve performance. Or, you can use this option if the contents of the file list are expected to change, as in the case of system log files.

2  Examine the contents of the manifest.
   For an explanation of the format, see Example 6–1.

3  (Optional) Protect the manifest.
   One way to protect system manifests is to place them in a directory that only the root account can access.
   # mkdir /var/adm/log/bartlogs
   # chmod 700 /var/adm/log/bartlogs
   # mv control-manifest /var/adm/log/bartlogs

   Choose a meaningful name for the manifest. For example, use the system name and date that the manifest was created, as in mach1-120312.

Example 6–1  Explanation of the BART Manifest Format

In this example, an explanation of the manifest format follows the sample output.

```
# bart create
! Version 1.1
! HASH SHA256
! Friday, September 07, 2012 (22:22:27)
# Format:
# fname D size mode acl dirmtime uid gid
# fname P size mode acl mtime uid gid
# fname S size mode acl mtime uid gid contents
# fname L size mode acl lmtime uid gid dest
# fname B size mode acl mtime uid gid devnode
#fname C size mode acl mtime uid gid devnode
/ D 1024 40755 user::rwx,group::r-x,mask::r-x,other::r-x
3ebc418e5be3729ffe7e54053be2d33ee884205502c81ae9689cd8cca5b0090 0 0
    ...
/zone D 512 40755 user::rwx group::r-x,other::r-x 3f81e892
154de3e7bdfd0d57a074c9fae0896a9e2e04bebfe5e872d273b063319e57f334 0 0
    ...
```

Each manifest consists of a header and file entries. Each file entry is a single line, depending on the file type. For example, for each file entry in the preceding output, type F specifies a file and type D specifies a directory. Also listed is information about size, content, user ID, group ID, and
permissions. File entries in the output are sorted by the encoded versions of the file names to correctly handle special characters. All entries are sorted in ascending order by file name. All nonstandard file names, such as those that contain embedded newline or tab characters, quote the nonstandard characters before sorting.

Lines that begin with ! supply metadata about the manifest. The manifest version line indicates the manifest specification version. The hash line indicates the hash mechanism that was used. For more information about the SHA256 hash that is used as a checksum, see the `sha2(3EXT)` man page.

The date line shows the date on which the manifest was created, in date form. See the `date(1)` man page. Some lines are ignored by the manifest comparison tool. Ignored lines include metadata, blank lines, lines that consist only of white space, and comments that begin with #.

## How to Customize a Manifest

You can customize a manifest in one of the following ways:

- **By specifying a subtree**
  Specifying an individual subtree is an efficient way to monitor changes to selected, important files, such as all files in the `/etc` directory.

- **By specifying a file name**
  Specifying a file name is an efficient way of monitoring particularly sensitive files, such as the files that configure and run a database application.

- **By using a rules file**
  By using a rules file to create and compare manifests gives you the flexibility to specify multiple attributes for more than one file or subtree. From the command line, you can specify a global attribute definition that applies to all files in a manifest or report. From a rules file, you can specify attributes that do not apply globally.

**Before You Begin**

You must assume the root role. For more information, see "How to Use Your Assigned Administrative Rights" on page 157.

1. **Determine which files to catalog and monitor.**

2. **Create a custom manifest by using one of the following options:**
   - By specifying a subtree:
     
     ```bash
     # bart create -R subtree
     ```
   - By specifying a file name or file names:
     
     ```bash
     # bart create -I filename...
     ```
For example:

```bash
# bart create -I /etc/system /etc/passwd /etc/shadow
```

- By using a rules file:

  ```bash
  # bart create -r rules-file
  ```

3 Examine the contents of the manifest.

4 **(Optional)** Save the manifest in a protected directory for future use.
   For an example, see Step 3 in “How to Create a Control Manifest” on page 94.

---

**Tip** – If you used a rules file, save the rules file with the manifest. For a useful comparison, you must run the comparison with the rules file.

### How to Compare Manifests for the Same System Over Time

By comparing manifests over time, you can locate corrupted or unusual files, detect security breaches, and troubleshoot performance issues on a system.

**Before You Begin**

To create and compare manifests that include system files, you must assume the root role.

1 **Create a control manifest of the files to monitor on the system.**
   
   ```bash
   # bart create -R /etc > control-manifest
   ```

2 **(Optional)** Save the manifest in a protected directory for future use.
   For an example, see Step 3 in “How to Create a Control Manifest” on page 94.

3 **At a later time, prepare an identical manifest to the control manifest.**
   
   ```bash
   # bart create -R /etc > test-manifest
   ```

4 **Protect the second manifest.**
   
   ```bash
   # mv test-manifest /var/adm/log/bartlogs
   ```

5 **Compare the two manifests.**
   Use the same command-line options and rules file to compare the manifests that you used to create them.
   
   ```bash
   # bart compare options control-manifest test-manifest > bart-report
   ```

6 **Examine the BART report for oddities.**
Tracking File Changes for the Same System Over Time

This example shows how to track the changes in the /etc directory over time. This type of comparison enables you to locate important files on the system that have been compromised.

- Create a control manifest.

```bash
# cd /var/adm/logs/manifests
# bart create -R /etc > system1.control.090712

Version 1.1
!
HASH SHA256
!
Friday, September 07, 2012 (11:11:17)
#
# Format:
# fname  D  size  mode  acl  dirmtime  uid  gid
# fname  P  size  mode  acl  mtime  uid  gid
# fname  S  size  mode  acl  mtime  uid  gid
# fname  F  size  mode  acl  mtime  uid  gid  contents
# fname  L  size  mode  acl  lmtime  uid  gid  dest
# fname  B  size  mode  acl  mtime  uid  gid  devmode
# fname  C  size  mode  acl  mtime  uid  gid  devmode

/.cpr_config  F  2236  100644  owner@:read_data/write_data/append_data/read_xattr/write_xattr/read_attributes/write_attributes/read_acl/write_acl/write_acl/write_acl/write_owner/synchronize:allow,group@:read_data/read_xattr/read_attributes/read_acl/synchronize:allow,everyone@:read_data/read_xattr/read_attributes/read_acl/synchronize:allow,everyone@:read_data/read_xattr/read_attributes/read_acl/synchronize:allow,group@:read_data/read_xattr/read_attributes/read_acl/synchronize:allow,everyone@:read_data/read_xattr/read_attributes/read_acl/synchronize:allow,group@:read_data/read_xattr/read_attributes/read_acl/synchronize:allow,group@:read_data/read_xattr/read_attributes/read_acl/synchronize:allow,everyone@:read_data/read_xattr/read_attributes/read_acl/synchronize:allow,group@:read_data/read_xattr/read_attributes/read_acl/synchronize:allow
4e271c59 0 0 3ebc418be3729ff7e54053be2d33ee8b4205502c81ae9689cd8caca5b0090
.
.
.
```

- Later, create a test manifest by using the same command-line options.

```bash
# bart create -R /etc > system1.test.101012

Version 1.1
!
HASH SHA256
!
Wednesday, October 10, 2012 (10:10:17)
#
# Format:
# fname  D  size  mode  acl  dirmtime  uid  gid
# fname  P  size  mode  acl  mtime  uid  gid
# fname  S  size  mode  acl  mtime  uid  gid
# fname  F  size  mode  acl  mtime  uid  gid  contents
# fname  L  size  mode  acl  lmtime  uid  gid  dest
# fname  B  size  mode  acl  mtime  uid  gid  devmode
# fname  C  size  mode  acl  mtime  uid  gid  devmode

/.cpr_config  F  2236  100644  owner@:read_data/write_data/append_data/read_xattr/write_xattr/read_attributes/write_attributes/read_acl/write_acl/write_acl/write_acl/write_owner/synchronize:allow,group@:read_data/read_xattr/read_attributes/read_acl/synchronize:allow,everyone@:read_data/read_xattr/read_attributes/read_acl/synchronize:allow,everyone@:read_data/read_xattr/read_attributes/read_acl/synchronize:allow,everyone@:read_data/read_xattr/read_attributes/read_acl/synchronize:allow,everyone@:read_data/read_xattr/read_attributes/read_acl/synchronize:allow
4e271c59 0 0 3ebc418be3729ff7e54053be2d33ee8b4205502c81ae9689cd8caca5b0090
.
.
.
```

- Compare the manifests.
The output indicates that the modification time on the audit_class file has changed since the control manifest was created. If this change is unexpected, you can investigate further.

How to Compare Manifests From Different Systems

By comparing manifests from different systems, you can determine if the systems are installed identically or have been upgraded in sync. For example, if you customized your systems to a particular security target, this comparison finds any discrepancies between the manifest that represents your security target, and the manifests from the other systems.

Before You Begin
To compare system manifests, you must assume the root role.

1 Create a control manifest.
   # bart create options > control-manifest
   For the options, see the bart(1M) man page.

2 (Optional) Save the manifest in a protected directory for future use.
   For an example, see Step 3 in “How to Create a Control Manifest” on page 94.

3 On the test system, use the same bart options to create a manifest.
   # bart create options > test1-manifest

4 (Optional) Save the manifest in a protected directory for future use.

5 To perform the comparison, copy the manifests to a central location.
   For example:
   # cp control-manifest /net/test-server/var/adm/logs/bartlogs
   If the test system is not an NFS-mounted system, use sftp or another reliable means to copy the manifests to a central location.

6 Compare the manifests and redirect the output to a file.
   # bart compare control-manifest test1-manifest > test1.report

7 Examine the BART report for oddities.
Example 6–3  Identifying a Suspect File in the /usr/bin Directory

This example compares the contents of the /usr/bin directory on two systems.

■ Create a control manifest.

```bash
# bart create -R /usr/bin > control-manifest.090711
! Version 1.1
! HASH SHA256
! Wednesday, September 07, 2011 (11:11:17)
# Format:
#fname D size mode acl dirmtime uid gid
#fname P size mode acl mtime uid gid
#fname S size mode acl mtime uid gid
#fname F size mode acl mtime uid gid contents
#fname L size mode acl lmtime uid gid dest
#fname B size mode acl mtime uid gid devnode
#fname C size mode acl mtime uid gid devnode
/2to3 F 105 100555 owner@:read_data/read_xattr/write_xattr/execute/read_attributes/write_attributes/read_acl/write_acl/write_owner/synchronize:allow,group@:read_data/read_xattr/execute/read_attributes/read_acl/synchronize:allow,everyone@:read_data/read_xattr/execute/read_attributes/read_acl/synchronize:allow 4bf9d261 0 2 154de3e7bdf057a074c9fae0896e9e2e04bebfe5e872d273b063319e57f334
/7z F 580228 100555 owner@:read_data/read_xattr/write_xattr/execute/read_attributes/write_attributes/read_acl/write_acl/write_owner/synchronize:allow,group@:read_data/read_xattr/execute/read_attributes/read_acl/synchronize:allow,everyone@:read_data/read_xattr/execute/read_attributes/read_acl/synchronize:allow 4dadc48a 0 2 3ecd418eb5be3729ffe7e54053be2d33ee884205502c81ae9689cd8cca5b0090
...```

■ Create an identical manifest for each system that you want to compare with the control system.

```bash
# bart create -R /usr/bin > system2-manifest.101011
! Version 1.1
! HASH SHA256
! Monday, October 10, 2011 (10:10:22)
# Format:
#fname D size mode acl dirmtime uid gid
#fname P size mode acl mtime uid gid
#fname S size mode acl mtime uid gid
#fname F size mode acl mtime uid gid contents
#fname L size mode acl lmtime uid gid dest
#fname B size mode acl mtime uid gid devnode
#fname C size mode acl mtime uid gid devnode
/2to3 F 105 100555 owner@:read_data/read_xattr/write_xattr/execute/read_attributes/write_attributes/read_acl/write_acl/write_owner/synchronize:allow,group@:read_data/read_xattr/execute/read_attributes/read_acl/synchronize:allow,everyone@:read_data/read_xattr/execute/read_attributes/read_acl/synchronize:allow 4bf9d261 0 2 154de3e7bdf057a074c9fae0896e9e2e04bebfe5e872d273b063319e57f334
...```

■ Copy the manifests to the same location.

```bash
# cp control-manifest.090711 /net/system2.central/bart/manifests
```

■ Compare the manifests.
# bart compare control-manifest.090711 system2.test.101011 > system2.report

/su:
  gid control:3 test:1
  /ypcat:
    mtime control:3fd72511 test:3fd9eb23

The output indicates that the group ID of the su file in the /usr/bin directory is not the same as that of the control system. This information might indicate that a different version of the software was installed on the test system. Because the GID is changed, the more likely reason is that someone has tampered with the file.

▼ How to Customize a BART Report by Specifying File Attributes

This procedure is useful to filter the output from existing manifests for specific file attributes.

**Before You Begin**

You must assume the root role. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1. **Determine which file attributes to check.**

2. **Compare two manifests that contain the file attributes to be checked.**
   
   For example:
   
   # bart compare -i lnmtime,mtime control-manifest.121511 \
   test-manifest.010512 > bart.report.010512
   
   Use a comma in the command-line syntax to separate each file attribute.

3. **Examine the BART report for oddities.**

▼ How to Customize a BART Report by Using a Rules File

By using a rules file, you can customize a BART manifest for particular files and file attributes of interest. By using different rules files on default BART manifests, you can run different comparisons for the same manifests.

**Before You Begin**

You must assume the root role. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1. **Determine which files and file attributes to monitor.**

2. **Create a rules file with the appropriate directives.**
3 Create a control manifest with the rules file that you created.
   # bart create -r myrules1-file > control-manifest

4 (Optional) Save the manifest in a protected directory for future use.
   For an example, see Step 3 in "How to Create a Control Manifest" on page 94.

5 Create an identical manifest on a different system, at a later time, or both.
   # bart create -r myrules1-file > test-manifest

6 Compare the manifests by using the same rules file.
   # bart compare -r myrules1-file control-manifest test-manifest > bart.report

7 Examine the BART report for oddities.

Example 6–4 Using a Rules File to Customize BART Manifests and the Comparison Report

The following rules file directs the bart create command to list all attributes of the files in the /usr/bin directory. In addition, the rules file directs the bart compare command to report only size and content changes in the same directory.

# Check size and content changes in the /usr/bin directory.
# This rules file only checks size and content changes.
# See rules file example.

IGNORE all
CHECK size contents
/usr/bin

■ Create a control manifest with the rules file that you created.

   # bart create -r usbrinrules.txt > usr_bin.control-manifest.121011

■ Prepare an identical manifest whenever you want to monitor changes to the /usr/bin directory.

   # bart create -r usbrinrules.txt > usr_bin.test-manifest.121111

■ Compare the manifests by using the same rules file.

   # bart compare -r usbrinrules.txt usr_bin.control-manifest.121011 \ usr_bin.test-manifest.121111

■ Examine the output of the bart compare command.

   /usr/bin/gunzip: add
   /usr/bin/ypcat: delete

The preceding output indicates that the /usr/bin/ypcat file was deleted, and the /usr/bin/gunzip file was added.
BART Manifests, Rules Files, and Reports (Reference)

This section describes the format of files that BART uses and creates.

**BART Manifest File Format**

Each manifest file entry is a single line, depending on the file type. Each entry begins with `fname`, which is the name of the file. To prevent parsing problems that are caused by special characters embedded in file names, the file names are encoded. For more information, see “BART Rules File Format” on page 104.

Subsequent fields represent the following file attributes:

- **type**: Type of file with the following possible values:
  - B for a block device node
  - C for a character device node
  - D for a directory
  - F for a file
  - L for a symbolic link
  - P for a pipe
  - S for a socket

- **size**: File size in bytes.

- **mode**: Octal number that represents the permissions of the file.

- **acl**: ACL attributes for the file. For a file with ACL attributes, this contains the output from `acl to text()`.

- **uid**: Numerical user ID of the owner of this entry.

- **gid**: Numerical group ID of the owner of this entry.

- **dirmtime**: Last modification time, in seconds, since 00:00:00 UTC, January 1, 1970, for directories.

- **lnmtime**: Last modification time, in seconds, since 00:00:00 UTC, January 1, 1970, for links.

- **mtime**: Last modification time, in seconds, since 00:00:00 UTC January 1, 1970, for files.

- **contents**: Checksum value of the file. This attribute is only specified for regular files. If you turn off context checking, or if checksums cannot be computed, the value of this field is -.

- **dest**: Destination of a symbolic link.

- **devnode**: Value of the device node. This attribute is for character device files and block device files only.
For more information, see the `bart_manifest(4)` man page.

**BART Rules File Format**

Rules files are text files that consist of lines that specify which files are to be included in the manifest and which file attributes are to be included in the manifest or the report. Lines that begin with #, blank lines, and lines that contain white space are ignored by the tool.

The input files have three types of directives:

- Subtree directive, with optional pattern matching modifiers
- CHECK directive
- IGNORE directive

**EXAMPLE 6–5  Rules File Format**

```
<Global CHECK/IGNORE Directives>
<subtree1> [pattern1..]
<IGNORE/CHECK Directives for subtree1>

<subtree2> [pattern2..]
<subtree3> [pattern3..]
<subtree4> [pattern4..]
<IGNORE/CHECK Directives for subtree2, subtree3, subtree4>
```

**Note** – All directives are read in order. Later directives can override earlier directives.

A subtree directive *must* begin with an absolute pathname, followed by zero or more pattern matching statements.

**Rules File Attributes**

The CHECK and IGNORE statements define which file attributes to track or ignore. The metadata that begins each manifest lists the attribute *keywords* per file type. For an example, see Example 6–1.

The `all` keyword indicates all file attributes.

**Quoting Syntax**

The rules file specification language that BART uses is the standard UNIX quoting syntax for representing nonstandard file names. Embedded tab, space, newline, or special characters are encoded in their octal forms to enable the tool to read file names. This nonuniform quoting syntax prevents certain file names, such as those containing an embedded carriage return, from
being processed correctly in a command pipeline. The rules specification language allows the expression of complex file name filtering criteria that would be difficult and inefficient to describe by using shell syntax alone.

For more information, see the `bart_rules(4)` man page.

**BART Reporting**

In default mode, a BART report checks all the files installed on the system, with the exception of modified directory timestamps (`dirmtime`):

```
CHECK all
IGNORE dirmtime
```

If you supply a rules file, then the global directives of `CHECK all` and `IGNORE dirmtime`, in that order, are automatically prepended to the rules file.

**BART Output**

The following exit values are returned:

- **0** Success
- **1** Nonfatal error when processing files, such as permission problems
- **>1** Fatal error, such as an invalid command-line option

The reporting mechanism provides two types of output: verbose and programmatic:

- **Verbose output** is the default output and is localized and presented on multiple lines. Verbose output is internationalized and is human-readable. When the `bart compare` command compares two system manifests, a list of file differences is generated. The structure of the output is as follows:

  ```
  filename attribute control-val test-val
  filename Name of the file that differs between the control manifest and the test manifest.
  attribute Name of the file attribute that differs between the manifests that are compared.
  The control-val precedes the test-val. When discrepancies for multiple attributes occur in the same file, each difference is noted on a separate line.
  ```

  Following is an example of attribute differences for the `/etc/passwd` file. The output indicates that the size, mtime, and contents attributes have changed.

  ```
  /etc/passwd:
  size control:74 test:81
  mtime control:3c165879 test:3c165979
  ```
Programmatic output is generated with the -p option to the bart compare command. This output is suitable for programmatic manipulation.

The structure of the output is as follows:

```
filename attribute control-val test-val  [attribute control-val test-val]*
filename               Same as the filename attribute in the default format
attribute control-val test-val  A description of the file attributes that differ between the control and test manifests for each file
```

For a list of attributes that are supported by the bart command, see “Rules File Attributes” on page 104.

For more information, see the bart(1M) man page.
This chapter describes how to protect files in Oracle Solaris. The chapter also describes how to protect the system from files whose permissions could compromise the system.

Note – To protect ZFS files with access control lists (ACLs), see Chapter 7, “Using ACLs and Attributes to Protect Oracle Solaris ZFS Files,” in Oracle Solaris 11.1 Administration: ZFS File Systems.

The chapter covers the following topics:

- “Using UNIX Permissions to Protect Files” on page 107
- “Protecting Executable Files From Compromising Security” on page 114
- “Protecting Files With UNIX Permissions (Task Map)” on page 115
- “Protecting Against Programs With Security Risk (Task Map)” on page 121

## Using UNIX Permissions to Protect Files

Files can be secured through UNIX file permissions and through ACLs. Files with sticky bits, and files that are executable, require special security measures.

## Commands for Viewing and Securing Files

This table describes the commands for monitoring and securing files and directories.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
<th>Man Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ls</td>
<td>Lists the files in a directory and information about the files.</td>
<td>ls(1)</td>
</tr>
</tbody>
</table>
TABLE 7-1 Commands for Securing Files and Directories (Continued)

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
<th>Man Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>chown</td>
<td>Changes the ownership of a file.</td>
<td>chown(1)</td>
</tr>
<tr>
<td>chgrp</td>
<td>Changes the group ownership of a file.</td>
<td>chgrp(1)</td>
</tr>
<tr>
<td>chmod</td>
<td>Changes permissions on a file. You can use either symbolic mode, which uses letters and symbols, or absolute mode, which uses octal numbers, to change permissions on a file.</td>
<td>chmod(1)</td>
</tr>
</tbody>
</table>

**File and Directory Ownership**

Traditional UNIX file permissions can assign ownership to three classes of users:

- **user** – The file or directory owner, which is usually the user who created the file. The owner of a file can decide who has the right to read the file, to write to the file (make changes to it), or, if the file is a command, to execute the file.

- **group** – Members of a group of users.

- **others** – All other users who are not the file owner and are not members of the group.

The owner of the file can usually assign or modify file permissions. Additionally, the root account can change a file's ownership. To override system policy, see Example 7–2.

A file can be one of seven types. Each type is displayed by a symbol:

- (Minus symbol) Text or program
- b Block special file
- c Character special file
- d Directory
- l Symbolic link
- s Socket
- D Door
- P Named pipe (FIFO)

**UNIX File Permissions**

The following table lists and describes the permissions that you can give to each class of user for a file or directory.


TABLE 7–2  File and Directory Permissions

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Permission</th>
<th>Object</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>r</td>
<td>Read</td>
<td>File</td>
<td>Designated users can open and read the contents of a file.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Directory</td>
<td>Designated users can list files in the directory.</td>
</tr>
<tr>
<td>w</td>
<td>Write</td>
<td>File</td>
<td>Designated users can modify the contents of the file or delete the file.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Directory</td>
<td>Designated users can add files or add links in the directory. They can also</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>remove files or remove links in the directory.</td>
</tr>
<tr>
<td>x</td>
<td>Execute</td>
<td>File</td>
<td>Designated users can execute the file, if it is a program or shell script.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Directory</td>
<td>Designated users can open files or execute files in the directory. They also</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>can make the directory and the directories beneath it current.</td>
</tr>
<tr>
<td>-</td>
<td>Denied</td>
<td>File and</td>
<td>Designated users cannot read, write, or execute the file.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Directory</td>
<td></td>
</tr>
</tbody>
</table>

These file permissions apply to regular files, and to special files such as devices, sockets, and named pipes (FIFOs).

For a symbolic link, the permissions that apply are the permissions of the file that the link points to.

You can protect the files in a directory and its subdirectories by setting restrictive file permissions on that directory. Note, however, that the root role has access to all files and directories on the system.

**Special File Permissions (setuid, setgid and Sticky Bit)**

Three special types of permissions are available for executable files and public directories: setuid, setgid, and sticky bit. When these permissions are set, any user who runs that executable file assumes the ID of the owner (or group) of the executable file.

You must be extremely careful when you set special permissions, because special permissions constitute a security risk. For example, a user can gain root capabilities by executing a program that sets the user ID (UID) to 0, which is the UID of root. Also, all users can set special permissions for files that they own, which constitutes another security concern.

You should monitor your system for any unauthorized use of the setuid permission and the setgid permission to gain root capabilities. A suspicious permission grants ownership of an administrative program to a user rather than to root or bin. To search for and list all files that use this special permission, see “How to Find Files With Special File Permissions” on page 121.
**setuid Permission**

When setuid permission is set on an executable file, a process that runs this file is granted access on the basis of the owner of the file. The access is *not* based on the user who is running the executable file. This special permission allows a user to access files and directories that are normally available only to the owner.

For example, the setuid permission on the passwd command makes it possible for users to change passwords. A passwd command with setuid permission would resemble the following:

```
-r-sr-sr-x 3 root sys 28144 Jun 17 12:02 /usr/bin/passwd
```

This special permission presents a security risk. Some determined users can find a way to maintain the permissions that are granted to them by the setuid process even after the process has finished executing.

**Note** – The use of setuid permissions with the reserved UIDs (0-100) from a program might not set the effective UID correctly. Use a shell script, or avoid using the reserved UIDs with setuid permissions.

**setgid Permission**

The setgid permission is similar to the setuid permission. The process's effective group ID (GID) is changed to the group that owns the file, and a user is granted access based on the permissions that are granted to that group. The /usr/bin/mail command has setgid permissions:

```
-r-x--s--x 1 root mail 67504 Jun 17 12:01 /usr/bin/mail
```

When the setgid permission is applied to a directory, files that were created in this directory belong to the group to which the directory belongs. The files do not belong to the group to which the creating process belongs. Any user who has write and execute permissions in the directory can create a file there. However, the file belongs to the group that owns the directory, not to the group that the user belongs to.

You should monitor your system for any unauthorized use of the setgid permission to gain root capabilities. A suspicious permission grants group access to such a program to an unusual group rather than to root or bin. To search for and list all files that use this permission, see “How to Find Files With Special File Permissions” on page 121.

**Sticky Bit**

The sticky bit is a permission bit that protects the files within a directory. If the directory has the sticky bit set, a file can be deleted only by the file owner, the directory owner, or by a privileged user. The root user is an example of a privileged user. The sticky bit prevents a user from deleting other users' files from public directories such as /tmp:
Be sure to set the sticky bit manually when you set up a public directory on a TMPFS file system. For instructions, see Example 7–5.

**Default umask Value**

When you create a file or directory, you create it with a default set of permissions. The system defaults are open. A text file has 666 permissions, which grants read and write permission to everyone. A directory and an executable file have 777 permissions, which grants read, write, and execute permission to everyone. Typically, users override the system defaults in their shell initialization files, such as .bashrc and .kshrc.user. An administrator can also set defaults in the /etc/profile file.

The value assigned by the umask command is subtracted from the default. This process has the effect of denying permissions in the same way that the chmod command grants them. For example, the chmod 022 command grants write permission to group and others. The umask 022 command denies write permission to group and others.

The following table shows some typical umask values and their effect on an executable file.

<table>
<thead>
<tr>
<th>Level of Security</th>
<th>umask Setting</th>
<th>Permissions Disallowed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Permissive (744)</td>
<td>022</td>
<td>w for group and others</td>
</tr>
<tr>
<td>Moderate (741)</td>
<td>026</td>
<td>w for group, rw for others</td>
</tr>
<tr>
<td>Strict (740)</td>
<td>027</td>
<td>w for group, rwx for others</td>
</tr>
<tr>
<td>Severe (700)</td>
<td>077</td>
<td>rwx for group and others</td>
</tr>
</tbody>
</table>

For more information about setting the umask value, see the umask(1) man page.

**File Permission Modes**

The chmod command enables you to change the permissions on a file. You must be root or the owner of a file or directory to change its permissions.
You can use the `chmod` command to set permissions in either of two modes:

- **Absolute Mode** – Use numbers to represent file permissions. When you change permissions by using the absolute mode, you represent permissions for each triplet by an octal mode number. Absolute mode is the method most commonly used to set permissions.

- **Symbolic Mode** – Use combinations of letters and symbols to add permissions or remove permissions.

The following table lists the octal values for setting file permissions in absolute mode. You use these numbers in sets of three to set permissions for owner, group, and other, in that order. For example, the value 644 sets read and write permissions for owner, and read-only permissions for group and other.

**TABLE 7–4  Setting File Permissions in Absolute Mode**

<table>
<thead>
<tr>
<th>Octal Value</th>
<th>File Permissions Set</th>
<th>Permissions Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>- - -</td>
<td>No permissions</td>
</tr>
<tr>
<td>1</td>
<td>- - x</td>
<td>Execute permission only</td>
</tr>
<tr>
<td>2</td>
<td>- w -</td>
<td>Write permission only</td>
</tr>
<tr>
<td>3</td>
<td>- wx</td>
<td>Write and execute permissions</td>
</tr>
<tr>
<td>4</td>
<td>r - -</td>
<td>Read permission only</td>
</tr>
<tr>
<td>5</td>
<td>r - x</td>
<td>Read and execute permissions</td>
</tr>
<tr>
<td>6</td>
<td>r w -</td>
<td>Read and write permissions</td>
</tr>
<tr>
<td>7</td>
<td>rwx</td>
<td>Read, write, and execute permissions</td>
</tr>
</tbody>
</table>

The following table lists the symbols for setting file permissions in symbolic mode. Symbols can specify whose permissions are to be set or changed, the operation to be performed, and the permissions that are being assigned or changed.

**TABLE 7–5  Setting File Permissions in Symbolic Mode**

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>u</td>
<td>who</td>
<td>User (owner)</td>
</tr>
<tr>
<td>g</td>
<td>who</td>
<td>Group</td>
</tr>
<tr>
<td>o</td>
<td>who</td>
<td>Others</td>
</tr>
<tr>
<td>a</td>
<td>who</td>
<td>All</td>
</tr>
<tr>
<td>=</td>
<td>operator</td>
<td>Assign</td>
</tr>
</tbody>
</table>
### TABLE 7–5 Settings File Permissions in Symbolic Mode (Continued)

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>+</td>
<td>operator</td>
<td>Add</td>
</tr>
<tr>
<td>-</td>
<td>operator</td>
<td>Remove</td>
</tr>
<tr>
<td>r</td>
<td>permissions</td>
<td>Read</td>
</tr>
<tr>
<td>w</td>
<td>permissions</td>
<td>Write</td>
</tr>
<tr>
<td>x</td>
<td>permissions</td>
<td>Execute</td>
</tr>
<tr>
<td>l</td>
<td>permissions</td>
<td>Mandatory locking, setgid bit is on, group execution bit is off</td>
</tr>
<tr>
<td>s</td>
<td>permissions</td>
<td>setuid or setgid bit is on</td>
</tr>
<tr>
<td>t</td>
<td>permissions</td>
<td>Sticky bit is on, execution bit for others is on</td>
</tr>
</tbody>
</table>

The *who* operator permissions designations in the function column specify the symbols that change the permissions on the file or directory.

- **who**: Specifies whose permissions are to be changed.
- **operator**: Specifies the operation to be performed.
- **permissions**: Specifies what permissions are to be changed.

You can set special permissions on a file in absolute mode or symbolic mode. However, you must use symbolic mode to set or remove *setuid* permissions on a directory. In absolute mode, you set special permissions by adding a new octal value to the left of the permission triplet. The following table lists the octal values for setting special permissions on a file.

### TABLE 7–6 Setting Special File Permissions in Absolute Mode

<table>
<thead>
<tr>
<th>Octal Value</th>
<th>Special File Permissions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Sticky bit</td>
</tr>
<tr>
<td>2</td>
<td>setgid</td>
</tr>
<tr>
<td>4</td>
<td>setuid</td>
</tr>
</tbody>
</table>

---

**Using Access Control Lists to Protect UFS Files**

Traditional UNIX file protection provides read, write, and execute permissions for the three user classes: file owner, file group, and other. In a UFS file system, an access control list (ACL) provides better file security by enabling you to do the following:

- Define file permissions for the file owner, the group, other, specific users and groups
- Define default permissions for each of the preceding categories
Note – For ACLs in the ZFS file system and ACLs on NFSv4 files, see Chapter 7, “Using ACLs and Attributes to Protect Oracle Solaris ZFS Files,” in Oracle Solaris 11.1 Administration: ZFS File Systems.

For example, if you want everyone in a group to be able to read a file, you can simply grant group read permissions on that file. However, if you want only one person in the group to be able to write to that file, you can use an ACL.

For more information about ACLs on UFS file systems, see System Administration Guide: Security Services for the Oracle Solaris 10 release.

Protecting Executable Files From Compromising Security

Programs read and write data on the stack. Typically, they execute from read-only portions of memory that are specifically designated for code. Some attacks that cause buffers on the stack to overflow try to insert new code on the stack and cause the program to execute it. Removing execute permission from the stack memory prevents these attacks from succeeding. That is, most programs can function correctly without using executable stacks.

64-bit processes always have non-executable stacks. By default, 32-bit SPARC processes have executable stacks. The noexec_user_stack variable enables you to specify whether the stacks of 32-bit processes are executable.

Once this variable is set, programs that attempt to execute code on their stack are sent a SIGSEGV signal. This signal usually results in the program terminating with a core dump. Such programs also generate a warning message that includes the name of the offending program, the process ID, and the real UID of the user who ran the program. For example:

a.out[347] attempt to execute code on stack by uid 555

The message is logged by the syslog daemon when the syslog kern facility is set to notice level. This logging is set by default in the syslog.conf file, which means that the message is sent to both the console and the /var/adm/messages file. For more information, see the syslogd(1M) and syslog.conf(4) man pages.

The syslog message is useful for observing potential security problems. The message also identifies valid programs that depend upon executable stacks that have been prevented from correct operation by setting the noexec_user_stack variable. If you do not want any messages logged, then set the log variable, noexec_user_stack_log, to zero in the /etc/system file. Even though messages are not being logged, the SIGSEGV signal can continue to cause the executing program to terminate with a core dump.

Programs can explicitly mark or prevent stack execution. The mprotect() function in programs explicitly marks the stack as executable. For more information, see the mprotect(2)
Protecting Files (Tasks)

The following procedures protect files with UNIX permissions, locate files with security risks, and protect the system from compromise by these files.

Protecting Files With UNIX Permissions (Task Map)

The following task map points to procedures that list file permissions, change file permissions, and protect files with special file permissions.

<table>
<thead>
<tr>
<th>Task</th>
<th>For Instructions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Display file information.</td>
<td>“How to Display File Information” on page 115</td>
</tr>
<tr>
<td>Change local file ownership.</td>
<td>“How to Change the Owner of a File” on page 116</td>
</tr>
<tr>
<td></td>
<td>“How to Change Group Ownership of a File” on page 117</td>
</tr>
<tr>
<td>Change local file permissions.</td>
<td>“How to Change File Permissions in Symbolic Mode” on page 118</td>
</tr>
<tr>
<td></td>
<td>“How to Change File Permissions in Absolute Mode” on page 119</td>
</tr>
<tr>
<td></td>
<td>“How to Change Special File Permissions in Absolute Mode” on page 120</td>
</tr>
</tbody>
</table>

▶ How to Display File Information

Display information about all the files in a directory by using the `ls` command.

- **Type the following command to display a long listing of all files in the current directory.**
  
  `ls -la`

  - `-l` Displays the long format that includes user ownership, group ownership, and file permissions.
  - `-a` Displays all files, including hidden files that begin with a dot (`.`).

  For all options to the `ls` command, see the `ls(1)` man page.

**Example 7–1  Displaying File Information**

In the following example, a partial list of the files in the `/sbin` directory is displayed.
% cd /sbin
% ls -la

ls -la

```
total 4960
-dwrxr-xr-x 2 root  sys  64 Dec  8 11:57 ./
-dwrxr-xr-x 39 root root  41 Dec  8 15:20 ../
-r-xr-xr-x  1 root bin 21492 Dec  1 20:55 autopush*
-r-xr-xr-x  1 root bin  33680 Oct  1 11:36 beadm*
-r-xr-xr-x  1 root bin 184360 Dec  1 20:55 bootadm*
lwrxrwxrwx  1 root root  21 Jun  7 2010 bpgetfile -> ...
-r-xr-xr-x  1 root bin  8048 Dec  1 20:55 cryptoadm*
-r-xr-xr-x  1 root bin 12828 Dec  1 20:55 devprop*
-r-xr-xr-x  1 root bin 130132 Dec  1 20:55 dhcpagent*
-r-xr-xr-x  1 root bin  13076 Dec  1 20:55 dhcpcfg*
```

Each line displays information about a file in the following order:

- **Type of file** – For example, d. For list of file types, see "File and Directory Ownership" on page 108.
- **Permissions** – For example, r-xr-xr-x. For description, see "File and Directory Ownership" on page 108.
- **Number of hard links** – For example, 2.
- **Owner of the file** – For example, root.
- **Group of the file** – For example, bin.
- **Size of the file, in bytes** – For example, 13076.
- **Date the file was created or the last date that the file was changed** – For example, Dec 1 20:55.
- **Name of the file** – For example, dhcpinfo.

## How to Change the Owner of a File

### Before You Begin

If you are not the owner of the file or directory, you must be assigned the Object Access Management rights profile. To change a file that is a public object, you must assume the root role.

For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

### Display the permissions on a local file.

% ls -l example-file

```
-rw-r--r-- 1 janedoe staff 112640 May 24 10:49 example-file
```
2 Change the owner of the file.
   `# chown stacey example-file`

3 Verify that the owner of the file has changed.
   `# ls -l example-file`
   
   `-rw-r--r-- 1 stacey staff 112640 May 26 08:50 example-file`

   To change permissions on NFS-mounted files, see Chapter 3, "Accessing Network File Systems (Reference)," in Managing Network File Systems in Oracle Solaris 11.1.

### Example 7–2 Enabling Users to Change the Ownership of Their Own Files

**Security Consideration** – You need a good reason to change the setting of the `rstchown` variable to zero. The default setting prevents users from listing their files as belonging to others so as to bypass space quotas.

In this example, the value of the `rstchown` variable is set to zero in the `/etc/system` file. This setting enables the owner of a file to use the `chown` command to change the file's ownership to another user. This setting also enables the owner to use the `chgrp` command to set the group ownership of a file to a group that the owner does not belong to. The change goes into effect when the system is rebooted.

```
set rstchown = 0
```

For more information, see the `chown(1)` and `chgrp(1)` man pages.

### How to Change Group Ownership of a File

**Before You Begin** If you are not the owner of the file or directory, you must be assigned the Object Access Management rights profile. To change a file that is a public object, you must assume the root role.

For more information, see "How to Use Your Assigned Administrative Rights” on page 157.

1 **Change the group ownership of a file.**
   ```
   $ chgrp scifi example-file
   ```
   For information about setting up groups, see Chapter 1, “Managing User Accounts and User Environments (Overview),” in Managing User Accounts and User Environments in Oracle Solaris 11.1.

2 **Verify that the group ownership of the file has changed.**
   ```
   $ ls -l example-file
   `-rw-r--r-- 1 stacey scifi 112640 June 20 08:55 example-file`
Also see Example 7–2.

▼ **How to Change File Permissions in Symbolic Mode**

In the following procedure, a user changes permissions on a file that the user owns.

1. **Change permissions in symbolic mode.**
   
   % chmod who operator permissions filename

   - **who**: Specifies whose permissions are to be changed.
   - **operator**: Specifies the operation to be performed.
   - **permissions**: Specifies what permissions are to be changed. For the list of valid symbols, see Table 7–5.
   - **filename**: Specifies the file or directory.

2. **Verify that the permissions of the file have changed.**
   
   % ls -l filename

   **Note** – If you are not the owner of the file or directory, you must be assigned the Object Access Management rights profile. To change a file that is a **public object**, you must assume the root role.

---

**Example 7–3** Changing Permissions in Symbolic Mode

In the following example, read permission is taken away from others.

% chmod o-r example-file1

In the following example, read and execute permissions are added to a local file for user, group, and others.

$ chmod a+rx example-file2

In the following example, read, write, and execute permissions for group are assigned to a local file.

$ chmod g+wx example-file3
How to Change File Permissions in Absolute Mode

In the following procedure, a user changes permissions on a file that the user owns.

1. Change permissions in absolute mode.
   
   `% chmod nnn filename`
   
   * `nnn` Specifies the octal values that represent the permissions for the file owner, file group, and others, in that order. For the list of valid octal values, see Table 7–4.
   
   * `filename` Specifies the file or directory.

   **Note** – If you use the `chmod` command to change file or directory permissions on objects that have existing ACL entries, the ACL entries might change as well. The exact changes are dependent upon the `chmod` permission operation changes and the file system's aclmode and aclinherit property values.

   For more information, see Chapter 7, “Using ACLs and Attributes to Protect Oracle Solaris ZFS Files,” in *Oracle Solaris 11.1 Administration: ZFS File Systems*.

2. Verify that the permissions of the file have changed.
   
   `% ls -l filename`

   **Note** – If you are not the owner of the file or directory, you must be assigned the Object Access Management rights profile. To change a file that is a public object, you must assume the root role.

Example 7–4  Changing Permissions in Absolute Mode

In the following example, the permissions of a directory that is open to the public are changed from 744 (read, write, execute; read-only; and read-only) to 755 (read, write, execute; read and execute; and read and execute).

```
# ls -ld public_dir
drwxr--r-- 1 jdoe staff 6023 Aug 5 12:06 public_dir
# chmod 755 public_dir
# ls -ld public_dir
drwxr-xr-x 1 jdoe staff 6023 Aug 5 12:06 public_dir
```

In the following example, the permissions of an executable shell script are changed from read and write to read, write, and execute.

```
% ls -l my_script
-rw------- 1 jdoe staff 6023 Aug 5 12:06 my_script
% chmod 700 my_script
```
How to Change Special File Permissions in Absolute Mode

**Before You Begin**
If you are not the owner of the file or directory, you must be assigned the Object Access Management rights profile. To change a file that is a public object, you must assume the root role.

For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1. **Change special permissions in absolute mode.**
   ```
   chmod nnnn filename
   ``
   - `nnnn` Specifies the octal values that change the permissions on the file or directory. The leftmost octal value sets the special permissions on the file. For the list of valid octal values for special permissions, see Table 7–6.
   - `filename` Specifies the file or directory.

   **Note** – When you use the `chmod` command to change the file group permissions on a file with ACL entries, both the file group permissions and the ACL mask are changed to the new permissions. Be aware that the new ACL mask permissions can change the permissions for additional users and groups who have ACL entries on the file. Use the `getfacl` command to make sure that the appropriate permissions are set for all ACL entries. For more information, see the `getfacl(1)` man page.

2. **Verify that the permissions of the file have changed.**
   ```
   ls -l filename
   ```

**Example 7–5** Setting Special File Permissions in Absolute Mode

In the following example, the `setuid` permission is set on the `dbprog` file.

```
# chmod 4555 dbprog
# ls -l dbprog
-r-xr-xr-x 1 db staff 12095 May 6 09:29 dbprog
```

In the following example, the `setgid` permission is set on the `dbprog2` file.

```
# chmod 2551 dbprog2
# ls -l dbprog2
-r-xr-xr-x 1 db staff 24576 May 6 09:30 dbprog2
```
In the following example, the sticky bit permission is set on the `public_dir` directory.

```bash
# chmod 1777 public_dir
# ls -ld public_dir
drwxrwxrwt 2 jdoe staff 512 May 15 15:27 public_dir
```

### Protecting Against Programs With Security Risk (Task Map)

The following task map points to procedures that find risky executables on the system, and that prevent programs from exploiting an executable stack.

<table>
<thead>
<tr>
<th>Task</th>
<th>Description</th>
<th>For Instructions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Find files with special permissions.</td>
<td>Locates files with the <code>setuid</code> bit set, but that are not owned by the root user.</td>
<td>&quot;How to Find Files With Special File Permissions&quot; on page 121</td>
</tr>
<tr>
<td>Prevent executable stack from overflowing.</td>
<td>Prevents programs from exploiting an executable stack.</td>
<td>&quot;How to Disable Programs From Using Executable Stacks&quot; on page 122</td>
</tr>
<tr>
<td>Prevent logging of executable stack messages.</td>
<td>Turns off logging of executable stack messages.</td>
<td>Example 7–7</td>
</tr>
</tbody>
</table>

#### How to Find Files With Special File Permissions

This procedure locates potentially unauthorized use of the `setuid` and `setgid` permissions on programs. A suspicious executable file grants ownership to a user rather than to `root` or `bin`.

**Before You Begin**

You must assume the `root` role. For more information, see "How to Use Your Assigned Administrative Rights" on page 157.

1. **Find files with `setuid` permissions by using the `find` command.**

   ```bash
   # find directory -user root -perm -4000 -exec ls -ldb {} \; >/tmp/filename
   find directory  Checks all mounted paths starting at the specified directory, which can be root (/), sys, bin, or mail.
   -user root      Displays files owned only by root.
   -perm -4000     Displays files only with permissions set to 4000.
   -exec ls -ldb   Displays the output of the `find` command in `ls -ldb` format.
   /tmp/filename   Is the file that contains the results of the `find` command.
   ```
2 Display the results in /tmp/filename.
   # more /tmp/filename
For background information about setuid permissions, see “setuid Permission” on page 110.

Example 7–6 Finding Files With setuid Permissions

The output from the following example shows that a user in a group called rar has made a personal copy of /usr/bin/sh, and has set the permissions as setuid to root. As a result, the /usr/rar/bin/sh program runs with root permissions.

This output was saved for future reference by moving the /var/tmp/chkprm file to an archive.

```
# find / -user root -perm -4000 -exec ls -ldb {} \; > /var/tmp/ckprm
# cat /var/tmp/ckprm
-rsr-xr-x 1 root bin 38836 Aug 10 16:16 /usr/bin/at
-rsr-xr-x 1 root bin 19812 Aug 10 16:16 /usr/bin/crontab
---sx-x-x 1 root sys 46040 Aug 10 15:18 /usr/bin/ct
-rsr-xr-x 1 root sys 12092 Aug 11 01:29 /usr/lib/mv_dir
-rsr-xr-x 1 root bin 33208 Aug 10 15:55 /usr/lib/lpadmin
-rsr-xr-x 1 root bin 38696 Aug 10 15:55 /usr/lib/lpsched
---sx--- 1 root rar 45376 Aug 18 15:11 /usr/rar/bin/sh
-rsr-xr-x 1 root bin 12574 Aug 11 01:27 /usr/bin/df
-rwxr-xr-x 1 root sys 21780 Aug 11 01:27 /usr/bin/newgrp
-rsr-xr-x 1 root sys 23000 Aug 11 01:27 /usr/bin/passwd
-rsr-xr-x 1 root sys 23824 Aug 11 01:27 /usr/bin/su
# mv /var/tmp/ckprm /export/sysreports/ckprm
```

▼ How to Disable Programs From Using Executable Stacks

For a description of the security risks of 32-bit executable stacks, see “Protecting Executable Files From Compromising Security” on page 114.

Before You Begin

You must assume the root role. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1 Edit the /etc/system file, and add the following line:
   set noexec_user_stack=1

2 Reboot the system.
   # reboot
Disabling the Logging of Executable Stack Messages

In this example, the logging of executable stack messages is disabled, and then the system is rebooted.

```bash
# cat /etc/system
set noexec_user_stack=1
set noexec_user_stack_log=0
# reboot
```

See Also For more information, read the following:

- [http://blogs.oracle.com/gbrunett/entry/solaris_non_executable_stack_overview](http://blogs.oracle.com/gbrunett/entry/solaris_non_executable_stack_overview)
- [http://blogs.oracle.com/gbrunett/entry/solaris_non_executable_stack_continued](http://blogs.oracle.com/gbrunett/entry/solaris_non_executable_stack_continued)
- [http://blogs.oracle.com/gbrunett/entry/solaris_non_executable_stack_concluded](http://blogs.oracle.com/gbrunett/entry/solaris_non_executable_stack_concluded)
Roles, Rights Profiles, and Privileges

This section covers role-based access control (RBAC) and process rights management. RBAC components include roles, rights profiles, and authorizations. Process rights management is implemented through privileges. Privileges work with RBAC to provide a more secure administration alternative than administration of a system by a superuser.

- Chapter 8, "Using Roles and Privileges (Overview)"
- Chapter 9, "Using Role-Based Access Control (Tasks)"
- Chapter 10, "Security Attributes in Oracle Solaris (Reference)"
Using Roles and Privileges (Overview)

The role-based access control (RBAC) feature of Oracle Solaris and the privileges feature of Oracle Solaris provide a more secure administrative alternative to superuser. This chapter provides overview information about RBAC and about privileges.

- “Role-Based Access Control (Overview)” on page 127
- “Privileges (Overview)” on page 138
- “About RBAC in This Release” on page 146

Role-Based Access Control (Overview)

Role-based access control (RBAC) is a security feature for controlling user access to tasks that would normally be restricted to the root role. By applying security attributes to processes and to users, RBAC can divide up superuser capabilities among several administrators. Process rights management is implemented through privileges. User rights management is implemented through RBAC.

- For a discussion of process rights management, see “Privileges (Overview)” on page 138.
- For information about RBAC tasks, see Chapter 9, “Using Role-Based Access Control (Tasks).”
- For reference information, see Chapter 10, “Security Attributes in Oracle Solaris (Reference).”

RBAC: An Alternative to the Superuser Model

In conventional UNIX systems, the root user, also referred to as superuser, is all-powerful. Programs that run as root, or setuid programs, are all-powerful. The root user has the ability to read and write to any file, run all programs, and send kill signals to any process. Effectively, anyone who can become superuser can modify a site’s firewall, alter the audit trail, read confidential records, and shut down the entire network. A setuid program that is hijacked can do anything on the system.
Role-based access control (RBAC) provides a more secure alternative to the all-or-nothing superuser model. With RBAC, you can enforce security policy at a more fine-grained level. RBAC uses the security principle of least privilege. Least privilege means that a user has precisely the amount of privilege that is necessary to perform a job. Regular users have enough privilege to use their applications, check the status of their jobs, print files, create new files, and so on. Capabilities beyond regular user capabilities are grouped into rights profiles. Users who are expected to do jobs that require some of the capabilities of superuser assume a role that includes the appropriate rights profile.

RBAC collects superuser capabilities into rights profiles. These rights profiles are assigned to special user accounts that are called roles. A user can then assume a role to do a job that requires some of superuser’s capabilities. Predefined rights profiles are supplied with Oracle Solaris software. You create the roles and assign the profiles.

Rights profiles can provide broad capabilities. For example, the System Administrator rights profile enables an account to perform tasks that are not related to security, such as printer management and cron jobs. Rights profiles can also be narrowly defined. For example, the Cron Management rights profile manages at and cron jobs. When you create roles, the roles can be assigned broad capabilities or narrow capabilities or both.

The following figure illustrates how RBAC can distribute rights to trusted parties.

FIGURE 8–1  RBAC Distribution of Rights

---

Oracle Solaris 11.1 Administration: Security Services • February 2014
In the RBAC model, superuser creates one or more roles. The roles are based on rights profiles. Superuser then assigns the roles to users who are trusted to perform the tasks of the role. Users log in with their user name. After login, users assume roles that can run restricted administrative commands and graphical user interface (GUI) tools.

The flexibility in setting up roles enables a variety of security policies. Although few roles are shipped with Oracle Solaris, a variety of roles can easily be configured. You can base most roles on rights profiles of the same name:

- **root** – A powerful role that is equivalent to the root user. However, this root cannot log in. A regular user must log in, then assume the assigned root role. This role is configured by default.
- **System Administrator** – A less powerful role for administration that is not related to security. This role can manage file systems, mail, and software installation. However, this role cannot set passwords.
- **Operator** – A junior administrator role for operations such as backups and printer management.

**Note** – The Media Backup rights profile provides access to the entire root file system. Therefore, while the Media Backup and Operator rights profiles are designed for a junior administrator, you must ensure that the user can be trusted.

You might also want to configure one or more security roles. Three rights profiles and their supplementary profiles handle security: Information Security, User Security, and Zone Security. Network security is a supplementary profile in the Information Security rights profile.

These roles do not have to be implemented. Roles are a function of an organization’s security needs. One strategy is to set up roles for special-purpose administrators in areas such as security, networking, or firewall administration. Another strategy is to create a single powerful administrator role along with an advanced user role. The advanced user role would be for users who are permitted to fix portions of their own systems.

The superuser model and the RBAC model can co-exist. The following table summarizes the gradations from superuser to restricted regular user that are possible in the RBAC model. The table includes the administrative actions that can be tracked in both models. For a summary of the effect of privileges alone on a system, see Table 8–2.

**TABLE 8–1** Superuser Model Contrasted With the RBAC With Privileges Model

<table>
<thead>
<tr>
<th>User Capabilities on a System</th>
<th>Superuser Model</th>
<th>RBAC Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Can become superuser with full superuser capability</td>
<td>Can</td>
<td>Can</td>
</tr>
</tbody>
</table>
TABLE 8–1 Superuser Model Contrasted With the RBAC With Privileges Model (Continued)

<table>
<thead>
<tr>
<th>User Capabilities on a System</th>
<th>Superuser Model</th>
<th>RBAC Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Can log in as a user with full user capabilities</td>
<td>Can</td>
<td>Can</td>
</tr>
<tr>
<td>Can become superuser with limited capabilities</td>
<td>Cannot</td>
<td>Can</td>
</tr>
<tr>
<td>Can log in as a user, and have superuser capabilities, sporadically</td>
<td>Can, with setuid programs only</td>
<td>Can, with setuid programs and with RBAC</td>
</tr>
<tr>
<td>Can log in as a user with administrative capabilities, but without full superuser capability</td>
<td>Cannot</td>
<td>Can, with RBAC and with directly-assigned privileges and authorizations</td>
</tr>
<tr>
<td>Can log in as a user with fewer capabilities than a regular user</td>
<td>Cannot</td>
<td>Can, with RBAC and with removed privileges</td>
</tr>
<tr>
<td>Can track superuser actions</td>
<td>Can, by auditing the su command</td>
<td>Can, by auditing calls to pfexec() Also, the name of the user who has assumed the root role is in the audit trail</td>
</tr>
</tbody>
</table>

RBAC Elements and Basic Concepts

The RBAC model in Oracle Solaris introduces the following elements:

- **Authorization** – A permission that enables a user or role to perform a class of actions that require additional rights. For example, security policy at installation gives regular users the `solaris.device.cdrw` authorization. This authorization enables users to read and write to a CD-ROM device. For a list of authorizations, see the `/etc/security/auth_attr` file.

- **Privilege** – A discrete right that can be granted to a command, a user, a role, or a system. Privileges enable a process to succeed. For example, the `proc_exec` privilege allows a process to call `execve()`. Regular users have basic privileges. To see your basic privileges, run the `ppriv -v1` basic command. For more command options, see "Administrative Commands for Handling Privileges" on page 204.

- **Security attributes** – An attribute that enables a process to perform an operation. In a typical UNIX environment, a security attribute enables a process to perform an operation that is otherwise forbidden to regular users. For example, `setuid` and `setgid` programs have security attributes. In the RBAC model, authorizations and privileges are security attributes in addition to `setuid` and `setgid` programs. These attributes can be assigned to a user. For example, a user with the `solaris.device.allocate` authorization can allocate a device for exclusive use. Privileges can be placed on a process. For example, a process with the `file_flag_set` privilege can set immutable, no-unlink, or append-only file attributes.
- **Privileged application** – An application or command that can override system controls by checking for security attributes. In a typical UNIX environment and in the RBAC model, programs that use `setuid` and `setgid` are privileged applications. In the RBAC model, programs that require privileges or authorizations to succeed are also privileged applications. For more information, see “Privileged Applications and RBAC” on page 134.

- **Rights profile** – A collection of security attributes that can be assigned to a role or to a user. A rights profile can include authorizations, directly assigned privileges, commands with security attributes, and other rights profiles. Profiles that are within another profile are called supplementary rights profiles. Rights profiles offer a convenient way to group security attributes.

- **Role** – A special identity for running privileged applications. The special identity can be assumed by assigned users only. In a system that is run by roles, including the root role, superuser is unnecessary. Superuser capabilities are distributed to different roles. For example, in a two-role system, security tasks would be handled by a security role. The second role would handle system administration tasks that are not security-related. Roles can be more fine-grained. For example, a system could include separate administrative roles for handling the Cryptographic Framework, printers, system time, file systems, and auditing.

The following figure shows how the RBAC elements work together.

**FIGURE 8-2** RBAC Element Relationships

The following figure uses the Network Security role and the Network Security rights profile to demonstrate RBAC relationships.
The Network Security role is used to manage IPsec, wifi, and network links. The role is assigned to the user jdoe. jdoe can assume the role by switching to the role, and then supplying the role password. The administrator can customize the role to accept the user password rather than the role password.

In Figure 8–3, the Network Security rights profile is assigned to the Network Security role. The Network Security rights profile contains supplementary profiles that are evaluated in order, Network Wifi Security, Network Link Security, and Network IPsec Management. These supplementary profiles fill out the role’s primary tasks.

The Network Security rights profile has three directly assigned authorizations, no directly assigned privileges, and two commands with security attributes. The supplementary rights profiles have directly assigned authorizations, and two of them have commands with security attributes. In the Network Security role, jdoe has all assigned authorizations in these profiles, and can run all the commands with security attributes in these profiles. jdoe can administer network security.
Privilege Escalation

Oracle Solaris provides administrators with a great deal of flexibility when configuring security. As installed, the software does not allow for privilege escalation. Privilege escalation occurs when a user or process gains more administrative rights than were intended to be granted. In this sense, privilege means any security attribute, not just privileges.

Oracle Solaris software includes security attributes that are assigned to the root role only. With other security protections in place, an administrator might assign attributes that are designed for the root role to other accounts, but such assignment must be made with care.

The following rights profile and set of authorizations can escalate the privileges of a non-root account.

- **Media Restore rights profile** – This profile exists, but is not part of any other rights profile. Because Media Restore provides access to the entire root file system, its use is a possible escalation of privilege. Deliberately altered files or substitute media could be restored. By default, the root role includes this rights profile.

- **solaris.*.assign authorizations** – These authorizations exist, but are not assigned to any rights profile or account. An account with a solaris.*.assign authorization can assign security attributes to others that the account itself is not assigned. For example, a role with the solaris.profile.assign authorization can assign rights profiles to other accounts that the role itself is not assigned. By default, only the root role has solaris.*.assign authorizations.

  Best practice is to assign solaris.*.delegate authorizations, not solaris.*.assign authorizations. A solaris.*.delegate authorization enables the delegator to assign other accounts only those security attributes that the delegator possesses. For example, a role that is assigned the solaris.profile.delegate authorization can assign rights profiles that the role itself is assigned to other users and roles.

For escalations that affect the privilege security attribute, see “Prevention of Privilege Escalation” on page 206.

RBAC Authorizations

An authorization is a discrete right that can be granted to a role or to a user. Authorizations enforce policy at the user application level.

While authorizations can be assigned directly to a role or to a user, best practice is to include authorizations in a rights profile. The rights profile is then added to a role, and the role is assigned to a user. For an example, see Figure 8–3.

Authorizations that include the words delegate or assign enable the user or role to assign security attributes to others.
To prevent escalation of privilege, do not assign an account an **assign** authorization.

- A **delegate** authorization enables the delegator to assign others only those security attributes that the delegator possesses. For example, a role that is assigned the `solaris.profile.delegate` authorization can assign to others rights profiles that the role itself is assigned.

- An **assign** authorization enables the assigner to assign others security attributes that the account does not possess. For example, a role with the `solaris.profile.assign` authorization can assign to others any rights profile.

The `solaris.*.assign` authorizations are delivered, but are not included in any profile. By default, only the root role has the `solaris.*.assign` authorizations.

RBAC-compliant applications can check a user’s authorizations prior to granting access to the application or specific operations within the application. This check replaces the check in conventional UNIX applications for `UID=0`. For more information about authorizations, see the following sections:

- “Authorizations” on page 198
- “auth_attr Database” on page 200
- “Selected Commands That Require Authorizations” on page 203

### Authorizations and Privileges

Privileges enforce security policy in the kernel. The difference between authorizations and privileges concerns the level at which the security policy is enforced. Without the proper privilege, a process can be prevented from performing privileged operations by the kernel. Without the proper authorizations, a user might be prevented from using a privileged application or from performing security-sensitive operations within a privileged application. For a fuller discussion of privileges, see “Privileges (Overview)” on page 138.

### Privileged Applications and RBAC

Applications and commands that can override system controls are considered privileged applications. Security attributes such as `UID=0`, privileges, and authorizations make an application privileged.

**Applications That Check UIDs and GIDs**

Privileged applications that check for `root (UID=0)` or some other special UID or GID have long existed in the UNIX environment. The rights profile mechanism enables you to isolate commands that require a specific ID. Instead of changing the ID on a command that anyone can access, you can place the command with assigned security attributes in a rights profile. A user or role with that rights profile can then run the program without having to become superuser.
IDs can be specified as real or effective. Assigning effective IDs is preferred over assigning real IDs. Effective IDs are equivalent to the setuid feature in the file permission bits. Effective IDs also identify the UID for auditing. However, because some shell scripts and programs require a real UID of root, real UIDs can be set as well. For example, the reboot command requires a real rather than an effective UID. If an effective ID is not sufficient to run a command, you need to assign the real ID to the command.

**Applications That Check for Privileges**

Privileged applications can check for the use of privileges. The RBAC rights profile mechanism enables you to specify the privileges for specific commands that require security attributes. Then, you can isolate the command with assigned security attributes in a rights profile. A user or role with that rights profile can then run the command with just the privileges that the command requires to succeed.

Commands that check for privileges include the following:

- Kerberos commands, such as kadmin, kprop, and kdb5_util
- Network commands, such as ipadm, routeadm, and snoop
- File and file system commands, such as chmod, chgrp, and mount
- Commands that control processes, such as kill, pcred, and rcapadm

To add commands with privileges to a rights profile, see “How to Create a Rights Profile” on page 167 and the profiles(1) man page. To determine which commands check for privileges in a particular profile, see “How to View All Defined Security Attributes” on page 150.

**Applications That Check Authorizations**

Oracle Solaris additionally provides commands that check authorizations. By definition, the root user has all authorizations. Therefore, the root user can run any application. Applications that check for authorizations include the following:

- Audit administration commands, such as auditconfig and auditreduce
- Printer administration commands, such as lpadm and lpfiler
- The batch job-related commands, such as at, atq, batch, and crontab
- Device-oriented commands, such as allocate, deallocate, list_devices, and cdrw

To test a script or program for authorizations, see Example 9-23. To write a program that requires authorizations, see "About Authorizations" in Developer’s Guide to Oracle Solaris 11 Security.

**RBAC Rights Profiles**

A rights profile is a collection of security attributes that can be assigned to a role or user to perform tasks that require administrative rights. A rights profile can include authorizations,
privileges, commands with assigned security attributes, and other rights profiles. Privileges that are assigned in a rights profile are in effect for all commands. Rights profiles also contain entries to reduce or extend the initial inheritable set, and to reduce the limit set of privileges.

For more information about rights profiles, see the following sections:
- “Rights Profiles” on page 195
- “prof_attr Database” on page 201
- “exec_attr Database” on page 201

RBAC Roles

A role is a special type of user account from which you can run privileged applications. Roles are created in the same general manner as user accounts. Roles have a home directory, a group assignment, a password, and so on. Rights profiles and authorizations give the role administrative capabilities. Roles cannot inherit capabilities from other roles or other users. Discrete roles parcel out superuser capabilities, and thus enable more secure administrative practices.

When a user assumes a role, the role’s attributes replace all user attributes. Role information is stored in the passwd, shadow, and user_attr databases. The actions of roles can be audited. For detailed information about setting up roles, see the following sections:
- “How to Plan Your RBAC Implementation” on page 161
- “How to Create a Role” on page 163
- “How to Change the Security Attributes of a Role” on page 179

A role can be assigned to more than one user. All users who can assume the same role have the same role home directory, operate in the same environment, and have access to the same files. Users can assume roles from the command line by running the su command and supplying the role name and a password. By default, users authenticate to a role by supplying the role’s password. The administrator can configure the system to enable a user to authenticate by supplying the user’s password. For the procedure, see “How to Enable a User to Use Own Password to Assume a Role” on page 182.

A role cannot log in directly. A user logs in, and then assumes a role. Having assumed a role, the user cannot assume another role without first exiting their current role. Having exited the role, the user can then assume another role.

The fact that root is a role in Oracle Solaris prevents anonymous root login. If the profile shell command, pexec, is being audited, the audit trail contains the login user’s real UID, the roles that the user has assumed, and the actions that the role performed. To audit the system or a particular user for role operations, see “How to Audit Roles” on page 166.

The rights profiles that ship with the software are designed to map to roles. For example, the System Administrator rights profile can be used to create the System Administrator role. To configure a role, see “How to Create a Role” on page 163.
Profile Shells and RBAC

Users and roles can run privileged applications from a profile shell. A profile shell is a special shell that recognizes the security attributes that are included in a rights profile. Administrators can assign a profile shell to a specific user as a login shell, or the profile shell is started when that user runs the su command to assume a role. In Oracle Solaris every shell has a profile shell counterpart. For example, the profile shell counterparts to the Bourne shell (sh), bash shell (bash), and Korn shell (ksh) are the pfsh, pfbash, and pfksh shells, respectively. For the list of profile shells, see the pfexec(1) man page.

Users who have been directly assigned a rights profile and whose login shell is not a profile shell must invoke a profile shell to run the commands with security attributes. For usability and security considerations, see "Security Considerations When Directly Assigning Security Attributes" on page 137.

All commands that are executed in a profile shell can be audited. For more information, see "How to Audit Roles" on page 166.

Name Service Scope and RBAC

Name service scope is an important concept for understanding RBAC. The scope of a role might be limited to an individual host. Alternatively, the scope might include all hosts that are served by a naming service such as LDAP. The name service scope for a system is specified in the name switch service, svc:/system/name-service/switch. A lookup stops at the first match. For example, if a rights profile exists in two name service scopes, only the entries in the first name service scope are used. If files is the first match, then the scope of the role is limited to the local host.

Security Considerations When Directly Assigning Security Attributes

Typically, a user obtains administrative capabilities through a role. Authorizations, privileges, and privileged commands are grouped into a rights profile. The rights profile is included in a role, and the role is assigned to a user.

Direct assignment of rights profiles and security attributes is also possible:

- Rights profiles, privileges, and authorizations can be assigned directly to users.
- Privileges and authorizations can be assigned directly to users and roles.

However, direct assignment of privileges is not a secure practice. Users and roles with a directly assigned privilege could override security policy wherever this privilege is required by the...
kernel. A more secure practice is to assign the privilege as a security attribute of a command in a rights profile. Then, that privilege is available only for that command by someone who has that rights profile.

Since authorizations act at the user level, direct assignment of authorizations can be less dangerous than direct assignment of privileges. However, authorizations can enable a user to perform highly secure tasks, such as assign audit flags.

**Usability Considerations When Directly Assigning Security Attributes**

Direct assignment of rights profiles and security attributes can affect usability:

- Directly assigned privileges and authorizations, and the commands and authorizations in a directly assigned rights profile, must be interpreted by a profile shell to be effective. By default, users are not assigned a profile shell.
  The user must remember to open a profile shell and execute the commands in that shell.
- Singly assigning authorizations is not scalable. And, directly assigned authorizations might not be sufficient to perform a task. The task might require privileged commands.
  Rights profiles are designed to bundle authorizations and privileged commands together. They are also scalable.

**Privileges (Overview)**

Process rights management enables processes to be restricted at the command, user, role, or system level. Oracle Solaris implements process rights management through *privileges*. Privileges decrease the security risk that is associated with one user or one process having full superuser capabilities on a system. Privileges and RBAC provide a compelling alternative model to the traditional superuser model.

- For information about RBAC, see “Role-Based Access Control (Overview)” on page 127.
- For information about how to administer privileges, see “Using Privileges (Tasks)” on page 185.
- For reference information about privileges, see “Privileges” on page 204.

**Privileges Protect Kernel Processes**

A privilege is a discrete right that a process requires to perform an operation. The right is enforced in the kernel. A program that operates within the bounds of the basic set of privileges operates within the bounds of the system security policy. *setuid* programs are examples of programs that operate outside the bounds of the system security policy. By using privileges, programs eliminate the need for calls to *setuid*. 
Privileges discretely enumerate the kinds of operations that are possible on a system. Programs can be run with the exact privileges that enable the program to succeed. For example, a program that manipulates files might require the `file_dac_write` and `file_flag_set` privileges. These privileges on the process eliminates the need to run the program as `root`.

Historically, systems have not followed the privilege model. Rather, systems used the superuser model. In the superuser model, processes run as `root` or as a user. User processes were limited to acting on the user’s directories and files. `root` processes could create directories and files anywhere on the system. A process that required creation of a directory outside the user’s directory would run with a `UID=0`, that is, as `root`. Security policy relied on DAC, discretionary access control, to protect system files. Device nodes were protected by DAC. For example, devices owned by group `sys` could be opened only by members of group `sys`.

However, `setuid` programs, file permissions, and administrative accounts are vulnerable to misuse. The actions that a `setuid` process is permitted are more numerous than the process requires to complete its operation. A `setuid` program can be compromised by an intruder who then runs as the all-powerful `root` user. Similarly, any user with access to the `root` password can compromise the entire system.

In contrast, a system that enforces policy with privileges allows a gradation between user capabilities and `root` capabilities. A user can be granted privileges to perform activities that are beyond the capabilities of regular users, and `root` can be limited to fewer privileges than `root` currently possesses. With RBAC, a command that runs with privileges can be isolated in a rights profile and assigned to one user or role. Table 8–1 summarizes the gradation between user capabilities and `root` capabilities that the RBAC plus privileges model provides.

The privilege model provides greater security than the superuser model. Privileges that have been removed from a process cannot be exploited. Process privileges prevent a program or administrative account from gaining access to all capabilities. Process privileges can provide an additional safeguard for sensitive files, where DAC protections alone can be exploited to gain access.

Privileges, then, can restrict programs and processes to just the capabilities that the program requires. This capability is called the principle of least privilege. On a system that implements least privilege, an intruder who captures a process can access only those privileges that the process has. The rest of the system cannot be compromised.

### Privilege Descriptions

Privileges are logically grouped on the basis of the area of the privilege.

- **FILE privileges** – Privileges that begin with the string `file` operate on file system objects. For example, the `file_dac_write` privilege overrides discretionary access control when writing to files.
■ **IPC privileges** – Privileges that begin with the string ipc override IPC object access controls. For example, the ipc_dac_read privilege enables a process to read remote shared memory that is protected by DAC.

■ **NET privileges** – Privileges that begin with the string net give access to specific network functionality. For example, the net_rawaccess privilege enables a device to connect to the network.

■ **PROC privileges** – Privileges that begin with the string proc allow processes to modify restricted properties of the process itself. PROC privileges include privileges that have a very limited effect. For example, the proc_clock_highres privilege enables a process to use high resolution timers.

■ **SYS privileges** – Privileges that begin with the string sys give processes unrestricted access to various system properties. For example, the sys_linkdir privilege enables a process to make and break hard links to directories.

Other logical groups include CONTRACT, CPC, DTRACE, GRAPHICS, VIRT, and WIN.

Some privileges have a limited effect on the system, and some have a broad effect. The definition of the proc_taskid privilege indicates its limited effect:

```
proc_taskid
   Allows a process to assign a new task ID to the calling process.
```

The definition of the net_rawaccess privilege indicates its broad effect:

```
net_rawaccess
   Allow a process to have direct access to the network layer.
```

The `privileges(5)` man page provides descriptions of every privilege. The command `ppriv -lv` prints a description of every privilege to standard out.

**Administrative Differences on a System With Privileges**

A system that has privileges has several visible differences from a system that does not have privileges. The following table lists some of the differences.

<table>
<thead>
<tr>
<th>Feature</th>
<th>No Privileges</th>
<th>Privileges</th>
</tr>
</thead>
<tbody>
<tr>
<td>Daemons</td>
<td>Daemons run as root.</td>
<td>Daemons run as the user daemon.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For example, these daemons have been assigned appropriate privileges and run as daemon: lockd and rpcbind.</td>
</tr>
</tbody>
</table>
TABLE 8–2  Visible Differences Between a System With Privileges and a System Without Privileges  (Continued)

<table>
<thead>
<tr>
<th>Feature</th>
<th>No Privileges</th>
<th>Privileges</th>
</tr>
</thead>
<tbody>
<tr>
<td>Log File Ownership</td>
<td>Log files are owned by root.</td>
<td>Log files are now owned by daemon, who created the log file. The root user does not own the file.</td>
</tr>
<tr>
<td>Error Messages</td>
<td>Error messages refer to superuser.</td>
<td>Error messages reflect the use of privileges.</td>
</tr>
<tr>
<td></td>
<td>For example, chroot: not superuser.</td>
<td>For example, the equivalent error message for chroot failure is chroot: exec failed.</td>
</tr>
<tr>
<td>setuid Programs</td>
<td>Programs use setuid to complete tasks that regular users are not allowed to perform.</td>
<td>Many setuid programs have been changed to run with privileges.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For example, the following commands use privileges: audit, ikeadm, ipadm, ipsecconf, ping, traceroute, and newtask.</td>
</tr>
<tr>
<td>File Permissions</td>
<td>Device permissions are controlled by DAC.</td>
<td>File permissions (DAC) do not predict who can open a device.</td>
</tr>
<tr>
<td></td>
<td>For example, members of the group sys can open /dev/ip.</td>
<td>Devices are protected with DAC and device policy.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For example, the /dev/ip file has 666 permissions, but the device can only be opened by a process with the appropriate privileges. Raw sockets are still protected by DAC.</td>
</tr>
<tr>
<td>Audit Events</td>
<td>Auditing the use of the su command covers many administrative functions.</td>
<td>Auditing the use of privileges covers most administrative functions.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The pm, ps, ex, ua and as audit classes include audit events that monitor device policy and use of privilege.</td>
</tr>
<tr>
<td>Processes</td>
<td>Processes are protected by who owns the process.</td>
<td>Processes are protected by privileges. Process privileges and process flags are visible as a new entry in the /proc/&lt;pid&gt; directory, priv.</td>
</tr>
<tr>
<td>Debugging</td>
<td>No reference to privileges in core dumps.</td>
<td>The ELF note section of core dumps includes information about process privileges and flags in the NT_PRPRIV and NT_PRPRIVINFO notes.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The ppriv command and other commands show the proper number of properly sized sets. The commands correctly map the bits in the bit sets to privilege names.</td>
</tr>
</tbody>
</table>

Privileges and System Resources

In Oracle Solaris, the project.max-locked-memory and zone.max-locked-memory resource controls can be used to limit the memory consumption of processes that are assigned the PRIV_PROC_LOCK_MEMORY privilege. This privilege allows a process to lock pages in physical memory.

If you assign the PRIV_PROC_LOCK_MEMORY privilege to a rights profile, you can give the processes that have this privilege the ability to lock all memory. As a safeguard, set a resource control to prevent the user of the privilege from locking all memory. For privileged processes that run in a non-global zone, set the zone.max-locked-memory resource control. For privileged processes that run on a system, create a project and set the project.max-locked-memory resource control. For information about these resource controls,
How Privileges Are Implemented

Every process has four sets of privileges that determine whether a process can use a particular privilege. The kernel automatically calculates the effective set of privileges. You can modify the initial inheritable set of privileges. A program that is coded to use privileges can reduce the program's permitted set of privileges. You can shrink the limit set of privileges.

- **Effective privilege set, or E** – Is the set of privileges that is currently in effect. A process can add privileges that are in the permitted set to the effective set. A process can also remove privileges from E.

- **Permitted privilege set, or P** – Is the set of privileges that is available for use. Privileges can be available to a program from inheritance or through assignment. An execution profile is one way to assign privileges to a program. The setuid command assigns all privileges that root has to a program. Privileges can be removed from the permitted set, but privileges cannot be added to the set. Privileges that are removed from P are automatically removed from E.

  A privilege-aware program removes the privileges that a program never uses from the program's permitted set. In this way, unnecessary privileges cannot be exploited by the program or a malicious process. For more information about privilege-aware programs, see Chapter 2, “Developing Privileged Applications,” in Developer's Guide to Oracle Solaris 11 Security.

- **Inheritable privilege set, or I** – Is the set of privileges that a process can inherit across a call to exec. After the call to exec, the inherited privileges are placed in the permitted and the effective sets, thus making these sets equal, except in the special case of a setuid program.

  For a setuid program, after the call to exec, the inheritable set is first restricted by the limit set. Then, the set of privileges that were inherited (I), minus any privileges that were in the limit set (L), are assigned to P and E for that process.

- **Limit privilege set, or L** – Is the outside limit of what privileges are available to a process and its children. By default, the limit set is all privileges. Processes can shrink the limit set but can never extend the limit set. L is used to restrict I. Consequently, L restricts P and E at the time of exec.

  If a user has been assigned a profile that includes a program that has been assigned privileges, the user can usually run that program. On an unmodified system, the program's assigned privileges are within the user's limit set. The privileges that have been assigned to the program become part of the user's permitted set. To run the program that has been assigned privileges, the user must run the program from a profile shell.
The kernel recognizes a basic privilege set. On an unmodified system, each user’s initial inheritable set equals the basic set at login. While you cannot modify the basic set, you can modify which privileges a user inherits from the basic set.

On an unmodified system, a user’s privilege sets at login would appear similar to the following:

**E (Effective):** basic
**I (Inheritable):** basic
**P (Permitted):** basic
**L (Limit):** all

Therefore, at login, all users have the basic set in their inheritable set, their permitted set, and their effective set. A user’s limit set is equivalent to the default limit set for the zone, global or non-global. To put more privileges in the user’s effective set, you must assign a rights profile to the user. The rights profile would include commands to which you have added privileges. You can also assign privileges directly to the user or role, though such privilege assignment can be risky. For a discussion of the risks, see “Security Considerations When Directly Assigning Security Attributes” on page 137.

### How Processes Get Privileges

Processes can inherit privileges. Or, processes can be assigned privileges. A process inherits privileges from its parent process. At login, the user’s initial inheritable set of privileges determines what privileges are available to the user’s processes. All child processes of the user’s initial login inherit that set.

You can also directly assign privileges to programs, users, and roles. When a program requires privileges, you assign the privileges to the program’s executable in a rights profile. Users or roles that are permitted to run the program are assigned the profile that includes the program. At login or when a profile shell is entered, the program runs with privilege when the program’s executable is typed in the profile shell. For example, a role that includes the Object Access Management profile is able to run the `chmod` command with the `file_chown` privilege.

When a role or user runs a program that has been directly assigned an additional privilege, the assigned privilege is added to the role or user’s inheritable set. Child processes of the program that was assigned privileges inherit the privileges of the parent. If the child process requires more privileges than the parent process, the child process must be directly assigned those privileges.

Programs that are coded to use privileges are called privilege-aware programs. A privilege-aware program turns on the use of privilege and turns off the use of privilege during program execution. To succeed in a production environment, the program must be assigned the privileges that the program turns on and off.

For examples of privilege-aware code, see Chapter 2, “Developing Privileged Applications,” in *Developer’s Guide to Oracle Solaris 11 Security*. To assign privileges to a program that requires privileges, see Example 9–18.
Assigning Privileges

You, in your capacity as security administrator, are responsible for assigning privileges. Best practice is to assign the privilege to a command in a rights profile. The rights profile is then assigned to a role or to a user.

Privileges can also be assigned directly to a user, a role, or a rights profile. If you trust a subset of users to use a privilege responsibly throughout their sessions, you can assign the privilege directly. Good candidates for direct assignment are privileges that have a limited effect, such as `proc_clock_highres`. Poor candidates for direct assignment are privileges that have far-reaching effects, such as `file_dac_write`.

Privileges can also be denied to a user or to a system. Care must be taken when removing privileges from the initial inheritable set or the limit set of a user or a system.

Expanding a User or Role's Privileges

Users and roles have an inheritable set of privileges. The limit set cannot be expanded, since the limit set is initially all privileges. The initial inheritable set can be expanded for users, roles, and systems. A privilege that is not in the inheritable set can also be assigned to a process.

You can expand the privileges that are available in three ways.

- The initial inheritable set can be expanded for users, roles, and systems.
- A privilege that is not in the inheritable set can be explicitly assigned to a process.
- A privilege that is not in the inheritable set can be explicitly assigned to a network port, UID, or file object. This use of privilege is called extended policy.

The assignment of privileges per process is the most precise way to add privileges. You can expand the number of privileged operations that a user can perform by assigning the user a role. The role would be assigned rights profiles that include commands with added privileges. When the user assumes the role, the user gets the role’s profile shell. When commands from the rights profile are typed in the role’s shell, the commands execute with the added privileges.

You can also assign a rights profile to the user rather than to a role that the user assumes. When the user opens a profile shell, such as `pfksh`, the user can execute the commands in the user’s rights profile with privilege. In a regular shell, the commands do not execute with privilege. The privileged process can only execute in a privileged shell.

To expand the initial inheritable set of privileges for users, roles, or systems is a riskier way to assign privileges. All privileges in the inheritable set are in the permitted and effective sets. All commands that the user or role types in a shell can use the directly assigned privileges. Directly assigned privileges enable a user or role to easily perform operations that can be outside the bounds of their administrative responsibilities.
To mitigate this risk, you can assign privileges to users or roles so that they can operate on an object at a time. Possible objects are network ports, UIDs, and file objects. For example, you could assign to a user the file_dac_read privilege for files in the /var/core directory. The file_dac_read privilege must be in the effective set of the user’s process when this extended policy is applied. If so, the user can read all files in the /var/core directory when the extended policy is assigned to this user for this directory object.

When you add to the initial inheritable set of privileges on a system, all users who log on to the system have a larger set of basic privileges. Such direct assignment enables all users of the system to easily perform operations that are probably outside the bounds of regular users.

**Note** – The limit set cannot be expanded, because the limit set is initially all privileges.

### Restricting a User or Role's Privileges

By removing privileges, you can prevent users and roles from performing particular tasks. You can remove privileges from the initial inheritable set, and from the limit set. You should carefully test removal of privileges before you distribute an initial inheritable set or a limit set that is smaller than the default set. By removing privileges from the initial inheritable set, you might prevent users from logging in. When privileges are removed from the limit set, a legacy setuid program might fail because the program requires a privilege that was removed.

### Assigning Privileges to a Script

Scripts are executables, like commands. Therefore, in a rights profile, you can add privileges to a script just as you can add privileges to a command. The script runs with the added privileges when a user or role who has been assigned the rights profile executes the script in a profile shell. If the script contains commands that require privileges, the commands with added privileges must also be in an assigned rights profile.

Privilege-aware programs can restrict privileges per process. Your job with a privilege-aware program is to assign the executable just the privileges that the program needs. You then test the program to see that the program succeeds in performing its tasks. You also check that the program does not abuse its use of privileges.

### Privileges and Devices

The privilege model uses privileges to protect system interfaces that, in the superuser model, are protected by file permissions alone. In a system with privileges, file permissions are too weak to protect the interfaces. A privilege such as proc_owner could override file permissions and then give full access to all of the system.
Therefore, in Oracle Solaris, ownership of the device directory is not sufficient to open a device. For example, members of the group sys are no longer automatically allowed to open the /dev/ip device. The file permissions on /dev/ip are 0666, but the net_rawaccess privilege is required to open the device.

Device policy is controlled by privileges. The getdevpolicy command displays the device policy for every device. The device configuration command, devfsadm, installs the device policy. The devfsadm command binds privilege sets with open for reading or writing of devices. For more information, see the getdevpolicy(1M) and devfsadm(1M) man pages.

Device policy allows you more flexibility in granting permission to open devices. You can require different privileges or more privileges than the default device policy. The privilege requirements can be modified for the device policy and for the driver proper. You can modify the privileges when installing, adding, or updating a device driver.

The add_drv and update_drv commands are used to modify device policy entries and driver-specific privileges. You must be running a process that has the full set of privileges to change the device policy. For more information, see the add_drv(1M) and update_drv(1M) man pages.

**Privileges and Debugging**

Oracle Solaris provides tools to debug privilege failure. The ppriv command and the truss command provide debugging output. For examples, see the ppriv(1) man page. For a procedure, see “How to Determine Which Privileges a Program Requires” on page 191. You can also use the dtrace command. For more information, see the dtrace(1M) man page.

**About RBAC in This Release**

The RBAC features include the following:

- The pfedit command enables a non-root user or role to edit specified system files. The user or role must be assigned the solaris.admin.edit/path-to-system-file authorization. This command can be used by the root role to ensure that root actions are placed in the audit record. For more information, see the pfedit(1M) man page.

- Extended privilege policy enables specific privileges to be applied to specific filenames, port numbers, and user IDs. For more information, see the ppriv(1) and privileges(5) man pages. For an example of applying extended privilege policy to a port number, see “How to Apply Extended Privilege Policy to a Port” on page 192.

- The pam_policy security attribute enables an administrator to configure PAM policy at the system, rights profile, user, and module levels. For more information, see “Changes to PAM for This Release” on page 267 and “How to Assign a Customized PAM Policy to a User” on page 271.
The auths command is extended similar to the profiles command. Authorizations can be managed from the command line for the files and LDAP repositories. For more information, see “How to Create an Authorization” on page 171 and the auths(1) man page.

A User Manager GUI is available to manage users and roles. For more information, see Chapter 3, "Managing User Accounts by Using the User Manager GUI (Tasks)," in Managing User Accounts and User Environments in Oracle Solaris 11.1.
This chapter covers tasks for distributing the capabilities of superuser by using discrete roles. The mechanisms that roles can use include rights profiles, authorizations, and privileges. The chapter covers the following topics:

- “Using RBAC (Tasks)” on page 149
- “Using Privileges (Tasks)” on page 185

For an overview of RBAC, see “Role-Based Access Control (Overview)” on page 127. For reference information, see Chapter 10, “Security Attributes in Oracle Solaris (Reference).” To use privileges, see “Using Privileges (Tasks)” on page 185.

### Using RBAC (Tasks)

To use RBAC requires planning, configuring RBAC, and knowing how to assume a role. Once roles become familiar, you might further customize RBAC to handle new operations. The following task map points to these major tasks, including the use of privilege.

<table>
<thead>
<tr>
<th>Task</th>
<th>Description</th>
<th>For Instructions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use the default RBAC configuration.</td>
<td>Views and uses RBAC without modifying the initial installation.</td>
<td>“Viewing and Using RBAC Defaults (Task Map)” on page 150</td>
</tr>
<tr>
<td>Plan, configure, and use RBAC.</td>
<td>Configures RBAC for your site.</td>
<td>“Initially Configuring RBAC (Task Map)” on page 160</td>
</tr>
<tr>
<td>Administer RBAC.</td>
<td>Updates your site’s RBAC configuration.</td>
<td>“Managing RBAC (Task Map)” on page 177</td>
</tr>
</tbody>
</table>
Users are assigned rights by default. Rights for all users of a system are assigned in the /etc/security/policy.conf file.

### Viewing and Using RBAC Defaults (Task Map)

At Oracle Solaris installation, your system is configured with user rights and process rights. With no further configuration, use the following task map to view and use RBAC.

<table>
<thead>
<tr>
<th>Task</th>
<th>Description</th>
<th>For Instructions</th>
</tr>
</thead>
<tbody>
<tr>
<td>View the contents of the security attributes databases.</td>
<td>List all the authorizations, rights profiles, and commands with security attributes on the system.</td>
<td>“How to View All Defined Security Attributes” on page 150</td>
</tr>
<tr>
<td>View your rights.</td>
<td>Involves listing your rights profiles, authorizations, privileges, and assigned roles.</td>
<td>“How to View Your Assigned Rights” on page 151</td>
</tr>
<tr>
<td>Assume the root role.</td>
<td>The initial user gains administrative rights.</td>
<td>“How to Assume a Role” on page 154</td>
</tr>
<tr>
<td>Modify the rights of a user.</td>
<td>Adds security attributes to a regular user or removes them.</td>
<td>“How to Change the Security Attributes of a User” on page 155</td>
</tr>
<tr>
<td>Become an administrator.</td>
<td>Several methods are available to users who are assigned administrative rights to use those rights.</td>
<td>“How to Use Your Assigned Administrative Rights” on page 157</td>
</tr>
</tbody>
</table>

▼ **How to View All Defined Security Attributes**

Use the following commands to list all authorizations, rights profiles, and commands with security attributes on the system. To list all defined privileges, see “How to List the Privileges on the System” on page 186.

1. List all authorizations.
   - List the names of all authorizations in the naming service.
     ```
     % auths info
     solaris.account.activate
     solaris.account.setpolicy
     solaris.admin.edit
     ...
     solaris.zone.login
     solaris.zone.manage
     ```
   - List authorization names per rights profile.
     ```
     % getent auth_attr | more
     solaris.:::All Solaris Authorizations::help=AllSolAuthsHeader.html
     solaris.account.:::Account Management::help=AccountHeader.html
     ```
2 List all rights profiles.

- List the names of all rights profiles in the naming service.
  
  ```
  % profiles -a
  ```

  - Console User
  - CUPS Administration
  - Desktop Removable Media User
  - VSCAN Management
  - WUSB Management

- List the full definitions of all rights profiles.

  ```
  % getent prof_attr | more
  ```

  - All:::Execute any command as the user or role:help=RtAll.html
  - Audit Configuration:::Configure Solaris Audit:auths=solaris.smf.value.audit;help=RtAuditCfg.html
  - Zone Management:::Zones Virtual Application Environment Administration:help=RtZoneMngmnt.html
  - Zone Security:::Zones Virtual Application Environment Security:auths=solaris.zone.*,solaris.auth.delegation;help=RtZoneSecurity.html ...

3 List all commands with security attributes.

  ```
  % getent exec_attr | more
  ```

  - All:solaris:cmd:::*:
  - Audit Configuration:solaris:cmd:::/usr/sbin/auditconfig:privs=sys_audit
  - Zone Security:solaris:cmd:::/usr/sbin/txzonemgr:uid=0
  - Zone Security:solaris:cmd:::/usr/sbin/zonecfg:uid=0 ...

### How to View Your Assigned Rights

Use the following commands to view your RBAC assignments. To view all rights that can be assigned, see “How to View All Defined Security Attributes” on page 150.

1 List your rights profiles.

  ```
  % profiles
  ```

  Basic Solaris User
  All

  The preceding rights profiles are assigned to all users by default. If you are the initial user, you have a longer list.

  ```
  % profiles Initial user
  ```

  System Administrator
  Audit Review
CPU Power Management
Basic Solaris User
All

2 List your authorizations.

```bash
% auths
solaris.device.cdrw,solaris.device.mount.removable,solaris.mail.mailq
solaris.network.autoconf.read,solaris.admin.wusb.read
solaris.smf.manage.vbiosd,solaris.smf.value.vbiosd
```

These authorizations are included in the rights profiles that are assigned to all users by default.

3 List your assigned roles.

```bash
% roles
root
```

This role is assigned to the initial user by default. No roles indicates that you are not assigned a role.

4 List the privileges in your default shell.

```bash
% ppriv $$
1234: /bin/csh
flags = <none>
E: basic
I: basic
P: basic
L: all
```

Every user is assigned the basic privilege set by default. The default limit set is all privileges.

```bash
% ppriv -vl basic
file_link_any
    Allows a process to create hardlinks to files owned by a uid
        different from the process’ effective uid.
file_read
    Allows a process to read objects in the filesystem.
file_write
    Allows a process to modify objects in the filesystem.
net_access
    Allows a process to open a TCP, UDP, SDP or SCTP network endpoint.
proc_exec
    Allows a process to call execve().
proc_fork
    Allows a process to call fork1()/forkall()/vfork()
proc_info
    Allows a process to examine the status of processes other
        than those it can send signals to. Processes which cannot
        be examined cannot be seen in /proc and appear not to exist.
proc_session
    Allows a process to send signals or trace processes outside its session.
```
List the privileges on commands in your rights profiles.

```
% profiles -l
Basic Solaris User
...
/usr/bin/cdrecord.bin privs=file_dac_read,sys_devices,
    proc_lock_memory,proc_priocntl,net_privaddr
/usr/bin/readcd.bin privs=file_dac_read,sys_devices,net_privaddr
/usr/bin/cdda2wav.bin privs=file_dac_read,sys_devices,
    proc_priocntl,net_privaddr
All
```

A user's rights profile can include commands that run with particular privileges. The Basic Solaris User profile includes commands that enable users to read and write to CD-ROMs.

**Example 9–1** Listing a User’s Authorizations

```
% auths username
solaris.device.cdrw,solaris.device.mount.removable,solaris.mail.mailq
```

**Example 9–2** Listing a User or Role’s Rights Profiles

The following command lists the rights profiles of a specific user.

```
% profiles jdoe
jdoe:
    Basic Solaris User
    All
```

The following command lists the rights profiles of a the cryptomgt role.

```
% profiles cryptomgt
cryptomgt:
    Crypto Management
    Basic Solaris User
    All
```

The following command lists the rights profiles of the root role:

```
% profiles root
root:
    All
    Console User
    Network Wifi Info
    Desktop Removable Media User
    Suspend To RAM
    Suspend To Disk
    Brightness
    CPU Power Management
    Network Autoconf User
    Basic Solaris User
```
Example 9–3  
Listing a User’s Assigned Roles

The following command lists the assigned roles of a specific user.

```bash
% roles jdoe
root
```

Example 9–4  
Listing a User’s Privileges on Specific Commands

The following command lists the privileged commands in a regular user’s rights profiles.

```bash
% profiles -l jdoe
jdoe: Basic Solaris User
... /usr/bin/cdda2wav.bin privs=file_dac_read,sys_devices,
proc_priocntl.net_privaddr
/usr/bin/cdrecord.bin privs=file_dac_read,sys_devices,
proc_lock_memory,proc_priocntl.net_privaddr
/usr/bin/readcd.bin privs=file_dac_read,sys_devices,net_privaddr
...
```

▼ How to Assume a Role

Before You Begin  
The role must already be assigned to you. By default, only the root role exists.

1  
**In a terminal window, determine which roles you can assume.**

```bash
% roles
Comma-separated list of role names is displayed
```

2  
**Use the su command to assume a role.**

```bash
% su - rolename
Password: <Type rolename password>
$
```

The `su - rolename` command changes the shell to a profile shell for the role. A profile shell recognizes security attributes, such as authorizations, privileges, and set ID bits.

3  
**(Optional) Verify that you are now in a role.**

```bash
$ /usr/bin/whoami
rolename
```

You can now perform role tasks in this terminal window.

4  
**(Optional) View the capabilities of your role.**

For sample output, see “How to View Your Assigned Rights” on page 151.

```bash
$ profiles -l
verbose rights profiles output
```
Example 9–5  Assuming the root Role

In the following example, the initial user assumes the root role and lists the privileges in the role's shell.

```bash
% roles
root
% su - root
Password:  <Type root password>
#  Prompt changes to root prompt
#  ppriv $$
1200:  pfksh
flags = <none>
  E:  all
  I:  basic
  P:  all
  L:  all
```

For information about privileges, see “Privileges (Overview)” on page 138.

▼ How to Change the Security Attributes of a User

User properties include login shell, rights profiles, and roles. The most secure method of giving a user administrative capabilities is to assign a role to the user. For a discussion, see “Security Considerations When Directly Assigning Security Attributes” on page 137.

Before You Begin

In the default configuration, you must assume the root role to modify a user's security attributes.

After configuring RBAC for your site, you have other options. To change most security attributes of a user, including the password, you must become an administrator who is assigned the User Security rights profile. To assign audit flags or change a role's password, you must assume the root role. To change other user attributes, you must become an administrator who is assigned the User Management rights profile. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

- Use the `usermod` command.

This command modifies the attributes of a user that is defined in the local naming service or the LDAP naming service. The RBAC arguments to this command are similar to the arguments to the `useradd` and `rolemod` commands, as described on the `user_attr(4)` man page, and shown in Step 1 in “How to Change the Security Attributes of a User” on page 155.
The RBAC arguments to the command are the following:

```
# usermod [-e expire] [-f inactive] [-s shell] [-m] [-A authorization-list] \ 
```

- `-e expire` Is the date that a user login expires. Use this option to create temporary users.
- `-f inactive` Is the maximum number of days that is allowed between user logins. When the `inactive` value is exceeded, the user cannot log in. The default value is 0, no expiration date.
- `-m` Creates a home directory for `rolename` at the default location.
- `-s shell` Is the login shell for `rolename`. This shell must be a profile shell, such as `pfbash`. For a list of profile shells, see the `pfexec(1)` man page.

**Tip** – You can also list the profile shells from the `/usr/bin` directory on your system, as in `ls /usr/bin/pf*sh`.

- `-A authorizations-list` Is one or more authorizations separated by commas. For the list of available authorizations, see “How to View All Defined Security Attributes” on page 150.
- `-P profiles-list` Is one or more rights profiles separated by commas. For the list of rights profiles, see “How to View All Defined Security Attributes” on page 150.
- `-R roles-list` Is one or more roles separated by commas. To create roles, see “How to Create a Role” on page 163.
- `-K key=value` Is a `key=value` pair. This option can be repeated. The following keys are available: `audit_flags`, `auths`, `profiles`, `project`, `defaultpriv`, `limitpriv`, `lock_after_retries`, `pam_policy`, and `roleauth`. For information about the keys, their values, and the authorizations that are required to set the values, see the `user_attr(4)` man page.
- `-S repository` Is one of `files` or `ldap`. The default is local files.

`login` Is the user name.

To assign authorizations to a user, see Example 9–7.

To assign a rights profile to a user, see Example 9–6.

To assign an existing role to a user, see “How to Assign a Role” on page 165. In the default configuration, you can assign the root role to an existing user.

To modify the privileges of a user, see Example 9–13 and Example 9–9.
Example 9-6  Creating a User Who Can Manage DHCP

In this example, the security administrator creates a user in LDAP. At login, the jdoe-dhcp user is able to manage DHCP.

```
# useradd -P "DHCP Management" -s /usr/bin/pfbash -S ldap jdoe-dhcp
```

Because the user is assigned pfbash as the login shell, the security attributes in the DHCP Management rights profile are available to the user in the user's default shell.

Example 9-7  Assigning Authorizations Directly to a User

In this example, the security administrator creates a local user who can control screen brightness.

```
# useradd -c "Screened JDoe, local" -s /usr/bin/pfbash \
-A solaris.system.power.brightness jdoe-scr
```

This authorization is added to the user's existing authorization assignments.

Example 9-8  Removing Privileges From a User's Limit Set

In the following example, all sessions that originate from jdoe's initial login are prevented from using the sys_linkdir privilege. That is, the user cannot make hard links to directories, nor can the user unlink directories, even after the user runs the su command.

```
$ usermod -K 'limitpriv=all,!sys_linkdir' jdoe
$ userattr limitpriv jdoe
    all,!sys_linkdir
```

Example 9-9  Assigning Privileges Directly to a User

In this example, the security administrator trusts the user jdoe with a very specific privilege that affects system time.

```
$ usermod -K defaultpriv='basic,proc_clock_highres' jdoe
```

The values for the defaultpriv keyword replace the existing values. Therefore, for the user to retain the basic privileges, the value basic is specified. In the default configuration, all users have basic privileges. For the list of basic privileges, see Step 4.

▼ How to Use Your Assigned Administrative Rights

In the root role, the initial user has all administrative rights.
Step 1 shows how to administer the system if you are assigned administrative rights. Step 2 shows how non-root accounts can edit a system file.

**Before You Begin**

You have been assigned rights that regular users are not assigned. If you are not root, you must be assigned a role, an administrative rights profile, or specific privileges or authorizations.

1. **Choose one of the following methods to run administrative commands.**
   
   Open a terminal window.

   - **Become root.**
     
     `% su -
     Password: Type the root password
     #`

     **Note** – This method works whether root is a user or a role. The pound sign (#) prompt indicates that you are now root.

   - **Assume a role that you have been assigned.**
     
     In the following example, you assume an audit configuration role. This role includes the Audit Configuration rights profile.

     `% su - audadmin
     Password: Type the audadmin password
     $`

     The shell in which you typed this command is now in a profile shell. In this shell, you can run the `auditconfig` command. For more about profile shells, see "Profile Shells and RBAC" on page 137.

     **Tip** – Use the steps in “How to View Your Assigned Rights” on page 151 to view the capabilities of your role.

   - **As a user, use the `pfbash` command to create a shell that runs with administrative rights.**
     
     For example, the following set of commands enables you to view audit preselection values and audit policy in the `pfbash` shell:

     `% pfbash
     $ auditconfig -getflags
     active user default audit flags = ua,ap,lo(0x45000,0x45000)
     configured user default audit flags = ua,ap,lo(0x45000,0x45000)
     $ auditconfig -getpolicy
     configured audit policies = cnt
     active audit policies = cnt`
As a user, use the `pfexec` command to create a process that runs with administrative rights.

Run the `pfexec` command with the name of a privileged command from your rights profile. For example, the following command enables you to view the user's preselected audit flags:

```
% pfexec auditconfig -getflags
active user default audit flags = ua,ap,lo(0x45000,0x45000)
configured user default audit flags = ua,ap,lo(0x45000,0x45000)
```

The same privilege limitations apply to `pfexec` as to `pfbash`. However, to run another privileged command, you must type `pfexec` again before you type the privileged command.

```
% pfexec auditconfig -getpolicy
configured audit policies = cnt
active audit policies = cnt
```

As a user, use the `sudo` command to create a process that runs with administrative rights.

Run the `sudo` command with the name of an administrative command that you are assigned in the `sudoers` file. For more information, see the `sudo(1M)` and `sudoers(4)` man pages.

To edit a system file, use the `pfedit` command.

If you are not `root` with the UID of 0, by default you cannot edit system files. However, if you are assigned the `solaris.admin.edit/path-to-system-file` authorization, you can edit `system-file`. For example, if you are assigned the `solaris.admin.edit/etc/security/audit_warn` authorization, you can edit the `audit_warn` file.

```
$ pfedit /etc/security/audit_warn
```

The command uses the value of `$EDITOR` to determine the text editor. For more information, see the `pfedit(1M)` man page. The `pfedit` command is usefully run by the `root` role, if auditing is configured to audit `AUE_PFEXEC` events.

Example 9–10 Caching Authentication for Ease of Role Use

In this example, the administrator configures a role to manage audit configuration, but provides ease of use by caching the user's authentication. First, the administrator creates and assigns the role.

```
# roleadd -K roleauth=user -P "Audit Configuration" audadmin
# usermod -R +audadmin jdoe
```

When `jdoe` uses the `-c` option when switching to the role, a password is required before the auditconfig output is displayed:

```
% su - audadmin -c auditconfig option
Password:
  auditconfig output
```
If authentication is not being cached, and jdoe runs the command again immediately, a password prompt appears.

The administrator creates a file in the `pam.d` directory to hold an `su` stack that enables the caching of authentication, so that a password is initially required, but not thereafter until a certain amount of time has passed.

```
# pfedit /etc/pam.d/su
## Cache authentication for switched user
#
auth required      pam_unix_cred.so.1
auth sufficient    pam_ttyTickets.so.1
auth requisite     pam_authok_get.so.1
auth required      pam_dhkeys.so.1
auth required      pam_unix_auth.so.1
```

After creating the file, the administrator checks the entries for typos, omissions, or repetitions.

The administrator must provide the entire preceding `su` stack. The `pam_ttyTickets.so.1` module implements the cache. For more about PAM, see the `pam.conf(4)` man page and Chapter 14, “Using Pluggable Authentication Modules.”

After the administrator adds the `su` PAM file and reboots the system, all roles including the `audadmin` role are prompted only once for a password when running a series of commands.

```
% su - audadmin -c auditconfig option
Password: 
auditconfig output
% su - audadmin -c auditconfig option
auditconfig output
...
```

Customizing RBAC for Your Site (Tasks)

Initial configuration of RBAC includes creating users who can assume specific roles, creating the roles, and assigning them to the appropriate users.

Initially Configuring RBAC (Task Map)

Use the following task map to plan and initially implement RBAC at your site. Some tasks are ordered.
### How to Plan Your RBAC Implementation

RBAC can be an integral part of how an organization manages its information resources. Planning requires a thorough knowledge of the RBAC capabilities as well as the security requirements of your organization.

**Note** – Default rights are assigned in the `/etc/security/policy.conf` file.
1 Learn the basic RBAC concepts.

Read “Role-Based Access Control (Overview)” on page 127. Using RBAC to administer a system is very different from using conventional UNIX administrative practices. To be familiar with RBAC concepts before you start your implementation, see Chapter 10, “Security Attributes in Oracle Solaris (Reference).”

2 Examine your security policy.

Your organization’s security policy details the potential threats to your system, measures the risk of each threat, and provides strategies to counter these threats. Isolating the security-relevant tasks through RBAC can be a part of the strategy. Although you can use the installed RBAC configurations as is, you might need to customize it to adhere to your security policy.

3 Decide how much RBAC your organization needs.

Depending on your security needs, you can use varying degrees of RBAC, as follows:

- **Root as a role** – This method is provided by default. It prevents any user from logging in as root. Instead, a user must log in by using their assigned login prior to assuming the root role.

- **Discrete roles** – This method creates roles that are based on provided rights profiles. The roles can be assigned according to level of responsibility, scope of task, and type of task. For example, the System Administrator role can perform many tasks that superuser can perform, while the Network IPsec Management role can manage IPsec.

  You can also separate security responsibilities from other responsibilities, The User Management role can create users, while the User Security role can assign security attributes, such as passwords, roles, and rights profiles. However, the User Security role cannot create a user, and the User Management role cannot assign a rights profile to a user.

- **No root role** – This method requires you to change the default configuration of the system. In this configuration, any user who knows the password for root can log in and modify the system. You cannot tell which user was acting as superuser.

4 Decide which roles are appropriate for your organization.

Review the capabilities of the recommended roles and their default rights profiles. Default rights profiles enable administrators to configure a recommended role by using a single profile.

To further examine rights profiles, do one of the following:

- View the available rights profiles on your system, by using the `getent prof_attr` command.

- In this guide, read “Rights Profiles” on page 195 for summaries of some typical rights profiles.
5 Decide if any additional roles or rights profiles are appropriate for your organization.
Look for other applications or families of applications at your site that might benefit from restricted access. Applications that affect security, that can cause denial-of-service problems, or that require special administrator training are good candidates for RBAC. You can customize roles and rights profiles to handle the security requirements of your organization.

a. Determine which commands are needed for the new task.

b. Decide which rights profile is appropriate for this task.
Check if an existing rights profile can handle this task or if a separate rights profile needs to be created.

Note – The Media Backup and Media Restorerights profiles provide access to the entire root file system. Therefore, these rights profiles are appropriately assigned to trusted users only. Alternatively, you can choose to not assign these rights profiles. By default, only the root role is trusted to back up and restore.

c. Determine which role is appropriate for this rights profile.
Decide if the rights profile for this task should be assigned to an existing role or if a new role should be created. If you use an existing role, check that the role’s original rights profiles are appropriate for users who are assigned to this role. Order the new rights profile so that commands execute with their required privileges. For information about ordering, see “Order of Search for Assigned Security Attributes” on page 197.

6 Decide which users should be assigned to which roles.
According to the principle of least privilege, you assign users to roles that are appropriate to the user’s level of trust. When you prevent users from performing tasks that the users do not need to perform, you reduce potential problems.

▼ How to Create a Role

Roles can be created locally and in an LDAP repository.

Before You Begin
To create a role, you must become an administrator who is assigned the User Management rights profile. To assign security attributes to the role, including the initial password, you must become an administrator who is assigned the User Security rights profile. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1 To create a role, use the roleadd command.
For restrictions on acceptable strings, see the roleadd(1M) man page.

```
```
Tip – When the name of the role reflects the name of a rights profile, you can easily understand the purpose of the role. For example, assign the Audit Review rights profile to the auditreview role to enable the role to read, filter, and archive audit records.

The RBAC arguments to this command are similar to the arguments to the usermod command, as described on the user_attr(4) man page, and shown in Step 1 in “How to Change the Security Attributes of a User” on page 155. For example, the following command creates a local User Administrator role and a home directory:

```
# roleadd -c "User Administrator role, local" -s /usr/bin/pfbash \ 
-m -K profiles="User Security,User Management" useradm
80 blocks
# ls /export/home/useradm
local.cshrc   local.login   local.profile
```

2 Create the initial password for the role.

```
# passwd -r files useradm
Password:   <Type useradm password>
Confirm Password: <Retype useradm password>
#
```

Note – Typically, a role account is assigned to more than one user. Therefore, an administrator typically creates a role password and provides the users with the role password out of band.

3 To assign the role to a user, run the usermod command.

For the procedure, see “How to Assign a Role” on page 165 and Example 9–14.

Example 9–11 Creating a User Administrator Role in the LDAP Repository

In this example, the administrator’s site uses an LDAP repository. By running the following command, the administrator creates a User Administrator role in LDAP.

```
# roleadd -c "User Administrator role, LDAP" -s /usr/bin/pfbash \ 
-m -S ldap -K profiles="User Security,User Management" useradm
```

Example 9–12 Creating Roles for Separation of Duty

In this example, the administrator’s site uses an LDAP repository. By running the following commands, the administrator creates two roles. The usermgmt role can create users, give them home directories, and perform other non-security tasks. The usermgmt role cannot assign passwords or other security attributes. The usersec role cannot create users, but can assign passwords and change other security attributes.
# roleadd -c "User Management role, LDAP" -s /usr/bin/pfbash \
-m -S ldap -K profiles="User Management" usermgt
# roleadd -c "User Security role, LDAP" -s /usr/bin/pfbash \
-m -S ldap -K profiles="User Security" usersec

Creating a Device and File Security Role

In this example, the administrator creates a Device and File Security role for this system:

# roleadd -c "Device and File System Security admin, local" -s /usr/bin/pfbash \
-m -K profiles="Device Security,File System Security" devflsec

How to Assign a Role

This procedure assigns a role to a user, restarts the name cache daemon, and then shows how the user can assume the role.

Assign the role to a user.

usermod [-S repository] [RBAC-arguments] login

For example, assign the role to a local user:

# usermod -R +useradm jdoe-local

The changes are in effect at the user’s next login.

For the options to the usermod command, see the usermod(1M) man page or the description of the options to the rolemod in Step 1 in "How to Change the Security Attributes of a User" on page 155.

Creating and Assigning a Role to Administer Crypto

In this example, the administrator on an LDAP network creates a role to administer the Cryptographic Framework, and assigns the role to UID 1111.

# roleadd -c "Cryptographic Services manager" \
-g 14 -m -u 104 -s /usr/bin/pfksh \
-S ldap -K profiles="Crypto Management" cryptmgt
# passwd cryptmgt
New Password:     <Type cryptmgt password>
Confirm password: <Retype cryptmgt password>

# usermod -u 1111 -R +cryptmgt

The user with UID 1111 logs in, then assumes the role and displays the assigned security attributes.

% su - cryptmgt
Password:     <Type cryptmgt password>
Confirm Password: <Retype cryptmgt password>

$ profiles -l
   Crypto Management               euid=0
       /usr/bin/kmfcfg               euid=0
       /usr/sbin/cryptoadm           euid=0
       /usr/sfw/bin/CA.pl             euid=0
       /usr/sfw/bin/openssl           euid=0

$ For information about the Cryptographic Framework, see Chapter 11, “Cryptographic Framework (Overview).” To administer the framework, see “Administering the Cryptographic Framework (Task Map)” on page 230.

▼ How to Audit Roles

The actions that a role performs can be audited. Included in the audit record is the login name of the user who assumed the role, the rolename, and the action that the role performed. The 116:AUE_PFEXEC:execve(2) with pfexec enabled:ps,ex,ua,as audit event captures role actions,. By preselecting one of the as, ex, ps, or ua classes, role actions are audited.

Before You Begin To configure auditing, you must become an administrator who is assigned the Audit Configuration rights profile. To enable or refresh the audit service, you must become an administrator who is assigned the Audit Control rights profile. The root role can perform every task in this procedure. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1 Include the auditing of roles in your audit plan.
For planning information, see Chapter 27, “Planning for Auditing.”

2 Preselect one of the as, ex, ps, or ua classes.
   - If the audit service is enabled, review the preselected classes.
     # auditconfig -getflags
If one of the as, ex, ps, or ua classes is preselected, role actions are being audited. If not, add one of these classes to the existing classes.

```
# auditconfig -setflags existing preselections, as
```

- If auditing is not yet enabled, preselect a class that audits role actions.

```
# auditconfig -setflags as
```

In this example, the administrator chooses the as class. This class includes other audit events. To view the audit events that are included in a class, use the auditrecord command, as shown in Example 28–28.

3 Enable or refresh the audit service.

```
# audit -s
```

### How to Create a Rights Profile

You can create or change a rights profile when the provided rights profiles do not contain the collection of security attributes that you need. To learn more about rights profiles, see "RBAC Rights Profiles" on page 135.

**Before You Begin**

To create a rights profile, you must become an administrator who is assigned the File Security rights profile. For more information, see "How to Use Your Assigned Administrative Rights" on page 157.

1 **Create a rights profile.**

```
# profiles -p [-S repository] profile-name
```

You are prompted for a description.

2 **Add contents to the rights profile.**

Use the set subcommand for profile properties that have a single value, such as set desc. Use the add subcommand for properties that have more than one value, such as add cmd.

For example, the following command creates the custom PAM rights profile in "How to Assign a New Rights Policy to All Users" on page 272 interactively. The name is shortened for display purposes.

```
# profiles -p -S LDAP "Site PAM LDAP"
profiles:Site PAM LDAP> set desc="Profile which sets pam_policy=ldap"
....LDAP> set pam_policy=ldap
....LDAP> commit
....LDAP> end
....LDAP> exit
```
Creating a Sun Ray Users Rights Profile

In this example, the administrator creates a rights profile for Sun Ray users in the LDAP repository. The administrator has already created a Sun Ray version of the Basic Solaris User rights profile, and has removed all rights profiles from the policy.conf file on the Sun Ray server.

```
# profiles -p -S LDAP "Sun Ray Users"
profiles:Sun Ray Users> set desc="For all users of Sun Rays"
... Ray Users> add profiles="Sun Ray Basic User"
... Ray Users> set defaultpriv="basic,!proc_info"
... Ray Users> set limitpriv="basic,!proc_info"
... Ray Users> end
... Ray Users> exit
```

The administrator verifies the contents.

```
# profiles -p "Sun Ray Users"
Found profile in LDAP repository.
profiles:Sun Ray Users> info
  name=Sun Ray Users
  desc=For all users of Sun Rays
  defaultpriv=basic,!proc_info,
  limitpriv=basic,!proc_info,
  profiles=Sun Ray Basic User
```

Removing a Basic Privilege From a Rights Profile

In the following example, after thorough testing, the security administrator removes another basic privilege from the Sun Ray Users rights profile. In Example 9–15, the administrator removed one privilege. The rights profile is modified to remove two basic privileges. Users who are assigned this profile cannot examine any processes outside their current session, and they cannot add another session.

```
$ profiles -p "Sun Ray Users"
profiles:Sun Ray Users> set defaultpriv="basic,!proc_session,!proc_info"
profiles:Sun Ray Users> end
profiles:Sun Ray Users> exit
```

Removing Privileges From the Limit Set of a Rights Profile

In the following example, after thorough testing, the security administrator removes two limit privileges from the Sun Ray Users rights profile.

```
$ profiles -p "Sun Ray Users"
profiles:Sun Ray Users> set limitpriv="all,!proc_session,!proc_info"
profiles:Sun Ray Users> end
profiles:Sun Ray Users> exit
```
Example 9–18  Creating a Rights Profile That Includes Privileged Commands

In this example, the security administrator adds privileges to an application in a rights profile that the administrator creates. The application is privilege-aware.

```
# profiles -p SiteApp
profiles:SiteApp> set desc="Site application"
profiles:SiteApp> add cmd="/opt/site-app/bin/site-cmd"
profiles:SiteApp:site-cmd> add privs="proc_fork,proc_taskid"
profiles:SiteApp> exit
```

To verify, the administrator selects the site-cmd.

```
# profiles -p SiteApp "select cmd=/opt/site-app/bin/site-cmd; info;end"
Found profile in files repository.
   id=/opt/site-app/bin/site-cmd
   privs=proc_fork,proc_taskid
```

See Also  To troubleshoot security attribute assignment, see “How to Troubleshoot RBAC and Privilege Assignment” on page 174. For background, see “Order of Search for Assigned Security Attributes” on page 197.

▼ How to Clone and Modify a System Rights Profile

The rights profiles that Oracle Solaris provides are read-only. You can clone a provided rights profile for modification if its collection of security attributes is insufficient. For example, you might want to add the `solaris.admin.edit/path-to-system-file` authorization to a provided rights profile.

Before You Begin  To create or change a rights profile, you must become an administrator who is assigned the File Security rights profile. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1  Create a new rights profile from an existing profile.

   ```
   # profiles -p [-S repository] existing-profile-name
   ```

   - To enhance an existing rights profile, create a new profile.
     Add the existing rights profile as a supplementary rights profile, then add the enhancements. For an example, see Example 9–19.

   - To remove content from an existing rights profile, clone the profile.
     Then, rename it and modify. For an example, see Example 9–20.

2  Continue to modify the new rights profile.

   Add or remove supplementary rights profiles, authorizations, and other security attributes.
Cloning and Enhancing the Network IPsec Management Rights Profile

In this example, the administrator adds several `solaris.admin.edit` authorizations to a site IPsec Management rights profile.

The administrator verifies that the Network IPsec Management rights profile cannot be modified.

```
# profiles -p "Network IPsec Management"
profiles:Network IPsec Management> add auths="solaris.admin.edit/etc/hosts"
Cannot add. Profile cannot be modified
```

Then, the administrator creates a rights profile that includes the Network IPsec Management profile.

```
# profiles -p "Total IPsec Mgt"
... IPsec Mgt> set desc="Network IPsec Mgt plus edit authorization"
... IPsec Mgt> add profiles="Network IPsec Management"
... IPsec Mgt> add auths="solaris.admin.edit/etc/hosts"
... IPsec Mgt> add auths="solaris.admin.edit/etc/inet/ipsecinit.conf"
... IPsec Mgt> add auths="solaris.admin.edit/etc/inet/ike/config"
... IPsec Mgt> add auths="solaris.admin.edit/etc/inet/secret/ipseckeys"
... IPsec Mgt> end
... IPsec Mgt> exit
```

The administrator verifies the contents.

```
# profiles -p "Total IPsec Mgt" info
name=Total IPsec Mgt
desc=Network IPsec Mgt plus edit authorization
auths=solaris.admin.edit/etc/hosts,
solaris.admin.edit/etc/inet/ipsecinit.conf,
solaris.admin.edit/etc/inet/ike/config,
solaris.admin.edit/etc/inet/secret/ipseckeys
profiles=Network IPsec Management
```

Cloning and Removing Security Attributes From a Rights Profile

In this example, the administrator separates managing the properties of the VSCAN service from the ability to enable and disable the service.

First, the administrator lists the contents of the rights profile that Oracle Solaris provides.

```
% profiles -p "VSCAN Management" info
name=VSCAN Management
desc=Manage the VSCAN service
auths=solaris.smf.manage.vscan,solaris.smf.value.vscan,
solaris.smf.modify.application
help=RtVscanMngmnt.html
```

Then, the administrator creates a rights profile to enable and disable the service.
# profiles -p "VSCAN Management"
profiles:VSCAN Management> set name="VSCAN Control"
profiles:VSCAN Control> set desc="Start and stop the VSCAN service"
... VSCAN Control> remove auths="solaris.smf.modify.application"
... VSCAN Control> end
... VSCAN Control> exit

Then, the administrator creates a rights profile that can change the properties of the service.

# profiles -p "VSCAN Management"
profiles:VSCAN Management> set name="VSCAN Properties"
profiles:VSCAN Properties> set desc="Modify VSCAN service properties"
... VSCAN Properties> remove auths="solaris.smf.manage.vscan"
... VSCAN Properties> end
... VSCAN Properties> exit

The administrator verifies the contents of the new rights profiles.

# profiles -p "VSCAN Control" info
name=VSCAN Control
desc=Start and stop the VSCAN service
auths=solaris.smf.manage.vscan
# profiles -p "VSCAN Properties" info
name=VSCAN Properties
desc=Modify VSCAN service properties
auths=solaris.smf.value.vscan,solaris.smf.modify.application

See Also
To troubleshoot security attribute assignment, see “How to Troubleshoot RBAC and Privilege Assignment” on page 174. For background, see “Order of Search for Assigned Security Attributes” on page 197.

▼ How to Create an Authorization

You can create an authorization when the provided authorizations do not cover the authorizations that you need. To learn more about authorizations, see “RBAC Authorizations” on page 133.

Before You Begin
You have defined and used the authorization in the program you are protecting. For instructions, see Developer’s Guide to Oracle Solaris 11 Security and “About Authorizations” in Developer’s Guide to Oracle Solaris 11 Security.

1 (Optional) Create the help file for your new authorization.
For example, create the help file for an authorization to enable the user to modify the data in an application.

# pfedit /docs/helps/NewcoSiteAppModData.html
</HTML>
-- Copyright 2012 Newco. All rights reserved.
The `com.newco.siteapp.data.modify` authorization authorizes you to modify existing data in the application. Only authorized accounts are permitted to modify data. Use this authorization with care.

2 Create the authorization.
   
   For example, create the `com.newco.siteapp.data.modify` authorization on the local system.

   ```bash
   # auths add -t "SiteApp Data Modify Authorized" -h /docs/helps/NewcoSiteAppModData.html com.newco.siteapp.data.modify
   
   You can now add the authorization to a rights profile and assign the profile to a role or user.
   
Example 9–21 Adding Authorizations to a Rights Profile

   In this example, the administrator adds an authorization that a site application checks before allowing a user to run the application.

   After creating the authorization, the security administrator adds the `com.newco.siteapp.data.modify` authorization to an existing rights. The administrator created the profile in Example 9–18.

   ```bash
   # profiles -p “SiteApp”
   profiles:SiteApp> add auths=”com.newco.siteapp.data.modify”
   profiles:SiteApp> end
   profiles:SiteApp> exit
   
   To verify, the administrator lists the contents of the profile.

   ```bash
   # profiles -p SiteApp
   Found profile in files repository.
   id=/opt/site-app/bin/site-cmd
   auths=com.newco.siteapp.data.modify
   ```

How to Add RBAC Properties to Legacy Applications

   A legacy application is a command or set of commands. The security attributes are set for each command in a rights profile that is assigned to a role. A user who assumes the role can run the legacy application with the security attributes.
Before You Begin

To create the rights profile, you must become an administrator who is assigned the Information Security or Rights Management rights profile. To assign the rights profile, you must become an administrator who is assigned the User Security rights profile. The root role can perform every task in this procedure. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1 Add security attributes to the commands that implement the legacy application.
You add security attributes to a legacy application in the same way that you would for any command. You must add the command with security attributes to a rights profile. For a legacy command, give the command euid=0 or uid=0 security attributes. For details of the procedure, see “How to Create a Rights Profile” on page 167.

a. Create a new rights profile for your legacy application.
For the steps, see “How to Create a Rights Profile” on page 167.

b. Add the commands with their required security attributes.
For an example, see Example 9–18.

2 Include the rights profile in a role’s list of profiles.
To assign a rights profile to a role, see Example 9–14.

Example 9–22 Adding Security Attributes to Commands in a Script

If a command in a script needs to have the setuid bit or setgid bit set to succeed, the script executable and the command must have the security attributes added in a rights profile. Then, the rights profile is assigned to a role, and the role is assigned to a user. When the user assumes the role and executes the script, the command runs with the security attributes.

Example 9–23 Checking for Authorizations in a Script or Program

To check for authorizations, you write a test that is based on the auths command. For detailed information about this command, see the auths(1) man page.

For example, the following line tests if the user has the authorization that is supplied as the $1 argument:

```
if [ '/usr/bin/auths|/usr/xpg4/bin/grep $1' ]; then
   echo Auth granted
else
   echo Auth denied
fi
```
To be more complete, the test must include logic that checks for the use of wildcards. For example, to test if the user has the `solaris.system.date` authorization, you would need to check for the following strings:

- `solaris.system.date`
- `solaris.system.*`
- `solaris.*`

If you are writing a program, use the function `getauthattr()` to test for the authorization.

### How to Troubleshoot RBAC and Privilege Assignment

Several factors can affect why a user or role's processes do not run with assigned security attributes. This procedure helps you debug failed security attribute assignments. Several of the steps are based on “Order of Search for Assigned Security Attributes” on page 197.

**Before You Begin**

You must assume the root role. For more information, see "How to Use Your Assigned Administrative Rights" on page 157.

1. **Verify and restart the naming service.**

   a. **Verify that the security assignments for the user or role are in the naming service that is enabled on the system.**

   ```
   # svccfg -s name-service/switch
   svc:/system/name-service/switch> listprop config
   config  application
   config/value_authorization  astring  solaris.smf.value.name-service.switch
   config/default  astring  files ldap
   config/host  astring  "files dns mdns ldap"
   config/netgroup  astring  ldap
   config/printer  astring  "user files"
   ```

   In this output, all services that are not explicitly mentioned inherit the value of the default, files ldap. Therefore, `passwd` (and therefore `user_attr`), `auth_attr`, and `prof_attr` are searched first in files, then in LDAP.

   b. **Restart the name service cache, svc:/system/name-service/cache.**

   The `nscd` daemon can have a lengthy time-to-live interval. By restarting the daemon, you update the naming service with current data.

   ```
   # svcadm restart name-service/cache
   ```
2 Determine where a security attribute is assigned to the user.

Use the security attribute as the value to the `userattr -v` command. For example, the following commands indicate which security attributes are assigned and where the assignment was made for the user `jdoe`:

```
# userattr -v audit_flags jdoe
user_attr: fw:no
# userattr -v auths jdoe
Basic Solaris User :solaris.mail.mailq,solaris.network.autoconf.read,
solaris.admin.wusb.read
Console User :solaris.system.shutdown,solaris.device.cdrw,
solaris.device.mount.removable,solaris.smf.manage.vbiosd,solaris.smf.value.vbiosd
# userattr -v defaultpriv jdoe
# userattr -v limitpriv jdoe
# userattr -v lock_after_retries jdoe
# userattr -v profiles jdoe
user_attr: Audit Review,Stop
# userattr roles jdoe
user_attr : cryptomgt,infosec
```

The output indicates that `jdoe` is directly assigned audit flags, two rights profiles, and two roles. Therefore, any audit flag values in the rights profiles are ignored. Upon assuming a role, the audit flags in that role replace the audit flags for the user.

3 Verify that the assigned authorizations are spelled correctly.

The source of an authorization assignment is not important, because authorizations accumulate for users. However, a misspelled authorization fails silently.

4 For rights profiles that you have created, verify that you have assigned the appropriate security attributes to the commands in that profile.

For example, some commands require `uid=0` rather than `euid=0` to succeed. Also, the options to some commands can require authorizations.

5 Check the following if security attributes are not available to a user.

   a. Check if the security attributes are directly assigned to the user.

      Use the `userattr` command, as shown in Step 2.

   b. If the security attributes are not directly assigned, check the rights profiles that are directly assigned to the user.

      i. In order, check for the security attribute assignment in the list of rights profiles.

      The value of the attribute in the earliest rights profile in the list is the value in the kernel. If this value is incorrect, either change the value in that rights profile, or reassign the profiles in the correct order.
For privileged commands, check if a privilege is assigned in the `defaultpriv` keyword, or removed in the `limitpriv` keyword.

**ii. If no attribute assignment is listed, check the roles that the user is assigned.**

If the attribute is assigned to a role, the user must assume the role to obtain the security attributes. If the attribute is assigned to more than one role, the assignment in the earliest role in the list is effective. If this value is incorrect, either assign the correct value to the first role in the list, or reassign the roles in the correct order.

6 **If you assigned a privilege directly to a user or role, check if a failed command requires authorizations to succeed.**

a. **Check if an option to the command requires authorization.**

   Rather than assign a privilege directly, assign the privilege to the command that requires it, add the required authorizations, place the command and authorizations in a rights profile, and assign the profile to the user.

b. **Check if a rights profile that includes the required authorization exists.**

   If it exists, assign it to the user. Order the rights profile before any other rights profile that includes the command.

7 **Check the following if a command continues to fail for a user.**

a. **Verify that the user is executing the command in a profile shell.**

   Administrative commands must be executed in a profile shell. To mitigate user error, you can assign a profile shell as the user’s login shell. Or, you can remind the user to run administrative commands in a profile shell.

b. **Check if any security attributes that are directly assigned to the user prevent the command from succeeding.**

   In particular, check the values of the user’s `defaultpriv` and `limitpriv` attributes.

c. **Determine which rights profile or role includes the command.**

i. **In order, check for the command with security attributes in the list of rights profiles.**

   The earliest value in the list of rights profiles is the value in the kernel. If this value is incorrect, either change the value in that rights profile, or reassign the profiles in the correct order.

   In particular, check the values of the profile’s `defaultpriv` and `limitpriv` attributes.
ii. If no attribute assignment is listed, check the roles that the user is assigned.

If the command is assigned to a role, the user must assume the role to obtain the security attributes. If the attribute is assigned to more than one role, the assignment in the earliest role in the list is effective. If the value is incorrect, either assign the correct value to the first role in the list, or reassign the roles in the correct order.

8 Check the following if a command fails for a role.

Administrative commands require privileges to succeed. The options to some commands can require authorization. Best practice is to assign a rights profile that includes the administrative command.

a. Check if any security attributes that are directly assigned to the role prevent the command from succeeding.

In particular, check the values of the role’s defaultpriv and limitpriv attributes.

b. In order, check for the command with security attributes in the list of rights profiles.

The earliest value in the list of rights profiles is the value in the kernel. If this value is incorrect, either change the value in that rights profile, or reassign the profiles in the correct order.

Managing RBAC (Tasks)

After you have configured and are using RBAC, use the following procedures to maintain and modify RBAC on your systems.

Managing RBAC (Task Map)

The following task map points to procedures for maintaining role-based access control (RBAC) after RBAC has been initially implemented.

<table>
<thead>
<tr>
<th>Task</th>
<th>Description</th>
<th>For Instructions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Change the role password.</td>
<td>The root role changes the password of the role.</td>
<td>&quot;How to Change the Password of a Role&quot; on page 178</td>
</tr>
<tr>
<td>Modify the assigned rights of a role.</td>
<td>Modifies the security attributes of a role.</td>
<td>&quot;How to Change the Security Attributes of a Role&quot; on page 179 Example 9–26</td>
</tr>
<tr>
<td>Modify a rights profile.</td>
<td>Modifies security attribute values in a rights profile, such as limit privileges and default privileges.</td>
<td>Example 9–17 Example 9–8</td>
</tr>
</tbody>
</table>
Managing RBAC (Tasks)

<table>
<thead>
<tr>
<th>Task</th>
<th>Description</th>
<th>For Instructions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reorder rights profile assignment.</td>
<td>Ensures that assigned security attributes are available to a user or role.</td>
<td>“How to Reorder Assigned Security Attributes” on page 180</td>
</tr>
<tr>
<td>Create a restricted profile shell.</td>
<td>Prevents users or roles from full access to all commands in the software.</td>
<td>“How to Restrict an Administrator to Explicitly Assigned Rights” on page 181</td>
</tr>
<tr>
<td>Restrict a user’s privileges.</td>
<td>Limits the user’s basic or limit set of privileges.</td>
<td>Example 9–8</td>
</tr>
<tr>
<td>Remove default rights from a system.</td>
<td>Creates a system for special uses.</td>
<td>Example 9–28</td>
</tr>
<tr>
<td>Enable a user to supply the user’s password to assume a role.</td>
<td>Modifies a user’s security attributes to make the user’s password authenticate the user to a role. This behavior is similar to Linux role behavior.</td>
<td>“How to Enable a User to Use Own Password to Assume a Role” on page 182</td>
</tr>
<tr>
<td>Change the root role into a user.</td>
<td>Prior to decommissioning a system, change the root role into a user.</td>
<td>“How to Change the root Role Into a User” on page 183</td>
</tr>
<tr>
<td>Restrict user actions on the desktop.</td>
<td>While these actions do not require RBAC, they add security to your system.</td>
<td>Chapter 11, “Disabling Features in the Oracle Solaris Desktop System,” in Oracle Solaris 11.1 Desktop Administrator’s Guide</td>
</tr>
</tbody>
</table>

These procedures manage security attributes on users, roles, and rights profiles. For basic user management procedures, refer to Chapter 1, “Managing User Accounts and User Environments (Overview),” in Managing User Accounts and User Environments in Oracle Solaris 11.1.

**How to Change the Password of a Role**

Because a role can be assigned to many users, users who are assigned a role cannot change the role password.

**Before You Begin**

You must assume the root role. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

- **Run the passwd command.**

  ```
  # passwd [-r naming-service] rolename
  `-r naming-service` Applies the password change to the files or ldap repository. The default repository is files. If you do not specify a repository, the password is changed in all repositories.

  `rolename` Is the name of an existing role that you want to modify.
  ```

  For more command options, see the passwd(1) man page.
Example 9–24  Changing a Role’s Password

In this example, the root role changes the password of the local devmgt role.

```
# passwd -r files devmgt
New password: Type new password
Confirm password: Retype new password
```

In this example, the root role changes the password of the devmgt role in the LDAP directory service.

```
# passwd -r ldap devmgt
New password: Type new password
Confirm password: Retype new password
```

In this example, the root role changes the password of the devmgt role in file and LDAP.

```
# passwd devmgt
New password: Type new password
Confirm password: Retype new password
```

▼ How to Change the Security Attributes of a Role

Before You Begin  You must become an administrator who is assigned the User Security rights profile to change most of the security attributes of a role. To assign audit flags or change a role’s password, you must assume the root role. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1 Use the rolemod command.

This command modifies the attributes of a role that is defined in the local naming service or in LDAP. The values of the -A, -P, and -R options can be modified by - or +. The - sign indicates to subtract the value from the currently assigned values. The + sign indicates to add the value to the currently assigned values.

For more information about the rolemod command, see the following:

- For a short description, see the description of the roleadd command in “How to Create a Role” on page 163.
- For all arguments to this command, see the rolemod(1M) man page.
- For the list of key values for the -K option, see the user_attr(4) man page.

The following command adds two rights profiles to the devmgt role in the LDAP repository:

```
$ rolemod -P +"Device Management,File Management" -S ldap devadmin
```

2 To change a role’s password, see “How to Change the Password of a Role” on page 178.
Example 9–25 Changing a Local Role’s Security Attributes

In this example, the security administrator modifies the prtmgt role to include the VSCAN Management rights profile.

```
$ rolemod -c "Handles printers and virus scanning" \ 
-P "Printer Management,VSCAN Management,All" prtmgt
```

Example 9–26 Assigning Privileges Directly to a Role

In this example, the security administrator entrusts the systime role with a very specific privilege that affects system time.

```
$ rolemod -K defaultpriv='proc_clock_highres' systime
```

The values for the defaultpriv keyword are in the list of privileges in the role's processes at all times.

▼ How to Reorder Assigned Security Attributes

Oracle Solaris reads rights profiles in order of assignment, as described in “Order of Search for Assigned Security Attributes” on page 197. In this procedure, you reorder rights profiles.

Before You Begin

You must become an administrator who is assigned the User Security rights profile. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1 View the list of rights profiles that are currently assigned to the user or role.
   The list displays in order.
   `$ profiles username rolename`

2 Assign the rights profiles in the correct order.
   `$ usermod | rolemod -P "list-of-profiles"`

Example 9–27 Assigning Rights Profiles in a Specific Order

In this example, the administrator determines that a rights profile with privileged commands is listed after the All rights profile for the role devadmin.

```
$ profiles devadmin
  Basic Solaris User
  All
  Device Management
```
Therefore, the devadmin role cannot run the device management commands with their assigned privileges.

The administrator reassignments the rights profiles to devadmin. In the new order of assignment, devadmin can run the device management commands with their assigned privileges.

```
$ rolemod -P "Device Management,Basic Solaris User,All"
$ profiles devadmin
  Device Management
  Basic Solaris User
  All
```

▼ How to Restrict an Administrator to Explicitly Assigned Rights

You can restrict a role or user to a limited number of administrative actions in two ways.

- You can use the Stop rights profile.
  The Stop rights profile is the simplest way to create a restricted shell. The authorizations and rights profiles that are assigned in the policy.conf file are not consulted. Therefore, the role or user is not assigned the Basic Solaris User rights profile, the Console User rights profile, or the solaris.device.cdrw authorization.

- You can modify the policy.conf file on a system, and require the role or user to use that system for administrative tasks.

**Before You Begin** You must assume the root role. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

- **Add the Stop rights profile as the last profile in the list of profiles that you assign.**
For example, you could limit the auditrev role to performing only audit reviews.

```
# rolemod -P "Audit Review,Stop" auditrev
```

Because the auditrev role does not have the Console User rights profile, the auditor cannot shut down the system. Because this role does not have the solaris.device.cdrw authorization, the auditor cannot read from or write to the CD-ROM drive. Because this role does not have the Basic Solaris User rights profile, no commands other than the commands in the Audit Review rights profile can be run in this role. For example, the ls command will not run. The role uses the File Browser to view the audit files.

For more information, see “Rights Profiles” on page 195 and “Order of Search for Assigned Security Attributes” on page 197.
The `rolemod` command modifies the attributes of a role that is defined in the local naming service or in LDAP. For arguments to this command, see the `rolemod(1M)` man page. The list of RBAC arguments is similar to the list for the `roleadd` command, as described in "How to Create a Role" on page 163.

**Example 9–28  Modifying a System to Limit the Rights Available to Its Users**

In this example, the administrator creates a system that is useful only to administer the network. The administrator removes the Basic Solaris User rights profile and any authorizations from the `policy.conf` file. The Console User rights profile is not removed. The affected lines in the resulting `policy.conf` file are the following:

```plaintext
...
#AUTHS_GRANTED=
#PROFS_GRANTED=Basic Solaris User
CONSOLE_USER=Console User
...
```

Only a user who has been explicitly assigned authorizations, commands, or rights profiles is able to use this system. After login, the authorized user can perform administrative duties. If the authorized user is sitting at the system console, the user has the rights of the Console User.

**▼ How to Enable a User to Use Own Password to Assume a Role**

By default, users must type the role's password to assume a role. Perform this procedure to make assuming a role in Oracle Solaris similar to assuming a role in a Linux environment.

**Before You Begin**

You assume a role that includes the User Security rights profile. This role cannot be the role whose `roleauth` value you want to change.

- **Enable a user password to authenticate a role.**

  ```bash
  $ rolemod -K roleauth=user rolename
  ```

  To assume this role, the assigned users can now use their own password, not the password that was created specifically for the role.

**Example 9–29  Enabling a Role to Use the Assigned User’s Password When Using a Rights Profile**

In this example, the `root` role changes the value of `roleauth` for the role `secadmin` on the local system.
When a user who is assigned the Security Administrator rights profile wants to assume the role, the user is prompted for a password. In the following sequence, the role name is secadmin:

```bash
% su - secadmin
Password: 
$ /* You are now in a profile shell with administrative rights */
```

If the user has been assigned other roles, they use their own password to authenticate to those roles, too.

### Example 9–30  Changing the Value of roleauth for a Role in the LDAP Repository

In this example, the root role enables all users who can assume the role secadmin to use their own password when assuming a role. This capability is granted to these users for all systems that are managed by the LDAP server.

```bash
# rolemod -S ldap -K roleauth=user secadmin
```

### Troubleshooting

If roleauth=user is set for the role, the user password enables the authenticated role to access all rights that are assigned to that role. This keyword is search-dependent. For more information, see “Order of Search for Assigned Security Attributes” on page 197.

#### ▼ How to Change the root Role Into a User

An administrator might change root to a user when decommissioning a system that has been removed from the network. In this instance, logging into the system as root simplifies the cleanup.

### Before You Begin

You must become root. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1. **Remove the root role assignment from local users.**
   For example, remove the role assignment from two users.

   ```bash
   % su - root
   Password: a12@3#4$5%6^7
   # roles jdoe
   root
   # roles kdoe
   root
   ```
2. Change the root role into a user.
   
   `usermod -K type=normal root`

   Users who are currently in the root role remain so. Other users who have root access can su to root or log in to the system as the root user.

3. Verify the change.
   
   You can use one of the following commands.

   `getent user_attr root`

   If the type keyword is missing in the output or is equal to normal, the account is not a role.

   `userattr type root`

   If the output is empty or lists normal, the account is not a role.

---

### Example 9-31 Preventing the root Role From Being Used to Configure a System

In this example, site security policy requires that the root account be prevented from maintaining the system. The administrator has created and tested the roles which maintain the system. These roles include every security profile and the System Administrator rights profile. A trusted user has been assigned a role that can restore a backup. No role can change the audit flags for the system, a user, or a rights profile.

To prevent the root account from being used to maintain the system, the security administrator removes the root role assignment. Because the root account must be able to log in to the system in single-user mode, the account retains a password.

```
# usermod -K roles= jdoe
# userattr roles jdoe
```

### Example 9-32 Changing the root User Into the root Role

In this example, the root user turns the root user back into a role.

First, the root user changes the root account into a role and verifies the change.

```
# usermod -K type=role root
# getent user_attr root
root::::type=role;auths=solaris.*;profiles=All;audit_flags=lo\:no;
```
\[ \text{lock\_after\_retries=\text{no};min\_label=admin\_low;clearance=admin\_high} \]

Then, root assigns the root role to a local user.

\# usermod -R root jdoe

**Troubleshooting**  
In a desktop environment, you cannot directly log in as root when root is a role. A diagnostic message indicates that root is a role on your system.

If you do not have a local account that can assume the root role, create one. As root, log in to the system in single-user mode, create a local user account and password, and assign the root role to the new account. Then, log in as the new user and assume the root role.

---

### Using Privileges (Tasks)

Privileges can enable users to perform specific tasks with administrative rights. Privileges can also be used to limit users to just those tasks that they are permitted to perform.

The following task map points to step-by-step instructions for viewing, managing, and using privileges on your system.

<table>
<thead>
<tr>
<th>Task</th>
<th>Description</th>
<th>For Instructions</th>
</tr>
</thead>
<tbody>
<tr>
<td>View the defined privileges.</td>
<td>Lists the privileges and their definitions in Oracle Solaris.</td>
<td>“How to List the Privileges on the System” on page 186</td>
</tr>
<tr>
<td>View your privileges as a user in any shell.</td>
<td>Shows your directly assigned privileges. All of your processes run with these privileges.</td>
<td>“How to Determine the Privileges That You Have Been Directly Assigned” on page 187</td>
</tr>
<tr>
<td>View your privileged commands in a profile shell.</td>
<td>Shows the privileged commands that you can run through an assigned rights profile.</td>
<td>“How to Determine the Privileged Commands That You Can Run” on page 188</td>
</tr>
<tr>
<td>Limit attacker access to a system when an attack on an application is successful.</td>
<td>Protects a system from attacks by applying extended privilege policy to the NTP port.</td>
<td>“How to Apply Extended Privilege Policy to a Port” on page 192</td>
</tr>
<tr>
<td>Determine which privileges are in a process.</td>
<td>Lists the effective, inheritable, permitted, and limit privilege sets for a process.</td>
<td>“How to Determine the Privileges on a Process” on page 189</td>
</tr>
<tr>
<td>Determine which privileges are missing from a process.</td>
<td>Lists the privileges that a failed process requires to succeed.</td>
<td>“How to Determine Which Privileges a Program Requires” on page 191</td>
</tr>
<tr>
<td>Limit attacker access to a system when an attack on an application is successful.</td>
<td>Creates an extended policy for the NTP service.</td>
<td>“How to Apply Extended Privilege Policy to a Port” on page 192</td>
</tr>
</tbody>
</table>
### Using Privileges (Tasks)

<table>
<thead>
<tr>
<th>Task</th>
<th>Description</th>
<th>For Instructions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add privileges to a command.</td>
<td>Adds privileges to a command in a rights profile. Users or roles can be assigned the rights profile. The users can then run the command with the assigned privileges in a profile shell.</td>
<td>Example 9–18</td>
</tr>
<tr>
<td>Assign privileges to a user or role.</td>
<td>Expands a user’s or role’s inheritable set of privileges. Use this procedure with caution.</td>
<td>Example 9–9</td>
</tr>
<tr>
<td>Restrict a user's privileges.</td>
<td>Limits the user’s basic set of privileges. Use this procedure with caution.</td>
<td>Example 9–16</td>
</tr>
<tr>
<td>Run a privileged shell script.</td>
<td>Adds privilege to a shell script and to the commands in the shell script. Then, runs the script in a profile shell.</td>
<td>&quot;How to Run a Shell Script With Privileged Commands&quot; on page 193</td>
</tr>
</tbody>
</table>

### How to List the Privileges on the System

The following procedure shows how to view the privilege names and definitions.

- **In a terminal window, you can view privileges online.**
  - List all privileges by viewing the `privileges(5)` man page.
    ```bash
    % man privileges
    Standards, Environments, and Macros privileges(5)
    NAME
    privileges - process privilege model
    ...
    The defined privileges are:
    PRIV_CONTRACT_EVENT
    
    Allow a process to request reliable delivery of events to an event endpoint.
    
    Allow a process to include events in the critical event set term of a template which could be generated in volume by the user.
    ...
    
    This privilege format is used by developers.
    
    List the privileges by using the `ppriv` command.
    ```
    ```bash
    % ppriv -lv | more
    contract_event
    Allows a process to request critical events without limitation.
    Allows a process to request reliable delivery of all events on any event queue.
    ...
    win_upgrade_sl
    Allows a process to set the sensitivity label of a window
    ```
resource to a sensitivity label that dominates the existing sensitivity label. This privilege is interpreted only if the system is configured with Trusted Extensions.

This privilege format is used to assign privileges to users and roles with the useradd, roleadd, usermod, and rolemod commands, and to rights profiles with the profiles command.

▼ How to Determine the Privileges That You Have Been Directly Assigned

The following procedures show how to determine if you have been directly assigned privileges.

Caution – Inappropriate use of directly assigned privileges can result in unintentional breaches of security. For a discussion, see “Security Considerations When Directly Assigning Security Attributes” on page 137.

1 List the privileges that your processes can use.
   See “How to Determine the Privileges on a Process” on page 189 for the procedure.

2 Invoke actions and run commands in any shell.
   The privileges that are listed in the effective set are in effect throughout your session. If you have been directly assigned privileges in addition to the basic set, the privileges are listed in the effective set.

Example 9–33  Determining Your Directly Assigned Privileges

In this example, the user is directly assigned the proc_clock_highres privilege, so the privilege is available in every process that the user owns.

% ppriv -v $$
1800: pfksh
flags = <none>
E: file_link.any,...,proc_clock_highres,proc_session
I: file_link.any,...,proc_clock_highres,proc_session
P: file_link.any,...,proc_clock_highres,proc_session
L: cpc_cpu,dtrace_kernel,dtrace_proc,dtrace_user,...,sys_time
% ppriv -vl proc_clock_highres
   Allows a process to use high resolution timers.

Example 9–34  Determining a Role’s Directly Assigned Privileges

In the following example, the role realtime has been directly assigned privileges to handle date and time programs.
su - realtime
Password: <Type realtime password>
$ ppriv -v $$
1600: pfksh
flags = <none>
   E: file_link_any,...,proc_clock_highres,proc_session,sys_time
   I: file_link_any,...,proc_clock_highres,proc_session,sys_time
   P: file_link_any,...,proc_clock_highres,proc_session,sys_time
   L: cpc_cpu,dtrace_kernel,dtrace_proc,dtrace_user,...,sys_time

▼ How to Determine the Privileged Commands That You Can Run

Typically, users and roles get access to privileged commands through a rights profile. Commands in a rights profile must be executed in a profile shell.

1 Determine the rights profiles that you have been assigned.
In the following example, the user is assigned several rights profiles. The system reads the rights profiles and their contents in order. For all attributes except authorizations, the first explicitly set attribute value is the one that is used. For more information, see “Order of Search for Assigned Security Attributes” on page 197.

   % profiles
   Audit Review
   Console User
   Suspend To RAM
   Suspend To Disk
   Brightness
   CPU Power Management
   Network Autoconf
   Desktop Print Management
   Network Wifi Info
   Desktop Removable Media User
   Basic Solaris User
   All

2 Determine your rights from the Audit Review profile.

   profiles -l
   Audit Review
   solaris.audit.read
   /usr/sbin/auditreduce euid=0
   /usr/sbin/auditstat euid=0
   /usr/sbin/praudit euid=0

   The Audit Review rights profile enables you to run the auditreduce, auditstat, and praudit commands with the effective UID of 0, and assigns you the solaris.audit.read authorization.
Determining the Privileged Commands of a Role

In this example, a user assumes an assigned role and lists the commands that are included in one of the rights profiles.

```
% roles
devadmin
% su - devadmin
Password: Type devadmin password
$ profiles -l
```

```
Device Security
/usr/bin/kbd uid=0;gid=sys
/usr/sbin/add_allocatable euid=0
/usr/sbin/add_drv uid=0
/usr/sbin/devfsadm uid=0
/usr/sbin/eeprom uid=0
/usr/sbin/list_devices euid=0
/usr/sbin/rem_drv uid=0
/usr/sbin/remove_allocatable euid=0
/usr/sbin/strace euid=0
/usr/sbin/update_drv uid=0
```

Running the Privileged Commands in Your Role

In the following example, the admin role can change the permissions on the `useful.script` file.

```
% whoami
jdoe
% ls -l useful.script
-rwxr-xr-- 1 elsee eng 262 Apr 2 10:52 useful.script
% chgrp admin useful.script
chgrp: useful.script: Not owner
% su - admin
Password: <Type admin password>
$ chgrp admin useful.script
$ chown admin useful.script
$ ls -l useful.script
-rwxr-xr-- 1 admin admin 262 Apr 2 10:53 useful.script
```

How to Determine the Privileges on a Process

This procedure shows how to determine which privileges are available to your processes. The listing does not include privileges that have been assigned to particular commands.

- **List the privileges that are available to your shell’s process.**
  ```
  % ppriv [-v] pid
  pid  Is the process number. Use a double dollar sign ($$) to pass the process number of the parent shell to the command.
  ```
-v Provides a verbose listing of the privilege names.

Example 9–37 Determining the Privileges in Your Current Shell

In the following example, the privileges in the parent process of the user’s shell process are listed. In the second example, the full names of the privileges are listed. The single letters in the output refer to the following privilege sets:

E Is the effective privilege set.
I Is the inheritable privilege set.
P Is the permitted privilege set.
L Is the limit privilege set.

```bash
% ppriv $$
1200: -csh
flags = <none>
  E: basic
  I: basic
  P: basic
  L: all
% ppriv -v $$
1200: -csh
flags = <none>
  E: file_link_any,net_access,proc_exec,proc_fork,proc_info,proc_session
  I: file_link_any,net_access,proc_exec,proc_fork,proc_info,proc_session
  P: file_link_any,net_access,proc_exec,proc_fork,proc_info,proc_session
  L: cpc_cpu,dtrace_kernel,dtrace_proc,dtrace_user,...,sys_time
```

Example 9–38 Determining the Privileges of a Role That You Can Assume

In the following example, the role sysadmin has no directly assigned privileges.

```bash
% su - sysadmin
Password: <Type sysadmin password>
$ /usr/bin/whoami
sysadmin
$ ppriv -v $$
1400: pfksh
flags = <none>
  E: file_link_any,file_read,file_write,net_access,proc_exec,proc_fork,
      proc_info,proc_session
  I: file_link_any,file_read,file_write,net_access,proc_exec,proc_fork,
      proc_info,proc_session
  P: file_link_any,file_read,file_write,net_access,proc_exec,proc_fork,
      proc_info,proc_session
  L: cpc_cpu,dtrace_kernel,dtrace_proc,dtrace_user,...,win_upgrade_sl
```
**How to Determine Which Privileges a Program Requires**

**Before You Begin**
The command or process must fail for this debugging procedure to work.

1. **Type the command that is failing as an argument to the `ppriv` debugging command.**
   
   ```
   % ppriv -eD touch /etc/acct/yearly
   touch[5245]: missing privilege "file_dac_write"
   (euid = 130, syscall = 224) needed at zfs_zaccess+0x258
   touch: cannot create /etc/acct/yearly: Permission denied
   ```

2. **Determine which system call is failing by finding the syscall number in the `/etc/name_to_sysnum` file.**
   
   ```
   % grep 224 /etc/name_to_sysnum
   creat64 224
   ```

**Example 9–39**  **Using the `truss` Command to Examine Privilege Use**

The `truss` command can debug privilege use in a regular shell. For example, the following command debugs the failing `touch` process:

```
% truss -t creat touch /etc/acct/yearly
creat64("/etc/acct/yearly", 0666)  
   Err#13 EACCES [file_dac_write]
touch: /etc/acct/yearly cannot create
```

The extended `/proc` interfaces report the missing privilege after the error code in `truss` output.

**Example 9–40**  **Using the `ppriv` Command to Examine Privilege Use in a Profile Shell**

In this example, the `jdoe` user can assume the role `objadmin`. The `objadmin` role includes the Object Access Management rights profile. This rights profile allows the `objadmin` role to change permissions on files that `objadmin` does not own.

In the following excerpt, `jdoe` fails to change the permissions on the `useful.script` file:

```
jdoe% ls -l useful.script
-rw-r--r-- 1 aloe staff 2303 Apr 10 10:10 useful.script
jdoe% chown objadmin useful.script
chown: useful.script: Not owner
jdoe% ppriv -eD chown objadmin useful.script
chown[11444]: missing privilege "file_chown"
   (euid = 130, syscall = 16) needed at zfs_zaccess+0x258
chown: useful.script: Not owner
```

When `jdoe` assumes the `objadmin` role, the permissions on the file are changed:
Example 9-41 Changing a File Owned by the root User

This example illustrates the protections against privilege escalation. For a discussion, see "Prevention of Privilege Escalation" on page 206. The file is owned by the root user. The less powerful role, objadmin role needs all privileges to change the file’s ownership, so the operation fails.

How to Apply Extended Privilege Policy to a Port

The service for the Network Time Protocol (NTP) uses the privileged port 123 for udp traffic. This procedure protects other ports from being accessed by a malicious user who might gain the privileges that are assigned to this port.

1. **Read the default service manifest entry for the port.**

The removed privileges prevent the service from signaling or observing any other processes, and from creating hard links as a way of renaming files.
That is, the process that is started by the service is only able to bind to the specific port 123, not to any of the other privileged ports. If a hacker could exploit the service to start another process, then the child process would also not be able to bind to any other privileged port.

2 Limit the net_privaddr privilege to this port only.
   The extended privilege policy that is highlighted in the following excerpt prevents access from this service to other privileged ports:
   
   `privileges='basic,!file_link_any,!proc_info,!proc_session,
   {net_privaddr}:123/udp,proc_lock_memory,sys_time'`

▼ How to Run a Shell Script With Privileged Commands

When you create a shell script that runs commands that require privilege, the appropriate rights profile must contain the commands with privileges assigned to them.

Before You Begin

You must assume the root role. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1 Start the script with /bin/pfsh, or any other profile shell, on the first line.

```
#!/bin/pfsh
# Copyright (c) 2012 by Oracle
```

2 Determine the privileges that the commands in the script need.

```
% ppriv -eD script-full-path
```

3 Become an administrator with the required security attributes.

For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

4 Create or modify a rights profile for the script.

Add the shell script, and the commands in the shell script, with their required security attributes to the rights profile. For the steps, see “How to Create a Rights Profile” on page 167.

5 Add the rights profile to a role and assign the role to a user.

To run the script, the user assumes the role and runs the script in the role’s profile shell.

- To add the rights profile to a role, see “How to Change the Security Attributes of a Role” on page 179.
- To assign the role to a user, see “How to Change the Security Attributes of a User” on page 155.
This chapter provides reference material about RBAC and privileges:

- “Rights Profiles” on page 195
- “Order of Search for Assigned Security Attributes” on page 197
- “Authorizations” on page 198
- “RBAC Databases” on page 199
- “RBAC Commands” on page 202
- “Administrative Commands for Handling Privileges” on page 204
- “Files With Privilege Information” on page 205
- “Privileges and Auditing” on page 205
- “Prevention of Privilege Escalation” on page 206
- “Legacy Applications and the Privilege Model” on page 207

For information about using RBAC, see Chapter 9, “Using Role-Based Access Control (Tasks).” For overview information, see “Role-Based Access Control (Overview)” on page 127.

To use privileges, see “Using Privileges (Tasks)” on page 185. For overview information, see “Privileges (Overview)” on page 138.

**Rights Profiles**

This section describes some typical rights profiles. Rights profiles are convenient collections of authorizations and other security attributes, commands with security attributes, and supplementary rights profiles. Oracle Solaris provides many rights profiles. If they are not sufficient for your needs, you can modify existing ones and create new ones.

Rights profiles must be assigned in order, from most to least powerful. For more information, see “Order of Search for Assigned Security Attributes” on page 197.
- **System Administrator rights profile** – Provides a profile that can do most tasks that are not connected with security. This profile includes several other profiles to create a powerful role. Note that the All rights profile is assigned at the end of the list of supplementary rights profiles. The `profiles` command displays the contents of the profile.

  ```bash
  % profiles -p "System Administrator" info
  ```

- **Operator rights profile** – Provides limited capabilities to manage files and offline media. This profile includes supplementary rights profiles to create a simple role. The `profiles` command displays the contents of the profile.

  ```bash
  % profiles -p Operator info
  ```

- **Printer Management rights profile** – Provides a limited number of commands and authorizations to handle printing. This profile is one of several profiles that cover a single area of administration. The `profiles` command displays the contents of the profile.

  ```bash
  % profiles -p "Printer Management" info
  ```

- **Basic Solaris User rights profile** – Enables users to use the system within the bounds of security policy. This profile is listed by default in the `policy.conf` file. Note that the convenience that is offered by the Basic Solaris User rights profile must be balanced against site security requirements. Sites that need stricter security might prefer to remove this profile from the `policy.conf` file or assign the Stop rights profile. The `profiles` command displays the contents of the profile.

  ```bash
  % profiles -p "Basic Solaris User" info
  ```

- **Console User rights profile** – For the workstation owner, provides access to authorizations, commands, and actions for the person who is seated at the computer. The `profiles` command displays the contents of the profile.

  ```bash
  % profiles -p "Console User" info
  ```

- **All rights profile** – For roles, provides access to commands that do not have security attributes. This profile can be appropriate for users with limited rights. The `profiles` command displays the contents of the profile.

  ```bash
  % profiles -p All info
  ```

- **Stop rights profile** – Is a special rights profile that stops the evaluation of further profiles. This profile prevents the evaluation of the `AUTHS_GRANTED`, `PROFS_GRANTED`, and `CONSOLE_USER` variables in the `policy.conf` file. With this profile, you can provide roles and users with a restricted profile shell.

  ```bash
  % profiles -p All info
  ```

**Note** – The Stop profile affects privilege assignment indirectly. Rights profiles that are listed after the Stop profile are not evaluated. Therefore, the commands with privileges in those profiles are not in effect. To use this profile, see “How to Restrict an Administrator to Explicitly Assigned Rights” on page 181.
% profiles -p Stop info

Each rights profile has an associated help file. The help files are in HTML and are customizable. The files reside in the /usr/lib/help/profiles/locale/C directory.

### Viewing the Contents of Rights Profiles

You have three views into the contents of rights profiles.

- The `getent` command enables you to view the contents of all of the rights profiles on the system. For sample output, see “How to View All Defined Security Attributes” on page 150.
- The `profiles -p "Profile Name"` info command enables you to view the contents of a specific rights profile.
- The `profiles -l account-name` command enables you to view the contents of the rights profiles that are assigned to a specific user or role.

For more information, see the `getent(1M)` and `profiles(1)` man pages.

### Order of Search for Assigned Security Attributes

A user or role can be assigned security attributes directly or through a rights profile. The order of search affects which security attribute value is used. The value of the first found instance of the attribute is used.

**Note** – The order of authorizations is not important. Authorizations are cumulative.

When a user logs in, security attributes are assigned in the following search order:

- **security attributes** that are assigned to the user with the `useradd` and `usermod` commands. For a list, see "user_attr Database" on page 200.
- **rights profiles** that are assigned to the user with the `useradd` and `usermod` commands. These assignments are searched in order.
  
  The order is first profile in the list, then its list of rights profiles, second profile in the list, then its list of profiles, and so on. The first instance of a value is the one that the system uses, except for `auths` values, which are cumulative. The attributes in rights profiles include all the security attributes for users, plus supplementary profiles. For a list, see "user_attr Database" on page 200.
- **Console User rights profile** value. For a description, see “Rights Profiles” on page 195.
- If the **Stop rights profile** is assigned, the evaluation of security attributes stops. No attributes are assigned after the Stop profile is assigned. The Stop profile is evaluated after the Console User rights profile and before the other security attributes in the `policy.conf` file, including `AUTHS_GRANTED`. For a description, see "Rights Profiles" on page 195.
Authorizations

An RBAC authorization is a discrete right that can be granted to a role or a user. Authorizations are checked by RBAC-compliant applications before a user gets access to the application or specific operations within the application.

Authorizations are user-level, therefore extensible. You can write a program that requires authorization, add the authorizations to your system, create a rights profile for these authorizations, and assign the rights profile to users or roles who are allowed to use the program.

Authorization Naming Conventions

An authorization has a name that is used internally. For example, solaris.system.date is the name of an authorization. An authorization has a short description, which appears in the graphical user interfaces (GUIs). For example, Set Date & Time is the description of the solaris.system.date authorization.

By convention, authorization names consist of the reverse order of the Internet name of the supplier, the subject area, any subareas, and the function. The parts of the authorization name are separated by dots. An example would be com.xyzcorp.device.access. Exceptions to this convention are the authorizations from Oracle, which use the prefix solaris instead of an Internet name. The naming convention enables administrators to apply authorizations in a hierarchical fashion. A wildcard (*) can represent any strings to the right of a dot.

As an example of how authorizations are used, consider the following: The Network Link Security rights profile has the solaris.network.link.security authorization only, while the Network Security rights profile has the Network Link Security profile as a supplementary profile, plus the solaris.network.* and solaris.smf.manage.ssh authorizations.

Delegation Authority in Authorizations

An authorization that ends with the suffix delegate enables a user or a role to delegate to other users any assigned authorizations that begin with the same prefix.
The **solaris auth.delegate** authorization enables a user or a role to delegate to other users any authorizations that these users or roles are assigned.

For example, a role with the **solaris auth.delegate** and **solaris.network.wifi.wep** authorizations can delegate the **solaris.network.wifi.wep** authorization to another user or role.

### RBAC Databases

The following databases store the data for the RBAC elements:

- **Extended user attributes database** *(user_attr)* – Associates users and roles with authorizations, privileges, keywords, and rights profiles
- **Rights profile attributes database** *(prof_attr)* – Defines rights profiles, lists the profiles’ assigned authorizations, privileges, and keywords, and identifies the associated help file
- **Authorization attributes database** *(auth_attr)* – Defines authorizations and their attributes, and identifies the associated help file
- **Execution attributes database** *(exec_attr)* – Identifies the commands with security attributes that are assigned to specific rights profiles

The **policy.conf** database contains authorizations, privileges, and rights profiles that are applied to all users. For more information, see “**policy.conf File**” on page 201.

### RBAC Databases and the Naming Services

The name service scope of the RBAC databases is defined in the SMF service for the naming service switch, *svc:/system/name-service/switch*. The properties in this service for the RBAC databases are **auth_attr**, **password**, and **prof_attr**. The **password** property sets the naming service precedence for the **passwd** and **user_attr** databases. The **prof_attr** property sets the naming service precedence for the **prof_attr** and **exec_attr** databases.

In the following output, the **auth_attr**, **password**, and **prof_attr** entries are not listed. Therefore, the RBAC databases are using the **files** naming service.

```bash
# svccfg -s name-service/switch listprop config
config  application
config/value_authorization astring  solaris.smf.value.name-service.switch
config/default astring  files
config/host astring  "files ldap dns"
config/printer astring  "user files ldap"
```
user_attr Database

The user_attr database contains user and role information that supplements the passwd and shadow databases.

The following security attributes can be set by using the roleadd, rolemod, useradd, usermod, and profiles commands:

- For a user, the roles keyword assigns one or more defined roles.
- For a role, the user value to the roleauth keyword enables the role to authenticate with the user password rather than with the role password. By default, the value is role.
- For a user or role, the following attributes can be set:
  - auditflags keyword - Modifies the audit mask. For reference, see the auditflags(5) man page.
  - auths keyword - Assigns authorizations. For reference, see the auths(1) man page.
  - defaultpriv keyword - Adds privileges or removes them from the default basic set of privileges. For reference, see “How Privileges Are Implemented” on page 142.
  - limitpriv keyword - Adds privileges or removes them from the default limit set of privileges. For reference, see “How Privileges Are Implemented” on page 142.
  - project keyword - Adds a default project. For reference, see the project(4) man page.
  - lock_after_retries keyword - If the value is yes, the system is locked after the number of retries exceeds the number that is allowed in the /etc/default/login file.
  - profiles keyword - Assigns rights profiles.

For more information, see the user_attr(4) man page. To view the contents of this database, use the getent user_attr command. For more information, see the getent(1M) man page and “How to View All Defined Security Attributes” on page 150.

auth_attr Database

All authorizations are stored in the auth_attr database. Authorizations can be assigned to users, to roles, or to rights profiles. The preferred method is to place authorizations in a rights profile, to include the profile in a role’s list of profiles, and then to assign the role to a user.

To view the contents of this database, use the getent auth_attr command. For more information, see the getent(1M) man page and “How to View All Defined Security Attributes” on page 150.
prof_attr Database

The prof_attr database stores the name, description, help file location, privileges, and authorizations that are assigned to rights profiles. The commands and security attributes that are assigned to rights profiles are stored in the exec_attr database. For more information, see “exec_attr Database” on page 201.

For more information, see the prof_attr(4) man page. To view the contents of this database, use the getent exec_attr command. For more information, see the getent(1M) man page and “How to View All Defined Security Attributes” on page 150.

exec_attr Database

The exec_attr database defines commands that require security attributes to succeed. The commands are part of a rights profile. A command with its security attributes can be run by roles or users to whom the profile is assigned.

For more information, see the exec_attr(4) man page. To view the contents of this database, use the getent command. For more information, see the getent(1M) man page and “How to View All Defined Security Attributes” on page 150.

policy.conf File

The policy.conf file provides a way of granting specific rights profiles, specific authorizations, and specific privileges to all users. The relevant entries in the file consist of key=value pairs:

- AUTHS_GRANTED=authorizations – Refers to one or more authorizations.
- PROFS_GRANTED=rights profiles – Refers to one or more rights profiles.
- CONSOLE_USER=Console User – Refers to the Console User rights profile. This profile is delivered with a convenient set of authorizations for the console user. You can customize this profile. To view the profile contents, see “Rights Profiles” on page 195.
- PRIV_DEFAULT=privileges – Refers to one or more privileges.
- PRIV_LIMIT=privileges – Refers to all privileges.

The following example shows some typical values from a policy.conf database:

```bash
# grep AUTHS /etc/security/policy
AUTHS_GRANTED=solaris.device.cdrw

# grep PROFS /etc/security/policy
PROFS_GRANTED=Basic Solaris User

# grep PRIV /etc/security/policy
#PRIV_DEFAULT=basic
#PRIV_LIMIT=all
```
For more information about privileges, see "Privileges (Overview)" on page 138.

**RBAC Commands**

This section lists commands that are used to administer RBAC. Also provided is a table of commands whose access can be controlled by authorizations.

**Commands That Manage RBAC**

The following commands retrieve and set RBAC information.

<table>
<thead>
<tr>
<th>TABLE 10-1 RBAC Administration Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Man Page for Command</strong></td>
</tr>
<tr>
<td>auths(1)</td>
</tr>
<tr>
<td>getent(1M)</td>
</tr>
<tr>
<td>nsccd(1M)</td>
</tr>
<tr>
<td>pam_roles(5)</td>
</tr>
<tr>
<td>pfedit(1M)</td>
</tr>
<tr>
<td>pfexec(1)</td>
</tr>
<tr>
<td>policy.conf(4)</td>
</tr>
<tr>
<td>profiles(1)</td>
</tr>
<tr>
<td>roles(1)</td>
</tr>
<tr>
<td>roleadd(1M)</td>
</tr>
<tr>
<td>roleadd(1M)</td>
</tr>
<tr>
<td>rolemod(1M)</td>
</tr>
<tr>
<td>userattr(1)</td>
</tr>
<tr>
<td>useradd(1M)</td>
</tr>
<tr>
<td>userdel(1M)</td>
</tr>
</tbody>
</table>
TABLE 10–1  RBAC Administration Commands  (Continued)

<table>
<thead>
<tr>
<th>Man Page for Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>usermod(1M)</td>
<td>Modifies a user's account properties on the system.</td>
</tr>
</tbody>
</table>

Selected Commands That Require Authorizations

The following table provides examples of how authorizations are used to limit command options on an Oracle Solaris system. For more discussion of authorizations, see “Authorizations” on page 198.

TABLE 10–2  Commands and Associated Authorizations

<table>
<thead>
<tr>
<th>Man Page for Command</th>
<th>Authorization Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>at(1)</td>
<td>solaris.jobs.user required for all options (when neither at.allow nor at.deny files exist)</td>
</tr>
<tr>
<td>atq(1)</td>
<td>solaris.jobs.admin required for all options</td>
</tr>
<tr>
<td>cdrw(1)</td>
<td>solaris.device.cdrw required for all options, and is granted by default in the policy.conf file</td>
</tr>
<tr>
<td>crontab(1)</td>
<td>solaris.jobs.user required for the option to submit a job (when neither crontab.allow nor crontab.deny files exist)</td>
</tr>
<tr>
<td></td>
<td>solaris.jobs.admin required for the options to list or modify other users' crontab files</td>
</tr>
<tr>
<td>allocate(1)</td>
<td>solaris.device.allocate (or other authorization as specified in deviceAllocate file) required to allocate a device</td>
</tr>
<tr>
<td></td>
<td>solaris.device.revoke (or other authorization as specified in deviceAllocate file) required to allocate a device to another user (-F option)</td>
</tr>
<tr>
<td>deallocate(1)</td>
<td>solaris.device.allocate (or other authorization as specified in deviceAllocate file) required to deallocate another user's device</td>
</tr>
<tr>
<td></td>
<td>solaris.device.revoke (or other authorization as specified in deviceAllocate) required to force deallocation of the specified device (-F option) or all devices (-I option)</td>
</tr>
<tr>
<td>list_devices(1)</td>
<td>solaris.device.revoke required to list another user's devices (-U option)</td>
</tr>
<tr>
<td>roleadd(1M)</td>
<td>solaris.user.manage required to create a role. solaris.account.activate required to set the initial password. solaris.account.setpolicy required to set password policy, such as account locking and password aging.</td>
</tr>
<tr>
<td>roledel(1M)</td>
<td>solaris.passwd.assign authorization required to delete the password.</td>
</tr>
</tbody>
</table>
TABLE 10–2 Commands and Associated Authorizations

<table>
<thead>
<tr>
<th>Command</th>
<th>Authorization Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>rolemod(1M)</td>
<td>solaris.passwd.assign authorization required to change the password.</td>
</tr>
<tr>
<td></td>
<td>solaris.account.setpolicy required to change password policy, such as account locking and password aging.</td>
</tr>
<tr>
<td>sendmail(1M)</td>
<td>solaris.mail required to access mail subsystem functions; solaris.mail.mailq required to view mail queue</td>
</tr>
<tr>
<td>useradd(1M)</td>
<td>solaris.user.manage required to create a user. solaris.account.activate required to set the initial password. solaris.account.setpolicy required to set password policy, such as account locking and password aging.</td>
</tr>
<tr>
<td>userdel(1M)</td>
<td>solaris.passwd.assign authorization required to delete the password.</td>
</tr>
<tr>
<td>usermod(1M)</td>
<td>solaris.passwd.assign authorization required to change the password.</td>
</tr>
<tr>
<td></td>
<td>solaris.account.setpolicy required to change password policy, such as account locking and password aging.</td>
</tr>
</tbody>
</table>

Privileges

Privileges restrict processes are implemented in the kernel, and can restrict processes at the command, user, role, or system level.

Administrative Commands for Handling Privileges

The following table lists the commands that are available to handle privileges.

<table>
<thead>
<tr>
<th>Purpose</th>
<th>Command</th>
<th>ManPage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Examine process privileges</td>
<td>ppriv -v pid</td>
<td>ppriv(1)</td>
</tr>
<tr>
<td>Set process privileges</td>
<td>ppriv -s spec</td>
<td></td>
</tr>
<tr>
<td>List the privileges on the system</td>
<td>ppriv -l</td>
<td></td>
</tr>
<tr>
<td>List a privilege and its description</td>
<td>ppriv -lv priv</td>
<td></td>
</tr>
<tr>
<td>Add extended privilege policy to a UID, process, or port.</td>
<td>ppriv -r rule</td>
<td>Use -X to remove the policy. privileges(5)</td>
</tr>
<tr>
<td>List extended privilege policy on a UID, process, or port.</td>
<td>ppriv -lv extended-policy</td>
<td>ppriv(1)</td>
</tr>
<tr>
<td>Debug privilege failure</td>
<td>ppriv -e0 failed-operation</td>
<td></td>
</tr>
<tr>
<td>Assign privileges to a new user</td>
<td>useradd</td>
<td>useradd(1M)</td>
</tr>
</tbody>
</table>
TABLE 10–3 Commands for Handling Privilege (Continued)

<table>
<thead>
<tr>
<th>Purpose</th>
<th>Command</th>
<th>ManPage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add privileges to an existing user</td>
<td>usermod</td>
<td>usermod(1M)</td>
</tr>
<tr>
<td>Assign privileges to a rights profile</td>
<td>profiles</td>
<td>profiles(1)</td>
</tr>
<tr>
<td>Assign privileges to a new role</td>
<td>roleadd</td>
<td>roleadd(1M)</td>
</tr>
<tr>
<td>Add privileges to an existing role</td>
<td>rolemod</td>
<td>rolemod(1M)</td>
</tr>
<tr>
<td>View device policy</td>
<td>getdevpolicy</td>
<td>getdevpolicy(1M)</td>
</tr>
<tr>
<td>Set device policy</td>
<td>devfsadm</td>
<td>devfsadm(1M)</td>
</tr>
<tr>
<td>Update device policy on open devices</td>
<td>update_drv -p policy driver</td>
<td>update_drv(1M)</td>
</tr>
<tr>
<td>Add device policy to a device</td>
<td>add_drv -p policy driver</td>
<td>add_drv(1M)</td>
</tr>
</tbody>
</table>

Files With Privilege Information

The following files contain information about privileges.

TABLE 10–4 Files That Contain Privilege Information

<table>
<thead>
<tr>
<th>File and Man Page</th>
<th>Privilege Information</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>/etc/security/policy.conf</td>
<td>PRIV_DEFAULT</td>
<td>Inheritable set of privileges for the system</td>
</tr>
<tr>
<td>syslog.conf</td>
<td>PRIV_LIMIT</td>
<td>Limit set of privileges for the system</td>
</tr>
<tr>
<td>syslog.conf(4)</td>
<td>System log file for debug messages</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Path set in priv. debug entry</td>
<td></td>
</tr>
</tbody>
</table>

Privileges and Auditing

Privilege use can be audited. Any time that a process uses a privilege, the use of privilege is recorded in the audit trail in the upriv audit token. When privilege names are part of the record, their textual representation is used. The following audit events record use of privilege:

- **AUE_SETPPRIV audit event** – The event generates an audit record when a privilege set is changed. The AUE_SETPPRIV audit event is in the pm class.
- **AUE_MODALLOCPRIV audit event** – The audit event generates an audit record when a privilege is added from outside the kernel. The AUE_MODALLOCPRIV audit event is in the ad class.
- **AUE_MODDEVLCPY audit event** – The audit event generates an audit record when the device policy is changed. The AUE_MODDEVLCPY audit event is in the ad class.
- **AUE_PFEXEC audit event** — The audit event generates an audit record when a call is made to `execve()` with `pfexec()` enabled. The AUE_PFEXEC audit event is in the as, ex, ps, and ua audit classes. The names of the privileges are included in the audit record.

The successful use of privileges that are in the basic set is not audited. The attempt to use a basic privilege that has been removed from a user’s basic set is audited.

## Prevention of Privilege Escalation

The kernel prevents *privilege escalation*. Privilege escalation is when a privilege enables a process to do more than the process should be able to do. To prevent a process from gaining more privileges than the process should have, vulnerable system modifications require the full set of privileges. For example, a file or process that is owned by `root` (UID=0) can only be changed by a process with the full set of privileges. The `root` account does not require privileges to change a file that `root` owns. However, a non-root user must have all privileges in order to change a file that is owned by `root`.

Similarly, operations that provide access to devices require all privileges in the effective set.

The `file_chown_self` and `proc_owner` privileges are subject to privilege escalation. The `file_chown_self` privilege allows a process to give away its files. The `proc_owner` privilege allows a process to inspect processes that the process does not own.

The `file_chown_self` privilege is limited by the `r stomach` system variable. When the `r stomach` variable is set to zero, the `file_chown_self` privilege is removed from the initial inheritable set of the system and of all users. For more information about the `r stomach` system variable, see the `chown(1)` man page.

The `file_chown_self` privilege is most safely assigned to a particular command, placed in a profile, and assigned to a role for use in a profile shell.

The `proc_owner` privilege is not sufficient to switch a process UID to 0. To switch a process from any UID to UID=0 requires all privileges. Because the `proc_owner` privilege gives unrestricted read access to all files on the system, the privilege is most safely assigned to a particular command, placed in a profile, and assigned to a role for use in a profile shell.

---

**Caution** — A user’s account can be modified to include the `file_chown_self` privilege or the `proc_owner` privilege in the user’s initial inheritable set. You should have overriding security reasons for placing such powerful privileges in the inheritable set of privileges for any user, role, or system.

For details of how privilege escalation is prevented for devices, see “Privileges and Devices” on page 145.
Legacy Applications and the Privilege Model

To accommodate legacy applications, the implementation of privileges works with both the superuser and the privilege models. The kernel automatically tracks the PRIV_AWARE flag, which indicates that a program has been designed to work with privileges. Consider a child process that is not aware of privileges. Any privileges that were inherited from the parent process are available in the child’s permitted and effective sets. If the child process sets a UID to 0, the child process might not have full superuser capabilities. The process’s effective and permitted sets are restricted to those privileges in the child’s limit set. Thus, the limit set of a privilege-aware process restricts the root privileges of child processes that are not aware of privileges.
Cryptographic Services

This section describes the centralized cryptographic and public key technology features that Oracle Solaris provides.

- Chapter 11, “Cryptographic Framework (Overview)”
- Chapter 12, “Cryptographic Framework (Tasks)”
- Chapter 13, “Key Management Framework”
This chapter describes the Cryptographic Framework feature of Oracle Solaris, and covers the following topics:

- “Introduction to the Cryptographic Framework” on page 211
- “Terminology in the Cryptographic Framework” on page 213
- “Scope of the Cryptographic Framework” on page 215
- “Cryptographic Services and Zones” on page 217
- “Cryptographic Framework and FIPS-140” on page 217
- “Cryptographic Framework and the SPARC T-Series Servers in This Release” on page 217

To administer and use the Cryptographic Framework, see Chapter 12, “Cryptographic Framework (Tasks).”

**Introduction to the Cryptographic Framework**

The Cryptographic Framework provides a common store of algorithms and PKCS #11 libraries to handle cryptographic requirements. The PKCS #11 libraries are implemented according to the following standard: RSA Security Inc. PKCS #11 Cryptographic Token Interface (Cryptoki).
At the kernel level, the framework currently handles cryptographic requirements for Kerberos and IPsec. User-level consumers include `libsasl` and IKE. The kernel SSL (`kssl`) proxy uses the Cryptographic Framework. For more information, see "SSL Kernel Proxy Encrypts Web Server Communications" in *Securing the Network in Oracle Solaris 11.1* and the `ksslcfg(1M)` man page.
Export law in the United States requires that the use of open cryptographic interfaces be licensed. The Cryptographic Framework satisfies the current law by requiring that kernel cryptographic providers and PKCS #11 cryptographic providers be signed. For further discussion, see "Binary Signatures for Third-Party Software" on page 216.

The framework enables providers of cryptographic services to have their services used by many consumers in Oracle Solaris. Another name for providers is plugins. The framework allows three types of plugins:

- **User-level plugins** – Shared objects that provide services by using PKCS #11 libraries, such as pkcs11_softtoken.so.1.
- **Kernel-level plugins** – Kernel modules that provide implementations of cryptographic algorithms in software, such as AES.
  
  Many of the algorithms in the framework are optimized for x86 with the SSE2 instruction set and for SPARC hardware.
- **Hardware plugins** – Device drivers and their associated hardware accelerators. The Niagara chips, the ncp and n2cp device drivers, are one example. A hardware accelerator offloads expensive cryptographic functions from the operating system. The Sun Crypto Accelerator 6000 board is one example.

The framework implements a standard interface, the PKCS #11, v2.20 amendment 3 library, for user-level providers. The library can be used by third-party applications to reach providers. Third parties can also add signed libraries, signed kernel algorithm modules, and signed device drivers to the framework. These plugins are added when the pkgadd utility installs the third-party software. For a diagram of the major components of the framework, see Chapter 8, "Introduction to the Oracle Solaris Cryptographic Framework," in *Developer's Guide to Oracle Solaris 11 Security*.

---

**Terminology in the Cryptographic Framework**

The following list of definitions and examples is useful when working with the Cryptographic Framework.

- **Algorithms** – Cryptographic algorithms. These are established, recursive computational procedures that encrypt or hash input. Encryption algorithms can be symmetric or asymmetric. Symmetric algorithms use the same key for encryption and decryption. Asymmetric algorithms, which are used in public-key cryptography, require two keys. Hashing functions are also algorithms.

Examples of algorithms include:

- Symmetric algorithms, such as AES and ARCFOUR
- Asymmetric algorithms, such as Diffie-Hellman and RSA
- Hashing functions, such as MD5
- **Consumers** – Are users of the cryptographic services that come from providers. Consumers can be applications, end users, or kernel operations.

  Examples of consumers include:
  - Applications, such as IKE
  - End users, such as a regular user who runs the `encrypt` command
  - Kernel operations, such as IPsec

- **Keystore** – In the Cryptographic Framework, is persistent storage for token objects, often used interchangeably with `token`. For information about a reserved keystore, see **Metaslot** in this list of definitions.

- **Mechanism** – Is the application of a mode of an algorithm for a particular purpose. For example, a DES mechanism that is applied to authentication, such as CKM_DES_MAC, is a separate mechanism from a DES mechanism that is applied to encryption, CKM_DES_CBC_PAD.

- **Metaslot** – Is a single slot that presents a union of the capabilities of other slots which are loaded in the framework. The metaslot eases the work of dealing with all of the capabilities of the providers that are available through the framework. When an application that uses the metaslot requests an operation, the metaslot figures out which actual slot should perform the operation. Metaslot capabilities are configurable, but configuration is not required. The metaslot is on by default. To configure the metaslot, see the `cryptoadm(1M)` man page.

  The metaslot does not have its own keystore. Rather, the metaslot reserves the use of a keystore from one of the actual slots in the Cryptographic Framework. By default, the metaslot reserves the Sun Crypto Softtoken keystore. The keystore that is used by the metaslot is not shown as one of the available slots.

  Users can specify an alternate keystore for metaslot by setting the environment variables `${METASLOT_OBJECTSTORE_SLOT}` and `${METASLOT_OBJECTSTORE_TOKEN}`, or by running the `cryptoadm` command. For more information, see the `libpkcs11(3LIB)`, `pkcs11_softtoken(5)`, and `cryptoadm(1M)` man pages.

- **Mode** – Is a version of a cryptographic algorithm. For example, CBC (Cipher Block Chaining) is a different mode from ECB (Electronic Code Book). The AES algorithm has two modes, CKM_AES_ECB and CKM_AES_CBC.

- **Policy** – Is the choice, by an administrator, of which mechanisms to make available for use. By default, all providers and all mechanisms are available for use. The disabling of any mechanism would be an application of policy. The enabling of a disabled mechanism would also be an application of policy.

- **Providers** – Are cryptographic services that consumers use. Providers plug in to the framework, so are also called plugins.

  Examples of providers include:
  - PKCS #11 libraries, such as `pkcs11_softtoken.so`
  - Modules of cryptographic algorithms, such as `aes` and `arcfour`
Device drivers and their associated hardware accelerators, such as the mca driver for the Sun Crypto Accelerator 6000

**Slot** – Is an interface to one or more cryptographic devices. Each slot, which corresponds to a physical reader or other device interface, might contain a token. A token provides a logical view of a cryptographic device in the framework.

**Token** – In a slot, a token provides a logical view of a cryptographic device in the framework.

---

**Scope of the Cryptographic Framework**

The framework provides commands for administrators, for users, and for developers who supply providers:

- **Administrative commands** – The `cryptoadm` command provides a `list` subcommand to list the available providers and their capabilities. Regular users can run the `cryptoadm list` and the `cryptoadm --help` commands.
  
  All other `cryptoadm` subcommands require you to assume a role that includes the Crypto Management rights profile, or to become superuser. Subcommands such as `disable`, `install`, and `uninstall` are available for administering the framework. For more information, see the `cryptoadm(1M)` man page.

  The `svcadm` command is used to manage the `kcfd` daemon, and to refresh cryptographic policy in the kernel. For more information, see the `svcadm(1M)` man page.

- **User-level commands** – The `digest` and `mac` commands provide file integrity services. The `encrypt` and `decrypt` commands protect files from eavesdropping. To use these commands, see “Protecting Files With the Cryptographic Framework (Task Map)” on page 219.

---

**Administrative Commands in the Cryptographic Framework**

The `cryptoadm` command administers a running Cryptographic Framework. The command is part of the Crypto Management rights profile. This profile can be assigned to a role for secure administration of the Cryptographic Framework. The `cryptoadm` command manages the following:

- Displaying cryptographic provider information
- Disabling or enabling provider mechanisms
- Disabling or enabling the metaslot

The `svcadm` command is used to enable, refresh, and disable the cryptographic services daemon, `kcfd`. This command is part of the Service Management Facility (SMF) feature of Oracle Solaris. `svc:/system/cryptosvcs` is the service instance for the Cryptographic Framework. For more information, see the `smf(5)` and `svcadm(1M)` man pages.
User-Level Commands in the Cryptographic Framework

The Cryptographic Framework provides user-level commands to check the integrity of files, to encrypt files, and to decrypt files. A separate command, `elfsign`, enables providers to sign binaries for use with the framework.

- **`digest` command** – Computes a message digest for one or more files or for stdin. A digest is useful for verifying the integrity of a file. SHA1 and MD5 are examples of digest functions.

- **`mac` command** – Computes a message authentication code (MAC) for one or more files or for stdin. A MAC associates data with an authenticated message. A MAC enables a receiver to verify that the message came from the sender and that the message has not been tampered with. The `sha1_mac` and `md5_hmac` mechanisms can compute a MAC.

- **`encrypt` command** – Encrypts files or stdin with a symmetric cipher. The `encrypt -l` command lists the algorithms that are available. Mechanisms that are listed under a user-level library are available to the `encrypt` command. The framework provides AES, DES, 3DES (Triple-DES), and ARCFOUR mechanisms for user encryption.

- **`decrypt` command** – Decrypts files or stdin that were encrypted with the `encrypt` command. The `decrypt` command uses the identical key and mechanism that were used to encrypt the original file.

Binary Signatures for Third-Party Software

The `elfsign` command provides a means to sign providers to be used with the Cryptographic Framework. Typically, this command is run by the developer of a provider.

The `elfsign` command has subcommands to request a certificate, sign binaries, and verify the signature on a binary. Unsigned binaries cannot be used by the Cryptographic Framework. Providers that have verifiable signed binaries can use the framework.

Plugins to the Cryptographic Framework

Third parties can plug their providers into the Cryptographic Framework. A third-party provider can be one of the following objects:

- PKCS #11 shared library
- Loadable kernel software module, such as an encryption algorithm, MAC function, or digest function
- Kernel device driver for a hardware accelerator

The objects from a provider must be signed with a certificate from Oracle. The certificate request is based on a private key that the third party selects, and a certificate that Oracle provides. The certificate request is sent to Oracle, which registers the third party and then issues the certificate. The third party then signs its provider object with the certificate from Oracle.
The loadable kernel software modules and the kernel device drivers for hardware accelerators must also register with the kernel. Registration is through the Cryptographic Framework SPI (service provider interface).

Cryptographic Services and Zones

The global zone and each non-global zone has its own /system/cryptosvc service. When the cryptographic service is enabled or refreshed in the global zone, the kcfd daemon starts in the global zone, user-level policy for the global zone is set, and kernel policy for the system is set. When the service is enabled or refreshed in a non-global zone, the kcfd daemon starts in the zone, and user-level policy for the zone is set. Kernel policy was set by the global zone.

For more information about zones, see Part II, “Oracle Solaris Zones,” in Oracle Solaris 11.1 Administration: Oracle Solaris Zones, Oracle Solaris 10 Zones, and Resource Management. For more information about using SMF to manage persistent applications, see Chapter 1, “Managing Services (Overview),” in Managing Services and Faults in Oracle Solaris 11.1 and the smf(5) man page.

Cryptographic Framework and FIPS-140

FIPS-140 is a U.S. Government computer security standard for cryptography modules. To use the Cryptographic Framework in FIPS-140 (FIPS) mode, see “How to Use the Cryptographic Framework in FIPS-140 Mode” on page 237.

Note – The presence of this mode does not mean that you are operating in a FIPS-140 compliant matter. You must follow the NIST security policy for this implementation to be compliant. For the policy, click the Security Policy link in the Cryptographic Framework entry for Oracle on the Validated FIPS 140-1 and FIPS 140-2 Cryptographic Modules (http://csrc.nist.gov/groups/STM/cmvp/documents/140-1/140val-all.htm) web site.

Cryptographic Framework and the SPARC T-Series Servers in This Release

The Cryptographic Framework supplies the SPARC T-Series systems with cryptographic mechanisms, and optimizes some mechanisms for these servers. Three cryptographic mechanisms are optimized for data at rest and in motion: AES-CBC, AES-CFB128, and ARCFOUR. Several cryptographic mechanisms are optimized for OpenSSL: DES, and by optimizing arbitrary-precision arithmetic (bignum), RSA and DSA. Other optimizations include small packet performance for handshakes and data in motion.
The following cryptographic mechanisms are available in this release:

- **AES-XTS** – Used for data at rest
- **SHA-224** – SHA2 mechanism
- **AES-XCBC-MAC** – Used for IPsec
This chapter describes how to use the Cryptographic Framework, and covers the following topics:

- “Protecting Files With the Cryptographic Framework (Tasks)” on page 219
- “Administering the Cryptographic Framework (Tasks)” on page 230

### Protecting Files With the Cryptographic Framework (Tasks)

This section describes how to generate symmetric keys, how to create checksums for file integrity, and how to protect files from eavesdropping. The commands in this section can be run by regular users. Developers can write scripts that use these commands.

### Protecting Files With the Cryptographic Framework (Task Map)

The Cryptographic Framework can help you protect your files. The following task map points to procedures for listing the available algorithms, and for protecting your files cryptographically.

<table>
<thead>
<tr>
<th>Task</th>
<th>Description</th>
<th>For Instructions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Generate a symmetric key.</td>
<td>Generates a key of user-specified length. Optionally, stores the key in a file, a PKCS #11 keystore, or an NSS keystore.</td>
<td>&quot;How to Generate a Symmetric Key by Using the pktool Command&quot; on page 220</td>
</tr>
<tr>
<td>Provide a checksum that ensures the integrity of a file.</td>
<td>Verifies that the receiver's copy of a file is identical to the file that was sent.</td>
<td>&quot;How to Compute a Digest of a File&quot; on page 224</td>
</tr>
<tr>
<td>Protect a file with a message authentication code (MAC).</td>
<td>Verifies to the receiver of your message that you were the sender.</td>
<td>&quot;How to Compute a MAC of a File&quot; on page 225</td>
</tr>
</tbody>
</table>
How to Generate a Symmetric Key by Using the pktool Command

Some applications require a symmetric key for encryption and decryption of communications. In this procedure, you create a symmetric key and store it.

If your site has a random number generator, you can use the generator to create a random number for the key. This procedure does not use your site's random number generator.

1 (Optional) If you plan to use a keystore, create it.

- To create and initialize a PKCS #11 keystore, see “How to Generate a Passphrase by Using the pktool setpin Command” on page 254.

- To create and initialize an NSS database, see Example 13–5.

2 Generate a random number for use as a symmetric key.

Use one of the following methods.

- Generate a key and store it in a file.

  The advantage of a file-stored key is that you can extract the key from this file for use in an application’s key file, such as the /etc/inet/secret/ipseckey file or IPsec. The usage statement shows the arguments.

  ```
  % pktool genkey keystore=file
  ...genkey keystore=file
  outkey=key-fn
  [ keytype=aes|arcfour|des|3des|generic ]
  [ keylen=key-size (AES, ARCFOUR or GENERIC only) ]
  [ print=y|n ]
  outkey=key-fn
  ```

  Is the filename where the key is stored.

  keytype=specific-symmetric-algorithm

  For a symmetric key of any length, the value is generic. For a particular algorithm, specify aes, arcfour, des, or 3des.

  keylen=size-in-bits

  Is the length of the key in bits. The number must be divisible by 8. Do not specify for des or 3des.
print=n
Prints the key to the terminal window. By default, the value of print is n.

- **Generate a key and store it in a PKCS #11 keystore.**

  The advantage of the PKCS #11 keystore is that you can retrieve the key by its label. This method is useful for keys that encrypt and decrypt files. You must complete Step 1 before using this method. The usage statement shows the arguments. The brackets around the keystore argument indicate that when the keystore argument is not specified, the key is stored in the PKCS #11 keystore.

  $pktool genkey
  ...genkey  [ keystore=pkcs11 ]
  [ label=key-label ]
  [ keytype=aes|arcfour|des|3des|generic ]
  [ keylen=key-size (AES, ARCFOUR or GENERIC only)]
  [ token=token[:manuf[:serial]]]
  [ sensitive=y|n ]
  [ extractable=y|n ]
  [ print=y|n ]

  label=key-label
  Is a user-specified label for the key. The key can be retrieved from the keystore by its label.

  keytype=specific-symmetric-algorithm
  For a symmetric key of any length, the value is generic. For a particular algorithm, specify aes, arcfour, des, or 3des.

  keylen=size-in-bits
  Is the length of the key in bits. The number must be divisible by 8. Do not specify for des or 3des.

  token=token
  Is the token name. By default, the token is Sun Software PKCS#11 softtoken.

  sensitive=n
  Specifies the sensitivity of the key. When the value is y, the key cannot be printed by using the print=y argument. By default, the value of sensitive is n.

  extractable=y
  Specifies that the key can be extracted from the keystore. Specify n to prevent the key from being extracted.

  print=n
  Prints the key to the terminal window. By default, the value of print is n.

- **Generate a key and store it in an NSS keystore.**

  You must complete Step 1 before using this method. The usage statement shows the arguments.

  $ pktool genkey keystore=nss
  ...genkey  keystore=nss
  [ label=key-label ]
Protecting Files With the Cryptographic Framework (Tasks)

- label=
  Is a user-specified label for the key. The key can be retrieved from the keystore by its label.

- keytype=
  specific-symmetric-algorithm
  For a symmetric key of any length, the value is generic. For a particular algorithm, specify aes, arcfour, des, or 3des.

- keylen=
  size-in-bits
  Is the length of the key in bits. The number must be divisible by 8. Do not specify for des or 3des.

- token=
  token
  Is the token name. By default, the token is the NSS internal token.

- dir=
  directory
  Is the directory path to the NSS database. By default, directory is the current directory.

- prefix=
  directory
  Is the prefix to the NSS database. The default is no prefix.

3 (Optional) Verify that the key exists.

Use one of the following commands, depending on where you stored the key.

- Verify the key in the key-fn file.
  % pktool list keystore=file objtype=key [infile=key-fn]
  Found n keys.
  Key #1 - keytype:location (keylen)

- Verify the key in the PKCS #11 or the NSS keystore.
  $ pktool list objtype=key
  Enter PIN for keystore:
  Found n keys.
  Key #1 - keytype:location (keylen)

Example 12–1 Creating a Symmetric Key by Using the pktool Command

In the following example, a user creates a PKCS #11 keystore for the first time, and then generates a large symmetric key for an application. Finally, the user verifies that the key is in the keystore.

# pktool setpin
Create new passphrase: Type easily-remembered-hard-to-detect-password
Re-enter new passphrase: Retype password
Passphrase changed.
pktool genkey label=specialappkey keytype=generic keylen=1024
Enter PIN for Sun Software PKCS#11 softtoken : Type password

pktool list objtype=key
Enter PIN for Sun Software PKCS#11 softtoken : Type password

Found 1 keys.
Key #1 - symmetric: specialappkey (1024 bits)

Creating a DES Key by Using the pktool Command

In the following example, a secret key for the DES algorithm is created. The key is stored in a local file for later decryption. The command protects the file with 400 permissions. When the key is created, the print=y option displays the generated key in the terminal window.

DES mechanisms use a 64-bit key. The user who owns the keyfile retrieves the key by using the od command.

pktool genkey keystore=file outkey=64bit.file1 keytype=des print=y
Key Value = "a3237b2c0a8ff9b3"

od -x 64bit.file1
0000000 a323 7b2c 0a8f f9b3

Creating a Symmetric Key for IPsec Security Associations

In the following example, the administrator manually creates the keying material for IPsec SAs and stores them in files. Then, the administrator copies the keys to the /etc/inet/secret/ipseckeys file and destroys the original files.

- First, the administrator creates and displays the keys that the IPsec policy requires:

```
# pktool genkey keystore=file outkey=ipencrin1 keytype=generic keylen=192 print=y
Key Value = "294979e512cb879370dbacedc37fcbb49e7e2d6b2d49"
# pktool genkey keystore=file outkey=ipencrout1 keytype=generic keylen=192 print=y
Key Value = "9678f0e33406c86e3d1686e5b0406bd0434619c2b09d204"
# pktool genkey keystore=file outkey=ipspi1 keytype=generic keylen=32 print=y
Key Value = "acbeaa20"
# pktool genkey keystore=file outkey=ipspi2 keytype=generic keylen=32 print=y
Key Value = "19174215"
# pktool genkey keystore=file outkey=ipsha21 keytype=generic keylen=256 print=y
Key Value = "659c20f2d6c3f9570bce9e96d95e2263aca4eb3360f7e1c786af4177fe9e"
# pktool genkey keystore=file outkey=ipsha22 keytype=generic keylen=256 print=y
Key Value = "b041975a0e1f0e0503665c3966684d73la3bb12cf87b0a837b2da5d82c810"
```  

- Then, the administrator creates the following /etc/inet/secret/ipseckeys file:

```
## SPI values require a leading 0x.
## Backslashes indicate command continuation.
##
## for outbound packets on this system
add esp spi 0xacbeaa20
  src 192.168.1.1 dst 192.168.2.1
```
After verifying that the syntax of the ipseckeys file is valid, the administrator destroys the original key files.

# ipseckey -c /etc/inet/secret/ipseckeys
# rm ipencrin1 ipencrout1 ipspi1 ipspi2 ipsha21 ipsha22

The administrator copies the ipseckeys file to the communicating system by using the ssh command or another secure mechanism. On the communicating system, the protections are reversed. The first entry in the ipseckeys file protects inbound packets, and the second entry protects outbound packets. No keys are generated on the communicating system.

### How to Compute a Digest of a File

When you compute a digest of a file, you can check to see that the file has not been tampered with by comparing digest outputs. A digest does not alter the original file.

1. **List the available digest algorithms.**
   
   `% digest -l`
   
   - md5
   - sha1
   - sha224
   - sha256
   - sha384
   - sha512

2. **Compute the digest of the file and save the digest listing.**

   Provide an algorithm with the `digest` command.
   
   `% digest -v -a algorithm input-file > digest-listing`
   
   `-v` Displays the output in the following format:
   
   ```
   algorithm (input-file) = digest
   ```
   
   `-a algorithm` Is the algorithm to use to compute a digest of the file. Type the algorithm as the algorithm appears in the output of Step 1.
   
   `input-file` Is the input file for the `digest` command.
   
   `digest-listing` Is the output file for the `digest` command.
Computing a Digest With the MD5 Mechanism

In the following example, the `digest` command uses the MD5 mechanism to compute a digest for an email attachment.

```
% digest -v -a md5 email.attach >> $HOME/digest.emails.05.07
% cat ~/digest.emails.05.07
md5 (email.attach) = 85c0a53d1a5cc71ea34d9ee7b1b28b01
```

When the `-v` option is not used, the digest is saved with no accompanying information:

```
% digest -a md5 email.attach >> $HOME/digest.emails.05.07
% cat ~/digest.emails.05.07
85c0a53d1a5cc71ea34d9ee7b1b28b01
```

Computing a Digest With the SHA1 Mechanism

In the following example, the `digest` command uses the SHA1 mechanism to provide a directory listing. The results are placed in a file.

```
% digest -v -a sha1 docs/* > $HOME/digest.docs.legal.05.07
% more ~/digest.docs.legal.05.07
sha1 (docs/legal1) = 1df50e8ad219e34f0b911e097b7b588e31f9b435
sha1 (docs/legal2) = 68efa5a636291bde8f33e046eb33508c94842c38
sha1 (docs/legal3) = 085d99125861b0c8fa2946c183be8e32cccf6c9
sha1 (docs/legal4) = f3085eae7e2c8d008816564f0df20827d10e1d083
```

How to Compute a MAC of a File

A message authentication code, or MAC, computes a digest for the file and uses a secret key to further protect the digest. A MAC does not alter the original file.

1 List the available mechanisms.

```
% mac -l
```

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Keysize: Min</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>des_mac</td>
<td>64</td>
<td>64</td>
</tr>
<tr>
<td>sha1_hmac</td>
<td>8</td>
<td>512</td>
</tr>
<tr>
<td>md5_hmac</td>
<td>8</td>
<td>512</td>
</tr>
<tr>
<td>sha224_hmac</td>
<td>8</td>
<td>512</td>
</tr>
<tr>
<td>sha256_hmac</td>
<td>8</td>
<td>512</td>
</tr>
<tr>
<td>sha384_hmac</td>
<td>8</td>
<td>1024</td>
</tr>
<tr>
<td>sha512_hmac</td>
<td>8</td>
<td>1024</td>
</tr>
</tbody>
</table>

2 Generate a symmetric key of the appropriate length.

You have two options. You can provide a passphrase from which a key will be generated. Or you can provide a key.
If you provide a passphrase, you must store or remember the passphrase. If you store the passphrase online, the passphrase file should be readable only by you.

If you provide a key, it must be the correct size for the mechanism. You can use the pktool command. For the procedure and some examples, see "How to Generate a Symmetric Key by Using the pktool Command" on page 220.

3 Create a MAC for a file.

Provide a key and use a symmetric key algorithm with the mac command.

% mac [-v] -a algorithm [-k keyfile | -K key-label [-T token]] input-file

- v Displays the output in the following format:
  algorithm (input-file) = mac
- a algorithm Is the algorithm to use to compute the MAC. Type the algorithm as the algorithm appears in the output of the mac -l command.
- k keyfile Is the file that contains a key of algorithm-specified length.
- K key-label Is the label of a key in the PKCS#11 keystore.
- T token Is the token name. By default, the token is Sun Software PKCS#11 softtoken.
  Is used only when the -K key-label option is used.
  input-file Is the input file for the MAC.

Example 12–6 Computing a MAC With DES_MAC and a Passphrase

In the following example, the email attachment is authenticated with the DES_MAC mechanism and a key that is derived from a passphrase. The MAC listing is saved to a file. If the passphrase is stored in a file, the file should not be readable by anyone but the user.

% mac -v -a des_mac email.attach
Enter passphrase: <Type passphrase>
des_mac (email.attach) = dd27870a
% echo "des_mac (email.attach) = dd27870a" >> /desmac.daily.05.07

Example 12–7 Computing a MAC With MD5_HMAC and a Key File

In the following example, the email attachment is authenticated with the MD5_HMAC mechanism and a secret key. The MAC listing is saved to a file.

% mac -v -a md5_hmac -k $HOME/keyf/05.07.mack64 email.attach
md5_hmac (email.attach) = 02df6eb6c123ff25d78877eb1d55710c
% echo "md5_hmac (email.attach) = 02df6eb6c123ff25d78877eb1d55710c" \
>> ~/mac.daily.05.07
Computing a MAC With SHA1_HMAC and a Key File

In the following example, the directory manifest is authenticated with the SHA1_HMAC mechanism and a secret key. The results are placed in a file.

```bash
% mac -v -a sha1_hmac \
-k $HOME/keyf/05.07.mack64 docs/* > $HOME/mac.docs.legal.05.07
% more ~/mac.docs.legal.05.07
sha1_hmac (docs/legal1) = 9b31536d3b3c0c6b25d653418db8e765e17fe07a
sha1_hmac (docs/legal2) = 865af61a300f8a457462a428db1a88c1b51ff5
sha1_hmac (docs/legal3) = 7aede27602ef6e4454748c0b32e0152e45beb4
```

Computing a MAC With SHA1_HMAC and a Key Label

In the following example, the directory manifest is authenticated with the SHA1_HMAC mechanism and a secret key. The results are placed in the user’s PKCS#11 keystore. The user initially created the keystore and the password to the keystore by using the `pktool setpin` command.

```bash
% mac -a sha1_hmac -K legaldocs0507 docs/*
Enter pin for Sun Software PKCS#11 softtoken: Type password
```

To retrieve the MAC from the keystore, the user uses the verbose option, and provides the key label and the name of the directory that was authenticated.

```bash
% mac -v -a sha1_hmac -K legaldocs0507 docs/*
Enter pin for Sun Software PKCS#11 softtoken: Type password
```

How to Encrypt and Decrypt a File

When you encrypt a file, the original file is not removed or changed. The output file is encrypted.

For solutions to common errors from the `encrypt` command, see the section that follows the examples.

1. Create a symmetric key of the appropriate length.

You have two options. You can provide a passphrase from which a key will be generated. Or you can provide a key.
If you provide a passphrase, you must store or remember the passphrase. If you store the passphrase online, the passphrase file should be readable only by you.

If you provide a key, it must be the correct size for the mechanism. You can use the pktool command. For the procedure and some examples, see "How to Generate a Symmetric Key by Using the pktool Command" on page 220.

2 Encrypt a file.

Provide a key and use a symmetric key algorithm with the encrypt command.

```
% encrypt -a algorithm [-v] [-k keyfile |-K key-label [-T token]] [-i input-file] [-o output-file] 
```

- **algorithm** Is the algorithm to use to encrypt the file. Type the algorithm as the algorithm appears in the output of the encrypt -l command.
- **keyfile** Is the file that contains a key of algorithm-specified length. The key length for each algorithm is listed, in bits, in the output of the encrypt -l command.
- **key-label** Is the label of a key in the PKCS #11 keystore.
- **token** Is the token name. By default, the token is Sun Software PKCS#11 softtoken. Is used only when the -K key-label option is used.
- **input-file** Is the input file that you want to encrypt. This file is left unchanged by the command.
- **output-file** Is the output file that is the encrypted form of the input file.

**Example 12–10 Creating an AES Key for Encrypting Your Files**

In the following example, a user creates and stores an AES key in an existing PKCS #11 keystore for use in encryption and decryption. The user can verify that the key exists and can use the key, but cannot view the key itself.

```
% pktool genkey label=MyAESkeynumber1 keytype=aes keylen=256
Enter PIN for Sun Software PKCS#11 softtoken : Type password

% pktool list objtype=key
Enter PIN for Sun Software PKCS#11 softtoken :< Type password>
Found 1 key
  Key #1 - Sun Software PKCS#11 softtoken: MyAESkeynumber1 (256)
```

To use the key to encrypt a file, the user retrieves the key by its label.

```
% encrypt -a aes -K MyAESkeynumber1 -i encryptthisfile -o encryptedthisfile
```

To decrypt the encryptedthisfile file, the user retrieves the key by its label.

```
% decrypt -a aes -K MyAESkeynumber1 -i encryptedthisfile -o sameasencryptthisfile
```
Encrypting and Decrypting With AES and a Passphrase

In the following example, a file is encrypted with the AES algorithm. The key is generated from the passphrase. If the passphrase is stored in a file, the file should not be readable by anyone but the user.

```
% encrypt -a aes -i ticket.to.ride -o ~/enc/e.ticket.to.ride
Enter passphrase: <Type passphrase>
Re-enter passphrase: Type passphrase again
```

The input file, ticket.to.ride, still exists in its original form.

To decrypt the output file, the user uses the same passphrase and encryption mechanism that encrypted the file.

```
% decrypt -a aes -i ~/enc/e.ticket.to.ride -o ~/d.ticket.to.ride
Enter passphrase: <Type passphrase>
```

Encrypting and Decrypting With AES and a Key File

In the following example, a file is encrypted with the AES algorithm. AES mechanisms use a key of 128 bits, or 16 bytes.

```
% encrypt -a aes -k ~/keyf/05.07.aes16 -i ticket.to.ride -o ~/enc/e.ticket.to.ride
```

The input file, ticket.to.ride, still exists in its original form.

To decrypt the output file, the user uses the same key and encryption mechanism that encrypted the file.

```
% decrypt -a aes -k ~/keyf/05.07.aes16 -i ~/enc/e.ticket.to.ride -o ~/d.ticket.to.ride
```

Encrypting and Decrypting With ARCFOUR and a Key File

In the following example, a file is encrypted with the ARCFOUR algorithm. The ARCFOUR algorithm accepts a key of 8 bits (1 byte), 64 bits (8 bytes), or 128 bits (16 bytes).

```
% encrypt -a arcfour -i personal.txt -k ~/keyf/05.07.rc4.8 -o ~/enc/e.personal.txt
```

To decrypt the output file, the user uses the same key and encryption mechanism that encrypted the file.

```
% decrypt -a arcfour -i ~/enc/e.personal.txt -k ~/keyf/05.07.rc4.8 -o ~/personal.txt
```
Encrypting and Decrypting With 3DES and a Key File

In the following example, a file is encrypted with the 3DES algorithm. The 3DES algorithm requires a key of 192 bits, or 24 bytes.

```bash
% encrypt -a 3des -k ~/keyf/05.07.des24 \ 
   -i ~/personal2.txt -o ~/enc/e.personal2.txt
```

To decrypt the output file, the user uses the same key and encryption mechanism that encrypted the file.

```bash
% decrypt -a 3des -k ~/keyf/05.07.des24 \ 
   -i ~/enc/e.personal2.txt -o ~/personal2.txt
```

**Troubleshooting**  
The following messages indicate that the key that you provided to the `encrypt` command is not permitted by the algorithm that you are using.

- `encrypt: unable to create key for crypto operation: CKR_ATTRIBUTE_VALUE_INVALID`
- `encrypt: failed to initialize crypto operation: CKR_KEY_SIZE_RANGE`

If you pass a key that does not meet the requirements of the algorithm, you must supply a better key.

- One option is to use a passphrase. The framework then provides a key that meets the requirements.
- The second option is to pass a key size that the algorithm accepts. For example, the DES algorithm requires a key of 64 bits. The 3DES algorithm requires a key of 192 bits.

Administering the Cryptographic Framework (Tasks)

This section describes how to administer the software providers and the hardware providers in the Cryptographic Framework. Software providers and hardware providers can be removed from use when desirable. For example, you can disable the implementation of an algorithm from one software provider. You can then force the system to use the algorithm from a different software provider.

**Administering the Cryptographic Framework (Task Map)**

The following task map points to procedures for administering software and hardware providers in the Cryptographic Framework.
### Task Description for Instructions

List the providers in the Cryptographic Framework.

<table>
<thead>
<tr>
<th>Task</th>
<th>Description</th>
<th>For Instructions</th>
</tr>
</thead>
<tbody>
<tr>
<td>List the providers in the Cryptographic Framework.</td>
<td>Lists the algorithms, libraries, and hardware devices that are available for use in the Cryptographic Framework.</td>
<td>“How to List Available Providers” on page 231</td>
</tr>
<tr>
<td>Enable FIPS-140 mode.</td>
<td>Runs the Cryptographic Framework to a U.S. government standard for cryptography modules.</td>
<td>“How to Use the Cryptographic Framework in FIPS-140 Mode” on page 237</td>
</tr>
<tr>
<td>Add a software provider.</td>
<td>Adds a PKCS #11 library or a kernel module to the Cryptographic Framework. The provider must be signed.</td>
<td>“How to Add a Software Provider” on page 235</td>
</tr>
<tr>
<td>Prevent the use of a user-level mechanism.</td>
<td>Removes a software mechanism from use. The mechanism can be enabled again.</td>
<td>“How to Prevent the Use of a User-Level Mechanism” on page 239</td>
</tr>
<tr>
<td>Temporarily disable mechanisms from a kernel module.</td>
<td>Temporarily removes a mechanism from use. Usually used for testing.</td>
<td>“How to Prevent the Use of a Kernel Software Provider” on page 240</td>
</tr>
<tr>
<td>Uninstall a library.</td>
<td>Removes a user-level software provider from use.</td>
<td>Example 12–21</td>
</tr>
<tr>
<td>Uninstall a kernel provider.</td>
<td>Removes a kernel software provider from use.</td>
<td>Example 12–23</td>
</tr>
<tr>
<td>List available hardware providers.</td>
<td>Shows the attached hardware, shows the mechanisms that the hardware provides, and shows which mechanisms are enabled for use.</td>
<td>“How to List Hardware Providers” on page 243</td>
</tr>
<tr>
<td>Disable mechanisms from a hardware provider.</td>
<td>Ensures that selected mechanisms on a hardware accelerator are not used.</td>
<td>“How to Disable Hardware Provider Mechanisms and Features” on page 244</td>
</tr>
<tr>
<td>Restart or refresh cryptographic services.</td>
<td>Ensures that cryptographic services are available.</td>
<td>“How to Refresh or Restart All Cryptographic Services” on page 245</td>
</tr>
</tbody>
</table>

### How to List Available Providers

The Cryptographic Framework provides algorithms for several types of consumers:

- User-level providers provide a PKCS #11 cryptographic interface to applications that are linked with the libpkcs11 library
- Kernel software providers provide algorithms for IPsec, Kerberos, and other Oracle Solaris kernel components
- Kernel hardware providers provide algorithms that are available to kernel consumers and to applications through the pkcs11_kernel library

1. List the providers in a brief format.
**Note** – The contents and format of the providers list varies for different Oracle Solaris releases and different platforms. Run the `cryptoadm list` command on your system to see the providers that your system supports.

Only those mechanisms at the user level are available for use by regular users.

```bash
% cryptoadm list
User-level providers:
Provider: /usr/lib/security/$ISA/pkcs11_kernel.so
Provider: /usr/lib/security/$ISA/pkcs11_softtoken.so
Provider: /usr/lib/security/$ISA/pkcs11_tpm.so

Kernel software providers:
  des
  aes
  arcfour
  blowfish
  ecc
  sha1
  sha2
  md4
  md5
  rsa
  swrand
  n2rng/0
  ncp/0
  n2cp/0
```

2 List the providers and their mechanisms in the Cryptographic Framework.

All mechanisms are listed in the following output. However, some of the listed mechanisms might be unavailable for use. To list only the mechanisms that the administrator has approved for use, see Example 12–16.

The output is truncated for display purposes.

```bash
% cryptoadm list -m
User-level providers:
=====================  
Provider: /usr/lib/security/$ISA/pkcs11_kernel.so

Mechanisms:
CKM_DSA
CKM_RSA_X_509
CKM_RSA_PKCS
...
CKM_SHA256_HMAC_GENERAL
CKM_SSL3_MD5_MAC

Provider: /usr/lib/security/$ISA/pkcs11_softtoken.so

Mechanisms:
CKM_DES_CBC
CKM_DES_CBC_PAD
CKM_DES_ECB
```
Finding the Existing Cryptographic Mechanisms

In the following example, all mechanisms that the user-level library, pkcs11_softtoken, offers are listed.

```
% cryptoadm list -m provider=/usr/lib/security/$ISA/pkcs11_softtoken.so
Mechanisms:
CKM DES CBC
CKM DES CBC_PAD
CKM DES ECB
CKM DES KEY GEN
CKM DES_MAC GENERAL
CKM DES_MAC
... 
CKM ECDSA
CKM ECDSA_SHA1
CKM_ECDH1_DERIVE
```

Finding the Available Cryptographic Mechanisms

Policy determines which mechanisms are available for use. The administrator sets the policy. An administrator can choose to disable mechanisms from a particular provider. The -p option displays the list of mechanisms that are permitted by the policy that the administrator has set.
% cryptoadm list -p
User-level providers:
=======================================
/usr/lib/security/$ISA/pkcs11_kernel.so: all mechanisms are enabled. random is enabled.
/usr/lib/security/$ISA/pkcs11_softtoken.so: all mechanisms are enabled, random is enabled.
/usr/lib/security/$ISA/pkcs11_tpm.so: all mechanisms are enabled.

Kernel providers:
=======================================
des: all mechanisms are enabled.
aes: all mechanisms are enabled.
arcfour: all mechanisms are enabled.
blowfish: all mechanisms are enabled.
ecc: all mechanisms are enabled.
sha1: all mechanisms are enabled.
sha2: all mechanisms are enabled.
md4: all mechanisms are enabled.
md5: all mechanisms are enabled.
rsa: all mechanisms are enabled.
swrand: random is enabled.
n2rng/0: all mechanisms are enabled, random is enabled.
ncp/0: all mechanisms are enabled.
n2cp/0: all mechanisms are enabled.

Example 12–17 Determining Which Cryptographic Mechanisms Perform Which Functions

Mechanisms perform specific cryptographic functions, such as signing or key generation. The -v -m options display every mechanism and its functions.

In this instance, the administrator wants to determine for which functions the CKM_ECDSA* mechanisms can be used.

% cryptoadm list -vm
User-level providers:
=======================================
Provider: /usr/lib/security/$ISA/pkcs11_kernel.so
Number of slots: 3
Slot #2
Description: ncp/0 Crypto Accel Asym 1.0
...
CKM_ECDSA 163 571 X . . . X . X . . . . . . . .
...
Provider: /usr/lib/security/$ISA/pkcs11_softtoken.so
...
CKM_ECDSA 112 571 . . . X . X . . . . . . . .
CKM_ECDSA_SHA1 112 571 . . . X . X . . . . . . . .
...
Kernel providers:
=======================================
...
ec: CKM_EC_KEY_PAIR_GEN,CKM_ECDH1_DERIVE,CKM_ECDSA,CKM_ECDSA_SHA1
...
The listing indicates that these mechanisms are available from the following user-level providers:

- **CKM_ECDSA and CKM_ECDSA_SHA1** – As software implementation in
  `/usr/lib/security/$ISA/pkcs11_softtoken.so` library
- **CKM_ECDSA** – Accelerated by ncp/0 Crypto Accel Asym 1.0 in
  `/usr/lib/security/$ISA/pkcs11_kernel.so` library

Each item in an entry represents a piece of information about the mechanism. For these ECC mechanisms, the listing indicates the following:

- Minimum length – 112 bytes
- Maximum length – 571 bytes
- Hardware – Is or is not available on hardware.
- Encrypt – Is not used to encrypt data.
- Decrypt – Is not used to decrypt data.
- Digest – Is not used to create message digests.
- Sign – Is used to sign data.
- Sign + Recover – Is not used to sign data, where the data can be recovered from the signature.
- Verify – Is used to verify signed data.
- Verify + Recover – Is not used to verify data that can be recovered from the signature.
- Key generation – Is not used to generate a private key.
- Pair generation – Is not used to generate a key pair.
- Wrap – Is not used to wrap, that is, encrypt, an existing key.
- Unwrap – Is not used to unwrap a wrapped key.
- Derive – Is not used to derive a new key from a base key.
- EC Caps – Absent EC capabilities that are not covered by previous items

### How to Add a Software Provider

**Before You Begin**

You must become an administrator who is assigned the Crypto Management rights profile. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1. List the software providers that are available to the system.

   ```bash
   % cryptoadm list
   User-level providers:
   Provider: /usr/lib/security/$ISA/pkcs11_kernel.so
   Provider: /usr/lib/security/$ISA/pkcs11_softtoken.so
   /usr/lib/security/$ISA/pkcs11_tpm.so: all mechanisms are enabled.
   ```
Kernel software providers:
  des
  aes
  arcfour
  blowfish
  sha1
  sha2
  md4
  md5
  rsa
  swrand
  n2rng/0
  ncp/0
  n2cp/0

2 **Add the provider from a repository.**

Existing provider software has been issued a certificate by Oracle.

3 **Refresh the providers.**

You need to refresh providers if you added a software provider, or if you added hardware and specified policy for the hardware.

```
# svcadm refresh svc:/system/cryptosvc
```

4 **Locate the new provider on the list.**

In this case, a new kernel software provider was installed.

```
# cryptoadm list
...
Kernel software providers:
  des
  aes
  arcfour
  blowfish
  ecc
  sha1
  sha2
  md4
  md5
  rsa
  swrand
  sha3  <-- added provider
...
```

**Example 12–18 Adding a User-Level Software Provider**

In the following example, a signed PKCS #11 library is installed.

```
# pkgadd -d /cdrom/cdrom0/SolarisNew
Answer the prompts
# svcadm refresh system/cryptosvc
# cryptoadm list
```
Developers who are testing a library with the Cryptographic Framework can install the library manually.

```
# cryptoadm install provider=/opt/lib/$ISA/libpkcs11.so.1
```

## How to Use the Cryptographic Framework in FIPS-140 Mode

By default, FIPS-140 mode is disabled in Oracle Solaris. In this procedure, you create a new boot environment (BE) for FIPS-140 mode, then enable FIPS-140 and boot into the new BE. This method enables you to recover from system panics that can result from FIPS-140 compliance tests. For more information, see the `cryptoadm(1M)` man page and “Cryptographic Framework and FIPS-140” on page 217.

**Before You Begin**

You must assume the root role. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1. **Determine if the system is in FIPS-140 mode.**

   ```
   % cryptoadm list fips-140
   User-level providers:
   ==============================================================
   /usr/lib/security/$ISA/pkcs11_softtoken: FIPS-140 mode is disabled.
   
   Kernel software providers:
   ==============================================================
   des: FIPS-140 mode is disabled.
   aes: FIPS-140 mode is disabled.
   ecc: FIPS-140 mode is disabled.
   sha1: FIPS-140 mode is disabled.
   sha2: FIPS-140 mode is disabled.
   rsa: FIPS-140 mode is disabled.
   swrand: FIPS-140 mode is disabled.
   
   Kernel hardware providers:
   ==============================================================
   ```

2. **Create a new BE for your FIPS-140 version of the Cryptographic Framework.**

   Before you enable FIPS-140 mode, you must first create, activate, and boot a new BE by using the `beadm` command. A FIPS-140-enabled system runs compliance tests that can cause a panic if
they fail. Therefore, it is important to have an available BE that you can boot to get your system up and running while you debug issues with the FIPS-140 boundary.

a. **Create a BE based on your current BE.**  
   In this example, you create a BE named `S11.1-FIPS`.
   ```
   # beadm create S11.1-FIPS-140
   ```

b. **Activate that BE.**
   ```
   # beadm activate S11.1-FIPS-140
   ```

c. **Reboot the system.**

d. **Enable FIPS-140 mode in the new BE.**
   ```
   # cryptoadm enable fips-140
   ```

   **Note:** This subcommand does not disable the non-FIPS-140 approved algorithms from the user-level `pkcs11_softtoken` library and the kernel software providers. The consumers of the framework are responsible for using only FIPS-140-approved algorithms.

   For more information about the effects of FIPS-140 mode, see the `cryptoadm(1M)` man page.

3 **When you want to run without FIPS-140 enabled, disable FIPS-140 mode.**

You can reboot to the original BE or disable FIPS-140 in the current BE.

- **Boot to the original BE.**

  ```
  # beadm list
  ```

  ```
  BE Active Mountpoint Space Policy Created
  -- ------ ---------- ----- ------ -------
  S11.1 - - 48.22G static 2012-10-10 10:10
  S11.1-FIPS-140 NR / 287.01M static 2012-11-18 18:18
  # beadm activate S11.1
  ```

- **Disable FIPS-140 mode in the current BE and reboot.**

  ```
  # cryptoadm disable fips-140
  ```

  FIPS-140 mode remains in operation until the system is rebooted.

  ```
  # reboot
  ```
How to Prevent the Use of a User-Level Mechanism

If some of the cryptographic mechanisms from a library provider should not be used, you can remove selected mechanisms. This procedure uses the DES mechanisms in the pkcs11_softtoken library as an example.

Before You Begin

You must become an administrator who is assigned the Crypto Management rights profile. For more information, see "How to Use Your Assigned Administrative Rights" on page 157.

1 List the mechanisms that are offered by a particular user-level software provider.

   % cryptoadm list -m provider=/usr/lib/security/$ISA/pkcs11_softtoken.so
   /usr/lib/security/$ISA/pkcs11_softtoken.so:
   CKM_DES_CBC, CKM_DES_CBC_PAD, CKM_DES ECB, CKM DES KEY GEN,
   CKM_DES3_CBC, CKM_DES3_CBC_PAD, CKM_DES3 ECB, CKM_DES3_KEY GEN,
   CKM_AES_CBC, CKM_AES_CBC_PAD, CKM_AES ECB, CKM_AES_KEY GEN,
   ...

2 List the mechanisms that are available for use.

   $ cryptoadm list -p
   user-level providers:
   =====================
   ...
   /usr/lib/security/$ISA/pkcs11_softtoken.so: all mechanisms are enabled.
   random is enabled.
   ...

3 Disable the mechanisms that should not be used.

   $ cryptoadm disable provider=/usr/lib/security/$ISA/pkcs11_softtoken.so
   > mechanism=CKM_DES_CBC, CKM_DES_CBC_PAD, CKM_DES ECB

4 List the mechanisms that are available for use.

   $ cryptoadm list -p provider=/usr/lib/security/$ISA/pkcs11_softtoken.so
   /usr/lib/security/$ISA/pkcs11_softtoken.so: all mechanisms are enabled,
   except CKM_DES ECB, CKM_DES_CBC_PAD, CKM_DES_CBC. random is enabled.

Example 12–19

Enabling a User-Level Software Provider Mechanism

In the following example, a disabled DES mechanism is again made available for use.

   $ cryptoadm list -m provider=/usr/lib/security/$ISA/pkcs11_softtoken.so
   /usr/lib/security/$ISA/pkcs11_softtoken.so:
   CKM_DES_CBC, CKM_DES_CBC_PAD, CKM_DES ECB, CKM_DES KEY GEN,
   CKM_DES3_CBC, CKM_DES3_CBC_PAD, CKM_DES3 ECB, CKM_DES3_KEY GEN,
   ...
   $ cryptoadm list -p provider=/usr/lib/security/$ISA/pkcs11_softtoken.so
   /usr/lib/security/$ISA/pkcs11_softtoken.so: all mechanisms are enabled,
   except CKM_DES ECB, CKM_DES_CBC_PAD, CKM_DES_CBC. random is enabled.
   $ cryptoadm enable provider=/usr/lib/security/$ISA/pkcs11_softtoken.so
   > mechanism=CKM_DES ECB
Example 12–20 Enabling All User-Level Software Provider Mechanisms
In the following example, all mechanisms from the user-level library are enabled.

```bash
$ cryptoadm enable provider=/usr/lib/security/$ISA/pkcs11_softtoken.so all
$ cryptoadm list -p provider=/usr/lib/security/$ISA/pkcs11_softtoken.so
/usr/lib/security/$ISA/pkcs11_softtoken.so: all mechanisms are enabled. random is enabled.
```

Example 12–21 Permanently Removing User-Level Software Provider Availability
In the following example, the libpkcs11.so.1 library is removed.

```bash
$ cryptoadm uninstall provider=/opt/lib/$ISA/libpkcs11.so.1
$ cryptoadm list
user-level providers:
  /usr/lib/security/$ISA/pkcs11_kernel.so
  /usr/lib/security/$ISA/pkcs11_softtoken.so
  /usr/lib/security/$ISA/pkcs11_tpm.so

kernel providers:
  ...
```


⚠️ How to Prevent the Use of a Kernel Software Provider
If the Cryptographic Framework provides multiple modes of a provider such as AES, you might remove a slow mechanism from use, or a corrupted mechanism. This procedure uses the AES algorithm as an example.

Before You Begin
You must become an administrator who is assigned the Crypto Management rights profile. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1 List the mechanisms that are offered by a particular kernel software provider.

   ```bash
   $ cryptoadm list -m provider=aes
   aes: CKM_AES_ECB,CKM_AES_CBC,CKM_AES_CTR,CKM_AES_CCM,CKM_AES_GCM,CKM_AES_GMAC,
   CKM_AES_CFB128,CKM_AES_XTS,CKM_AES_XCBC_MAC
   ```

2 List the mechanisms that are available for use.

   ```bash
   $ cryptoadm list -p provider=aes
   aes: all mechanisms are enabled.
   ```

3 Disable the mechanism that should not be used.

   ```bash
   $ cryptoadm disable provider=aes mechanism=CKM_AES_ECB
   ```
4 List the mechanisms that are available for use.

```
cryptoadm list -p provider=aes
aes: all mechanisms are enabled, except CKM_AES_ECB.
```

**Example 12–22** Enabling a Kernel Software Provider Mechanism

In the following example, a disabled AES mechanism is again made available for use.

```
cryptoadm list -m provider=aes
aes: CKM_AES_ECB, CKM_AES_CBC, CKM_AES_CTR, CKM_AES_CCM, CKM_AES_GCM, CKM_AES_GMAC, CKM_AES_CFB128, CKM_AES_XTS, CKM_AES_XCBC_MAC
```

```
cryptoadm enable provider=aes mechanism=CKM_AES_ECB
```

```
cryptoadm list -p provider=aes
aes: all mechanisms are enabled.
```

**Example 12–23** Temporarily Removing Kernel Software Provider Availability

In the following example, the AES provider is temporarily removed from use. The unload subcommand is useful to prevent a provider from being loaded automatically while the provider is being uninstalled. For example, the unload subcommand would be used when installing a patch that affects the provider.

```
cryptoadm unload provider=aes
```

```
cryptoadm list
...
Kernel software providers:
  des
  aes (inactive)
  arcfour
  blowfish
  ecc
  sha1
  sha2
  md4
  md5
  rsa
  swrand
  n2rng/0
  ncp/0
  n2cp/0
```

The AES provider is unavailable until the Cryptographic Framework is refreshed.

```
svcadm refresh system/cryptosvc
```

```
cryptoadm list
...
Kernel software providers:
  des
```
If a kernel consumer is using the kernel software provider, the software is not unloaded. An error message is displayed and the provider continues to be available for use.

**Example 12–24** Permanently Removing Software Provider Availability

In the following example, the AES provider is removed from use. Once removed, the AES provider does not appear in the policy listing of kernel software providers.

```
$ cryptoadm uninstall provider=aes
$ cryptoadm list
...
Kernel software providers:
  des
  arcfour
  blowfish
  ecc
  sha1
  sha2
  md4
  md5
  rsa
  swrand
  n2rng/0
  ncp/0
  n2cp/0
```

If a kernel consumer is using the kernel software provider, an error message is displayed and the provider continues to be available for use.

**Example 12–25** Reinstalling a Removed Kernel Software Provider

In the following example, the AES kernel software provider is reinstalled.

```
$ cryptoadm install provider=aes \ mechanism=CKM_AES_ECB,CKM_AES_CBC,CKM_AES_CTR,CKM_AES_CCM, CKM_AES_GCM,CKM_AES_GMAC,CKM_AES_CFB128,CKM_AES_XTS,CKM_AES_XCBC_MAC
```
$ cryptoadm list
...
Kernel software providers:
des
aes
arcfour
blowfish
ecc
sha1
sha2
md4
md5
rsa
swrand
n2rng/0
ncp/0
n2cp/0

▼ How to List Hardware Providers

Hardware providers are automatically located and loaded. For more information, see driver.conf(4) man page.

Before You Begin
When you have hardware that expects to be used within the Cryptographic Framework, the hardware registers with the SPI in the kernel. The framework checks that the hardware driver is signed. Specifically, the framework checks that the object file of the driver is signed with a certificate that Sun issues.

For example, the Sun Crypto Accelerator 6000 board (mca), the ncp driver for the cryptographic accelerator on the UltraSPARC T1 and T2 processors (ncp), and the n2cp driver for the UltraSPARC T2 processors (n2cp) plug hardware mechanisms into the framework.

For information about getting your provider signed, see "Binary Signatures for Third-Party Software" on page 216.

1 List the hardware providers that are available on the system.
   % cryptoadm list
   ...
kernel hardware providers:
   ncp/0

2 List the mechanisms that the chip or the board provides.
   % cryptoadm list -m provider=ncp/0
   ncp/0:
   CKM_DSA
   CKM_RSA_X_509
   ...  
   CKM_ECDH1_DERIVE
   CKM_ECDSA
3  List the mechanisms that are available for use on the chip or the board.
   % cryptoadm list -p provider=ncp/0
   ncp/0: all mechanisms are enabled.

▼ How to Disable Hardware Provider Mechanisms and Features

You can selectively disable mechanisms and the random number feature from a hardware provider. To enable them again, see Example 12–26. The hardware in this example, the Sun Crypto Accelerator 1000 board, provides a random number generator.

Before You Begin

You must become an administrator who is assigned the Crypto Management rights profile. For more information, see "How to Use Your Assigned Administrative Rights" on page 157.

● Choose the mechanisms or feature to disable.

List the hardware provider.
   # cryptoadm list
   ...
   Kernel hardware providers:
   dca/0

   ▪ Disable selected mechanisms.
   
   # cryptoadm list -m provider=dca/0
   dca/0: CKM_RSA_PKCS, CKM_RSA_X_509, CKM_DSA, CKM_DES_CBC, CKM_DES3_CBC
   random is enabled.
   # cryptoadm disable provider=dca/0 mechanism=CKM_DES_CBC,CKM_DES3_CBC
   # cryptoadm list -p provider=dca/0
   dca/0: all mechanisms are enabled except CKM_DES_CBC,CKM_DES3_CBC.
   random is enabled.

   ▪ Disable the random number generator.
   
   # cryptoadm list -p provider=dca/0
   dca/0: all mechanisms are enabled. random is enabled.
   # cryptoadm disable provider=dca/0 random
   # cryptoadm list -p provider=dca/0
   dca/0: all mechanisms are enabled. random is disabled.

   ▪ Disable all mechanisms. Do not disable the random number generator.
   
   # cryptoadm list -p provider=dca/0
   dca/0: all mechanisms are enabled. random is enabled.
   # cryptoadm disable provider=dca/0 mechanism=all
   # cryptoadm list -p provider=dca/0
   dca/0: all mechanisms are disabled. random is enabled.

   ▪ Disable every feature and mechanism on the hardware.
   
   # cryptoadm list -p provider=dca/0
   dca/0: all mechanisms are enabled. random is enabled.
Enabling Mechanisms and Features on a Hardware Provider

In the following examples, disabled mechanisms on a piece of hardware are selectively enabled.

```
# cryptoadm list -p provider=dca/0
  dca/0: all mechanisms are enabled except CKM_DES_ECB, CKM_DES3_ECB

  random is enabled.
```

```
# cryptoadm enable provider=dca/0 mechanism=CKM_DES3_ECB
# cryptoadm list -p provider=dca/0
  dca/0: all mechanisms are enabled except CKM_DES_ECB.
  random is enabled.
```

In the following example, only the random generator is enabled.

```
# cryptoadm list -p provider=dca/0
  dca/0: all mechanisms are enabled, except CKM_MD5, CKM_MD5_HMAC, ....
  random is disabled.
```

```
# cryptoadm enable provider=dca/0 random
# cryptoadm list -p provider=dca/0
  dca/0: all mechanisms are enabled, except CKM_MD5, CKM_MD5_HMAC, ....
  random is enabled.
```

In the following example, only the mechanisms are enabled. The random generator continues to be disabled.

```
# cryptoadm list -p provider=dca/0
  dca/0: all mechanisms are enabled, except CKM_MD5, CKM_MD5_HMAC, ....
  random is disabled.
```

```
# cryptoadm enable provider=dca/0 mechanism=all
# cryptoadm list -p provider=dca/0
  dca/0: all mechanisms are enabled. random is disabled.
```

In the following example, every feature and mechanism on the board is enabled.

```
# cryptoadm list -p provider=dca/0
  dca/0: all mechanisms are enabled, except CKM_DES_ECB, CKM_DES3_ECB.
  random is disabled.
```

```
# cryptoadm enable provider=dca/0 all
# cryptoadm list -p provider=dca/0
  dca/0: all mechanisms are enabled. random is enabled.
```

How to Refresh or Restart All Cryptographic Services

By default, the Cryptographic Framework is enabled. When the kcfd daemon fails for any reason, the Service Management Facility (SMF) can be used to restart cryptographic services. For more information, see the `smf(5)` and `svcadm(1M)` man pages. For the effect on zones of restarting cryptographic services, see “Cryptographic Services and Zones” on page 217.
Before You Begin  You must become an administrator who is assigned the Crypto Management rights profile. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1  **Check the status of cryptographic services.**

   % svcs cryptosvc  
   STATE    STIME    PMRI  
   offline  Dec_09  svc:/system/cryptosvc:default

2  **Enable cryptographic services.**

   # svcadm enable svc:/system/cryptosvc

**Example 12–27  Refreshing Cryptographic Services**

In the following example, cryptographic services are refreshed in the global zone. Therefore, kernel-level cryptographic policy in every non-global zone is also refreshed.

   # svcadm refresh system/cryptosvc
Key Management Framework

The Key Management Framework (KMF) feature of Oracle Solaris provides tools and programming interfaces for managing public key objects. Public key objects include X.509 certificates and public/private key pairs. The formats for storing these objects can vary. KMF also provides a tool for managing policies that define the use of X.509 certificates by applications. KMF supports third-party plugins.

- “Managing Public Key Technologies (Overview)” on page 247
- “Key Management Framework Utilities” on page 248
- “Using the Key Management Framework (Tasks)” on page 250

Managing Public Key Technologies (Overview)

The Key Management Framework (KMF) provides a unified approach to managing public key technologies (PKI). Oracle Solaris has several different applications that make use of PKI technologies. Each application provides its own programming interfaces, key storage mechanisms, and administrative utilities. If an application provides a policy enforcement mechanism, the mechanism applies to that application only. With KMF, applications use a unified set of administrative tools, a single set of programming interfaces, and a single policy enforcement mechanism. These features manage the PKI needs of all applications that adopt these interfaces.

KMF unifies the management of public key technologies with the following interfaces:

- **pktool command** – This command manages PKI objects, such as certificates, in a variety of keystores.
- **kmfCfg command** – This command manages the PKI policy database and third-party plugins.

PKI policy decisions include operations such as the validation method for an operation. Also, PKI policy can limit the scope of a certificate. For example, PKI policy might assert that a certificate can be used only for specific purposes. Such a policy would prevent that certificate from being used for other requests.
**KMF library** – This library contains programming interfaces that abstract the underlying keystore mechanism. Applications do not have to choose one particular keystore mechanism, but can migrate from one mechanism to another mechanism. The supported keystores are PKCS #11, NSS, and OpenSSL. The library includes a pluggable framework so that new keystore mechanisms can be added. Therefore, applications that use the new mechanisms would require only minor modifications to use a new keystore.

**Note** – To determine the version of OpenSSL that is running, type `openssl version`. The output is similar to the following:

```
OpenSSL 1.0.0i 19 Apr 2012
```

---

**Key Management Framework Utilities**

KMF provides methods for managing the storage of keys and provides the overall policy for the use of those keys. KMF manages the policy, keys, and certificates for three public key technologies:

- Tokens from PKCS #11 providers, that is, from the Cryptographic Framework
- NSS, that is, Network Security Services
- OpenSSL, a file-based keystore

The `kmfcfg` tool can create, modify, or delete KMF policy entries. The tool also manages plugins to the framework. KMF manages keystores through the `pktool` command. For more information, see the `kmfcfg(1)` and `pktool(1)` man pages, and the following sections.

**KMF Policy Management**

KMF policy is stored in a database. This policy database is accessed internally by all applications that use the KMF programming interfaces. The database can constrain the use of the keys and certificates that are managed by the KMF library. When an application attempts to verify a certificate, the application checks the policy database. The `kmfcfg` command modifies the policy database.
**KMF Plugin Management**

The `kmfcfg` command provides the following subcommands for plugins:

- **list plugin** – Lists plugins that are managed by KMF.
- **install plugin** – Installs the plugin by the module's path name and creates a keystore for the plugin. To remove the plugin from KMF, you remove the keystore.
- **uninstall plugin** – Removes the plugin from KMF by removing its keystore.
- **modify plugin** – Enables the plugin to be run with an option that is defined in the code for the plugin, such as `debug`.

For more information, see the `kmfcfg(1)` man page. For the procedure, see “How to Manage Third-Party Plugins in KMF” on page 260.

**KMF Keystore Management**

KMF manages the keystores for three public key technologies, PKCS #11 tokens, NSS, and OpenSSL. For all of these technologies, the `pktool` command enables you to do the following:

- Generate a self-signed certificate.
- Generate a certificate request.
- Generate a symmetric key.
- Generate a public/private key pair.
- Generate a PKCS #10 certificate signing request (CSR) to be sent to an external certificate authority (CA) to be signed.
- Sign a PKCS #10 CSR.
- Import objects into the keystore.
- List the objects in the keystore.
- Delete objects from the keystore.
- Download a CRL.

For the PKCS #11 and NSS technologies, the `pktool` command also enables you to set a PIN by generating a passphrase:

- Generate a passphrase for the keystore.
- Generate a passphrase for an object in the keystore.

For examples of using the `pktool` utility, see the `pktool(1)` man page and “Using the Key Management Framework (Task Map)” on page 250.
Using the Key Management Framework (Tasks)

This section describes how to use the `pktool` command to manage your public key objects, such as passwords, passphrases, files, keystores, certificates, and CRLs.

### Using the Key Management Framework (Task Map)

The Key Management Framework (KMF) enables you to centrally manage public key technologies.

<table>
<thead>
<tr>
<th>Task</th>
<th>Description</th>
<th>For Instructions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create a certificate.</td>
<td>Creates a certificate for use by PKCS #11, NSS, or OpenSSL.</td>
<td>“How to Create a Certificate by Using the pktool gencert Command” on page 251</td>
</tr>
<tr>
<td>Export a certificate.</td>
<td>Creates a file with the certificate and its supporting keys. The file can be protected with a password.</td>
<td>“How to Export a Certificate and Private Key in PKCS #12 Format” on page 253</td>
</tr>
<tr>
<td>Import a certificate.</td>
<td>Imports a certificate from another system.</td>
<td>“How to Import a Certificate Into Your Keystore” on page 252</td>
</tr>
<tr>
<td>Import a certificate in PKCS #12 format from another system.</td>
<td></td>
<td>Example 13–2</td>
</tr>
<tr>
<td>Generate a passphrase.</td>
<td>Generates a passphrase for access to a PKCS #11 keystore or an NSS keystore.</td>
<td>“How to Generate a Passphrase by Using the pktool setpin Command” on page 254</td>
</tr>
<tr>
<td>Generate a symmetric key.</td>
<td>Generates symmetric keys for use in encrypting files, creating a MAC of a file, and for applications.</td>
<td>“How to Generate a Symmetric Key by Using the pktool Command” on page 220</td>
</tr>
<tr>
<td>Generate a key pair.</td>
<td>Generates a public/private key pair for use with applications.</td>
<td>“How to Generate a Key Pair by Using the pktool genkeypair Command” on page 255</td>
</tr>
<tr>
<td>Generate a PKCS #10 CSR.</td>
<td>Generates a PKCS #10 certificate signing request (CSR) for an external certificate authority (CA) to sign.</td>
<td><code>pktool(1)</code> man page</td>
</tr>
<tr>
<td>Sign a PKCS #10 CSR.</td>
<td>Signs a PKCS #10 CSR.</td>
<td>“How to Sign a Certificate Request by Using the pktool signcsr Command” on page 259</td>
</tr>
<tr>
<td>Add a plugin to KMF.</td>
<td>Installs, modifies, and lists a plugin. Also, removes the plugin from the KMF.</td>
<td>“How to Manage Third-Party Plugins in KMF” on page 260</td>
</tr>
</tbody>
</table>
How to Create a Certificate by Using the \texttt{pktool gencert} Command

This procedure creates a self-signed certificate and stores the certificate in the PKCS #11 keystore. As a part of this operation, an RSA public/private key pair is also created. The private key is stored in the keystore with the certificate.

1. **Generate a self-signed certificate.**
   
   \texttt{\% pktool gencert [keystore=keystore] label=label-name \subject=subject-DN serial=hex-serial-number}
   
   - \texttt{keystore=keystore}  
     Specifies the keystore by type of public key object. The value can be \texttt{nss}, \texttt{pkcs11}, or \texttt{file}. This keyword is optional.
   - \texttt{\label=label-name}  
     Specifies a unique name that the issuer gives to the certificate.
   - \texttt{subject=subject-DN}  
     Specifies the distinguished name for the certificate.
   - \texttt{serial=hex-serial-number}  
     Specifies the serial number in hexadecimal format. The issuer of the certificate chooses the number, such as \texttt{0x0102030405}.

2. **Verify the contents of the keystore.**

   \texttt{\% pktool list}
   
   Found \texttt{number} certificates.
   
   1. (X.509 certificate)
      
      Label: \texttt{label-name}
      ID: \texttt{Fingerprint that binds certificate to private key}
      Subject: \texttt{subject-DN}
      Issuer: \texttt{distinguished-name}
      Serial: \texttt{hex-serial-number}
      
      \texttt{n. ...}

   This command lists all certificates in the keystore. In the following example, the keystore contains one certificate only.

**Example 13–1 Creating a Self-Signed Certificate by Using \texttt{pktool}**

In the following example, a user at \textit{My Company} creates a self-signed certificate and stores the certificate in a keystore for PKCS #11 objects. The keystore is initially empty. If the keystore has not been initialized, the PIN for the softtoken is \texttt{changeme}.

\texttt{\% pktool gencert keystore=pkcs11 label="My Cert" \subject="C=US, O=My Company, OU=Security Engineering Group, CN=MyCA" \serial=0x0000000001}

Enter pin for Sun Software PKCS#11 softtoken: \texttt{Type PIN for token}

\texttt{\% pktool list}

Found \texttt{1} certificates.

1. (X.509 certificate)
How to Import a Certificate Into Your Keystore

This procedure describes how to import a file with PKI information that is encoded with PEM or with raw DER into your keystore. For an export procedure, see Example 13–4.

1 Import the certificate.

   % pktool import keyystore=keystore infile=infile-name label=label-name

2 If you are importing private PKI objects, provide passwords when prompted.

   a. At the prompt, provide the password for the file.

      If you are importing PKI information that is private, such as an export file in PKCS #12 format, the file requires a password. The creator of the file that you are importing provides you with the PKCS #12 password.

      Enter password to use for accessing the PKCS12 file: Type PKCS #12 password

   b. At the prompt, type the password for your keystore.

      Enter pin for Sun Software PKCS#11 softtoken: Type PIN for token

3 Verify the contents of the keystore.

   % pktool list
   Found number certificates.
   1. (X.509 certificate)
      Label: label-name
      ID: Fingerprint that binds certificate to private key
      Subject: distinguished-name
      Issuer: distinguished-name
      Serial: hex-serial-number

Example 13–2 Importing a PKCS #12 File Into Your Keystore

In the following example, the user imports a PKCS #12 file from a third party. The pktool import command extracts the private key and the certificate from the gracedata.p12 file, and stores them in the user’s preferred keystore.

   % pktool import keyystore=pkcs11 infile=gracedata.p12 label=GraceCert
   Enter password to use for accessing the PKCS12 file: Type PKCS #12 password
   Enter pin for Sun Software PKCS#11 softtoken: Type PIN for token
Found 1 certificate(s) and 1 key(s) in gracedata.p12

% pktool list
Found 1 certificates.
1. (X.509 certificate)
   Label: GraceCert
   Subject: C=US, O=My Company, OU=Security Engineering Group, CN=MyCA
   Issuer: C=US, O=My Company, OU=Security Engineering Group, CN=MyCA
   Serial: 0x01

Example 13–3 Importing an X.509 Certificate Into Your Keystore

In the following example, the user imports an X.509 certificate in PEM format into the user’s preferred keystore. This public certificate is not protected with a password. The user’s public keystore is also not protected by a password.

% pktool import keystore=pkcs11 infile=somecert.pem label="TheirCompany Root Cert"
% pktool list
Found 1 certificates.
1. (X.509 certificate)
   Label: TheirCompany Root Cert
   Subject: C=US, O=TheirCompany, OU=Security, CN=TheirCompany Root CA
   Issuer: C=US, O=TheirCompany, OU=Security, CN=TheirCompany Root CA
   Serial: 0x01

▼ How to Export a Certificate and Private Key in PKCS #12 Format

You can create a file in PKCS #12 format to export private keys and their associated X.509 certificate to other systems. Access to the file is protected by a password.

1 Find the certificate to export.
   % pktool list
   Found number certificates.
   1. (X.509 certificate)
      Label: label-name
      ID: Fingerprint that binds certificate to private key
      Subject: subject-DN
      Issuer: distinguished-name
      Serial: hex-serial-number
   2. ...
2 Export the keys and certificate.

Use the keystore and label from the `pktool list` command. Provide a file name for the export file. When the name contains a space, surround the name with double quotes.

```
% pktool export keystore=keystore outfile=outfile-name label=label-name
```

3 Protect the export file with a password.

At the prompt, type the current password for the keystore. At this point, you create a password for the export file. The receiver must provide this password when importing the file.

```
Enter pin for Sun Software PKCS#11 softtoken: Type PIN for token
Enter password to use for accessing the PKCS12 file: Create PKCS#12 password
```

Tip – Send the password separately from the export file. Best practice suggests that you provide the password out of band, such as during a telephone call.

---

Example 13–4 Exporting a Certificate and Private Key in PKCS #12 Format

In the following example, a user exports the private keys with their associated X.509 certificate into a standard PKCS #12 file. This file can be imported into other keystores. The PKCS #11 password protects the source keystore. The PKCS #12 password is used to protect private data in the PKCS #12 file. This password is required to import the file.

```
% pktool list
Found 1 certificates.
1. (X.509 certificate)
   Label: My Cert
   Subject: C=US, O=My Company, OU=Security Engineering Group, CN=MyCA
   Issuer: C=US, O=My Company, OU=Security Engineering Group, CN=MyCA
   Serial: 0x01

% pktool export keystore=pkcs11 outfile=mydata.p12 label="My Cert"
Enter pin for Sun Software PKCS#11 softtoken: Type PIN for token
Enter password to use for accessing the PKCS12 file: Create PKCS#12 password
```

The user then telephones the recipient and provides the PKCS #12 password.

---

∇ How to Generate a Passphrase by Using the `pktool setpin` Command

You can generate a passphrase for an object in a keystore, and for the keystore itself. The passphrase is required to access the object or keystore. For an example of generating a passphrase for an object in a keystore, see Example 13–4.
1 Generate a passphrase for access to a keystore.

   % pktool setpin keystore=nss|pkcs11 dir=directory

   The initial password for a PKCS #11 keystore is change Me. The initial password for an NSS keystore is an empty password.

2 Answer the prompts.

   When prompted for the current token passphrase, type the token PIN for a PKCS #11 keystore, or press the Return key for an NSS keystore.

   Enter current token passphrase: Type PIN or press the Return key
   Create new passphrase: Type the passphrase that you want to use
   Re-enter new passphrase: Retype the passphrase
   Passphrase changed.

   The keystore is now protected by passphrase. If you lose the passphrase, you lose access to the objects in the keystore.

Example 13–5 Protecting a Keystore With a Passphrase

   The following example shows how to set the passphrase for an NSS database. Because no passphrase has been created, the user presses the Return key at the first prompt.

   % pktool setpin keystore=nss dir=/var/nss
   Enter current token passphrase: Press the Return key
   Create new passphrase: has8n0NdaH
   Re-enter new passphrase: has8n0NdaH
   Passphrase changed.

How to Generate a Key Pair by Using the pktool genkeypair Command

   Some applications require a public/private key pair. In this procedure, you create these key pairs and store them.

   1 (Optional) If you plan to use a keystore, create the keystore.

   ▪ To create and initialize a PKCS #11 keystore, see “How to Generate a Passphrase by Using the pktool setpin Command” on page 254.

   ▪ To create and initialize an NSS keystore, see Example 13–5.
2 Create the key pair.

Use one of the following methods.

- Create the key pair, and store the key pair in a file.
  File-based keys are created for applications that read keys directly from files on the disk. Typically, applications that directly use OpenSSL cryptographic libraries require that you store the keys and certificates for the application in files.

  Note – The file keystore does not support elliptic curve (ec) keys and certificates.

  ```bash
  % pktool genkeypair keystore=file outkey=key-filename
  [format=der|pem] [keytype=rsa|dsa] [keylen=key-size]
  keystore=file
  The value file specifies the file type of storage location for the key.
  outkey=key-filename
  Specifies the name of the file where the key pair is stored.
  format=der|pem
  Specifies the encoding format of the key pair. der output is binary, and pem output is ASCII.
  keytype=rsa|dsa
  Specifies the type of key pair that can be stored in a file keystore. For definitions, see DSA and RSA.
  keylen=key-size
  Specifies the length of the key in bits. The number must be divisible by 8. To determine possible key sizes, use the cryptoadm list -vm command.

- Create the key pair, and store it in a PKCS #11 keystore.
  You must complete Step 1 before using this method.

  The PKCS #11 keystore is used to store objects on a hardware device. The device could be a Sun Crypto Accelerator 6000 card, a trusted platform module (TPM) device, or a smart card that is plugged into the Cryptographic Framework. PKCS #11 can also be used to store objects in the soft token, or software-based token, which stores the objects in a private subdirectory on the disk. For more information, see the pkcs11_softtoken(5) man page.

  You can retrieve the key pair from the keystore by a label that you specify.

  ```bash
  % pktool genkeypair label=key-label
  [token=token[:manuf:serial]]
  [keytype=rsa|dsa|ec] [curve=ECC-Curve-Name]
  [keylen=key-size] [listcurves]
Generate the key pair, and store it in an NSS keystore.

The NSS keystore is used by servers that rely on NSS as their primary cryptographic interface.

You must complete Step 1 before using this method.

```bash
% pktool keystore=nss genkeypair
   label=key-nickname
   [token=token[:manuf[:serial]]]
   [dir=directory-path] [prefix=database-prefix]
   [keytype=rsa|dsa|ec [curve=ECC-Curve-Name]]
   [keylen=key-size] [listcurves]

keystore=nss
   The value nss specifies the NSS type of storage location for the key.

label=nickname
   Specifies a label for the key pair. The key pair can be retrieved from the keystore by its label.

token=token[:manuf[:serial]]
   Specifies the token name. By default, the token is Sun Software PKCS#11 softtoken.

dir=directory
   Specifies the directory path to the NSS database. By default, directory is the current directory.

prefix=database-prefix
   Specifies the prefix to the NSS database. The default is no prefix.

keytype=rsa|dsa|ec [curve=ECC-Curve-Name]
   Specifies the keypair type. For the elliptic curve (ec) type, optionally specifies a curve name. Curve names are listed as output to the listcurves option.
```
keylen=key-size
    Specifies the length of the key in bits. The number must be divisible by 8.

listcurves
    Lists the elliptic curve names that can be used as values to the curve= option for an ec key type.

3 (Optional) Verify that the key exists.
Use one of the following commands, depending on where you stored the key:

- **Verify the key in the key-filename file.**
  
  ```
  % pktool list keystore=file objtype=key infile=key-filename
  Found n keys.
  Key #1 - keytype:location (keylen)
  ```

- **Verify the key in the PKCS #11 keystore.**
  
  ```
  $ pktool list objtype=key
  Enter PIN for keystore:
  Found n keys.
  Key #1 - keytype:location (keylen)
  ```

- **Verify the key in the NSS keystore.**
  
  ```
  % pktool list keystore=nss dir=directory objtype=key
  ```

**Example 13–6 Creating a Key Pair by Using the pktool Command**

In the following example, a user creates a PKCS #11 keystore for the first time. After determining the key sizes for RSA key pairs, the user then generates a key pair for an application. Finally, the user verifies that the key pair is in the keystore. The user notes that the second instance of the RSA key pair can be stored on hardware. Because the user does not specify a token argument, the key pair is stored as a Sun Software PKCS#11 softtoken.

```
# pktool setpin
Create new passphrase: Easily remembered, hard-to-detect password
Re-enter new passphrase: Retype password
Passphrase changed.
% cryptoadm list -vm | grep PAIR
...  
CM_DSA_KEY_PAIR_GEN 512 3072 . . . . . . . . X . . .
CM_RSA_PKCS_KEY_PAIR_GEN 256 8192 . . . . . . . X . . .
...  
CM_RSA_PKCS_KEY_PAIR_GEN 256 2048 X . . . . . . . . X . . .

ecc: CMK_EC_KEYPAIR_GEN,CMK_ECDH1_DERIVE,CMK_ECDSA,CMK_ECDSA_SHA1
% pktool genkeypair label=specialappkeypair keytype=rsa keylen=2048
Enter PIN for Sun Software PKCS#11 softtoken : Type password
```

```
Creating a Key Pair That Uses the Elliptic Curve Algorithm

In the following example, a user adds an elliptic curve (ec) key pair to the key store, specifies a curve name, and verifies that the key pair is in the key store.

```bash
% pktool genkeypair listcurves
secp112r1, secp112r2, secp128r1, secp128r2, secp160k1
.
.
c2pnb304w1, c2tnb359v1, c2pnb368w1, c2tnb431r1, prime192v2
prime192v3
% pktool genkeypair label=eckeypair keytype=ec curves=c2tnb431r1
% pktool list
Enter PIN for Sun Software PKCS#11 softtoken : Type password

Found 2 keys.
Key #1 · keypair: specialappkeypair (2048 bits)
Key #2 · keypair: eckeypair (c2tnb431r1)
```

How to Sign a Certificate Request by Using the `pktool signcsr` Command

This procedure is used to sign a PKCS #10 certificate signing request (CSR). The CSR can be in PEM or DER format. The signing process issues an X.509 v3 certificate. To generate a PKCS #10 CSR, see the `pktool(1)` man page.

Before You Begin

You are a certificate authority (CA), you have received a CSR, and it is stored in a file.

1. Collect the following information for the required arguments to the `pktool signcsr` command:
   - `signkey`: If you have stored the signer’s key in a PKCS #11 key store, `signkey` is the `label` that retrieves this private key.
     - If you have stored the signer’s key in an NSS key store or a file key store, `signkey` is the file name that holds this private key.
   - `csr`: Specifies the file name of the CSR.
   - `serial`: Specifies the serial number of the signed certificate.
   - `outcer`: Specifies the file name for the signed certificate.
   - `issuer`: Specifies your CA issuer name in distinguished name (DN) format.
Sign the request and issue the certificate.

For example, the following command signs the certificate with the signer’s key from the PKCS #11 repository:

```bash
# pktool signcsr signkey=CASigningKey \
csr=fromExampleCoCSR \
serial=0x12345678 \
outcert=ExampleCoCert2010 \
issuer="O=Oracle Corporation, \n     OU=Oracle Solaris Security Technology, L=Redwood City, ST=CA, C=US, \n     CN=rootsign Oracle"
```

The following command signs the certificate with the signer’s key from a file:

```bash
# pktool signcsr signkey=CASigningKey \
csr=fromExampleCoCSR \
serial=0x12345678 \
outcert=ExampleCoCert2010 \
issuer="O=Oracle Corporation, \n     OU=Oracle Solaris Security Technology, L=Redwood City, ST=CA, C=US, \n     CN=rootsign Oracle"
```

Send the certificate to the requester.

You can use email, a web site, or other mechanism to deliver the certificate to the requester.

For example, you could use email to send the `ExampleCoCert2010` file to the requester.

## How to Manage Third-Party Plugins in KMF

You identify your plugin by giving it a keystore name. When you add the plugin to KMF, the software identifies it by its keystore name. The plugin can be defined to accept an option. This procedure includes how to remove the plugin from KMF.

1 Install the plugin.

   ```bash
   % /usr/bin/kmfcfg install keystore=keystore-name \
   modulepath=path-to-plugin [option="option-string"]
   ```

   where

   - `keystore-name` – Specifies a unique name for the keystore that you provide.
   - `path-to-plugin` – Specifies the full path to the shared library object for the KMF plugin.
   - `option-string` – Specifies an optional argument to the shared library object.
2 List the plugins.
   
   `% kmfcfg list plugin
   
   keystore-name:path-to-plugin [(built-in)] | [:option=option-string]

3 To remove the plugin, uninstall it and verify its removal.

   `% kmfcfg uninstall keystore=keystore-name
   `% kmfcfg plugin list

Example 13–8 Calling a KMF Plugin With an Option

In the following example, the administrator stores a KMF plugin in a site-specific directory. The plugin is defined to accept a debug option. The administrator adds the plugin and verifies that the plugin is installed.

```
# /usr/bin/kmfcfg install keystore=mykmfplug modulepath=/lib/security/site-modules/mykmfplug.so
# kmfcfg list plugin
KMF plugin information: 
-----------------------
  pkcs11:kmf_pkcs11.so.1 (built-in)
  file:kmf_openssl.so.1 (built-in)
  nss:kmf_nss.so.1 (built-in)
  mykmfplug:/lib/security/site-modules/mykmfplug.so
# kmfcfg modify plugin keystore=mykmfplug option="debug"
# kmfcfg list plugin
KMF plugin information: 
-----------------------
  pkcs11:kmf_pkcs11.so.1 (built-in)
  file:kmf_openssl.so.1 (built-in)
  nss:kmf_nss.so.1 (built-in)
  mykmfplug:/lib/security/site-modules/mykmfplug.so;option=debug
```

The plugin now runs in debugging mode.
PART V

Authentication Services and Secure Communication

This section discusses authentication services that can be configured on a non-networked system, or between two systems.

- Chapter 14, “Using Pluggable Authentication Modules”
- Chapter 15, “Using Secure Shell”
- Chapter 16, “Secure Shell (Reference)”
- Chapter 17, “Using Simple Authentication and Security Layer”
- Chapter 18, “Network Services Authentication (Tasks)”

To configure a network of authenticated users and systems, see Part VI, “Kerberos Service.”
Using Pluggable Authentication Modules

This chapter covers the Pluggable Authentication Module (PAM) framework. PAM provides a method to “plug in” authentication services into the Oracle Solaris OS. PAM provides support for multiple authentication services when accessing a system.

- “PAM (Overview)” on page 265
- “PAM (Tasks)” on page 267
- “PAM Configuration (Reference)” on page 272

PAM (Overview)

The Pluggable Authentication Module (PAM) framework lets you “plug in” new authentication services without changing system services, such as login, su, and ssh. You can also use PAM to integrate UNIX login with other security mechanisms such as Kerberos. Mechanisms for account, credential, session, and password management can also be “plugged in” by using this framework.

Benefits of Using PAM

The PAM framework enables you to configure the use of system services (such as, su, login, or ssh) for user authentication. Some benefits that PAM provides are as follows:

- Flexible configuration policy
  - Per-application authentication policy
  - Per-user authentication policy
  - The ability to choose a default authentication mechanism
  - The ability to require multiple authorizations on high-security systems
- Ease of use for the end user
  - No retying of passwords if the passwords are the same for different authentication services
The ability to prompt the user for passwords for multiple authentication services without requiring the user to type multiple commands

The ability to pass options to the user authentication services

The ability to implement a site-specific security policy without having to change the system entry services

**Introduction to the PAM Framework**

The PAM framework consists of four parts:

- PAM consumers
- PAM library
- PAM configuration
- PAM service modules, also referred to as providers

The framework provides a uniform way for authentication-related activities to take place. This approach enables application developers to use PAM services without having to know the semantics of the policy. Algorithms are centrally supplied. The algorithms can be modified independently of the individual applications. With PAM, administrators can tailor the authentication process to the needs of a particular system without having to change any applications. Adjustments are made through the PAM configuration.

The following figure illustrates the PAM architecture. Applications communicate with the PAM library through the PAM application programming interface (API). PAM modules communicate with the PAM library through the PAM service provider interface (SPI). Thus, the PAM library enables applications and modules to communicate with each other.
Changes to PAM for This Release

The PAM framework included in the Oracle Solaris 11.1 release includes several new features including:

- definitive control flag
- per-service configuration
- per-user configuration

PAM (Tasks)

This section discusses some tasks that might be required to make the PAM framework use a particular security policy. You should be aware of some security issues that are associated with the PAM configuration files. For information about the security issues, see "Planning for Your PAM Implementation" on page 268.
# PAM (Task Map)

<table>
<thead>
<tr>
<th>Task</th>
<th>Description</th>
<th>For Instructions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Plan for your PAM installation.</td>
<td>Consider configuration issues and make decisions about them before you start the software configuration process.</td>
<td>“Planning for Your PAM Implementation” on page 268</td>
</tr>
<tr>
<td>Add new PAM modules.</td>
<td>Sometimes, site-specific modules must be written and installed to cover requirements that are not part of the generic software. This procedure explains how to install these new PAM modules.</td>
<td>“How to Add a PAM Module” on page 269</td>
</tr>
<tr>
<td>Assign a new PAM policy to a user.</td>
<td>Establish specific authentication requirements for multiple services to be assigned to a specific user.</td>
<td>“How to Assign a Customized PAM Policy to a User” on page 271</td>
</tr>
<tr>
<td>Assign a new rights profile to all users.</td>
<td>Establish specific authentication requirements for multiple services to be assigned to all users on the system.</td>
<td>“How to Assign a New Rights Policy to All Users” on page 272</td>
</tr>
<tr>
<td>Block access through ~/.rhosts.</td>
<td>Further increase security by preventing access through ~/.rhosts.</td>
<td>“How to Prevent Rhost-Style Access From Remote Systems With PAM” on page 270</td>
</tr>
<tr>
<td>Initiate error logging.</td>
<td>Start the logging of PAM error messages through syslog.</td>
<td>“How to Log PAM Error Reports” on page 270</td>
</tr>
</tbody>
</table>

## Planning for Your PAM Implementation

As delivered, the PAM configuration implements the standard security policy. This policy should work in many situations. If you need to implement a different security policy, here are the issues that you should focus on:

- Determine what your needs are, especially which PAM service modules you should select.
- Identify the services that need special configuration options. Use the service name `other` to provide defaults for services, if appropriate.
- Decide the order in which the modules should be run.
- Select the control flag for each module. See “How PAM Stacking Works” on page 275 for more information about all of the control flags.
- Choose any options that are necessary for each module. The man page for each module should list any special options.
Here are some suggestions to consider before you change the PAM configuration:

- Use service name other entries for each module type so that every application does not have to be included in the PAM configuration.
- Make sure to consider the security implications of the control flags.
- Review the man pages that are associated with the modules. These man pages can help you understand how each module functions, what options are available, and the interactions between stacked modules.

**Caution** – If the PAM configuration is misconfigured or becomes corrupted, no user might be able to log in. Because the `su log in` command does not use PAM, the root password would then be required to boot the machine into single-user mode and fix the problem.

After you change the PAM configuration, review the changes as much as possible while you still have system access to correct problems. Test all the commands that might have been affected by your changes.

▼ **How to Add a PAM Module**

This procedure shows how to add a new PAM module. New modules can be created to cover site-specific security policies or to support third party applications.

**Before You Begin**

You must assume the root role. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1. **Determine which control flags and which options should be used.**
   Refer to “How PAM Stacking Works” on page 275 for information on the control flags.

2. **Ensure that the ownership and permissions are set so that the module file is owned by root and the permissions are 555.**

3. **Use p edit to edit an appropriate PAM configuration file and add this module to the appropriate services.**
   Changes can be made to either `/etc/pam.conf` or `/etc/pam.d/service`.

4. **Verify that the module has been added properly.**
   You must test in case the configuration file is misconfigured. Login using a direct service, such as ssh, and run the su command.
How to Prevent Rhost-Style Access From Remote Systems With PAM

Note – The rsh service is not enabled by default. To provide a more secure connection, use the ssh command instead.

Before You Begin

You must assume the root role. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1 Remove all of the lines that include rhosts_auth.so.1 from the PAM configuration files.

This step prevents the reading of the ~/.rhosts files during an rlogin session. Therefore, this step prevents unauthenticated access to the local system from remote systems. All rlogin access requires a password, regardless of the presence or contents of any ~/.rhosts or /etc/hosts.equiv files.

2 Disable the rsh service.

To prevent unauthenticated access to the system, remember to disable the rsh service.

# svcadm disable network/shell:default

3 Disable the rlogin service.

Disable the rlogin service as well, if necessary.

# svcadm disable network/login:rlogin

How to Log PAM Error Reports

Before You Begin

You must assume the root role. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1 Determine which system-log service instance is online.

# svcsc system-log

<table>
<thead>
<tr>
<th>STATE</th>
<th>STIME</th>
<th>FMRI</th>
</tr>
</thead>
<tbody>
<tr>
<td>disabled</td>
<td>13:11:55</td>
<td>svc:/system/system-log:rsyslog</td>
</tr>
<tr>
<td>online</td>
<td>13:13:27</td>
<td>svc:/system/system-log:default</td>
</tr>
</tbody>
</table>

Note – If the rsyslog service instance is online, modify the rsyslog.conf file.

2 Configure the /etc/syslog.conf file for the level of logging that you need.

See the syslog.conf(4) man page for more information about the logging levels. Most PAM error reporting is done to the LOG_AUTH facility.
3 Refresh the configuration information for the system-log service.

    # svcadm refresh system-log:default

---

Note – Refresh the system-log:rsyslog service instance if the rsyslog service is online.

How to Assign a Customized PAM Policy to a User

**Before You Begin**

You must assume the root role. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1 Create a new PAM policy configuration file.

   See the comments in the text below for a description of the effects of the file.

   ```
   # cat /etc/opt/pam_policy/custom-config
   
   # PAM configuration which uses UNIX authentication for console logins, # LDAP for SSH keyboard-interactive logins, and denies telnet logins. #
   #
   login auth requisite pam_authtok_get.so.1
   login auth required pam_dhkeys.so.1
   login auth required pam_unix_auth.so.1
   login auth required pam_unix_cred.so.1
   login auth required pam_dial_auth.so.1
   
   sshd-kbdint auth requisite pam_authtok_get.so.1
   sshd-kbdint auth binding pam_unix_auth.so.1 server_policy
   sshd-kbdint auth required pam_unix_cred.so.1
   sshd-kbdint auth required pam_ldap.so.1
   
   telnet auth requisite pam_deny.so.1
   telnet account requisite pam_deny.so.1
   telnet session requisite pam_deny.so.1
   telnet password requisite pam_deny.so.1
   ```

2 Check the file permissions on the new file.

   The file must be owned by root and cannot be group or world writable.

   ```
   # ls -l /etc/opt/pam_policy
   total 5
   -r--r--r-- 1 root 4570 Jun 21 12:00 custom-config
   ```

3 Assign the new PAM policy to a user.

   The custom-config file in /etc/opt/pam_policy is assigned to the user named jill.

   ```
   # useradd -K pam_policy=/etc/opt/pam_policy/custom-config jill
   ```
How to Assign a New Rights Policy to All Users

Before You Begin
You must assume the root role. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1 Create a new rights policy.

   In this example, the ldap PAM policy is used.

   # profiles -p "PAM Per-User Policy of LDAP" \ 
   'set desc="Profile which sets pam_policy=ldap"; \ 
   set pam_policy=ldap; exit;'

2 Assign the new rights profile to all users.

   Use pfedit to add the new policy to the PROFS_GRANTED declaration.

   # cat /etc/security/policy.conf

   AUTHS_GRANTED=
   PROFS_GRANTED=Basic Solaris User,PAM Per-User Policy of LDAP
   CONSOLE_USER=Console User

Example 14–1 Assigning a Rights Profile to a User

   If a profile has been created as in step 1 in the previous procedure, that rights profile can be assigned to a user using the following command:

   # usermod -P +"PAM Per-User Policy of LDAP" jill

PAM Configuration (Reference)

System applications, like ssh, that use the PAM service are configured in the PAM configuration files. See the pam.conf(4) man page for more information. These configuration files include the /etc/pam.conf file, as well as service specific files placed in /etc/pam.d. Changes to these files affect all users on the system. The service specific PAM configuration files are the preferred mechanism for configuring PAM, since their granularity means a mistake in a file only affects that service. Also, adding new PAM services is simplified to copying a single file. The service specific files allow for better interoperability with other cross-platform PAM applications, since /etc/pam.d is the default configuration in most PAM implementations.

In addition, PAM policy files can be used to create authentication policies for individual services and assign those policies either to an individual, a group of individuals, or all users, as needed. The default PAM policy files are located in /etc/security/pam_policy. The PAM policy files provide the ability to set or change the authentication policy for one or more users in a safe and reliable manner.
The system administrator manages the PAM configuration files. An incorrect order of entries in these files can cause unforeseen side effects. For example, a badly configured file can lock out users so that single-user mode becomes necessary for repair. For a description of setting the order, see “How PAM Stacking Works” on page 275.

### PAM Configuration Search Order

The PAM configuration information in the PAM configuration files is collected by the PAM library in the following order:

1. The service name is looked for in /etc/pam.conf.
2. /etc/pam.d/service is checked.
3. The service name other is checked in /etc/pam.conf.
4. The /etc/pam.d/other file is checked.

This order allows for an existing /etc/pam.conf file to work with the per-service PAM configuration files located in /etc/pam.d.

### PAM Configuration File Syntax

The /etc/pam.conf file and the PAM policy files use a syntax that is different than the service specific files. The syntax of each of these files is described below.

The entries in /etc/pam.conf or in PAM policy files are in one of the following formats:

```
service-name module-type control-flag module-path module-options

service-name module-type include path-to-included-PAM-configuration

service-name include
```

- **service-name**: The case insensitive name of the service, for example, login or passwd. An application can use different service names for the services that the application provides. For example, the Oracle Solaris secure shell daemon uses these service names: sshd-none, sshd-password, sshd-kbdint, sshd-pubkey, and sshd-hostbased.

- **module-type**: The service name other is a predefined name that is used as a wildcard service-name. If a particular service-name is not found in the configuration file, the configuration for other is used.

- **module-path**: The type of service, that is, auth, account, session, or password.
control-flag

Indicates the role of the module in determining the integrated success or failure value for the service. Valid control flags are binding, definitive, include, optional, required, requisite, and sufficient. See “How PAM Stacking Works” on page 275 for information on the use of these flags.

module-path

The path to the library object that implements the service. If the path name is not absolute, the path name is assumed to be relative to /usr/lib/security/$ISA/. Use the architecture-dependent macro $ISA to cause libpam to look in the directory for the particular architecture of the application.

module-options

Options that are passed to the service modules. A module's man page describes the options that are accepted by that module. Typical module options include nowarn and debug.

path-to-included-PAM-configuration

Gives the full path to a separate PAM configuration file or a path name relative to the /usr/lib/security directory.

The per-service configuration files located in /etc/pam.d use the same syntax as pam.conf, but don't include the service name. When using the per-service configuration files, the name of the file is the service name. For instance, /etc/pam.d/cron includes the PAM configuration for the cron command.

Per User Authentication Policy

The pam_user_policy PAM module allows system administrators to specify PAM configurations on a per-user basis. The pam_policy key for the user needs to provide the path to a user-specific PAM configuration file. See the pam_user_policy(5) man page for more information.

Here are some ways to establish a per-user authentication policy:

- Create a new PAM policy file for a user and then use the usermod command to assign the policy to the user. This procedure is documented in “How to Assign a Customized PAM Policy to a User” on page 271.
- Use the usermod command to assign a pre-defined policy to a user.
- Assign a rights profile that includes a pam_policy key to a user using the -P option to the usermod command.
Assign a rights profile that includes a `pam_policy` key to all users by adding it to the `PROFS_GRANTED` key in `/etc/security/policy.conf`. This procedure is documented in "How to Assign a New Rights Policy to All Users" on page 272

**How PAM Stacking Works**

When an application calls one of the following functions, `libpam` reads the PAM configuration files to determine which modules participate in the operation for this service:

- `pam_authenticate`(3PAM)
- `pam_acct_mgmt`(3PAM)
- `pam_setcred`(3PAM)
- `pam_open_session`(3PAM)
- `pam_close_session`(3PAM)
- `pam_chauthtok`(3PAM)

If the configuration files contain only one module for an operation for this service such as authentication or account management, the result of that module determines the outcome of the operation. For example, the default authentication operation for the `passwd` application contains one module, `pam_passwd_auth.so.1`:

```bash
passwd auth required pam_passwd_auth.so.1
```

If, on the other hand, there are multiple modules defined for the service's operation, those modules are said to be stacked and a PAM stack exists for that service. For example, consider the case where `/etc/pam.d/login` contains the following entries:

```bash
auth definitive pam_user_policy.so.1
auth requisite pam_authtok_get.so.1
auth required pam_unix_auth.so.1
auth required pam_dhkeys.so.1
auth required pam_unix_cred.so.1
auth required pam_dial_auth.so.1
```

These entries represent a sample auth stack for the `login` service. To determine the outcome of this stack, the result codes of the individual modules require an integration process. In the integration process, the modules are executed in order as specified in the file. Each success or failure code is integrated in the overall result depending on the module's control flag. The control flag can cause early termination of the stack. For example, a requisite or definitive module might fail, or a sufficient, definitive, or binding module might succeed. After the stack has been processed, the individual results are combined into a single, overall result that is delivered to the application.
The control flag indicates the role that a PAM module plays in determining access to the service. The control flags and their effects are:

- **Binding** – Success in meeting a binding module’s requirements returns success immediately to the application if no previous required modules have failed. If these conditions are met, then no further execution of modules occurs. Failure causes a required failure to be recorded and the processing of modules to be continued.

- **Definitive** – Success in meeting a definitive module’s requirements returns success immediately to the application if no previous required modules have failed. If a previous required module failed, that failure is immediately returned to the application with no further execution of modules. Failure results in an immediate error return with no further execution of modules.

- **Include** – Adds lines from a separate PAM configuration file to be used at this point in the PAM stack. This flag does not control success or failure behaviors. When a new file is read, the PAM include stack is incremented. When the stack check in the new file finishes, the include stack value is decremented. When the end of a file is reached and the PAM include stack is 0, then the stack processing ends. The maximum number for the PAM include stack is 32.

- **Optional** – Success in meeting an optional module’s requirements is not necessary for using the service. Failure causes an optional failure to be recorded.

- **Required** – Success in meeting a required module’s requirements is necessary for using the service. Failure results in an error return after the remaining modules for this service have been executed. Final success for the service is returned only if no binding or required modules have reported failures.

- **Requisite** – Success in meeting a requisite module’s requirements is necessary for using the service. Failure results in an immediate error return with no further execution of modules. All requisite modules for a service must return success for the function to be able to return success to the application.

- **Sufficient** – If no previous required failures have occurred, success in a sufficient module returns success to the application immediately with no further execution of modules. Failure causes an optional failure to be recorded.

The following two diagrams show how access is determined in the integration process. The first diagram indicates how success or failure is recorded for each type of control flag. The second diagram shows how the integrated value is determined.
If no previous required failures, record required success.
- If there are previous required failures, record required success.

- Record required failure if no other required failures.

Increment include level.

- Record required success.
- Record requisite success.

- Record required failure if no other required failures.

If there are previous required failures, record required success.

- Record requisite failure or first required failure.

- If no previous required failures, record required success.

- If there are previous required failures, record required success.

- Record optional failure if no other required failures.
This example shows the default definitions for authentication management in the `/etc/pam.d/other` file. These definitions are used if no service-specific definitions have been configured.

```bash
# Default definitions for Authentication management
# Used when service name is not explicitly mentioned for authentication
#
auth definitive  pam_user_policy.so.1
auth requisite    pam_authtok_get.so.1
auth required    pam_dhkeys.so.1
auth required    pam_unix_auth.so.1
auth required    pam_unix_cred.so.1

First, the security policy for the user is checked using the `pam_user_policy` module. The definitive control flag selects that if the evaluation of the security policy succeeds, the service
returns success to the application, since no other modules have been checked at this point. If the request fails, then a failure message is sent to the application and no further checking is done. If no policy is set for the user, then the next module is executed.

If a per-user PAM policy isn’t specified for this user, then the `pam_authok_get` module is executed. The control flag for this module is set to `requisite`. If `pam_authok_get` fails, then the authentication process ends and the failure is returned to the service.

If `pam_authok_get` does not fail, then the next three modules are executed. These modules are configured with the required control flag, so that the process continues regardless of whether an individual failure is returned. After `pam_unix_cred` is executed, no modules remain. At this point, if all the modules succeeded, the user is granted access. If either `pam_dhkeys`, `pam_unix_auth`, or `pam_unix_cred` has returned a failure, the user is denied access.
The Secure Shell feature of Oracle Solaris provides secure access to a remote host over an unsecured network. The shell provides commands for remote login and remote file transfer. The chapter covers the following topics:

- “Secure Shell (Overview)” on page 281
- “Secure Shell and the OpenSSH Project” on page 283
- “Secure Shell and FIPS-140” on page 285
- “Configuring Secure Shell (Tasks)” on page 285
- “Using Secure Shell (Tasks)” on page 292

For reference information, see Chapter 16, “Secure Shell (Reference).”

**Secure Shell (Overview)**

Secure Shell in Oracle Solaris is built on top of the Open Source toolkit, OpenSSL, which implements the Secure Sockets Layer and Transport Layer Security.

Two distinct versions of the toolkit are available in Oracle Solaris.

- Version 1.0.0 is the default version that Secure Shell runs on.
- Version 0.9.8 implements FIPS-140, a U.S. government computer security standard for cryptography modules.

To use Secure Shell in FIPS-140 (FIPS) mode, see "Secure Shell and FIPS-140” on page 285.

In Secure Shell, authentication is provided by the use of passwords, public keys, or both. All network traffic is encrypted. Thus, Secure Shell prevents a would-be intruder from being able to read an intercepted communication. Secure Shell also prevents an adversary from spoofing the system.

Secure Shell can also be used as an on-demand virtual private network (VPN). A VPN can forward X Window system traffic or can connect individual port numbers between the local machines and remote machines over an encrypted network link.
With Secure Shell, you can perform these actions:

- Log in to another host securely over an unsecured network.
- Copy files securely between the two hosts.
- Run commands securely on the remote host.

On the server side, Secure Shell supports Version 2 (v2) of the Secure Shell protocol. On the client side, in addition to v2, the client supports Version 1 (v1). For information about v1, see *System Administration Guide: Security Services*.

### Secure Shell Authentication

Secure Shell provides public key and password methods for authenticating the connection to the remote host. Public key authentication is a stronger authentication mechanism than password authentication, because the private key never travels over the network.

The authentication methods are tried in the following order. When the configuration does not satisfy an authentication method, the next method is tried.

- **GSS-API** – Uses credentials for GSS-API mechanisms such as `mech_krb5` (Kerberos V) and `mech_dh` (AUTH_DH) to authenticate clients and servers. For more information about GSS-API, see "Introduction to GSS-API" in *Developer’s Guide to Oracle Solaris 11 Security*.
- **Host-based authentication** – Uses host keys and rhosts files. Uses the client’s RSA and DSA public/private host keys to authenticate the client. Uses the rhosts files to authorize clients to users.
- **Public key authentication** – Authenticates users with their RSA and DSA public/private keys.
- **Password authentication** – Uses PAM to authenticate users. Keyboard authentication method in v2 allows for arbitrary prompting by PAM. For more information, see the `SECURITY` section in the `sshd(1M)` man page.

The following table shows the requirements for authenticating a user who is trying to log into a remote host. The user is on the local host, the client. The remote host, the server, is running the `sshd` daemon. The table shows the Secure Shell authentication methods, the compatible protocol versions, and the host requirements.

<table>
<thead>
<tr>
<th>Authentication Method</th>
<th>Local Host (Client) Requirements</th>
<th>Remote Host (Server) Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>GSS-API</td>
<td>Initiator credentials for the GSS mechanism.</td>
<td>Acceptor credentials for the GSS mechanism. For more information, see &quot;Acquiring GSS Credentials in Secure Shell&quot; on page 304.</td>
</tr>
</tbody>
</table>
### Secure Shell in the Enterprise


### Secure Shell and the OpenSSH Project

The Secure Shell is a fork of the OpenSSH ([http://www.openssh.com](http://www.openssh.com)) project. Security fixes for vulnerabilities that are discovered in later versions of OpenSSH are integrated into Secure Shell, as are individual bug fixes and features. Internal development continues on the Secure Shell fork.

#### TABLE 15-1  Authentication Methods for Secure Shell  (Continued)

<table>
<thead>
<tr>
<th>Authentication Method</th>
<th>Local Host (Client) Requirements</th>
<th>Remote Host (Server) Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>Host-based</td>
<td>User account</td>
<td>User account</td>
</tr>
<tr>
<td></td>
<td>HostbasedAuthentication yes in</td>
<td>HostbasedAuthentication yes in</td>
</tr>
<tr>
<td></td>
<td>/etc/ssh/ssh_config</td>
<td>/etc/ssh/sshd_config</td>
</tr>
<tr>
<td></td>
<td>IgnoreRhosts no in /etc/ssh/sshd_config</td>
<td>IgnoreRhosts no in /etc/ssh/sshd_config</td>
</tr>
<tr>
<td></td>
<td>Local host entry in /etc/ssh/hosts.equiv, /etc/hosts.equiv, ~/.rhosts, or ~/.shosts</td>
<td>Local host entry in /etc/ssh/hosts.equiv, /etc/hosts.equiv, ~/.rhosts, or ~/.shosts</td>
</tr>
<tr>
<td>RSA or DSA public key</td>
<td>User account</td>
<td>User account</td>
</tr>
<tr>
<td></td>
<td>Private key in ~/.ssh/id_rsa or ~/.ssh/id_dsa</td>
<td>User's public key in ~/.ssh/authorized_keys</td>
</tr>
<tr>
<td></td>
<td>User's public key in ~/.ssh/id_rsa.pub or ~/.ssh/id_dsa.pub</td>
<td></td>
</tr>
<tr>
<td>Password-based</td>
<td>User account</td>
<td>User account</td>
</tr>
<tr>
<td></td>
<td>Supports PAM.</td>
<td></td>
</tr>
<tr>
<td>.rhosts with RSA (v1) on server only</td>
<td>User account</td>
<td>User account</td>
</tr>
<tr>
<td></td>
<td>Local host public key in /etc/ssh/ssh_host_rsa1_key</td>
<td>Local host public key in /etc/ssh/ssh_known_hosts or ~/.ssh/known_hosts</td>
</tr>
<tr>
<td></td>
<td>IgnoreRhosts no in /etc/ssh/sshd_config</td>
<td>IgnoreRhosts no in /etc/ssh/sshd_config</td>
</tr>
<tr>
<td></td>
<td>Local host entry in /etc/ssh/hosts.equiv, /etc/hosts.equiv, ~/.rhosts, or ~/.shosts</td>
<td>Local host entry in /etc/ssh/hosts.equiv, /etc/hosts.equiv, ~/.rhosts, or ~/.shosts</td>
</tr>
</tbody>
</table>
The following features are implemented for the v2 protocol in this release of Secure Shell:

- **ForceCommand keyword** – Forces the execution of the specified command regardless of what the user types on the command line. This keyword is very useful inside a Match block. This `sshd_config` configuration option is similar to the `command="..."` option in `$HOME/.ssh/authorized_keys`.

- **AES-128 passphrase protection** – In this release, private keys that are generated by the `ssh-keygen` command are protected with the AES-128 algorithm. This algorithm protects newly-generated keys and re-encrypted keys, such as when a passphrase is changed.

- **-u option to sftp-server command** – Enables user to set an explicit umask on files and directories. This option overrides the user's default umask. For an example, see the description of Subsystem on the `sshd_config(4)` man page.

- **Additional keywords for Match blocks** – `AuthorizedKeysFile`, `ForceCommand`, and `HostbasedUsesNameFromPacketOnly` are supported inside Match blocks. By default, the value of `AuthorizedKeysFile` is `$HOME/.ssh/authorized_keys` and `HostbasedUsesNameFromPacketOnly` is no. To use Match blocks, see “How to Create User and Host Exceptions to Secure Shell Defaults” on page 289.

While Oracle Solaris engineers provide bug fixes to the project, they have also integrated the following Oracle Solaris features into the fork of Secure Shell:

- **PAM** - Secure Shell uses PAM. The `UsePAM` configuration option is not supported.

- **Privilege separation** - Secure Shell does not use the privilege separation code from the OpenSSH project. Secure Shell separates the processing of auditing, record keeping and re-keying from the processing of the session protocols. Secure Shell privilege separation code is always on and cannot be switched off. The `UsePrivilegeSeparation` option is not supported.

- **Locale** - Secure Shell fully supports language negotiation as defined in RFC 4253, *Secure Shell Transfer Protocol*. After the user logs in, the user's login shell profile can override the Secure Shell negotiated locale settings.

- **Auditing** - Secure Shell is fully integrated into the Solaris audit service. For information about the audit service, see Part VII, "Auditing in Oracle Solaris.”

- **GSS-API support** - GSS-API can be used for user authentication and for initial key exchange. The GSS-API is defined in RFC4462, *Generic Security Service Application Program Interface*.

- **Proxy commands** - Secure Shell provides proxy commands for SOCKS5 and HTTP protocols. For an example, see “How to Set Up Default Secure Shell Connections to Hosts Outside a Firewall” on page 301.

In the Oracle Solaris releases, Secure Shell resyncs the `SSH_OLD_FORWARD_ADDR` compatibility flag from the OpenSSH project. As of March 2011, the Secure Shell version is 1.5.
Secure Shell and FIPS-140

Oracle Solaris provides a FIPS-140 option for the server side and the client side. FIPS mode, where Secure Shell uses the FIPS-140 mode of OpenSSL, is not the default. You can invoke FIPS mode on the command line, as in `ssh -o "UseFIPS140 yes" remote-host`. As an alternative, you can set a keyword in the configuration files.

Briefly, the implementation consists of the following:

- The following FIPS-approved ciphers are available on the server and client side:
  - `aes128-cbc`, `aes192-cbc`, and `aes256-cbc`.
  - `3des-cbc` is available by default on the client side, but it is not in the server side cipher list because of potential security risks.
- The following FIPS-approved Message Authentication Codes (MAC) are available:
  - `hmac-sha1`
  - `hmac-sha1-96`
  - `hmac-sha2-256`, `hmac-sha2-256-96`
  - `hmac-sha2-512`, `hmac-sha2-512-96`
- Four server-client configurations are supported:
  - No FIPS mode on either client or server side
  - FIPS mode on both the client and server side
  - FIPS mode on the server side, but no FIPS on client side
  - No FIPS mode on the server side, but FIPS mode on the client side
- The `ssh-keygen` command has an option to generate the user’s private key in the PKCS #8 format that Secure Shell clients in FIPS mode require. For more information, see the `ssh-keygen(1)` man page.

For more information about FIPS operations in Secure Shell, see the `sshd(1M)`, `sshd_config(4)`, `ssh(1)`, and `ssh_config(4)` man pages.

When you use a Sun Crypto Accelerator 6000 card for Secure Shell operations, Secure Shell runs with FIPS-140 support at Level 3. Level 3 hardware is certified to resist physical tampering, use identity-based authentication, and isolate the interfaces that handle critical security parameters from the hardware’s other interfaces.

Configuring Secure Shell (Tasks)

Secure Shell is configured at installation. To change the defaults requires administrative intervention. The following tasks demonstrate how to change some of the defaults.

Configuring Secure Shell (Task Map)

The following task map points to procedures for configuring Secure Shell.
### How to Set Up Host-Based Authentication for Secure Shell

The following procedure sets up a public key system where the client’s public key is used for authentication on the server. The user must also create a public/private key pair.

In the procedure, the terms *client* and *local host* refer to the machine where a user types the *ssh* command. The terms *server* and *remote host* refer to the machine that the client is trying to reach.

#### Before You Begin

You must assume the *root* role. For more information, see "How to Use Your Assigned Administrative Rights" on page 157.

1. **On the client, enable host-based authentication.**
   In the client configuration file, */etc/ssh/ssh_config*, type the following entry:
   ```
   HostbasedAuthentication yes
   
   For the syntax of the file, see the *ssh_config(4)* man page
   ```

2. **On the server, enable host-based authentication.**
   In the server configuration file, */etc/ssh/sshd_config*, type the same entry:
   ```
   HostbasedAuthentication yes
   
   For the syntax of the file, see the *sshd_config(4)* man page
   ```
3 On the server, configure a file that enables the client to be recognized as a trusted host.
For more information, see the FILES section of the sshd(1M) man page.

- Add the client as an entry to the server's /etc/ssh/hosts.equiv file.
  client-host

- Or, you can instruct users to add an entry for the client to their ~/.ssh/hosts file on the server.
  client-host

4 On the server, ensure that the sshd daemon can access the list of trusted hosts.
Set IgnoreRhosts to no in the /etc/ssh/sshd_config file.

```
## sshd_config
IgnoreRhosts no
```

5 Ensure that users of Secure Shell at your site have accounts on both hosts.

6 Do one of the following to put the client's public key on the server.

- Modify the sshd_config file on the server, then instruct your users to add the client's public
  host keys to their ~/.ssh/known_hosts file.

```
## sshd_config
IgnoreUserKnownHosts no
```

For user instructions, see “How to Generate a Public/Private Key Pair for Use With Secure
Shell” on page 292.

- Copy the client's public key to the server.
  The host keys are stored in the /etc/ssh directory. The keys are typically generated by the
  sshd daemon on first boot.

  a. Add the key to the /etc/ssh/ssh_known_hosts file on the server.
     On the client, type the command on one line with no backslash.

     ```
     # cat /etc/ssh/ssh_host_dsa_key.pub | ssh RemoteHost \
     'cat >> /etc/ssh/ssh_known_hosts && echo "Host key copied"
     ```

  b. When you are prompted, supply your login password.
     When the file is copied, the message “Host key copied” is displayed.
     Each line in the /etc/ssh/ssh_known_hosts file consists of fields that are separated by
     spaces:

     ```
     hostnames algorithm-name publickey comment
     ```
c. Edit the `/etc/ssh/ssh_known_hosts` file and add `RemoteHost` as the first field in the copied entry.

```
    ## /etc/ssh/ssh_known_hosts File
    RemoteHost <copied entry>
```

**Example 15–1 Setting Up Host-based Authentication**

In the following example, each host is configured as a server and as a client. A user on either host can initiate an `ssh` connection to the other host. The following configuration makes each host a server and a client:

- On each host, the Secure Shell configuration files contain the following entries:

  ```
  ## /etc/ssh/ssh_config
  HostBasedAuthentication yes
  #
  ## /etc/ssh/sshd_config
  HostBasedAuthentication yes
  IgnoreRhosts no
  ```

- On each host, the `hosts.equiv` file contains an entry for the other host:

  ```
  ## /etc/ssh/shosts.equiv on machine2
  machine1
  ## /etc/ssh/shosts.equiv on machine1
  machine2
  ```

- The public key for each host is in the `/etc/ssh/ssh_known_hosts` file on the other host:

  ```
  ## /etc/ssh/ssh_known_hosts on machine2
  ... machine1
  ## /etc/ssh/ssh_known_hosts on machine1
  ... machine2
  ```

- Users have an account on both hosts:

  ```
  ## /etc/passwd on machine1
  jdoe:x:3111:10:J Doe:/home/jdoe:/bin/sh
  ## /etc/passwd on machine2
  jdoe:x:3111:10:J Doe:/home/jdoe:/bin/sh
  ```

▼ **How to Configure Port Forwarding in Secure Shell**

Port forwarding enables a local port be forwarded to a remote host. Effectively, a socket is allocated to listen to the port on the local side. Similarly, a port can be specified on the remote side.

**Note** – Secure Shell port forwarding must use TCP connections. Secure Shell does not support UDP connections for port forwarding.
You must assume the root role. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1. **Configure a Secure Shell setting on the remote server to allow port forwarding.**
   
   Change the value of `AllowTcpForwarding` to `yes` in the `/etc/ssh/sshd_config` file.
   
   ```
   # Port forwarding
   AllowTcpForwarding yes
   ```
   
   Restart the Secure Shell service.
   
   ```
   remoteHost# svcadm restart network/ssh:default
   ```
   
   For information about managing persistent services, see Chapter 1, “Managing Services (Overview),” in Managing Services and Faults in Oracle Solaris 11.1 and the `svcadm(1M)` man page.

2. **Verify that port forwarding can be used.**
   
   ```
   remoteHost# /usr/bin/pgrep -lf sshd
   1296 ssh -L 2001:remoteHost:23 remoteHost
   ```

### How to Create User and Host Exceptions to Secure Shell Defaults

This procedure adds a conditional `Match` block after the global section of the `/etc/ssh/sshd_config` file. Keyword-value pairs that follow the `Match` block specify exceptions for the user, group, host, or address that is specified as the match.

**Before You Begin**

You must become an administrator who is assigned the `solaris.admin.edit/etc/ssh/sshd_config` authorization. By default, the root role has this authorization. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1. **Edit the `/etc/ssh/sshd_config` file.**

2. **Configure a user, group, host, or address to use different Secure Shell settings from the default settings.**

   Place the `Match` blocks after the global settings.

**Note** – The global section of the file might or might not list the default settings. For the defaults, see the `sshd_config(4)` man page.
You might have users who should not be allowed to use TCP forwarding. In the following example, any user in the group public, and any user name that begins with test cannot use TCP forwarding:

```bash
## sshd_config file
## Global settings

# Example (reflects default settings):
#
# Host *
# ForwardAgent no
# ForwardX11 no
# PubkeyAuthentication yes
# PasswordAuthentication yes
# FallBackToRsh no
# UseRsh no
# BatchMode no
# CheckHostIP yes
# StrictHostKeyChecking ask
# EscapeChar ~
Match Group public
   AllowTcpForwarding no
Match User test*
   AllowTcpForwarding no
```

For information about the syntax of the Match block, see the `sshd_config(4)` man page.

### How to Create an Isolated Directory for sftp Files

This procedure configures an `sftponly` directory that is created specifically for sftp transfers. Users cannot see any files or directories outside the transfer directory.

**Before You Begin**

You must assume the root role. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1. **On the Secure Shell server, create the isolated directory as a `chroot` environment.**

   ```bash
   # groupadd sftp
   # useradd -m -G sftp -s /bin/false sftponly
   # chown root:root /export/home/sftponly
   # mkdir /export/home/sftponly/WWW
   # chown sftponly:staff /export/home/sftponly/WWW
   
   In this configuration, /export/home/sftponly is the chroot directory that only the root account has access to. The user has write permission to the sftponly/WWW subdirectory.
   ```
Still on the server, configure a match block for the sftp group.

In the /etc/ssh/sshd_config file, locate the sftp subsystem entry and modify the file as follows:

```
# pfedit /etc/ssh/sshd_config
...
# sftp subsystem
#Subsystem sftp /usr/lib/ssh/sftp-server
Subsystem sftp internal-sftp
...
## Match Group for Subsystem
## At end of file, to follow all global options
Match Group sftp
  ChrootDirectory %h
  ForceCommand internal-sftp
  AllowTcpForwarding no
```

You can use the following variables to specify the chroot path:

- `%h` – Specifies the home directory.
- `%u` – Specifies the username of the authenticated user.
- `%%` – Escapes the `%` sign.

On the client, verify that the configuration works correctly.

The files in your chroot environment might be different.

```
root@client:~# ssh sftponly@server
This service allows sftp connections only.
Connection to server closed. No shell access, sftp is enforced.
root@client:~# sftp sftponly@server
sftp> pwd
sftp access granted
Remote working directory: /
  chroot directory looks like root directory
sftp> ls
WWW local.cshrc local.login local.profile
sftp> get local.cshrc
Fetching /local.cshrc to local.cshrc
/local.cshrc 100% 166 0.2KB/s 00:00 user can read contents
sftp> put /etc/motd
Uploading /etc/motd to /motd
Couldn't get handle: Permission denied user cannot write to / directory
sftp> cd WWW
sftp> put /etc/motd
Uploading /etc/motd to /WWW/motd
/etc/motd 100% 118 0.1KB/s 00:00 user can write to WWW directory
sftp> ls -l
-rw-r--r-- 1 101 10 118 Jul 20 09:07 motd successful transfer
sftp>
```
Using Secure Shell (Tasks)

Secure Shell provides secure access between a local shell and a remote shell. For more information, see the `ssh_config(4)` and `ssh(1)` man pages.

Using Secure Shell (Task Map)

The following task map points to user procedures for using Secure Shell.

<table>
<thead>
<tr>
<th>Task</th>
<th>Description</th>
<th>For Instructions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create a public/private key pair.</td>
<td>Enables access to Secure Shell for sites that require public-key authentication.</td>
<td>&quot;How to Generate a Public/Private Key Pair for Use With Secure Shell&quot; on page 292</td>
</tr>
<tr>
<td>Change your passphrase.</td>
<td>Changes the phrase that authenticates your private key.</td>
<td>&quot;How to Change the Passphrase for a Secure Shell Private Key&quot; on page 294</td>
</tr>
<tr>
<td>Log in with Secure Shell.</td>
<td>Provides encrypted Secure Shell communication when logging in remotely.</td>
<td>&quot;How to Log In to a Remote Host With Secure Shell&quot; on page 295</td>
</tr>
<tr>
<td>Log in to Secure Shell without being prompted for a password.</td>
<td>Enables login by using an agent which provides your password to Secure Shell.</td>
<td>&quot;How to Reduce Password Prompts in Secure Shell&quot; on page 296</td>
</tr>
<tr>
<td>Log in to Secure Shell as root.</td>
<td>Enables login as root for ZFS send and receive commands.</td>
<td>&quot;How to Remotely Administer ZFS With Secure Shell&quot; on page 297</td>
</tr>
<tr>
<td>Use port forwarding in Secure Shell.</td>
<td>Specifies a local port or a remote port to be used in a Secure Shell connection over TCP.</td>
<td>&quot;How to Use Port Forwarding in Secure Shell&quot; on page 299</td>
</tr>
<tr>
<td>Copy files with Secure Shell.</td>
<td>Securely copies files between hosts.</td>
<td>&quot;How to Copy Files With Secure Shell&quot; on page 300</td>
</tr>
<tr>
<td>Securely connect from a host inside a firewall to a host outside the firewall.</td>
<td>Uses Secure Shell commands that are compatible with HTTP or SOCKS5 to connect hosts that are separated by a firewall.</td>
<td>&quot;How to Set Up Default Secure Shell Connections to Hosts Outside a Firewall&quot; on page 301</td>
</tr>
</tbody>
</table>

▼ How to Generate a Public/Private Key Pair for Use With Secure Shell

Users must generate a public/private key pair when their site implements host-based authentication or user public-key authentication. For additional options, see the `ssh-keygen(1)` man page.

Before You Begin Determine from your system administrator if host-based authentication is configured.
1 Start the key generation program.
   myLocalHost% ssh-keygen -t rsa
   Generating public/private rsa key pair.
   ...
   where -t is the type of algorithm, one of rsa, dsa, or rsa1.

2 Specify the path to the file that will hold the key.
By default, the file name id_rsa, which represents an RSA v2 key, appears in parentheses. You can select this file by pressing the Return key. Or, you can type an alternative file name.
   Enter file in which to save the key (/home/jdoe/.ssh/id_rsa): <Press Return>
   The filename of the public key is created automatically by appending the string .pub to the name of the private key file.

3 Type a passphrase for using your key.
   This passphrase is used for encrypting your private key. A null entry is strongly discouraged. Note that the passphrase is not displayed when you type it in.
   Enter passphrase (empty for no passphrase): <Type passphrase>

4 Retype the passphrase to confirm it.
   Enter same passphrase again: <Type passphrase>
   Your identification has been saved in /home/jdoe/.ssh/id_rsa.
   Your public key has been saved in /home/jdoe/.ssh/id_rsa.pub.
   The key fingerprint is:

5 Check the results.
   Check that the path to the key file is correct.
   % ls ~/.ssh
   id_rsa
   id_rsa.pub
   At this point, you have created a public/private key pair.

6 Choose the appropriate option:

- If your administrator has configured host-based authentication, you might need to copy the local host's public key to the remote host.
  You can now log in to the remote host. For details, see "How to Log In to a Remote Host With Secure Shell" on page 295.

  a. Type the command on one line with no backslash.
     % cat /etc/ssh/ssh_host_dsa_key.pub | ssh RemoteHost 'cat >> ~/.ssh/known_hosts && echo "Host key copied"'
b. When you are prompted, supply your login password.

Enter password: <Type password>

Host key copied
%

- If your site uses user authentication with public keys, populate your authorized_keys file on the remote host.

a. Copy your public key to the remote host.

Type the command on one line with no backslash.

myLocalHost% cat $HOME/.ssh/id_rsa.pub | ssh myRemoteHost 'cat >> .ssh/authorized_keys && echo "Key copied"'

b. When you are prompted, supply your login password.

When the file is copied, the message “Key copied” is displayed.

Enter password: Type login password

Key copied
myLocalHost%

7 (Optional) Reduce the prompting for passphrases.

For a procedure, see “How to Reduce Password Prompts in Secure Shell” on page 296. For more information, see the ssh-agent(1) and ssh-add(1) man pages.

▼ How to Change the Passphrase for a Secure Shell Private Key

The following procedure does not change the private key. The procedure changes the authentication mechanism for the private key, the passphrase. For more information, see the ssh-keygen(1) man page.

- Change your passphrase.

Type the ssh-keygen command with the -p option, and answer the prompts.

myLocalHost% ssh-keygen -p
Enter file which contains the private key (/home/jdoe/.ssh/id_rsa): <Press Return>
Enter passphrase (empty for no passphrase): <Type passphrase>
Enter same passphrase again: <Type passphrase>

where -p requests changing the passphrase of a private key file.
How to Log In to a Remote Host With Secure Shell

1 Start a Secure Shell session.
Type the ssh command, and specify the name of the remote host and your login.

```
myLocalHost% ssh myRemoteHost -l username
```

A prompt questions the authenticity of the remote host:

```
The authenticity of host 'myRemoteHost' can’t be established.
Are you sure you want to continue connecting(yes/no)?
```

This prompt is normal for initial connections to remote hosts.

2 If prompted, verify the authenticity of the remote host key.
- If you cannot confirm the authenticity of the remote host, type no and contact your system administrator.
  
  ```
  Are you sure you want to continue connecting(yes/no)? no
  ```
  
  The administrator is responsible for updating the global /etc/ssh/ssh_known_hosts file. An updated ssh_known_hosts file prevents this prompt from appearing.

- If you confirm the authenticity of the remote host, answer the prompt and continue to the next step.
  
  ```
  Are you sure you want to continue connecting(yes/no)? yes
  ```

3 Authenticate yourself to Secure Shell.

a. When prompted, type your passphrase.

  ```
Enter passphrase for key '/home/jdoe/.ssh/id_rsa': <Type passphrase>
  ```

b. When prompted, type your account password.

  ```
jdoe@myRemoteHost’s password: <Type password>
Last login: Wed Sep  7 09:07:49 2011 from myLocalHost
Oracle Corporation SunOS 5.11 September 2011
myRemoteHost%
```

4 Conduct transactions on the remote host.
The commands that you send are encrypted. Any responses that you receive are encrypted.

5 Close the Secure Shell connection.
When you are finished, type exit or use your usual method for exiting your shell.

```
myRemoteHost% exit
myRemoteHost% logout
```
Connection to myRemoteHost closed
myLocalHost%

Example 15–2 Displaying a Remote GUI in Secure Shell

In this example, jdoe is the initial user on both systems, so is assigned the Software Installation rights profile. jdoe wants to use the Package Manager GUI on the remote system. The default value of the X11Forwarding keyword is still yes, and the xauth package is installed on the remote system.

% ssh -l jdoe -X myRemoteHost
jdoe@myRemoteHost’s password:  <Type password>
Last login: Wed Sep  7 09:07:49 2011 from myLocalHost
Oracle Corporation SunOS 5.11 September 2011
myRemoteHost% packagemanager &

▼ How to Reduce Password Prompts in Secure Shell

If you do not want to type your passphrase and your password to use Secure Shell, you can use the agent daemon. Start the daemon at the beginning of the session. Then, store your private keys with the agent daemon by using the ssh-add command. If you have different accounts on different hosts, add the keys that you need for the session.

You can start the agent daemon manually when needed, as described in the following procedure.

1  Start the agent daemon.
   myLocalHost% eval `ssh-agent`
   Agent pid 9892

2  Verify that the agent daemon has been started.
   myLocalHost% pgrep ssh-agent
   9892

3  Add your private key to the agent daemon.
   Type the ssh-add command.
   myLocalHost% ssh-add
   Enter passphrase for /home/jdoe/.ssh/id_rsa:  <Type passphrase>
   Identity added: /home/jdoe/.ssh/id_rsa(/home/jdoe/.ssh/id_rsa)
   myLocalHost%

4  Start a Secure Shell session.
   myLocalHost% ssh myRemoteHost -l jdoe
   You are not prompted for a passphrase.
Using ssh-add Options

In this example, jdoe adds two keys to the agent daemon. The -l option is used to list all keys that are stored in the daemon. At the end of the session, the -D option is used to remove all the keys from the agent daemon.

```bash
myLocalHost% ssh-agent
myLocalHost% ssh-add
Enter passphrase for /home/jdoe/.ssh/id_rsa: <Type passphrase>
Identity added: /home/jdoe/.ssh/id_rsa
myLocalHost% ssh-add /home/jdoe/.ssh/id_dsa
Enter passphrase for /home/jdoe/.ssh/id_dsa: <Type passphrase>
Identity added: /home/jdoe/.ssh/id_dsa
myLocalHost% ssh-add -l
/home/jdoe/.ssh/id_rsa (RSA)
/home/jdoe/.ssh/id_dsa (DSA)

User conducts Oracle Solaris Secure Shell transactions

myLocalHost% ssh-add -D
Identity removed:
/home/jdoe/.ssh/id_rsa
/home/jdoe/.ssh/id_dsa
```

How to Remotely Administer ZFS With Secure Shell

By default, the root role cannot log in remotely with Secure Shell. Historically, root has used Secure Shell for important tasks, such as sending ZFS pool data to storage on a remote system. In this procedure, the root role creates a user who can act as a remote ZFS administrator.

**Before You Begin**

You must assume the root role. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1. Create the user on the both systems.

   For example, create the zfsroot user and provide a password.

   ```bash
   source # useradd -c "Remote ZFS Administrator" -u 1201 -d /home/zfsroot zfsroot
   source # passwd zfsroot
   Enter password:
   Retype password:
   #
   dest # useradd -c "Remote ZFS Administrator" -u 1201 -d /home/zfsroot zfsroot
   dest # passwd zfsroot
   ... ```
The zfsroot user must be identically defined on both systems.

2 **Create the user’s key pair for Secure Shell authentication.**

The key pair is created on the source system. Then, the public key is copied to the zfsroot user on the destination system.

a. **Generate the key pair and put it in the file `id_migrate`.**

```bash
# ssh-keygen -t rsa -P "" -f ~/id_migrate
Generating public/private rsa key pair.
Your identification has been saved in /root/id_migrate.
Your public key has been saved in /root/id_migrate.pub.
The key fingerprint is:
```

b. **Send the public part of the key pair to the destination system.**

```bash
# scp ~/id_migrate.pub zfsroot@dest: 
The authenticity of host ‘dest (10.134.76.126)’ can’t be established.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added ‘dest,10.134.76.126’ (RSA) to the list of known hosts.
Password: 
id_migrate.pub 100% |*****************************| 399 00:00
```

3 **On both systems, assign the ZFS File Management rights profile to zfsroot.**

```bash
source # usermod -P +’ZFS File System Management’ -S files zfsroot
dest # usermod -P +’ZFS File System Management’ -S files zfsroot
```

4 **Verify that the destination system is assigned the rights profile.**

```bash
dest # profiles zfsroot
zfsroot:
ZFS File System Management
Basic Solaris User
All
```

5 **On the destination system, move the public part of the key pair to the private**

/`home/zfsroot/.ssh` directory.

```bash
root@dest # su - zfsroot
Oracle Corporation SunOS 5.11 11.1 May 2012
zfsroot@dest $ mkdir -m 700 .ssh
zfsroot@dest $ cat id_migrate.pub >> .ssh/authorized_keys
```

6 **Verify that the configuration works.**

```bash
root@source# ssh -l zfsroot -i ~/id_migrate dest 
pfexec /usr/sbin/zfs snapshot zones@test
root@source# ssh -l zfsroot -i ~/id_migrate dest 
pfexec /usr/sbin/zfs destroy zones@test
```

7 **(Optional) Verify that you can create a snapshot and replicate the data.**

```bash
root@source# zfs snapshot -r rpool/zones@migrate-all
root@source# zfs send -rc rpool/zones@migrate-all |
```

```
Using Secure Shell (Tasks)
Oracle Solaris 11.1 Administration: Security Services • February 2014
```
ssh -l zfsroot -i ~/id_migrate dest pfexec /usr/sbin/zfs recv -F zones

8 (Optional) Remove the ability to use the zfsroot account for ZFS administration.
root@dest# usermod -P -'ZFS File System Management' zfsroot
root@dest# su - zfsroot
zfsroot@dest# cp .ssh/authorized_keys .ssh/authorized_keys.bak
zfsroot@dest# grep -v root@source .ssh/authorized_keys.bak> .ssh/authorized_keys

▼ How to Use Port Forwarding in Secure Shell

You can specify that a local port be forwarded to a remote host. Effectively, a socket is allocated to listen to the port on the local side. The connection from this port is made over a secure channel to the remote host. For example, you might specify port 143 to obtain email remotely with IMAP4. Similarly, a port can be specified on the remote side.

Before You Begin

To use port forwarding, the administrator must have enabled port forwarding on the remote Secure Shell server. For details, see “How to Configure Port Forwarding in Secure Shell” on page 288.

To use secure port forwarding, choose one of the following options:

- To set a local port to receive secure communication from a remote port, specify both ports.
  Specify the local port that listens for remote communication. Also, specify the remote host and the remote port that forward the communication.
  myLocalHost% ssh -L localPort:remoteHost:remotePort

- To set a remote port to receive a secure connection from a local port, specify both ports.
  Specify the remote port that listens for remote communication. Also, specify the local host and the local port that forward the communication.
  myLocalHost% ssh -R remotePort:localhost:localPort

Example 15–4 Using Local Port Forwarding to Receive Mail

The following example demonstrates how you can use local port forwarding to receive mail securely from a remote server.

myLocalHost% ssh -L 9143:myRemoteHost:143 myRemoteHost

This command forwards connections from port 9143 on myLocalHost to port 143. Port 143 is the IMAP v2 server port on myRemoteHost. When the user launches a mail application, the user specifies the local port number for the IMAP server, as in localhost:9143.

Do not confuse localhost with myLocalHost. myLocalHost is a hypothetical host name.
localhost is a keyword that identifies your local system.
Using Remote Port Forwarding to Communicate Outside of a Firewall

This example demonstrates how a user in an enterprise environment can forward connections from a host on an external network to a host inside a corporate firewall.

```bash
myLocalHost% ssh -R 9022:myLocalHost:22 myOutsideHost
```

This command forwards connections from port 9022 on `myOutsideHost` to port 22, the `sshd` server, on the local host.

```bash
myOutsideHost% ssh -p 9022 localhost
```

How to Copy Files With Secure Shell

The following procedure shows how to use the `scp` command to copy encrypted files between hosts. You can copy encrypted files either between a local host and a remote host, or between two remote hosts. The `scp` command prompts for authentication. For more information, see the `scp(1)` man page.

You can also use the `sftp` secure file transfer program. For more information, see the `sftp(1)` man page. For an example, see Example 15–6.

Note – The audit service can audit `sftp` transactions through the `ft` audit class. For `scp`, the audit service can audit access and exit for the `ssh` session.

1. **Start the secure copy program.**
   Specify the source file, the user name at the remote destination, and the destination directory.
   ```bash
   myLocalHost% scp myfile.1 jdoe@myRemoteHost:~
   ```

2. **Supply your passphrase when prompted.**
   Enter passphrase for key `/home/jdoe/.ssh/id_rsa`:
   ```
   myfile.1 25% |******* | 640 KB 0:20 ETA
   ```
   After you type the passphrase, a progress meter is displayed. See the second line in the preceding output. The progress meter displays:
   - The file name
   - The percentage of the file that has been transferred
   - A series of asterisks that indicate the percentage of the file that has been transferred
   - The quantity of data transferred
   - The estimated time of arrival, or ETA, of the complete file (that is, the remaining amount of time)
Example 15–6  Specifying a Port When Using the `sftp` Command

In this example, the user wants the `sftp` command to use a specific port. The user uses the `-o` option to specify the port.

```
% sftp -o port=2222 guest@RemoteFileServer
```

\section*{How to Set Up Default Secure Shell Connections to Hosts Outside a Firewall}

You can use Secure Shell to make a connection from a host inside a firewall to a host outside the firewall. This task is done by specifying a proxy command for `ssh` either in a configuration file or as an option on the command line. For the command-line option, see Example 15–7.

In general, you can customize your `ssh` interactions through a configuration file.

- You can customize either your own personal file in `~/.ssh/config`.
- Or, you can use the settings in the administrative configuration file, `/etc/ssh/ssh_config`.

The file can be customized with two types of proxy commands. One proxy command is for HTTP connections. The other proxy command is for SOCKS5 connections. For more information, see the `ssh_config(4)` man page.

\section{Specify the proxy commands and hosts in a configuration file.}

Use the following syntax to add as many lines as you need:

```
[Host outside-host]
ProxyCommand proxy-command [-h proxy-server] \ [-p proxy-port] outside-host|%h outside-port|%p
Host outside-host
    Limits the proxy command specification to instances when a remote host name is specified on the command line. If you use a wildcard for `outside-host`, you apply the proxy command specification to a set of hosts.

proxy-command
    Specifies the proxy command.

    The command can be either of the following:
    - `/usr/lib/ssh/ssh-http-proxy-connect` for HTTP connections
    - `/usr/lib/ssh/ssh-socks5-proxy-connect` for SOCKS5 connections

-h proxy-server and -p proxy-port
    These options specify a proxy server and a proxy port, respectively. If present, the proxies override any environment variables that specify proxy servers and proxy ports, such as `HTTPPROXY`, `HTTPPROXYPORT`, `SOCKS5_PORT`, `SOCKS5_SERVER`, and `http_proxy`. The `http_proxy` variable specifies a URL. If the options are not used, then the relevant
environment variables must be set. For more information, see the
`ssh-socks5-proxy-connect(1)` and `ssh-http-proxy-connect(1)` man pages.

**outside-host**
Designates a specific host to connect to. Use the `%h` substitution argument to specify the host
on the command line.

**outside-port**
Designates a specific port to connect to. Use the `%p` substitution argument to specify the port
on the command line. By specifying `%h` and `%p` without using the `Host outside-host` option,
the proxy command is applied to the host argument whenever the `ssh` command is invoked.

2 Run Secure Shell, specifying the outside host.
For example, type the following:

```
myLocalHost% ssh myOutsideHost
```

This command looks for a proxy command specification for `myOutsideHost` in your personal
configuration file. If the specification is not found, then the command looks in the system-wide
configuration file, `/etc/ssh/ssh_config`. The proxy command is substituted for the `ssh`
command.

---

**Example 15–7 Connecting to Hosts Outside a Firewall From the Secure Shell Command Line**

“How to Set Up Default Secure Shell Connections to Hosts Outside a Firewall” on page 301
explains how to specify a proxy command in a configuration file. In this example, a proxy
command is specified on the `ssh` command line.

```
% ssh -o'Proxycommand=/usr/lib/ssh/ssh-http-proxy-connect \ 
-h myProxyServer -p 8080 myOutsideHost 22' myOutsideHost
```

The `-o` option to the `ssh` command provides a command-line method of specifying a proxy
command. This example command does the following:

- Substitutes the HTTP proxy command for `ssh`
- Uses port `8080` and `myProxyServer` as the proxy server
- Connects to port `22` on `myOutsideHost`
This chapter describes the configuration options in the Secure Shell feature of Oracle Solaris, and covers the following topics:

- “A Typical Secure Shell Session” on page 303
- “Client and Server Configuration in Secure Shell” on page 305
- “Keywords in Secure Shell” on page 306
- “Maintaining Known Hosts in Secure Shell” on page 311
- “Secure Shell Files” on page 312
- “Secure Shell Commands” on page 314

For procedures to configure Secure Shell, see Chapter 15, “Using Secure Shell.”

**A Typical Secure Shell Session**

The Secure Shell daemon (sshd) is normally started at boot time when network services are started. The daemon listens for connections from clients. A Secure Shell session begins when the user runs an ssh, scp, or sftp command. A new sshd daemon is forked for each incoming connection. The forked daemons handle key exchange, encryption, authentication, command execution, and data exchange with the client. These session characteristics are determined by client-side configuration files and server-side configuration files. Command-line arguments can override the settings in the configuration files.

The client and server must authenticate themselves to each other. After successful authentication, the user can execute commands remotely and copy data between hosts.

**Session Characteristics in Secure Shell**

The server-side behavior of the sshd daemon is controlled by keyword settings in the /etc/ssh/sshd_config file. For example, the sshd_config file controls which types of
authentication are permitted for accessing the server. The server-side behavior can also be controlled by the command-line options when the sshd daemon is started.

The behavior on the client side is controlled by Secure Shell keywords in this order of precedence:

- Command-line options
- User’s configuration file, ~/.ssh/config
- System-wide configuration file, /etc/ssh/ssh_config

For example, a user can override a system-wide configuration Ciphers setting that prefers aes128-ctr by specifying -c aes256-ctr,aes128-ctr,arcfour on the command line. The first cipher, aes256-ctr, is now preferred.

### Authentication and Key Exchange in Secure Shell

The Secure Shell protocol supports client user/host authentication and server host authentication. Cryptographic keys are exchanged for the protection of Secure Shell sessions. Secure Shell provides various methods for authentication and key exchange. Some methods are optional. Client authentication mechanisms are listed in Table 15–1. Servers are authenticated by using known host public keys.

For authentication, Secure Shell supports user authentication and generic interactive authentication, which usually involves passwords. Secure Shell also supports authentication with user public keys and with trusted host public keys. The keys can be RSA or DSA. Session key exchanges consist of Diffie-Hellman ephemeral key exchanges that are signed in the server authentication step. Additionally, Secure Shell can use GSS credentials for authentication.

### Acquiring GSS Credentials in Secure Shell

To use GSS-API for authentication in Secure Shell, the server must have GSS-API acceptor credentials and the client must have GSS-API initiator credentials. Support is available for mech_dh and for mech_krb5.

For mech_dh, the server has GSS-API acceptor credentials if root has run the keylogin command.

For mech_krb5, the server has GSS-API acceptor credentials when the host principal that corresponds to the server has a valid entry in /etc/krb5/krb5.keytab.

The client has initiator credentials for mech_dh if one of the following has been done:

- The keylogin command has been run.
- The pam_dhkeys module is used in the pam.conf file.
The client has initiator credentials for mech_krb5 if one of the following has been done:

- The kinit command has been run.
- The pam_krb5 module is used in the pam.conf file.

For the use of mech_dh in secure RPC, see Chapter 18, "Network Services Authentication (Tasks)." For the use of mech_krb5, see Chapter 19, "Introduction to the Kerberos Service." For more information about mechanisms, see the mech(4) and mech_spnego(5) man pages.

**Command Execution and DataForwarding in Secure Shell**

After authentication is complete, the user can use Secure Shell, generally by requesting a shell or executing a command. Through the ssh command options, the user can make requests. Requests can include allocating a pseudo-tty, forwarding X11 connections or TCP/IP connections, or enabling an ssh-agent authentication program over a secure connection.

The basic components of a user session are as follows:

1. The user requests a shell or the execution of a command, which begins the session mode. In this mode, data is sent or received through the terminal on the client side. On the server side, data is sent through the shell or a command.
2. When data transfer is complete, the user program terminates.
3. All X11 forwarding and TCP/IP forwarding is stopped, except for those connections that already exist. Existing X11 connections and TCP/IP connections remain open.
4. The server sends an exit status message to the client. When all connections are closed, such as forwarded ports that had remained open, the client closes the connection to the server. Then, the client exits.

**Client and Server Configuration in Secure Shell**

The characteristics of a Secure Shell session are controlled by configuration files. The configuration files can be overridden to a certain degree by options on the command line.

**Client Configuration in Secure Shell**

In most cases, the client-side characteristics of a Secure Shell session are governed by the system-wide configuration file, /etc/ssh/ssh_config. The settings in the ssh_config file can be overridden by the user’s configuration file, ~/.ssh/config. In addition, the user can override both configuration files on the command line.
The settings in the server's `/etc/ssh/sshd_config` file determine which client requests are permitted by the server. For a list of server configuration settings, see "Keywords in Secure Shell" on page 306. For detailed information, see the `sshd_config(4)` man page.

The keys in the client configuration file are listed in "Keywords in Secure Shell" on page 306. If the keyword has a default value, the value is given. These keywords are described in detail in the `ssh(1), scp(1), sftp(1), and ssh_config(4)` man pages. For a list of keywords in alphabetical order and their equivalent command-line overrides, see Table 16–8.

## Server Configuration in Secure Shell

The server-side characteristics of a Secure Shell session are governed by the `/etc/ssh/sshd_config` file. The keywords in the server configuration file are listed in "Keywords in Secure Shell" on page 306. If the keyword has a default value, the value is given. For a full description of the keywords, see the `sshd_config(4)` man page.

### Keywords in Secure Shell

The following tables list the keywords and their default values, if any. The keywords are in alphabetical order. Keywords that apply to the client are in the `ssh_config` file. Keywords that apply to the server are in the `sshd_config` file. Some keywords are set in both files. Keywords for a Secure Shell server that is running the v1 protocol are marked.

---

**TABLE 16–1   Keywords in Secure Shell Configuration Files (A to Escape)**

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Default Value</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>AllowGroups</td>
<td>No default.</td>
<td>Server</td>
</tr>
<tr>
<td>AllowTcpForwarding</td>
<td>yes</td>
<td>Server</td>
</tr>
<tr>
<td>AllowUsers</td>
<td>No default.</td>
<td>Server</td>
</tr>
<tr>
<td>AuthorizedKeysFile</td>
<td>~/.ssh/authorized_keys</td>
<td>Server</td>
</tr>
<tr>
<td>Banner</td>
<td>/etc/issue</td>
<td>Server</td>
</tr>
<tr>
<td>Batchmode</td>
<td>no</td>
<td>Client</td>
</tr>
<tr>
<td>BindAddress</td>
<td>No default.</td>
<td>Client</td>
</tr>
<tr>
<td>CheckHostIP</td>
<td>yes</td>
<td>Client</td>
</tr>
<tr>
<td>ChrootDirectory</td>
<td>no</td>
<td>Server</td>
</tr>
<tr>
<td>Cipher</td>
<td>blowfish, 3des</td>
<td>Client</td>
</tr>
</tbody>
</table>
### TABLE 16–1  Keywords in Secure Shell Configuration Files (A to Escape) (Continued)

<table>
<thead>
<tr>
<th>Keyword</th>
<th>DefaultValue</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ciphers</td>
<td>aes128-ctr, aes128-cbc, 3des-cbc, blowfish-cbc, arcfour</td>
<td>Both</td>
</tr>
<tr>
<td>ClearAllForwardings</td>
<td>no</td>
<td>Client</td>
</tr>
<tr>
<td>ClientAliveCountMax</td>
<td>3</td>
<td>Server</td>
</tr>
<tr>
<td>ClientAliveInterval</td>
<td>0</td>
<td>Server</td>
</tr>
<tr>
<td>Compression</td>
<td>no</td>
<td>Both</td>
</tr>
<tr>
<td>CompressionLevel</td>
<td>No default.</td>
<td>Client</td>
</tr>
<tr>
<td>ConnectionAttempts</td>
<td>1</td>
<td>Client</td>
</tr>
<tr>
<td>ConnectTimeout</td>
<td>System TCP timeout</td>
<td>Client</td>
</tr>
<tr>
<td>DenyGroups</td>
<td>No default</td>
<td>Server</td>
</tr>
<tr>
<td>DenyUsers</td>
<td>No default</td>
<td>Server</td>
</tr>
<tr>
<td>DisableBanner</td>
<td>no</td>
<td>Client</td>
</tr>
<tr>
<td>DynamicForward</td>
<td>No default.</td>
<td>Client</td>
</tr>
<tr>
<td>EscapeChar</td>
<td>~</td>
<td>Client</td>
</tr>
</tbody>
</table>

### TABLE 16–2  Keywords in Secure Shell Configuration Files (Fall to Local)

<table>
<thead>
<tr>
<th>Keyword</th>
<th>DefaultValue</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>FallBackToRsh</td>
<td>no</td>
<td>Client</td>
</tr>
<tr>
<td>ForwardAgent</td>
<td>no</td>
<td>Client</td>
</tr>
<tr>
<td>ForwardX11</td>
<td>no</td>
<td>Client</td>
</tr>
<tr>
<td>ForwardX11Trusted</td>
<td>yes</td>
<td>Client</td>
</tr>
<tr>
<td>GatewayPorts</td>
<td>no</td>
<td>Both</td>
</tr>
<tr>
<td>GlobalKnownHostsFile</td>
<td>/etc/ssh/ssh_known_hosts</td>
<td>Client</td>
</tr>
<tr>
<td>GSSAPIAuthentication</td>
<td>yes</td>
<td>Both</td>
</tr>
<tr>
<td>GSSAPIDelegateCredentials</td>
<td>no</td>
<td>Client</td>
</tr>
<tr>
<td>GSSAPIKeyExchange</td>
<td>yes</td>
<td>Both</td>
</tr>
<tr>
<td>GSSAPIStoreDelegateCredentials</td>
<td>yes</td>
<td>Server</td>
</tr>
<tr>
<td>HashKnownHosts</td>
<td>no</td>
<td>Client</td>
</tr>
</tbody>
</table>
Keywords in Secure Shell Configuration Files (Fall to Local) (Continued)

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Default Value</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>Host</td>
<td>* For more information, see “Host-Specific Parameters in Secure Shell” on page 310</td>
<td>Client</td>
</tr>
<tr>
<td>HostbasedAuthentication</td>
<td>no</td>
<td>Both</td>
</tr>
<tr>
<td>HostbasedUsesNameFromPacketOnly</td>
<td>no</td>
<td>Server</td>
</tr>
<tr>
<td>HostKey</td>
<td>/etc/ssh/ssh_host_key</td>
<td>Server, v1</td>
</tr>
<tr>
<td>HostKey</td>
<td>/etc/ssh/host_rsa_key, /etc/ssh/host_dsa_key</td>
<td>Server</td>
</tr>
<tr>
<td>HostKeyAlgorithms</td>
<td>ssh-rsa, ssh-dss</td>
<td>Client</td>
</tr>
<tr>
<td>HostKeyAlias</td>
<td>No default</td>
<td>Client</td>
</tr>
<tr>
<td>HostName</td>
<td>No default</td>
<td>Client</td>
</tr>
<tr>
<td>IdentityFile</td>
<td>~/.ssh/id_dsa, ~/.ssh/id_rsa</td>
<td>Client</td>
</tr>
<tr>
<td>IgnoreIfUnknown</td>
<td>No default</td>
<td>Client</td>
</tr>
<tr>
<td>IgnoreRhosts</td>
<td>yes</td>
<td>Server</td>
</tr>
<tr>
<td>IgnoreUserKnownHosts</td>
<td>yes</td>
<td>Server</td>
</tr>
<tr>
<td>KbdInteractiveAuthentication</td>
<td>yes</td>
<td>Both</td>
</tr>
<tr>
<td>KeepAlive</td>
<td>yes</td>
<td>Both</td>
</tr>
<tr>
<td>KeyRegenerationInterval</td>
<td>3600 (seconds)</td>
<td>Server</td>
</tr>
<tr>
<td>ListenAddress</td>
<td>No default</td>
<td>Server</td>
</tr>
<tr>
<td>LocalForward</td>
<td>No default</td>
<td>Client</td>
</tr>
</tbody>
</table>

Keywords in Secure Shell Configuration Files (Login to R)

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Default Value</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>LoginGraceTime</td>
<td>120 (seconds)</td>
<td>Server</td>
</tr>
<tr>
<td>LogLevel</td>
<td>info</td>
<td>Both</td>
</tr>
<tr>
<td>LookupClientHostnames</td>
<td>yes</td>
<td>Server</td>
</tr>
<tr>
<td>MACs</td>
<td>hmac-sha1, hmac-md5</td>
<td>Both</td>
</tr>
<tr>
<td>Match</td>
<td>No default</td>
<td>Server</td>
</tr>
<tr>
<td>MaxStartups</td>
<td>10:30:60</td>
<td>Server</td>
</tr>
</tbody>
</table>
### TABLE 16–3  Keywords in Secure Shell Configuration Files (Login to R)  
(Continued)

<table>
<thead>
<tr>
<th>Keyword</th>
<th>DefaultValue</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>NoHostAuthenticationForLocalHost</td>
<td>no</td>
<td>Client</td>
</tr>
<tr>
<td>NumberOfPasswordPrompts</td>
<td>3</td>
<td>Client</td>
</tr>
<tr>
<td>PAMServiceName</td>
<td>No default</td>
<td>Server</td>
</tr>
<tr>
<td>PAMServicePrefix</td>
<td>No default</td>
<td>Server</td>
</tr>
<tr>
<td>PasswordAuthentication</td>
<td>yes</td>
<td>Both</td>
</tr>
<tr>
<td>PermitEmptyPasswords</td>
<td>no</td>
<td>Server</td>
</tr>
<tr>
<td>PermitRootLogin</td>
<td>no</td>
<td>Server</td>
</tr>
<tr>
<td>PermitUserEnvironment</td>
<td>no</td>
<td>Server</td>
</tr>
<tr>
<td>PidFile</td>
<td>/system/volatile/sshd.pid</td>
<td>Server</td>
</tr>
<tr>
<td>Port</td>
<td>22</td>
<td>Both</td>
</tr>
<tr>
<td>PreferredAuthentications</td>
<td>hostbased,publickey,keyboard-interactive,password</td>
<td>Client</td>
</tr>
<tr>
<td>PreUserauthHook</td>
<td>No default</td>
<td>Server</td>
</tr>
<tr>
<td>PrintLastLog</td>
<td>yes</td>
<td>Server</td>
</tr>
<tr>
<td>PrintMotd</td>
<td>no</td>
<td>Server</td>
</tr>
<tr>
<td>Protocol</td>
<td>2,1</td>
<td>Both</td>
</tr>
<tr>
<td>ProxyCommand</td>
<td>No default.</td>
<td>Client</td>
</tr>
<tr>
<td>PubkeyAuthentication</td>
<td>yes</td>
<td>Both</td>
</tr>
<tr>
<td>RekeyLimit</td>
<td>1G to 4G</td>
<td>Client</td>
</tr>
<tr>
<td>RemoteForward</td>
<td>No default.</td>
<td>Client</td>
</tr>
<tr>
<td>RhostsAuthentication</td>
<td>no</td>
<td>Server,v1</td>
</tr>
<tr>
<td>RhostsRSAAuthentication</td>
<td>no</td>
<td>Server,v1</td>
</tr>
<tr>
<td>RSAAuthentication</td>
<td>no</td>
<td>Server,v1</td>
</tr>
</tbody>
</table>

### TABLE 16–4  Keywords in Secure Shell Configuration Files (S to X)

<table>
<thead>
<tr>
<th>Keyword</th>
<th>DefaultValue</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>ServerAliveCountMax</td>
<td>3</td>
<td>Client</td>
</tr>
<tr>
<td>ServerAliveInterval</td>
<td>0</td>
<td>Client</td>
</tr>
</tbody>
</table>
TABLE 16–4  Keywords in Secure Shell Configuration Files (S to X)  

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Default Value</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>ServerKeyBits</td>
<td>512 to 768</td>
<td>Server, v1</td>
</tr>
<tr>
<td>StrictHostKeyChecking</td>
<td>ask</td>
<td>Client</td>
</tr>
<tr>
<td>StrictModes</td>
<td>yes</td>
<td>Server</td>
</tr>
<tr>
<td>Subsystem</td>
<td>sftp /usr/lib/ssh/sftp-server</td>
<td>Server</td>
</tr>
<tr>
<td>SyslogFacility</td>
<td>auth</td>
<td>Server</td>
</tr>
<tr>
<td>UseFIPS140</td>
<td>no</td>
<td>Both</td>
</tr>
<tr>
<td>UseOpenSSLEngine</td>
<td>yes</td>
<td>Both</td>
</tr>
<tr>
<td>UsePrivilegedPort</td>
<td>no</td>
<td>Both</td>
</tr>
<tr>
<td>User</td>
<td>No default</td>
<td>Client</td>
</tr>
<tr>
<td>UserKnownHostsFile</td>
<td>~/.ssh/known_hosts</td>
<td>Client</td>
</tr>
<tr>
<td>UseRsh</td>
<td>no</td>
<td>Client</td>
</tr>
<tr>
<td>VerifyReverseMapping</td>
<td>no</td>
<td>Server</td>
</tr>
<tr>
<td>X11DisplayOffset</td>
<td>10</td>
<td>Server</td>
</tr>
<tr>
<td>X11Forwarding</td>
<td>yes</td>
<td>Server</td>
</tr>
<tr>
<td>X11UseLocalHost</td>
<td>yes</td>
<td>Server</td>
</tr>
<tr>
<td>XAuthLocation</td>
<td>/usr/bin/xauth</td>
<td>Both</td>
</tr>
</tbody>
</table>

**Host-Specific Parameters in Secure Shell**

If it is useful to have different Secure Shell characteristics for different local hosts, the administrator can define separate sets of parameters in the `/etc/ssh/ssh_config` file to be applied according to host or regular expression. This task is done by grouping entries in the file by `Host` keyword. If the `Host` keyword is not used, the entries in the client configuration file apply to whichever local host a user is working on.

**Secure Shell and Login Environment Variables**

When the following Secure Shell keywords are not set in the `sshd_config` file, they obtain their value from equivalent entries in the `/etc/default/login` file.
When the following variables are set by the initialization scripts from the user's login shell, the `sshd` daemon uses those values. When the variables are not set, the daemon uses the default value.

<table>
<thead>
<tr>
<th>TIMEZONE</th>
<th>Controls the setting of the TZ environment variable. When not set, the <code>sshd</code> daemon uses value of TZ when the daemon was started.</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALTSHELL</td>
<td>Controls the setting of the SHELL environment variable. The default is ALTSHELL=YES, where the <code>sshd</code> daemon uses the value of the user's shell. When ALTSHELL=NO, the SHELL value is not set.</td>
</tr>
<tr>
<td>PATH</td>
<td>Controls the setting of the PATH environment variable. When the value is not set, the default path is /usr/bin.</td>
</tr>
<tr>
<td>SUPATH</td>
<td>Controls the setting of the PATH environment variable for root. When the value is not set, the default path is /usr/sbin:/usr/bin.</td>
</tr>
</tbody>
</table>

For more information, see the `login(1)` and `sshd(1M)` man pages.

**Maintaining Known Hosts in Secure Shell**

Each host that needs to communicate securely with another host must have the server's public key stored in the local host's /etc/ssh/ssh_known_hosts file. Although a script could be used to update the /etc/ssh/ssh_known_hosts files, such a practice is heavily discouraged because a script opens a major security vulnerability.
The /etc/ssh/ssh_known_hosts file should only be distributed by a secure mechanism as follows:

- Over a secure connection, such as Secure Shell, IPsec, or Kerberized ftp from a known and trusted machine
- At system install time

To avoid the possibility of an intruder gaining access by inserting bogus public keys into a known_hosts file, you should use a known and trusted source of the ssh_known_hosts file. The ssh_known_hosts file can be distributed during installation. Later, scripts that use the scp command can be used to pull in the latest version.

## Secure Shell Files

The following table shows the important Secure Shell files and the suggested file permissions.

<table>
<thead>
<tr>
<th>File Name</th>
<th>Description</th>
<th>Suggested Permissions and Owner</th>
</tr>
</thead>
<tbody>
<tr>
<td>/etc/sshd_config</td>
<td>Contains configuration data for ssdh, the Secure Shell daemon.</td>
<td>-rw-r--r-- root</td>
</tr>
<tr>
<td>/etc/ssh_host_dsa_key or</td>
<td>Contains the host private key.</td>
<td>-rw------- root</td>
</tr>
<tr>
<td>/etc/ssh/ssh_host_rsa_key</td>
<td></td>
<td></td>
</tr>
<tr>
<td>host-private-key.pub</td>
<td>Contains the host public key, for example,</td>
<td>-rw-r-r-- root</td>
</tr>
<tr>
<td>/system/volatile/sshd.pid</td>
<td>Contains the process ID of the Secure Shell daemon, ssdh. If multiple</td>
<td>-rw-r--r-- root</td>
</tr>
<tr>
<td></td>
<td>daemons are running, the file contains the last daemon that was started.</td>
<td></td>
</tr>
<tr>
<td>~/.ssh/authorized_keys</td>
<td>Holds the public keys of the user who is allowed to log in to the user</td>
<td>-rw-r--r-- username</td>
</tr>
<tr>
<td></td>
<td>account.</td>
<td></td>
</tr>
<tr>
<td>/etc/ssh/ssh_known_hosts</td>
<td>Contains the host public keys for all hosts with which the client can</td>
<td>-rw-r--r-- root</td>
</tr>
<tr>
<td></td>
<td>communicate securely. The file is populated by the administrator.</td>
<td></td>
</tr>
<tr>
<td>~/.ssh/known_hosts</td>
<td>Contains the host public keys for all hosts with which the client can</td>
<td>-rw-r--r-- username</td>
</tr>
<tr>
<td></td>
<td>communicate securely. The file is maintained automatically. Whenever the</td>
<td></td>
</tr>
<tr>
<td></td>
<td>user connects with an unknown host, the remote host key is added to the file.</td>
<td></td>
</tr>
<tr>
<td>/etc/default/login</td>
<td>Provides defaults for the ssdh daemon when corresponding</td>
<td>-r----r-- root</td>
</tr>
<tr>
<td></td>
<td>sshd_config parameters are not set.</td>
<td></td>
</tr>
</tbody>
</table>
TABLE 16–5  Secure Shell Files

<table>
<thead>
<tr>
<th>File Name</th>
<th>Description</th>
<th>Suggested Permissions and Owner</th>
</tr>
</thead>
<tbody>
<tr>
<td>/etc/nologin</td>
<td>If this file exists, the sshd daemon only permits root to log in. The contents of this file are displayed to users who are attempting to log in.</td>
<td>-rw-r--r-- root</td>
</tr>
<tr>
<td>~/.rhosts</td>
<td>Contains the host-user name pairs that specify the hosts to which the user can log in without a password. This file is also used by the rlogin and rshd daemons.</td>
<td>-rw-r--r-- username</td>
</tr>
<tr>
<td>~/.ssh/hosts</td>
<td>Contains the host-user name pairs that specify the hosts to which the user can log in without a password. This file is not used by other utilities. For more information, see the sshd(1M) man page in the FILES section.</td>
<td>-rw-r--r-- username</td>
</tr>
<tr>
<td>/etc/hosts.equiv</td>
<td>Contains the hosts that are used in .rhosts authentication. This file is also used by the rlogin and rshd daemons.</td>
<td>-rw-r--r-- root</td>
</tr>
<tr>
<td>/etc/ssh/hosts.equiv</td>
<td>Contains the hosts that are used in host-based authentication. This file is not used by other utilities.</td>
<td>-rw-r--r-- root</td>
</tr>
<tr>
<td>~/.ssh/environment</td>
<td>Contains initial assignments at login. By default, this file is not read. The PermitUserEnvironment keyword in the sshd_config file must be set to yes for this file to be read.</td>
<td>-rw-r--r-- username</td>
</tr>
<tr>
<td>~/.ssh/rc</td>
<td>Contains initialization routines that run before the user shell starts. For a sample initialization routine, see the sshd(1M) man page.</td>
<td>-rw-r--r-- username</td>
</tr>
<tr>
<td>/etc/ssh/sshrc</td>
<td>Contains host-specific initialization routines that are specified by an administrator.</td>
<td>-rw-r--r-- root</td>
</tr>
<tr>
<td>/etc/ssh/ssh_config</td>
<td>Configures system settings on the client system.</td>
<td>-rw-r--r-- root</td>
</tr>
<tr>
<td>~/.ssh/config</td>
<td>Configures user settings which override system settings.</td>
<td>-rw-r--r-- username</td>
</tr>
</tbody>
</table>

Note – The sshd config file can be overridden by a file from a site-customized package. For more information, see the definition of the overlay file attribute in the pkg(5) man page.

The following table lists the Secure Shell files that can be overridden by keywords or command options.

TABLE 16–6  Overrides for the Location of Secure Shell Files

<table>
<thead>
<tr>
<th>File Name</th>
<th>Keyword Override</th>
<th>Command-Line Override</th>
</tr>
</thead>
<tbody>
<tr>
<td>/etc/ssh/ssh_config</td>
<td></td>
<td>ssh -F config-file</td>
</tr>
<tr>
<td></td>
<td></td>
<td>scp -F config-file</td>
</tr>
</tbody>
</table>
Secure Shell Commands

The following table summarizes the major Secure Shell commands.

<table>
<thead>
<tr>
<th>Man Page for Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ssh(1)</td>
<td>Logs a user in to a remote machine and securely executes commands on a remote machine. The ssh command enables secure encrypted communications between two untrusted hosts over an insecure network. X11 connections and arbitrary TCP/IP ports can also be forwarded over the secure channel.</td>
</tr>
<tr>
<td>sshd(1M)</td>
<td>Is the daemon for Secure Shell. The daemon listens for connections from clients and enables secure encrypted communications between two untrusted hosts over an insecure network.</td>
</tr>
<tr>
<td>ssh-add(1)</td>
<td>Adds RSA or DSA identities to the authentication agent, ssh-agent. Identities are also called keys.</td>
</tr>
<tr>
<td>ssh-agent(1)</td>
<td>Holds private keys that are used for public key authentication. The ssh-agent program is started at the beginning of an X-session or a login session. All other windows and other programs are started as clients of the ssh-agent program. Through the use of environment variables, the agent can be located and used for authentication when users use the ssh command to log in to other systems.</td>
</tr>
<tr>
<td>ssh-keygen(1)</td>
<td>Generates and manages authentication keys for Secure Shell.</td>
</tr>
<tr>
<td>ssh-keyscan(1)</td>
<td>Gathers the public keys of a number of Secure Shell hosts. Aids in building and verifying ssh_known_hosts files.</td>
</tr>
<tr>
<td>ssh-keysign(1M)</td>
<td>Is used by the ssh command to access the host keys on the local host. Generates the digital signature that is required during host-based authentication with Secure Shell v2. The command is invoked by the ssh command, not by the user.</td>
</tr>
</tbody>
</table>
TABLE 16–7  Commands in Secure Shell  (Continued)

<table>
<thead>
<tr>
<th>Man Page for Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>scp(1)</td>
<td>Securely copies files between hosts on a network over an encrypted ssh transport. Unlike the rcp command, the scp command prompts for passwords or passphrases, if password information is needed for authentication.</td>
</tr>
<tr>
<td>sftp(1)</td>
<td>Is an interactive file transfer program that is similar to the ftp command. Unlike the ftp command, the sftp command performs all operations over an encrypted ssh transport. The command connects, logs in to the specified host name, and then enters interactive command mode.</td>
</tr>
</tbody>
</table>

The following table lists the command options that override Secure Shell keywords. The keywords are specified in the ssh_config and sshd_config files.

TABLE 16–8  Command-Line Equivalents for Secure Shell Keywords

<table>
<thead>
<tr>
<th>Keyword</th>
<th>ssh Command-Line Override</th>
<th>scp Command-Line Override</th>
</tr>
</thead>
<tbody>
<tr>
<td>BatchMode</td>
<td>ssh -B</td>
<td>scp -B</td>
</tr>
<tr>
<td>BindAddress</td>
<td>ssh -b bind-addr</td>
<td>scp -a bind-addr</td>
</tr>
<tr>
<td>Cipher</td>
<td>ssh -c cipher</td>
<td>scp -c cipher</td>
</tr>
<tr>
<td>Ciphers</td>
<td>ssh -c cipher-spec</td>
<td>scp -c cipher-spec</td>
</tr>
<tr>
<td>Compression</td>
<td>ssh -C</td>
<td>scp -C</td>
</tr>
<tr>
<td>DynamicForward</td>
<td>ssh -D SOCKS4-port</td>
<td></td>
</tr>
<tr>
<td>EscapeChar</td>
<td>ssh -e escape-char</td>
<td></td>
</tr>
<tr>
<td>ForwardAgent</td>
<td>ssh -A to enable</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ssh -a to disable</td>
<td></td>
</tr>
<tr>
<td>ForwardX11</td>
<td>ssh -X to enable</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ssh -x to disable</td>
<td></td>
</tr>
<tr>
<td>GatewayPorts</td>
<td>ssh -g</td>
<td></td>
</tr>
<tr>
<td>IPv4</td>
<td>ssh -4</td>
<td>scp -4</td>
</tr>
<tr>
<td>IPv6</td>
<td>ssh -6</td>
<td>scp -6</td>
</tr>
<tr>
<td>LocalForward</td>
<td>ssh -L localport:remotehost:remoteport</td>
<td></td>
</tr>
<tr>
<td>MACS</td>
<td>ssh -m mac-spec</td>
<td></td>
</tr>
<tr>
<td>Port</td>
<td>ssh -p port</td>
<td>scp -P port</td>
</tr>
<tr>
<td>Protocol</td>
<td>ssh -2 for v2 only</td>
<td></td>
</tr>
<tr>
<td>RemoteForward</td>
<td>ssh -R remoteport:localhost:localport</td>
<td></td>
</tr>
</tbody>
</table>
CHAPTER 17

Using Simple Authentication and Security Layer

This chapter includes information about the Simple Authentication and Security Layer (SASL).

- “SASL (Overview)” on page 317
- “SASL (Reference)” on page 318

SASL (Overview)

The Simple Authentication and Security Layer (SASL) is a framework that provides authentication and optional security services to network protocols. An application calls the SASL library, /usr/lib/libsasl.so, which provides a glue layer between the application and the various SASL mechanisms. The mechanisms are used in the authentication process and in providing optional security services. The version of SASL is derived from the Cyrus SASL with a few changes.

SASL provides the following services:

- Loading of any plug-ins
- Determining the necessary security options from the application to aid in the choice of a security mechanism
- Listing of plug-ins that are available to the application
- Choosing the best mechanism from a list of available mechanisms for a particular authentication attempt
- Routing the authentication data between the application and the chosen mechanism
- Providing information about the SASL negotiation back to the application
The following section provides information about the implementation of SASL.

**SASL Plug-ins**

SASL plug-ins provide support for security mechanisms, user-canonicalization, and auxiliary property retrieval. By default, the dynamically loaded 32-bit plug-ins are installed in `/usr/lib/sasl`, and the 64-bit plug-ins are installed in `/usr/lib/sasl/$ISA`. The following security mechanism plug-ins are provided:

- `crammd5.so.1` CRAM-MD5, which supports authentication only, no authorization
- `digestmd5.so.1` DIGEST-MD5, which supports authentication, integrity, and privacy, as well as authorization
- `gssapi.so.1` GSSAPI, which supports authentication, integrity, and privacy, as well as authorization. The GSSAPI security mechanism requires a functioning Kerberos infrastructure.
- `plain.so.1` PLAIN, which supports authentication and authorization.

In addition, the EXTERNAL security mechanism plug-in and the INTERNAL user canonicalization plug-ins are built into `libsasl.so.1`. The EXTERNAL mechanism supports authentication and authorization. The mechanism supports integrity and privacy if the external security source provides it. The INTERNAL plug-in adds the realm name if necessary to the username.

The Oracle Solaris release is not supplying any auxprop plug-ins at this time. For the CRAM-MD5 and DIGEST-MD5 mechanism plug-ins to be fully operational on the server side, the user must provide an auxprop plug-in to retrieve clear text passwords. The PLAIN plug-in requires additional support to verify the password. The support for password verification can be one of the following: a callback to the server application, an auxprop plug-in, `saslauthd`, or `pwcheck`. The `saslauthd` and `pwcheck` daemons are not provided in the Oracle Solaris releases. For better interoperability, restrict server applications to those mechanisms that are fully operational by using the `mech_list` SASL option.

**SASL Environment Variable**

By default, the client authentication name is set to `getenv("LOGNAME")`. This variable can be reset by the client or by the plug-in.
**SASL Options**

The behavior of `libsasl` and the plug-ins can be modified on the server side by using options that can be set in the `/etc/sasl/app.conf` file. The variable `app` is the server-defined name for the application. The documentation for the server `app` should specify the application name.

The following options are supported:

- **auto_transition**: Automatically transitions the user to other mechanisms when the user does a successful plain text authentication.
- **auxprop_login**: Lists the name of auxiliary property plug-ins to use.
- **canon_user_plugin**: Selects the canon_user plug-in to use.
- **mech_list**: Lists the mechanisms that are allowed to be used by the server application.
- **pwcheck_method**: Lists the mechanisms used to verify passwords. Currently, auxprop is the only allowed value.
- **reauth_timeout**: Sets the length of time, in minutes, that authentication information is cached for a fast reauthentication. This option is used by the DIGEST-MD5 plug-in. Setting this option to 0 disables reauthentication.

The following options are not supported:

- **plugin_list**: Lists available mechanisms. Not used because the option changes the behavior of the dynamic loading of plugins.
- **saslauthd_path**: Defines the location of the saslauthd door, which is used for communicating with the saslauthd daemon. The saslauthd daemon is not included in the Oracle Solaris release. So, this option is also not included.
- **keytab**: Defines the location of the keytab file used by the GSSAPI plug-in. Use the `KRB5_KTNAME` environment variable instead to set the default keytab location.

The following options are options not found in Cyrus SASL. However, they have been added for the Oracle Solaris release:

- **use_authid**: Acquire the client credentials rather than use the default credentials when creating the GSS client security context. By default, the default client Kerberos identity is used.
- **log_level**: Sets the desired level of logging for a server.
This chapter provides information about how to use Secure RPC to authenticate a host and a user across an NFS mount, and covers the following topics:
- “Overview of Secure RPC” on page 321
- “Administering Authentication With Secure RPC (Tasks)” on page 326

Overview of Secure RPC

Secure RPC (Remote Procedure Call) protects remote procedures with an authentication mechanism. The Diffie-Hellman authentication mechanism authenticates both the host and the user who is making a request for a service. The authentication mechanism uses Data Encryption Standard (DES) encryption. Applications that use Secure RPC include NFS and the NIS naming service.

NFS Services and Secure RPC

NFS enables several hosts to share files over the network. Under the NFS service, a server holds the data and resources for several clients. The clients have access to the file systems that the server shares with the clients. Users who are logged in to the client systems can access the file systems by mounting the file systems from the server. To the user on the client system, it appears as if the files are local to the client. One of the most common uses of NFS allows systems to be installed in offices, while storing all user files in a central location. Some features of the NFS service, such as the -nosuid option to the mount command, can be used to prohibit the opening of devices and file systems by unauthorized users.
The NFS service uses Secure RPC to authenticate users who make requests over the network. This process is known as Secure NFS. The Diffie-Hellman authentication mechanism, AUTH_DH, uses DES encryption to ensure authorized access. The AUTH_DH mechanism has also been called AUTH_DES. For more information, see the following:

- To set up and administer Secure NFS, see “Administering the Secure NFS System” in Managing Network File Systems in Oracle Solaris 11.1.
- For an outline of the transactions that are involved in RPC authentication, see “Implementation of Diffie-Hellman Authentication” on page 323.

**Kerberos Authentication**

Kerberos is an authentication system that was developed at MIT. Some encryption in Kerberos is based on DES. Kerberos V4 support is no longer supplied as part of Secure RPC. However, a client-side and server-side implementation of Kerberos V5, which uses RPCSEC_GSS, is included with this release. For more information, see “How to Configure Kerberos NFS Servers” on page 382.

**DES Encryption With Secure NFS**

The Data Encryption Standard (DES) encryption functions use a 56-bit key to encrypt data. If two credential users or principals know the same DES key, they can communicate in private by using the key to encipher and decipher text. DES is a relatively fast encryption mechanism.

The risk of using just the DES key is that an intruder can collect enough cipher-text messages that were encrypted with the same key to be able to discover the key and decipher the messages. For this reason, security systems such as Secure NFS need to change the keys frequently.

**Diffie-Hellman Authentication and Secure RPC**

The Diffie-Hellman (DH) method of authenticating a user is nontrivial for an intruder to crack. The client and the server have their own private key, which they use with the public key to devise a common key. The private key is also known as the secret key. The client and the server use the common key to communicate with each other. The common key is encrypted with an agreed-upon encryption function, such as DES.

Authentication is based on the ability of the sending system to use the common key to encrypt the current time. Then, the receiving system can decrypt and check against its current time. The time on the client and the server must be synchronized. For more information, see “Managing Network Time Protocol (Tasks)” in Introduction to Oracle Solaris 11 Network Services.

The public keys and private keys are stored in an NIS database. NIS stores the keys in the publickey map. This file contains the public key and the private key for all potential users.
The system administrator is responsible for setting up NIS maps and for generating a public key and a private key for each user. The private key is stored in encrypted form with the user’s password. This process makes the private key known only to the user.

**Implementation of Diffie-Hellman Authentication**

This section describes the series of transactions in a client-server session that use Diffie-Hellman authentication (AUTH_DH).

**Generating the Public Keys and Secret Keys for Secure RPC**

Sometime prior to a transaction, the administrator runs either the newkey or the nisaddc red command to generate a public key and a secret key. Each user has a unique public key and secret key. The public key is stored in a public database. The secret key is stored in encrypted form in the same database. The chkey command changes the key pair.

**Running the keylogin Command for Secure RPC**

Normally, the login password is identical to the Secure RPC password. In this case, the keylogin command is not required. However, if the passwords are different, the users have to log in and then run the keylogin command.

The keylogin command prompts the user for a Secure RPC password. The command then uses the password to decrypt the secret key. The keylogin command then passes the decrypted secret key to the keyserv er program. The keyserv er is an RPC service with a local instance on every computer. The keyserv er saves the decrypted secret key and waits for the user to initiate a Secure RPC transaction with a server.

If both the login password and the RPC password are the same, the login process passes the secret key to the keyserv er. If the passwords are required to be different, then the user must always run the keylogin command. When the keylogin command is included in the user's environment configuration file, such as the ~/.login, ~/.cshrc, or ~/.profile file, the keylogin command runs automatically whenever the user logs in.

**Generating the Conversation Key for Secure RPC**

When the user initiates a transaction with a server, the following occurs:

1. The keyserv er randomly generates a conversation key.
2. The kernel uses the conversation key, plus other material, to encrypt the client’s timestamp.
3. The keyserv er looks up the server’s public key in the public key database. For more information, see the publickey(4) man page.
4. The keyserv er uses the client’s secret key and the server’s public key to create a common key.
5. The keyserv er encrypts the conversation key with the common key.
Initially Contacting the Server in Secure RPC

The transmission, which includes the encrypted timestamp and the encrypted conversation key, is then sent to the server. The transmission includes a credential and a verifier. The credential contains three components:

- The client’s network name
- The conversation key, which is encrypted with the common key
- A “window,” which is encrypted with the conversation key

The window is the difference in time that the client says should be allowed between the server’s clock and the client’s timestamp. If the difference between the server’s clock and the timestamp is greater than the window, the server rejects the client’s request. Under normal circumstances, this rejection does not happen, because the client first synchronizes with the server before starting the RPC session.

The client’s verifier contains the following:

- The encrypted timestamp
- An encrypted verifier of the specified window, which is decremented by 1

The window verifier is needed in case somebody wants to impersonate a user. The impersonator can write a program that, instead of filling in the encrypted fields of the credential and verifier, just inserts random bits. The server decrypts the conversation key into some random key. The server then uses the key to try to decrypt the window and the timestamp. The result is random numbers. After a few thousand trials, however, the random window/timestamp pair is likely to pass the authentication system. The window verifier lessens the chance that a fake credential could be authenticated.

Decrypting the Conversation Key in Secure RPC

When the server receives the transmission from the client, the following occurs:

1. The keyserver that is local to the server looks up the client’s public key in the public key database.
2. The keyserver uses the client’s public key and the server’s secret key to deduce the common key. The common key is the same common key that is computed by the client. Only the server and the client can calculate the common key because the calculation requires knowing one of the secret keys.
3. The kernel uses the common key to decrypt the conversation key.
4. The kernel calls the keyserver to decrypt the client’s timestamp with the decrypted conversation key.
Storing Information on the Server in Secure RPC

After the server decrypts the client’s timestamp, the server stores four items of information in a credential table:

- The client’s computer name
- The conversation key
- The window
- The client’s timestamp

The server stores the first three items for future use. The server stores the client’s timestamp to protect against replays. The server accepts only timestamps that are chronologically greater than the last timestamp seen. As a result, any replayed transactions are guaranteed to be rejected.

Note – Implicit in these transactions is the name of the caller, who must be authenticated in some manner. The key server cannot use DES authentication to authenticate the caller because the use of DES by the key server would create a deadlock. To avoid a deadlock, the key server stores the secret keys by user ID (UID) and grants requests only to local root processes.

Returning the Verifier to the Client in Secure RPC

The server returns a verifier to the client, which includes the following:

- The index ID, which the server records in its credential cache
- The client’s timestamp minus 1, which is encrypted by the conversation key

The reason for subtracting 1 from the client’s timestamp is to ensure that the timestamp is out of date. An out-of-date timestamp cannot be reused as a client verifier.

Authenticating the Server in Secure RPC

The client receives the verifier and authenticates the server. The client knows that only the server could have sent the verifier because only the server knows what timestamp the client sent.

Handling Transactions in Secure RPC

With every transaction after the first transaction, the client returns the index ID to the server in its next transaction. The client also sends another encrypted timestamp. The server sends back the client’s timestamp minus 1, which is encrypted by the conversation key.
Administering Authentication With Secure RPC (Tasks)

By requiring authentication for use of mounted NFS file systems, you increase the security of your network.

Administering Secure RPC (Task Map)

The following task map points to procedures that configure Secure RPC for NIS, and NFS.

<table>
<thead>
<tr>
<th>Task</th>
<th>Description</th>
<th>For Instructions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Start the keyserver. Ensures that keys can be created so that users can be authenticated.</td>
<td>“How to Restart the Secure RPC Keyserver” on page 326</td>
</tr>
<tr>
<td>2.</td>
<td>Set up credentials on an NIS host. Ensures that the root user on a host can be authenticated in an NIS environment.</td>
<td>“How to Set Up a Diffie-Hellman Key for an NIS Host” on page 326</td>
</tr>
<tr>
<td>3.</td>
<td>Give an NIS user a key. Enables a user to be authenticated in an NIS environment.</td>
<td>“How to Set Up a Diffie-Hellman Key for an NIS User” on page 327</td>
</tr>
<tr>
<td>4.</td>
<td>Share NFS files with authentication. Enables an NFS server to securely protect shared file systems using authentication.</td>
<td>“How to Share NFS Files With Diffie-Hellman Authentication” on page 328</td>
</tr>
</tbody>
</table>

▼ How to Restart the Secure RPC Keyserver

Before You Begin
You must assume the root role. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1. Verify that the keyserv daemon is running.
   
   # svcs \*keyserv\*
   
   STATE STIME FMRI
   
   disabled Dec_14 svc:/network/rpc/keyserv

2. Enable the keyserver service if the service is not online.
   
   # svcadm enable network/rpc/keyserv

▼ How to Set Up a Diffie-Hellman Key for an NIS Host

This procedure should be done on every host in the NIS domain.

Before You Begin
You must assume the root role. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.
If the default naming service is not NIS, add the publickey map to the naming service.

a. Verify that the value of config/default for the naming service is not nis.

```
# svccfg -s name-service/switch listprop config
```

```
config    application
config/value_authorization astring solaris.smf.value.name-service.switch
config/default    astring files
config/host    astring "files nis dns"
config/printer    astring "user files nis"
```

If the value of config/default is nis, you can stop here.

b. Set the naming service for publickey to nis.

```
# svccfg
# svccfg -s name-service/switch setprop config/publickey = astring: "nis"
# svccfg -s name-service/switch:default refresh
```

c. Confirm the publickey value.

```
# svccfg
# svccfg -s name-service/switch listprop
```

```
config    application
config/value_authorization astring solaris.smf.value.name-service.switch
config/default    astring files
config/host    astring "files nis dns"
config/printer    astring "user files nis"
config/publickey    astring nis
```

On this system, the value of publickey is listed because it differs from the default, files.

2 Create a new key pair by using the newkey command.

```
# newkey -h hostname
```

where hostname is the name of the client.

Example 18–1 Setting Up a New Key for root on an NIS Client

In the following example, earth is set up as a secure NIS client. The administrator is assigned the Name Service Security rights profile.

```
# newkey -h earth
Adding new key for unix.earth@example.com
New Password: <Type password>
Retype password: <Retype password>
Please wait for the database to get updated...
Your new key has been successfully stored away.
```

▼ How to Set Up a Diffie-Hellman Key for an NIS User

This procedure should be done for every user in the NIS domain.
Only system administrators, when logged in to the NIS master server, can generate a new key for a user. The administrators must be assigned the Name Service Security rights profile. Become an administrator with the required security attributes. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

Create a new key for a user.

```
  # newkey -u username
```

where `username` is the name of the user. The system prompts for a password. You can type a generic password. The private key is stored in an encrypted form by using the generic password.

Tell the user to log in and type the `chkey -p` command.

This command allows users to re-encrypt their private keys with a password known only to the user.

**Note** – The `chkey` command can be used to create a new key pair for a user.

**Example 18-2** Setting Up and Encrypting a New User Key in NIS

In this example, superuser sets up the key.

```
  # newkey -u jdoe

  Adding new key for unix.12345@example.com
  New Password: <Type password>
  Retype password: <Retype password>
  Please wait for the database to get updated...
  Your new key has been successfully stored away.

  #
```

Then the user `jdoe` re-encrypts the key with a private password.

```
  % chkey -p

  Updating nis publickey database.
  Reencrypting key for unix.12345@example.com
  Please enter the Secure-RPC password for jdoe: <Type password>
  Please enter the login password for jdoe: <Type password>
  Sending key change request to centralexample...
```

**How to Share NFS Files With Diffie-Hellman Authentication**

This procedure protects shared file systems on an NFS server by requiring authentication for access.
**Before You Begin**

Diffie-Hellman public key authentication must be enabled on the network. To enable authentication on the network, complete "How to Set Up a Diffie-Hellman Key for an NIS Host" on page 326.

You must become an administrator who is assigned the System Management rights profile to perform this task. For more information, see "How to Use Your Assigned Administrative Rights" on page 157.

1. **On the NFS server, share a file system with Diffie-Hellman authentication.**
   
   ```bash
   # share -F nfs -o sec=dh /filesystem
   ```
   
   where `filesystem` is the file system that is being shared.

   The `-o sec=dh` option means that AUTH_DH authentication is now required to access the file system.

2. **On an NFS client, mount a file system with Diffie-Hellman authentication.**
   
   ```bash
   # mount -F nfs -o sec=dh server:filesystem mount-point
   ```
   
   `server` Is the name of the system that is sharing `filesystem`

   `filesystem` Is the name of the file system that is being shared, such as `opt`

   `mount-point` Is the name of the mount point, such as `/opt`

   The `-o sec=dh` option mounts the file system with AUTH_DH authentication.
Kerberos Service

This section provides information about the configuration, management and use of the Kerberos service in the following chapters:

- Chapter 19, “Introduction to the Kerberos Service”
- Chapter 20, “Planning for the Kerberos Service”
- Chapter 21, “Configuring the Kerberos Service (Tasks)”
- Chapter 22, “Kerberos Error Messages and Troubleshooting”
- Chapter 23, “Administering Kerberos Principals and Policies (Tasks)”
- Chapter 24, “Using Kerberos Applications (Tasks)”
- Chapter 25, “The Kerberos Service (Reference)”
Introduction to the Kerberos Service

This chapter introduces the Kerberos service. The following is a list of the overview information in this chapter.

- “What Is the Kerberos Service?” on page 333
- “How the Kerberos Service Works” on page 334
- “Kerberos Security Services” on page 340
- “Components of Various Kerberos Releases” on page 341

What Is the Kerberos Service?

The Kerberos service is a client-server architecture that provides secure transactions over networks. The service offers strong user authentication, as well as integrity and privacy. Authentication guarantees that the identities of both the sender and the recipient of a network transaction are true. The service can also verify the validity of data being passed back and forth (integrity) and encrypt the data during transmission (privacy). Using the Kerberos service, you can log in to other machines, execute commands, exchange data, and transfer files securely. Additionally, the service provides authorization services, which allows administrators to restrict access to services and machines. Moreover, as a Kerberos user, you can regulate other people’s access to your account.

The Kerberos service is a single sign-on system, which means that you only need to authenticate yourself to the service once per session, and all subsequent transactions during the session are automatically secured. After the service has authenticated you, you do not need to authenticate yourself every time you use a Kerberos-based command such as ftp or ssh, or to access data on an NFS file system. Thus, you do not have to send your password over the network, where it can be intercepted, each time you use these services.

The Kerberos service in the Oracle Solaris release is based on the Kerberos V5 network authentication protocol that was developed at the Massachusetts Institute of Technology (MIT). People who have used the Kerberos V5 product will therefore find the Oracle Solaris version very familiar. Because the Kerberos V5 protocol is a de facto industry standard for
network security, the Oracle Solaris version promotes interoperability with other systems. In other words, because the Kerberos service in the Oracle Solaris release works with systems that use the Kerberos V5 protocol, the service allows for secure transactions even over heterogeneous networks. Moreover, the service provides authentication and security both between domains and within a single domain.

The Kerberos service allows for flexibility in running Oracle Solaris applications. You can configure the service to allow both Kerberos-based and non-Kerberos-based requests for network services such as the NFS service, telnet, and ftp. As a result, current applications still work even if they are running on systems on which the Kerberos service is not enabled. Of course, you can also configure the Kerberos service to allow only Kerberos-based network requests.

The Kerberos service provides a security mechanism which allows the use of Kerberos for authentication, integrity, and privacy when using applications that use the Generic Security Service Application Programming Interface (GSS-API). However, applications do not have to remain committed to the Kerberos service if other security mechanisms are developed. Because the service is designed to integrate modularly into the GSS-API, applications that use the GSS-API can utilize whichever security mechanism best suits their needs.

How the Kerberos Service Works

The following is an overview of the Kerberos authentication system. For a more detailed description, see “How the Kerberos Authentication System Works” on page 516.

From the user’s standpoint, the Kerberos service is mostly invisible after the Kerberos session has been started. Commands such as ssh or ftp work about the same. Initializing a Kerberos session often involves no more than logging in and providing a Kerberos password.

The Kerberos system revolves around the concept of a ticket. A ticket is a set of electronic information that identifies a user or a service such as the NFS service. Just as your driver’s license identifies you and indicates what driving privileges you have, so a ticket identifies you and your network access privileges. When you perform a Kerberos-based transaction (for example, if you remote log in to another machine), you transparently send a request for a ticket to a Key Distribution Center, or KDC. The KDC accesses a database to authenticate your identity and returns a ticket that grants you permission to access the other machine. “Transparency” means that you do not need to explicitly request a ticket. The request happens as part of the rlogin command. Because only an authenticated client can get a ticket for a specific service, another client cannot use rlogin under an assumed identity.

Tickets have certain attributes associated with them. For example, a ticket can be forwardable, which means that it can be used on another machine without a new authentication process. A ticket can also be postdated, which means that it is not valid until a specified time. How tickets can be used, for example, to specify which users are allowed to obtain which types of ticket, is set by policies. Policies are determined when the Kerberos service is installed or administered.
Note – You will frequently see the terms credential and ticket. In the greater Kerberos world, they are often used interchangeably. Technically, however, a credential is a ticket plus the session key for that session. This difference is explained in more detail in “Gaining Access to a Service Using Kerberos” on page 516.

The following sections further explain the Kerberos authentication process.

**Initial Authentication: the Ticket-Granting Ticket**

Kerberos authentication has two phases: an initial authentication that allows for all subsequent authentications, and the subsequent authentications themselves.

The following figure shows how the initial authentication takes place.

**FIGURE 19–1 Initial Authentication for a Kerberos Session**

1. At login (or with `kinit`), Client requests a TGT that allows it to obtain tickets for services.

2. KDC checks database, sends TGT

3. Client uses password to decrypt TGT, thus proving identity; can now use the TGT to obtain other tickets.

**TGT = Ticket-granting ticket**  
**KDC = Key Distribution Center**

1. A client (a user, or a service such as NFS) begins a Kerberos session by requesting a ticket-granting ticket (TGT) from the Key Distribution Center (KDC). This request is often done automatically at login.

   A ticket-granting ticket is needed to obtain other tickets for specific services. Think of the ticket-granting ticket as similar to a passport. Like a passport, the ticket-granting ticket identifies you and allows you to obtain numerous “visas,” where the “visas” (tickets) are not for foreign countries but for remote machines or network services. Like passports and visas,
the ticket-granting ticket and the other various tickets have limited lifetimes. The difference is that “Kerberized” commands notice that you have a passport and obtain the visas for you. You don’t have to perform the transactions yourself.

Another analogy for the ticket-granting ticket is that of a three-day ski pass that is good at four different ski resorts. You show the pass at whichever resort you decide to go to and you receive a lift ticket for that resort, as long as the pass has not expired. Once you have the lift ticket, you can ski all you want at that resort. If you go to another resort the next day, you once again show your pass, and you get an additional lift ticket for the new resort. The difference is that the Kerberos-based commands notice that you have the weekend ski pass, and they get the lift ticket for you. So you don’t have to perform the transactions yourself.

2. The KDC creates a ticket–granting ticket and sends it back, in encrypted form, to the client. The client decrypts the ticket-granting ticket by using the client’s password.

3. Now in possession of a valid ticket-granting ticket, the client can request tickets for all sorts of network operations, such as `rlogin` or `telnet`, for as long as the ticket-granting ticket lasts. This ticket usually lasts for a few hours. Each time the client performs a unique network operation, it requests a ticket for that operation from the KDC.

Subsequent Kerberos Authentications

After the client has received the initial authentication, each subsequent authentication follows the pattern that is shown in the following figure.
1. The client requests a ticket for a particular service, for example, to remote log in to another machine, from the KDC by sending the KDC its ticket-granting ticket as proof of identity.

2. The KDC sends the ticket for the specific service to the client.

For example, suppose user joe wants to access an NFS file system that has been shared with krb5 authentication required. Because he is already authenticated (that is, he already has a ticket-granting ticket), as he attempts to access the files, the NFS client system automatically and transparently obtains a ticket from the KDC for the NFS service.

For example, suppose the user joe uses rlogin on the server boston. Because he is already authenticated, that is, he already has a ticket-granting ticket, he automatically and transparently obtains a ticket as part of the rlogin command. This ticket allows him to remote log in to boston as often as he wants until the ticket expires. If joe wants to remote log in to the machine denver, he obtains another ticket, as in Step 1.

3. The client sends the ticket to the server.

When using the NFS service, the NFS client automatically and transparently sends the ticket for the NFS service to the NFS server.

4. The server allows the client access.
These steps make it appear that the server doesn't ever communicate with the KDC. The server
does, though; it registers itself with the KDC, just as the first client does. For simplicity’s sake,
that part has been left out.

**Kerberos Remote Applications**

The Kerberos-based (or “Kerberized”) commands that a user such as *joe* can use are the
following:

- `ftp`
- `rcp, rlogin, rsh`
- `ssh, scp, sftp`
- `telnet`

These applications are the same as the Solaris applications of the same name. However, they
have been extended to use Kerberos principals to authenticate transactions, thereby providing
Kerberos-based security. See “Kerberos Principals” on page 338 for information on principals.

These commands are discussed further in “Kerberos User Commands” on page 500.

**Kerberos Principals**

A client in the Kerberos service is identified by its *principal*. A principal is a unique identity to
which the KDC can assign tickets. A principal can be a user, such as *joe*, or a service, such as
*nfs* or *telnet*.

By convention, a principal name is divided into three components: the *primary*, the *instance*,
and the *realm*. A typical Kerberos principal would be, for example,

```
joe/admin@ENG.EXAMPLE.COM
```

In this example:

- *joe* is the primary. The primary can be a user name, as shown here, or a service, such as *nfs*.  
The primary can also be the word host, which signifies that this principal is a service principal that is set up to provide various network services, `ftp`, `scp`, `ssh`, and so on.
- *admin* is the instance. An instance is optional in the case of user principals, but it is required for service principals. For example, if the user *joe* sometimes acts as a system administrator, he can use `joe/admin` to distinguish himself from his usual user identity. Likewise, if *joe* has accounts on two different hosts, he can use two principal names with different instances, for example, `joe/denver.example.com` and `joe/boston.example.com`. Notice that the Kerberos service treats *joe* and *joe/admin* as two completely different principals.

In the case of a service principal, the instance is the fully qualified host name.

```
bigmachine.eng.example.com
```
is an example of such an instance. The primary/instance for this example might be `ftp/bigmachine.eng.example.com` or `host/bigmachine.eng.example.com`. 
ENG.EXAMPLE.COM is the Kerberos realm. Realms are discussed in "Kerberos Realms" on page 339.

The following are all valid principal names:

- joe
- joe/admin
- joe/admin@ENG.EXAMPLE.COM
- nfs/host.eng.example.com@ENG.EXAMPLE.COM
- host/eng.example.com@ENG.EXAMPLE.COM

**Kerberos Realms**

A realm is a logical network, similar to a domain, that defines a group of systems under the same master KDC. Figure 19–3 shows how realms can relate to one another. Some realms are hierarchical, where one realm is a superset of the other realm. Otherwise, the realms are nonhierarchical (or "direct") and the mapping between the two realms must be defined. A feature of the Kerberos service is that it permits authentication across realms. Each realm only needs to have a principal entry for the other realm in its KDC. This Kerberos feature is called cross-realm authentication.

**Kerberos Servers**

Each realm must include a server that maintains the master copy of the principal database. This server is called the master KDC server. Additionally, each realm should contain at least one slave KDC server, which contains duplicate copies of the principal database. Both the master KDC server and the slave KDC server create tickets that are used to establish authentication.

The realm can also include a Kerberos application server. This server provides access to Kerberized services (such as ftp, telnet, ssh and NFS). If you have installed SEAM 1.0 or 1.0.1, the realm might include a Kerberos network application server, but this software was not included with these releases.
The following figure shows what a hypothetical realm might contain.

**Figure 19–4  A Typical Kerberos Realm**

EXEMPLARY.COM

- Master KDC
- Slave KDCs
- Clients
- Application Servers

**Kerberos Security Services**

In addition to providing secure authentication of users, the Kerberos service provides two security services:

- **Integrity** – Just as authentication ensures that clients on a network are who they claim to be, integrity ensures that the data they send is valid and has not been tampered with during transit. Integrity is done through cryptographic checksumming of the data. Integrity also includes user authentication.

- **Privacy** – Privacy takes security a step further. Privacy not only includes verifying the integrity of transmitted data, but it encrypts the data before transmission, protecting it from eavesdroppers. Privacy authenticates users, as well.

Developers can design their RPC-based applications to choose a security service by using the RPCSEC_GSS programming interface.
Components of Various Kerberos Releases

Components of the Kerberos service have been included in many releases. Originally, the Kerberos service and changes to the base operating system to support the Kerberos service were released using the product name “Sun Enterprise Authentication Mechanism” which was shortened to SEAM. As more components of the SEAM product were included in the Oracle Solaris software, the contents of the SEAM release decreased. Starting with the Oracle Solaris 10 release, all components of the SEAM product are included, so there is no longer a need for the SEAM product. The SEAM product name exists in the documentation for historical reasons.

The following table describes which components are included in each release. Each product release is listed in chronological order. All components are described in the following sections.

<table>
<thead>
<tr>
<th>Release Name</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEAM 1.0 in Solaris Easy Access Server 3.0</td>
<td>Full release of the Kerberos service for the Solaris 2.6 and 7 releases</td>
</tr>
<tr>
<td>The Kerberos service in the Solaris 8 release</td>
<td>Kerberos client software only</td>
</tr>
<tr>
<td>SEAM 1.0.1 in the Solaris 8 Admin Pack</td>
<td>Kerberos KDC and remote applications for the Solaris 8 release</td>
</tr>
<tr>
<td>The Kerberos service in the Solaris 9 release</td>
<td>Kerberos KDC and client software only</td>
</tr>
<tr>
<td>SEAM 1.0.2</td>
<td>Kerberos remote applications for the Solaris 9 release</td>
</tr>
<tr>
<td>The Kerberos service starting in the Oracle Solaris 10 release</td>
<td>Full release of the Kerberos service with enhancements</td>
</tr>
</tbody>
</table>

For more information about enhancements included in the Oracle Solaris 10 release, see "Kerberos Components" in *System Administration Guide: Security Services*.

Kerberos Components

Similar to the MIT distribution of the Kerberos V5 product, the Kerberos service in the Oracle Solaris release includes the following:

- Key Distribution Center (KDC):
  - Kerberos database administration daemon – kadmin.
  - Kerberos ticket processing daemon – krb5kdc.
  - Database administration programs – kadmin (master only), kadmin.local and kdb5_util.
  - Database propagation software – kprop (slave only) and kpropd.
Components of Various Kerberos Releases

- User programs for managing credentials – kinit, klist, and kdestroy.
- User program for changing your Kerberos password – kpasswd.
- Remote applications – ftp, rcp, rlogin, rsh, scp, sftp, ssh, and telnet.
- Remote application daemons – ftpd, rlogind, rshd, sshd, and telnetd.
- Keytab administration utility – ktutil.

The Generic Security Service Application Programming Interface (GSS-API) – Enables applications to use multiple security mechanisms without requiring you to recompile the application every time a new mechanism is added. The GSS-API uses standard interfaces that allow applications to be portable to many operating systems. GSS-API provides applications with the ability to include the integrity and privacy security services, as well as authentication. Both ftp and ssh use the GSS-API.

The RPCSEC_GSS Application Programming Interface (API) – Enables NFS services to use Kerberos authentication. RPCSEC_GSS is a security flavor that provides security services that are independent of the mechanisms being used. RPCSEC_GSS sits on top of the GSS-API layer. Any pluggable GSS_API-based security mechanism can be used by applications that use RPCSEC_GSS.

In addition, the Kerberos service in the Oracle Solaris release includes the following:

- A Kerberos Administration GUI-based Tool (gkadmin) – Enables you to administer the principals and principal policies. This Java technology-based GUI is an alternative to the kadmin command.
- A Kerberos V5 service module for PAM – Provides authentication, account management, session management and password management for the Kerberos service. The module can be used to make Kerberos authentication transparent to the user.
- Kernel modules – Provides kernel-based implementations of the kerberos service for use by the NFS service, which greatly improves performance.

About Kerberos in this Release

This section lists the changes that are available in the Oracle Solaris 11.1 release.

- The Kerberos software has been synchronized with the MIT 1.8 release. The following features were included:
  - The arcfour-hmac-md5-exp, des-cbc-md5, and des-cbc-crc weak encryption types are disallowed by default. The allow_weak_crypto = true declaration in the /etc/krb5/krb5.conf file can be added to allow use of the weaker encryption algorithms.
  - In the /etc/krb5/krb5.conf file, the permitted_enctypes relation can take an optional DEFAULT keyword with + or - enctyp_family to add or remove a specific encryption type from the default set.
In most cases, you can eliminate the need for the `domain_realm` mapping table on the client side, by implementing minimal referral support in the KDC and providing the mapping information to clients through that protocol. Clients can function with no `domain_realm` mapping table, by sending requests for the service principal name `service/canonical fqdn@LOCAL.REALM` to the local KDC and requesting referrals. This capability can be limited to service principal names with specific name types or in specific forms. The KDC can use only its `domain_realm` mapping table. No blocking queries to DNS can be introduced.

You can create aliases for principal entries if you are using an LDAP backend for the Kerberos database. Principal alias support is useful if a service can be accessed by different host names or if DNS is not available to canonicalize the host name, meaning that the short form is being used. You can use an alias for the various principal names a service is known by and the system only needs one set of keys for the actual service principal in its keytab file.

You can use the `kvno` utility to diagnose issues with service principal keys that are stored in `/etc/krb5/krb5.keytab`.

The `kadmin ktadd` command supports the `-norandkey` option which prevents the `kadmin` command from creating a new randomized key. The `-norandkey` option can be useful when you want to create a keytab for a principal that has a password-derived key. You can create a keytab that can be used to run the `kinit` command without having to specify a password.

Principals can be locked out after a certain number of preauthentication failures within a given time limit. See “How to Configure Account Lockout” on page 404 for more information.

The `OK_AS_DELEGATE` flag enables the KDC to communicate the local realm policy to a client regarding whether an intermediate server is trusted to accept delegated credentials. See “Trusts of Services for Delegation” on page 352 for more information.

The Kerberos service has been updated to provide the ability for a Solaris client to make updates in a multiple KDC master environment. See Example 21–15 for more information.

The `ktkt_warnd` daemon has been enhanced to register principals with existing credentials and to allow mail to be sent to users without explicitly specify each user. In addition, the service is now user configurable. See “How to Automatically Renew All Ticket-Granting Tickets (TGTs)” on page 404 for working examples.
Planning for the Kerberos Service

This chapter should be studied by administrators who are involved in the installation and maintenance of the Kerberos service. The chapter discusses several installation and configuration options that administrators must resolve before they install or configure the service.

This is a list of the topics that a system administrator or other knowledgeable support staff should study:

- “Why Plan for Kerberos Deployments?” on page 345
- “Planning Kerberos Realms” on page 346
- “Mapping Host Names Onto Realms” on page 347
- “Client and Service Principal Names” on page 347
- “Ports for the KDC and Admin Services” on page 348
- “The Number of Slave KDCs” on page 348
- “Which Database Propagation System to Use” on page 350
- “Clock Synchronization Within a Realm” on page 350
- “Client Configuration Options” on page 350
- “Improving Client Login Security” on page 351
- “KDC Configuration Options” on page 351
- “Trusts of Services for Delegation” on page 352
- “Kerberos Encryption Types” on page 352
- “Online Help URL in the Graphical Kerberos Administration Tool” on page 353

Why Plan for Kerberos Deployments?

Before you install the Kerberos service, you must resolve several configuration issues. Although changing the configuration after the initial install is not impossible, some changes can be difficult to implement. In addition, some changes require that the KDC be rebuilt, so it is better to consider long-term goals when you plan your Kerberos configuration.
Deploying a Kerberos infrastructure involves such tasks as installing KDCs, creating keys for your hosts, and migrating users. Reconfiguring a Kerberos deployment can be as hard as performing an initial deployment, so plan a deployment carefully to avoid having to re-configure.

**Planning Kerberos Realms**

A *realm* is logical network, similar to a domain, that defines a group of systems that are under the same master KDC. As with establishing a DNS domain name, issues such as the realm name, the number and size of each realm, and the relationship of a realm to other realms for cross-realm authentication should be resolved before you configure the Kerberos service.

**Realm Names**

Realm names can consist of any ASCII string. Usually, the realm name is the same as your DNS domain name, except that the realm name is in uppercase. This convention helps differentiate problems with the Kerberos service from problems with the DNS namespace, while using a name that is familiar. If you do not use DNS or you choose to use a different string, then you can use any string. However, the configuration process requires more work. The use of realm names that follow the standard Internet naming structure is wise.

**Number of Realms**

The number of realms that your installation requires depends on several factors:

- The number of clients to be supported. Too many clients in one realm makes administration more difficult and eventually requires that you split the realm. The primary factors that determine the number of clients that can be supported are as follows:
  - The amount of Kerberos traffic that each client generates
  - The bandwidth of the physical network
  - The speed of the hosts

  Because each installation will have different limitations, no rule exists for determining the maximum number of clients.

- How far apart the clients are. Setting up several small realms might make sense if the clients are in different geographic regions.

- The number of hosts that are available to be installed as KDCs. Each realm should have at least two KDC servers, one master server and one slave server.

Alignment of Kerberos realms with administrative domains is recommended. Note that a Kerberos V realm can span multiple sub-domains of the DNS domain to which the realm corresponds.
Realm Hierarchy

When you are configuring multiple realms for cross-realm authentication, you need to decide how to tie the realms together. You can establish a hierarchical relationship among the realms, which provides automatic paths to the related domains. Of course, all realms in the hierarchical chain must be configured properly. The automatic paths can ease the administration burden. However, if there are many levels of domains, you might not want to use the default path because it requires too many transactions.

You can also choose to establish the trust relationship directly. A direct trust relationship is most useful when too many levels exist between two hierarchical realms or when no hierarchical relationship exists. The connection must be defined in the /etc/krb5/krb5.conf file on all hosts that use the connection. So, some additional work is required. The direct trust relationship is also referred to as a transitive relationship. For an introduction, see “Kerberos Realms” on page 339. For the configuration procedures for multiple realms, see “Configuring Cross-Realm Authentication” on page 376.

Mapping Host Names Onto Realms

The mapping of host names onto realm names is defined in the domain_realm section of the krb5.conf file. These mappings can be defined for a whole domain and for individual hosts, depending on the requirements.

DNS can also be used to look up information about the KDCs. Using DNS makes it easier to change the information because you will not need to edit the krb5.conf file on all of the clients each time you make a change. See the krb5.conf(4) man page for more information.

Solaris Kerberos clients can interoperate better with Active Directory servers. The Active Directory servers can be configured to provide the realm to host mapping.

Client and Service Principal Names

When you are using the Kerberos service, DNS must be enabled on all hosts. With DNS, the principal should contain the Fully Qualified Domain Name (FQDN) of each host. For example, if the host name is boston, the DNS domain name is example.com, and the realm name is EXAMPLE.COM, then the principal name for the host should be host/boston.example.com@EXAMPLE.COM. The examples in this book require that DNS is configured and use the FQDN for each host.

The Kerberos service canonicalizes host alias names through DNS, and uses the canonicalized form (cname) when constructing the service principal for the associated service. Therefore when creating a service principal, the host name component of service principal names should be the canonical form of the host name of the system hosting the service.
The following is an example of how the Kerberos service canonicalizes hostname. If a user runs the command "ssh alpha.example.com" where alpha.example.com is a DNS host alias for the
cname beta.example.com. When ssh calls Kerberos and requests a host service ticket for
alpha.example.com, the Kerberos service canonicalizes alpha.example.com to
beta.example.com and requests a ticket for the service principal “host/beta.example.com”
from the KDC.

For the principal names that include the FQDN of a host, it is important to match the string that
describes the DNS domain name in the /etc/resolv.conf file. The Kerberos service requires
that the DNS domain name be in lowercase letters when you are specifying the FQDN for a
principal. The DNS domain name can include uppercase and lowercase letters, but only use
lowercase letters when you are creating a host principal. For example, it doesn’t matter if the
DNS domain name is example.com, Example.COM, or any other variation. The principal name
for the host would still be host/boston.example.com@EXAMPLE.COM.

In addition, the Service Management Facility has been configured so that many of the daemons
or commands do not start if the DNS client service is not running. The kdb5_util, kadminl,
and kpropd daemons, as well as the kprop command all are configured to depend on the DNS
service. To fully utilize the features available using the Kerberos service and SMF, you must
enable the DNS client service on all hosts.

**Ports for the KDC and Admin Services**

By default, port 88 and port 750 are used for the KDC, and port 749 is used for the KDC
administration daemon. Different port numbers can be used. However, if you change the port
numbers, then the /etc/services and /etc/krb5/krb5.conf files must be changed on every
client. In addition to these files, the /etc/krb5/kdc.conf file on each KDC must be updated.

**The Number of Slave KDCs**

Slave KDCs generate credentials for clients just as the master KDC does. Slave KDCs provide
backup if the master becomes unavailable. Each realm should have at least one slave KDC.
Additional slave KDCs might be required, depending on these factors:

- The number of physical segments in the realm. Normally, the network should be set up so
  that each segment can function, at least minimally, without the rest of the realm. To do so, a
  KDC must be accessible from each segment. The KDC in this instance could be either a
  master or a slave.

- The number of clients in the realm. By adding more slave KDC servers, you can reduce the
  load on the current servers.

It is possible to add too many slave KDCs. Remember that the KDC database must be
propagated to each server, so the more KDC servers that are installed, the longer it can take to
get the data updated throughout the realm. Also, because each slave retains a copy of the KDC database, more slaves increase the risk of a security breach.

In addition, one or more slave KDCs can easily be configured to be swapped with the master KDC. The advantage of configuring at least one slave KDC in this way is that if the master KDC fails for any reason, you will have a system preconfigured that will be easy to swap as the master KDC. For instructions on how to configure a swappable slave KDC, see “Swapping a Master KDC and a Slave KDC” on page 407.

**Mapping GSS Credentials to UNIX Credentials**

The Kerberos service provides a default mapping of GSS credential names to UNIX user IDs (UIDs) for GSS applications that require this mapping, such as NFS. GSS credential names are equivalent to Kerberos principal names when using the Kerberos service. The default mapping algorithm is to take a one component Kerberos principal name and use that component, which is the primary name of the principal, to look up the UID. The look up occurs in the default realm or any realm that is allowed by using the auth_to_local_realm parameter in /etc/krb5/krb5.conf. For example, the user principal name bob@EXAMPLE.COM is mapped to the UID of the UNIX user named bob using the password table. The user principal name bob/admin@EXAMPLE.COM would not be mapped, because the principal name includes an instance component of admin. If the default mappings for the user credentials are sufficient, the GSS credential table does not need to be populated. In past releases, populating the GSS credential table was required to get the NFS service to work. If the default mapping is not sufficient, for example if you want to map a principal name which contains an instance component, then other methods should be used. For more information see:

- “How to Create a Credential Table” on page 383
- “How to Add a Single Entry to the Credential Table” on page 384
- “How to Provide Credential Mapping Between Realms” on page 384
- "Observing Mapping From GSS Credentials to UNIX Credentials” on page 450

**Automatic User Migration to a Kerberos Realm**

UNIX users who do not have valid user accounts in the default Kerberos realm can be automatically migrated using the PAM framework. Specifically, the pam_krb5_migrate module would be used in the authentication stack of the PAM service. Services would be setup up so that whenever a user, who does not have a Kerberos principal, performs a successful log in to a system using their password, a Kerberos principal would be automatically created for that user. The new principal password would be the same as the UNIX password. See “How to Configure Automatic Migration of Users in a Kerberos Realm” on page 401 for instructions on how to use the pam_krb5_migrate module.
Which Database Propagation System to Use

The database that is stored on the master KDC must be regularly propagated to the slave KDCs. You can configure the propagation of the database to be incremental. The incremental process propagates only updated information to the slave KDCs, rather than the entire database. For more information about database propagation, see "Administering the Kerberos Database" on page 411.

If you do not use incremental propagation, one of the first issues to resolve is how often to update the slave KDCs. The need to have up-to-date information that is available to all clients must be weighed against the amount of time it takes to complete the update.

In large installations with many KDCs in one realm, one or more slaves can propagate the data so that the process is done in parallel. This strategy reduces the amount of time that the update takes, but it also increases the level of complexity in administering the realm. For a complete description of this strategy, see "Setting Up Parallel Propagation" on page 423.

Clock Synchronization Within a Realm

All hosts that participate in the Kerberos authentication system must have their internal clocks synchronized within a specified maximum amount of time. Known as clock skew, this feature provides another Kerberos security check. If the clock skew is exceeded between any of the participating hosts, requests are rejected.

One way to synchronize all the clocks is to use the Network Time Protocol (NTP) software. See "Synchronizing Clocks Between KDCs and Kerberos Clients" on page 405 for more information. Other ways of synchronizing the clocks are available, so the use of NTP is not required. However, some form of synchronization should be used to prevent access failures because of clock skew.

Client Configuration Options

The \texttt{kclient} configuration utility can be run in interactive mode or noninteractive mode. In interactive mode, the user is prompted for Kerberos-specific parameter values, which allows the user to make changes to the existing installation when configuring the client. In noninteractive mode, a file with previously set parameter values is used. Also, command-line options can be used in the noninteractive mode. Both interactive and noninteractive modes require less steps than the manual process, which should make the process quicker and less prone to error.
Changes have been made to allow for a zero-configuration Kerberos client. If these rules are followed in your environment then no explicit configuration procedure is necessary for a Solaris Kerberos client:

- DNS is configured to return SRV records for KDCs.
- The realm name matches the DNS domain name or the KDC supports referrals.
- The Kerberos client does not require a keytab file.

In some cases it may be better to explicitly configure the Kerberos client:

- If referrals are not used, the zero-configuration logic depends on the DNS domain name of the host to determine the realm. This introduces a small security risk, but the risk is much smaller than enabling `dns_lookup_realm`.
- The `pam_krb5` module relies on a host key entry in the keytab. This requirement may be disabled in the `krb5.conf` file however it is not recommend for security reasons. See the `krb5.conf(4)` man page.
- The zero-configuration process is less efficient than direct configuration, and has a greater reliance on DNS. The process performs more DNS lookups than a directly configured client.

See “Configuring Kerberos Clients” on page 387 for a description of all the client configuration processes.

## Improving Client Login Security

On login, a client, using the `pam_krb5` module, verifies that the KDC that issued the latest TGT, is the same KDC that issued the client host principal that is stored in `/etc/krb5/krb5.keytab`. The `pam_krb5` module verifies the KDC when the module is configured in the authentication stack. For some configurations, like DHCP clients that do not store a client host principal, this check needs to be disabled. To turn off this check, you must set the `verify_ap_req_nofail` option in the `krb5.conf` file to be false. See “How to Disable Verification of the Ticket-Granting Ticket” on page 399 for more information.

## KDC Configuration Options

There are several ways to configure a KDC. The simplest ways use the `kdcsmgr` utility to configure the KDC automatically or interactively. The automatic version requires that you use command line options to define the configuration parameters. This method is especially useful for scripts. The interactive version prompts you for all information that is needed. See Table 21–1 for pointers to the instructions for using this command.

Also available is support for using LDAP to manage the database files for Kerberos. See “How to Configure a KDC to Use an LDAP Data Server” on page 363 for instructions. Using LDAP simplifies administration for sites that require better coordination between the Kerberos databases and their existing directory server setup.
For some applications, a client might need to delegate authority to a server to act on its behalf in contacting other services. The client must forward credentials to an intermediate server. The client’s ability to obtain a service ticket to a server conveys no information to the client about whether the server should be trusted to accept delegated credentials. The \texttt{ok\_to\_auth\_as\_delegate} option to the \texttt{kadmin} command provides a way for a KDC to communicate the local realm policy to a client regarding whether an intermediate server is trusted to accept such credentials.

The copy of the credential ticket flags in the encrypted part of the KDC reply might have the \texttt{ok\_to\_auth\_as\_delegate} option set to indicate to the client that the server specified in the ticket has been determined by the policy of the realm to be a suitable recipient of delegation. A client can use the presence of this information to determine whether to delegate credentials (by granting either a proxy or a forwarded TGT) to this server. When setting this option, an administrator must consider the security and placement of the server on which the service runs, as well as whether the service requires the use of delegated credentials.

**Kerberos Encryption Types**

An encryption type is an identifier that specifies the encryption algorithm, encryption mode, and hash algorithms used in the Kerberos service. The keys in the Kerberos service have an associated encryption type to identify the cryptographic algorithm and mode to be used when the service performs cryptographic operations with the key. Here are the supported encryption types:

- \texttt{des-cbc-md5}
- \texttt{des-cbc-crc}
- \texttt{des3-cbc-sha1-kd}
- \texttt{arcfour-hmac-md5}
- \texttt{arcfour-hmac-md5-exp}
- \texttt{aes128-cts-hmac-sha1-96}
- \texttt{aes256-cts-hmac-sha1-96}

**Note** – In releases prior to Solaris 10 8/07 release, the \texttt{aes256-cts-hmac-sha1-96} encryption type can be used with the Kerberos service if the unbundled Strong Cryptographic packages are installed.

If you want to change the encryption type, you should do so when creating a new principal database. Because of the interaction between the KDC, the server, and the client, changing the encryption type on an existing database is difficult. Leave these parameters unset unless you are re-creating the database. Refer to “Using Kerberos Encryption Types” on page 519 for more information.
Note – If you have a master KDC installed that is not running the Solaris 10 release, the slave KDCs must be upgraded to the Solaris 10 release before you upgrade the master KDC. A Solaris 10 master KDC will use the new encryption types, which an older slave will not be able to handle.

The arcfour-hmac-md5-exp, des-cbc-md5, and des-cbc-crc weak encryption types are disallowed by default in the Oracle Solaris 11 release. If you need to continue using these encryption types, especially for telnet, then set allow_weak_crypto = true in the libdefaults section of the /etc/krb5/krb5.conf file.

**Online Help URL in the Graphical Kerberos Administration Tool**

The online help URL is used by the Graphical Kerberos Administration Tool, gkadmin, so the URL should be defined properly to enable the “Help Contents” menu to work. The HTML version of this manual can be installed on any appropriate server. Alternately, you can decide to use the collections at [http://www.oracle.com/technetwork/indexes/documentation/index.html](http://www.oracle.com/technetwork/indexes/documentation/index.html).

The URL is specified in the krb5.conf file when configuring a host to use the Kerberos service. The URL should point to the section titled “SEAM Tool on page 454 in the Administering Kerberos Principals and Policies (Tasks) chapter in this book. You can choose another HTML page, if another location is more appropriate.
This chapter provides configuration procedures for KDC servers, network application servers, NFS servers, and Kerberos clients. Many of these procedures require superuser access, so they should be used by system administrators or advanced users. Cross-realm configuration procedures and other topics related to KDC servers are also covered.

The following topics are covered.

- “Configuring the Kerberos Service (Task Map)” on page 355
- “Configuring KDC Servers” on page 356
- “Configuring Kerberos Clients” on page 387
- “Configuring Cross-Realm Authentication” on page 376
- “Configuring Kerberos Network Application Servers” on page 378
- “Configuring Kerberos NFS Servers” on page 381
- “Synchronizing Clocks Between KDCs and Kerberos Clients” on page 405
- “Swapping a Master KDC and a Slave KDC” on page 407
- “Administering the Kerberos Database” on page 411
- “Increasing Security on Kerberos Servers” on page 429

Configuring the Kerberos Service (Task Map)

Parts of the configuration process depend on other parts and must be done in a specific order. These procedures often establish services that are required to use the Kerberos service. Other procedures are not dependent on any order, and can be done when appropriate. The following task map shows a suggested order for a Kerberos installation.

<table>
<thead>
<tr>
<th>Task</th>
<th>Description</th>
<th>For Instructions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Plan for your Kerberos installation.</td>
<td>Lets you resolve configuration issues before you start the software configuration process. Planning ahead saves you time and other resources in the long run.</td>
<td>Chapter 20, &quot;Planning for the Kerberos Service&quot;</td>
</tr>
</tbody>
</table>
Configuring Additional Kerberos Services (Task Map)

<table>
<thead>
<tr>
<th>Task</th>
<th>Description</th>
<th>For Instructions</th>
</tr>
</thead>
<tbody>
<tr>
<td>2. (Optional) Install NTP.</td>
<td>Configures the Network Time Protocol (NTP) software, or another clock synchronization protocol. In order for the Kerberos service to work properly, the clocks on all systems in the realm must be synchronized.</td>
<td>“Synchronizing Clocks Between KDCs and Kerberos Clients” on page 405</td>
</tr>
<tr>
<td>3. Configure the KDC servers.</td>
<td>Configures and builds the master KDC and the slave KDC servers and the KDC database for a realm.</td>
<td>“Configuring KDC Servers” on page 356</td>
</tr>
<tr>
<td>5. (Optional) Configure swappable KDC servers.</td>
<td>Makes the task of swapping the master KDC and a slave KDC easier.</td>
<td>“How to Configure a Swappable Slave KDC” on page 407</td>
</tr>
</tbody>
</table>

Configuring Additional Kerberos Services (Task Map)

Once the required steps have been completed, the following procedures can be used, when appropriate.

<table>
<thead>
<tr>
<th>Task</th>
<th>Description</th>
<th>For Instructions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configure cross-realm authentication.</td>
<td>Enables communications from one realm to another realm.</td>
<td>“Configuring Cross-Realm Authentication” on page 376</td>
</tr>
<tr>
<td>Configure Kerberos application servers.</td>
<td>Enables a server to support services such as ftp, telnet, and rsh using Kerberos authentication.</td>
<td>“Configuring Kerberos Network Application Servers” on page 378</td>
</tr>
<tr>
<td>Configure Kerberos clients.</td>
<td>Enables a client to use Kerberos services.</td>
<td>“Configuring Kerberos Clients” on page 387</td>
</tr>
<tr>
<td>Configure Kerberos NFS server.</td>
<td>Enables a server to share a file system that requires Kerberos authentication.</td>
<td>“Configuring Kerberos NFS Servers” on page 381</td>
</tr>
</tbody>
</table>

Configuring KDC Servers

After you install the Kerberos software, you must configure the KDC servers. Configuring a master KDC and at least one slave KDC provides the service that issues credentials. These credentials are the basis for the Kerberos service, so the KDCs must be installed before you attempt other tasks.

The most significant difference between a master KDC and a slave KDC is that only the master KDC can handle database administration requests. For instance, changing a password or adding a new principal must be done on the master KDC. These changes can then be propagated to the slave KDCs. Both the slave KDC and master KDC generate credentials. This feature provides redundancy in case the master KDC cannot respond.
TABLE 21–1 Configuring KDC Servers (Task Map)

<table>
<thead>
<tr>
<th>Task</th>
<th>Description</th>
<th>For Instructions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configure a master KDC.</td>
<td>Configures and builds the master KDC server and database for a realm by using an automatic process, which is good for scripts.</td>
<td>&quot;How to Automatically Configure a Master KDC“ on page 357</td>
</tr>
<tr>
<td></td>
<td>Configures and builds the master KDC server and database for a realm using an interactive process, which is sufficient for most installations</td>
<td>&quot;How to Interactively Configure a Master KDC“ on page 358</td>
</tr>
<tr>
<td></td>
<td>Configures and builds the master KDC server and database for a realm using a manual process, which is needed for more complex installations</td>
<td>&quot;How to Manually Configure a Master KDC“ on page 359</td>
</tr>
<tr>
<td></td>
<td>Configures and builds the master KDC server and database for a realm using a manual process and using LDAP for the KDC</td>
<td>&quot;How to Configure a KDC to Use an LDAP Data Server“ on page 363</td>
</tr>
<tr>
<td>Configure a slave KDC server.</td>
<td>Configures and builds a slave KDC server for a realm using an automatic process, which is good for scripts</td>
<td>&quot;How to Automatically Configure a Slave KDC“ on page 369</td>
</tr>
<tr>
<td></td>
<td>Configures and builds a slave KDC server for a realm using an interactive process, which is sufficient for most installations</td>
<td>&quot;How to Interactively Configure a Slave KDC“ on page 370</td>
</tr>
<tr>
<td></td>
<td>Configures and builds a slave KDC server for a realm using a manual process, which is needed for more complex installations</td>
<td>&quot;How to Manually Configure a Slave KDC“ on page 371</td>
</tr>
<tr>
<td>Refresh principal keys on a KDC server.</td>
<td>Updates the session key on a KDC server to use new encryption types.</td>
<td>&quot;How to Refresh the Ticket-Granting Service Keys on a Master Server“ on page 375</td>
</tr>
</tbody>
</table>

▼ How to Automatically Configure a Master KDC

In the Oracle Solaris 11 release, a master KDC can be automatically configured by using the following procedure.

**Before You Begin** You must assume the root role. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

- **Create the KDC.**
  
  Run the `kdcmgr` utility to create the KDC. You need to provide both the master key password and the password for the administrative principal.

  ```shell
  kdc1# kdcmgr -a kws/admin -r EXAMPLE.COM create master
  Starting server setup
  ```
Setting up /etc/krb5/kdc.conf
Setting up /etc/krb5/krb5.conf

Initializing database '/var/krb5/principal' for realm 'EXAMPLE.COM',
master key name 'K/M@EXAMPLE.COM'
You will be prompted for the database Master Password.
It is important that you NOT FORGET this password.
Enter KDC database master key: <Type the password>
Re-enter KDC database master key to verify: <Type it again>

Authenticating as principal root/admin@EXAMPLE.COM with password.
WARNING: no policy specified for kws/admin@EXAMPLE.COM; defaulting to no policy
Enter password for principal "kws/admin@EXAMPLE.COM": <Type the password>
Re-enter password for principal "kws/admin@EXAMPLE.COM": <Type it again>
Principal "kws/admin@EXAMPLE.COM" created.

Setting up /etc/krb5/kadm5.acl.

---------------------------------------------------
Setup COMPLETE.
kdc1#

How to Interactively Configure a Master KDC

In the Oracle Solaris release, a master KDC can be interactively configured by using the following procedure.

Before You Begin
You must assume the root role. For more information, see "How to Use Your Assigned Administrative Rights" on page 157.

- Create the KDC.
  Run the kdcmsgr utility to create the KDC. You need to provide both the master key password and the password for the administrative principal.
  kdc1# kdcmsgr create master
  Starting server setup
  ---------------------------------------
  Enter the Kerberos realm: EXAMPLE.COM
  Setting up /etc/krb5/kdc.conf
  Setting up /etc/krb5/krb5.conf
  Initializing database '/var/krb5/principal' for realm 'EXAMPLE.COM',
master key name ‘K/M@EXAMPLE.COM’
You will be prompted for the database Master Password. It is important that you NOT FORGET this password.
Enter KDC database master key: <Type the password>
Re-enter KDC database master key to verify: <Type it again>

Enter the krb5 administrative principal to be created: kws/admin

Authenticating as principal root/admin@EXAMPLE.COM with password.
WARNING: no policy specified for kws/admin@EXAMPLE.COM; defaulting to no policy
Enter password for principal "kws/admin@EXAMPLE.COM": <Type the password>
Re-enter password for principal "kws/admin@EXAMPLE.COM": <Type it again>
Principal "kws/admin@EXAMPLE.COM" created.

Setting up /etc/krb5/kadm5.acl.
---------------------------------------------------
Setup COMPLETE.
---------------------------------------------------
kdc1#

Example 21–1 Displaying the Status of a KDC Server
The kdcsmgr status command can be used to display information about either a master or a slave KDC server.

▼ How to Manually Configure a Master KDC
In this procedure, incremental propagation is configured. In addition, the following configuration parameters are used:

■ Realm name = EXAMPLE.COM
■ DNS domain name = example.com
■ Master KDC = kdc1.example.com
■ admin principal = kws/admin
■ Online help URL =

Note – Adjust the URL to point to the section, as described in “Online Help URL in the Graphical Kerberos Administration Tool” on page 353.

Before You Begin This procedure requires that the host is configured to use DNS. For specific naming instructions if this master is to be swappable, see “Swapping a Master KDC and a Slave KDC” on page 407.
You must assume the root role. For more information, see “How to Use YourAssigned Administrative Rights” on page 157.

1 **Edit the Kerberos configuration file (krb5.conf).**

You need to change the realm names and the names of the servers. See the *krb5.conf*(4) man page for a full description of this file.

```
kdc1 # cat /etc/krb5/krb5.conf
[libdefaults]
    default_realm = EXAMPLE.COM

[realms]
EXAMPLE.COM = {
    kdc = kdc1.example.com
    admin_server = kdc1.example.com
}

[domain_realm]
.example.com = EXAMPLE.COM
#
# if the domain name and realm name are equivalent,
# this entry is not needed
#
[logging]
default = FILE:/var/krb5/kdc.log
kdc = FILE:/var/krb5/kdc.log

[appdefaults]
gkadmin = {
}
```

In this example, the lines for default_realm, kdc, admin_server, and all domain_realm entries were changed. In addition, the line that defines the help_url was edited.

---

**Note** – If you want to restrict the encryption types, you can set the default_tkt_enctypes or default_tgs_enctypes lines. Refer to “Using Kerberos Encryption Types” on page 519 for a description of the issues involved with restricting the encryption types.

2 **Edit the KDC configuration file (kdc.conf).**

You need to change the realm name. See the *kdc.conf*(4) man page for a full description of this file.

```
kdc1 # cat /etc/krb5/kdc.conf
[kdcdefaults]
    kdc_ports = 88,750

[realms]
EXAMPLE.COM = {
    profile = /etc/krb5/krb5.conf
    database_name = /var/krb5/principal
    acl_file = /etc/krb5/kadm5.acl
    kadmind_port = 749
```

`
In this example, the realm name definition in the realms section was changed. Also, in the realms section, lines to enable incremental propagation and to select the number of updates the KDC master keeps in the log were added.

**Note** - If you want to restrict the encryption types, you can set the `permitted_enctypes`, `supported_enctypes`, or `master_key_type` lines. Refer to “Using Kerberos Encryption Types” on page 519 for a description of the issues involved with restricting the encryption types.

3 **Create the KDC database by using the kdb5_util command.**

The `kdb5_util` command creates the KDC database. Also, when used with the `-s` option, this command creates a stash file that is used to authenticate the KDC to itself before the `kadmind` and `krb5kdc` daemons are started.

```
kdc1 # /usr/sbin/kdb5_util create -s
Initializing database '/var/krb5/principal' for realm 'EXAMPLE.COM'
master key name 'K/M@EXAMPLE.COM'
You will be prompted for the database Master Password.
It is important that you NOT FORGET this password.
Enter KDC database master key:  
Re-enter KDC database master key to verify: 
```

4 **Edit the Kerberos access control list file (kadm5.acl).**

Once populated, the `/etc/krb5/kadm5.acl` file should contain all principal names that are allowed to administer the KDC.

```
kws/admin@EXAMPLE.COM *
```

The entry gives the `kws/admin` principal in the EXAMPLE.COM realm the ability to modify principals or policies in the KDC. The default installation includes an asterisk (*) to match all `admin` principals. This default could be a security risk, so it is more secure to include a list of all of the `admin` principals. See the `kadm5.acl(4)` man page for more information.

5 **Add administration principals to the database.**

You can add as many `admin` principals as you need. You must add at least one `admin` principal to complete the KDC configuration process. For this example, a `kws/admin` principal is added.

You can substitute an appropriate principal name instead of "kws."

```
kadmin.local: addprinc kws/admin
Enter password for principal kws/admin@EXAMPLE.COM:  
Re-enter password for principal kws/admin@EXAMPLE.COM:  
Principal "kws/admin@EXAMPLE.COM" created.
kadmin.local: 
```
6  **Start the Kerberos daemons.**

   kdc1 # svcadm enable -r network/security/krb5kdc
   kdc1 # svcadm enable -r network/security/kadmin

7  **Start kadmin and add more principals.**

   At this point, you can add principals by using the Graphical Kerberos Administration Tool. To do so, you must log in with one of the admin principal names that you created earlier in this procedure. However, the following command-line example is shown for simplicity.

   kdc1 # /usr/sbin/kadmin -p kws/admin
   Enter password:  <Type kws/admin password>
   kadmin:

   a. **Create the master KDC host principal.**

      The host principal is used by Kerberized applications, such as kprop to propagate changes to the slave KDCs. This principal is also used to provide secure remote access to the KDC server using applications, like ssh. Note that when the principal instance is a host name, the FQDN must be specified in lowercase letters, regardless of the case of the domain name in the name service.

      kadmin:  addprinc -randkey host/kdc1.example.com
      Principal "host/kdc1.example.com@EXAMPLE.COM" created.
      kadmin:

   b. **(Optional) Create the kclient principal.**

      This principal is used by the kclient utility during the installation of a Kerberos client. If you do not plan on using this utility, then you do not need to add the principal. The users of the kclient utility need to use this password.

      kadmin:  addprinc clntconfig/admin
      Enter password for principal clntconfig/admin@EXAMPLE.COM:  <Type the password>
      Re-enter password for principal clntconfig/admin@EXAMPLE.COM:  <Type it again>
      Principal "clntconfig/admin@EXAMPLE.COM" created.
      kadmin:

   c. **Add the master KDC's host principal to the master KDC's keytab file.**

      Adding the host principal to the keytab file allows this principal to be used by application servers, like ssh, automatically.

      kadmin:  ktadd host/kdc1.example.com
      Entry for principal host/kdc1.example.com with kvno 3, encryption type AES-256 CTS mode with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5/krb5.keytab.
      Entry for principal host/kdc1.example.com with kvno 3, encryption type AES-128 CTS mode with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5/krb5.keytab.
      Entry for principal host/kdc1.example.com with kvno 3, encryption type Triple DES cbc mode with HMAC/sha1 added to keytab WRFILE:/etc/krb5/krb5.keytab.
      Entry for principal host/kdc1.example.com with kvno 3, encryption type ArcFour with HMAC/md5 added to keytab WRFILE:/etc/krb5/krb5.keytab.
      Entry for principal host/kdc1.example.com with kvno 3, encryption type DES cbc mode with RSA-MDS added to keytab WRFILE:/etc/krb5/krb5.keytab.
      kadmin:
d. Quit kadmin.

```
kadmin: quit
```

8 **(Optional) Synchronize the master KDCs clock by using NTP or another clock synchronization mechanism.**

Installing and using the Network Time Protocol (NTP) is not required. However, every clock must be within the default time that is defined in the `libdefaults` section of the `krb5.conf` file for authentication to succeed. See “Synchronizing Clocks Between KDCs and Kerberos Clients” on page 405 for information about NTP.

9 **Configure Slave KDCs.**

To provide redundancy, make sure to install at least one slave KDC. See “How to Manually Configure a Slave KDC” on page 371 for specific instructions.

---

**How to Configure a KDC to Use an LDAP Data Server**

Use the following procedure to configure a KDC to use an LDAP data server.

In this procedure, the following configuration parameters are used:

- Realm name = `EXAMPLE.COM`
- DNS domain name = `example.com`
- Master KDC = `kdc1.example.com`
- Directory Server = `dsserver.example.com`
- `admin` principal = `kws/admin`
- FMRI for the LDAP service = `svc:/application/sun/ds:-var-opt-SUNWdsee:dsins1`
- Online help URL = 
  
  ```
  ```

*Note* – Adjust the URL to point to the section, as described in “Online Help URL in the Graphical Kerberos Administration Tool” on page 353.

---

**Before You Begin**

This procedure also requires that the host is configured to use DNS. For better performance, install the KDC and the LDAP Directory Service on the same server. In addition, a directory server should be running. The following procedure works with servers using the Sun Directory Server Enterprise Edition 7.0 release.

You must assume the root role on the KDC server. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.
Configure the master KDC to use SSL to reach the directory server.

The following steps configure an release KDC to use the Directory Server self-signed certificate. If the certificate has expired, follow the instructions for renewing a certificate in "To Manage Self-Signed Certificates".

a. On the directory server, export the self-signed directory server certificate.


b. On the master KDC, import the directory server certificate.

   # pktool setpin keystore=nss dir=/var/ldap
   # chmod a+r /var/ldap/*.db
   # pktool import keystore=nss objtype=cert trust="CT" infile=/tmp/defaultCert.certutil.der \
   label=defaultCert dir=/var/ldap

c. On the master KDC, test that SSL is working.

   This example assumes that the cn=directory manager entry has administration privileges.

   /usr/bin/ldapsearch -Z -P /var/ldap -D "cn=directory manager" \
   -h dsserver.example.com -b "" -s base objectclass='*

   Subject:
   "CN=dsserver.example.com,CN=636,CN=Directory Server,O=Example Corporation"

   Note that the CN=dsserver.example.com entry should include the fully qualified host name, not a short version.

2 PopulatetheLDAP directory, if necessary.

3 Add the Kerberos schema to the existing schema.

   # ldapmodify -h dsserver.example.com -D "cn=directory manager" -f /usr/share/lib/ldif/kerberos.ldif

4 Create the Kerberos container in the LDAP directory.

   Add the following entries to the krb5.conf file.

   a. Define the database type.

      Add an entry to define the database_module to the realms section.

      database_module = LDAP

   b. Define the database module.

      [dbmodules]
      LDAP = {
        ldap_kerberos_container_dn = "cn=krbcontainer,dc=example,dc=com"
        db_library = kldap
        ldap_kdc_dn = "cn=kdc service,ou=profile,dc=example,dc=com"
        ldap_kadmind_dn = "cn=kadmin service,ou=profile,dc=example,dc=com"
        ldap_cert_path = /var/ldap
        ldap_servers = ldaps://dsserver.example.com
      }

Oracle Solaris 11.1 Administration: Security Services • February 2014
c. **Create the KDC in the LDAP directory.**
   
   This command creates krbcontainer and several other objects. It also creates a 
   `/var/krb5/.k5.EXAMPLE.COM` master key stash file.
   
   ```bash
   # kdb5_ldap_util -D "cn=directory manager" create -P abcd1234 -r EXAMPLE.COM -s
   ```

5 **Stash the KDC bind Distinguished Name (DN) passwords.**

   These passwords are used by the KDC when it binds to the DS. The KDC uses different roles 
   depending on the type of access the KDC is using.

   ```bash
   # kdb5_ldap_util stashsrvpw "cn=kdc service,ou=profile,dc=example,dc=com"
   # kdb5_ldap_util stashsrvpw "cn=kadmin service,ou=profile,dc=example,dc=com"
   ```

6 **Add KDC service roles.**

   a. **Create a `kdc_roles.ldif` file with contents like this:**

   ```ldif
   dn: cn=kdc service,ou=profile,dc=example,dc=com
   cn: kdc service
   sn: kdc service
   objectclass: top
   objectclass: person
   userpassword: test123
   
   dn: cn=kadmin service,ou=profile,dc=example,dc=com
   cn: kadmin service
   sn: kadmin service
   objectclass: top
   objectclass: person
   userpassword: test123
   ```

   b. **Create the role entries in the LDAP directory**

   ```bash
   # ldapmodify -a -h dserver.example.com -D "cn=directory manager" -f kdc_roles.ldif
   ```

7 **Set the ACLs for the KDC-related roles.**

   ```bash
   # cat << EOF | ldapmodify -h dserver.example.com -D "cn=directory manager"
   
   # Set kadmin ACL for everything under krbcontainer.
   dn: cn=krbcontainer,dc=example,dc=com
   changetype: modify
   add: acl
   acl: {target="ldap://cn=krbcontainer,dc=example,dc=com")(targetattr="krb*")(version 3.0;
   acl kadmin_ACL; allow (all);)
   userdn = "ldap://cn=kadmin service,ou=profile,dc=example,dc=com";
   }
   
   # Set kadmin ACL for everything under the people subtree if there are
   # mix-in entries for krb princs:
   dn: ou=people,dc=example,dc=com
   changetype: modify
   add: acl
   acl: {target="ldap://ou=people,dc=example,dc=com")(targetattr="krb*")(version 3.0;
   acl kadmin_ACL; allow (all);)
   userdn = "ldap://cn=kadmin service,ou=profile,dc=example,dc=com";
   }
   
   EOF
   ```
8 Edit the Kerberos configuration file (`krb5.conf`).

You need to change the realm names and the names of the servers. See the `krb5.conf(4)` man page for a full description of this file.

```bash
kdc1 # cat /etc/krb5/krb5.conf
[libdefaults]
  default_realm = EXAMPLE.COM

[realms]
  EXAMPLE.COM = {
    kdc = kdc1.example.com
    admin_server = kdc1.example.com
  }

[domain_realm]
  .example.com = EXAMPLE.COM
  # if the domain name and realm name are equivalent,
  # this entry is not needed
  #
  [logging]
    default = FILE:/var/krb5/kdc.log
    kdc = FILE:/var/krb5/kdc.log

[appdefaults]
  gkadmin = {
  }
```

In this example, the lines for `default_realm`, `kdc`, `admin_server`, and all `domain_realm` entries were changed. In addition, the line that defines the `help_url` was edited.

**Note** – If you want to restrict the encryption types, you can set the `default_tkt_enctypes` or `default_tgs_enctypes` lines. Refer to “Using Kerberos Encryption Types” on page 519 for a description of the issues involved with restricting the encryption types.

9 Edit the KDC configuration file (`kdc.conf`).

You need to change the realm name. See the `kdc.conf(4)` man page for a full description of this file.

```bash
kdc1 # cat /etc/krb5/kdc.conf
[kdcdesigns]
  kdc_ports = 88,750

[realms]
  EXAMPLE.COM = {
    profile = /etc/krb5/krb5.conf
    database_name = /var/krb5/principal
    acl_file = /etc/krb5/kadm5.acl
    kadmind_port = 749
    max_life = 8h 0m 0s
    max_renewable_life = 7d 0h 0m 0s
    sunw_dbprop_enable = true
    sunw_dbprop_master_uologsize = 1000
  }
```

Oracle Solaris 11.1 Administration: Security Services • February 2014

366
In this example, the realm name definition in the `realms` section was changed. Also, in the `realms` section, lines to enable incremental propagation and to select the number of updates the KDC master keeps in the log were added.

**Note** – If you want to restrict the encryption types, you can set the `permitted_enctypes`, `supported_enctypes`, or `master_key_type` lines. Refer to “Using Kerberos Encryption Types” on page 519 for a description of the issues involved with restricting the encryption types.

10 **Edit the Kerberos access control list file (`kadm5.acl`).**

Once populated, the `/etc/krb5/kadm5.acl` file should contain all principal names that are allowed to administer the KDC.

```
kws/admin@EXAMPLE.COM *
```

The entry gives the `kws/admin` principal in the EXAMPLE.COM realm the ability to modify principals or policies in the KDC. The default installation includes an asterisk (*) to match all `admin` principals. This default could be a security risk, so it is more secure to include a list of all of the `admin` principals. See the `kadm5.acl(4)` man page for more information.

11 **Start the `kadmin.local` command and add principals.**

The next substeps create principals that are used by the Kerberos service.

```
kdc1 # /usr/sbin/kadmin.local
kadmin.local:
```

**a. Add administration principals to the database.**

You can add as many `admin` principals as you need. You must add at least one `admin` principal to complete the KDC configuration process. For this example, a `kws/admin` principal is added. You can substitute an appropriate principal name instead of “kws.”

```
kadmin.local: addprinc kws/admin
Enter password for principal kws/admin@EXAMPLE.COM: <Type the password>
Re-enter password for principal kws/admin@EXAMPLE.COM: <Type it again>
Principal "kws/admin@EXAMPLE.COM" created.
kadmin.local:
```

**b. Quit `kadmin.local`.**

You have added all of the required principals for the next steps.

```
kadmin.local: quit
```
12 (Optional) Configure LDAP dependencies for Kerberos services.

If the LDAP and KDC servers are running on the same host and if the LDAP service is configured with a SMF FMRI, add a dependency to the LDAP service for the Kerberos daemons. This dependency will restart the KDC service if the LDAP service is restarted.

a. Add the dependency to the krb5kdc service.

```
# svccfg -s security/krb5kdc
svc:/network/security/krb5kdc> addpg dsins1 dependency
svc:/network/security/krb5kdc> setprop dsins1/entities = \n     fmri: "svc:/application/sun/ds:ds--var-opt-SUNWdsee-dsins1"
svc:/network/security/krb5kdc> setprop dsins1/grouping = astring: "require_all"
svc:/network/security/krb5kdc> setprop dsins1/restart_on = astring: "restart"
svc:/network/security/krb5kdc> setprop dsins1/type = astring: "service"
svc:/network/security/krb5kdc> exit
```

b. Add the dependency to the kadmin service.

```
# svccfg -s security/kadmin
svc:/network/security/kadmin> addpg dsins1 dependency
svc:/network/security/kadmin> setprop dsins1/entities = \n     fmri: "svc:/application/sun/ds:ds--var-opt-SUNWdsee-dsins1"
svc:/network/security/kadmin> setprop dsins1/grouping = astring: "require_all"
svc:/network/security/kadmin> setprop dsins1/restart_on = astring: "restart"
svc:/network/security/kadmin> setprop dsins1/type = astring: "service"
svc:/network/security/kadmin> exit
```

13 Start the Kerberos daemons.

```
kdc1 # svcadm enable -r network/security/krb5kdc
kdc1 # svcadm enable -r network/security/kadmin
```

14 Start kadmin and add more principals.

At this point, you can add principals by using the GUI Kerberos Administration Tool. To do so, you must log in with one of the admin principal names that you created earlier in this procedure. However, the following command-line example is shown for simplicity.

```
kdc1 # /usr/sbin/kadmin -p kws/admin
Enter password: <Type kws/admin password>
kadmin:
```

a. Create the master KDC host principal.

The host principal is used by Kerberized applications, such as klist and kprop. Clients use this principal when mounting an authenticated NFS file system. Note that when the principal instance is a host name, the FQDN must be specified in lowercase letters, regardless of the case of the domain name in the name service.

```
kadmin: addprinc -randkey host/kdc1.example.com
Principal "host/kdc1.example.com@EXAMPLE.COM" created.
kadmin:
```
b. **(Optional) Create the kclient principal.**

This principal is used by the kclient utility during the installation of a Kerberos client. If you do not plan on using this utility, then you do not need to add the principal. The users of the kclient utility need to use this password.

```
kdadmin: addprinc clntconfig/admin
Enter password for principal clntconfig/admin@EXAMPLE.COM:    <Type the password>
Re-enter password for principal clntconfig/admin@EXAMPLE.COM:  <Type it again>
```

Principal "clntconfig/admin@EXAMPLE.COM" created.

kdadmin:

c. **Add the master KDC’s host principal to the master KDC’s keytab file.**

Adding the host principal to the keytab file allows this principal to be used automatically.

```
kadmin: ktadd host/kdc1.example.com
Entry for principal host/kdc1.example.com with kvno 3, encryption type AES-256 CTS mode with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal host/kdc1.example.com with kvno 3, encryption type AES-128 CTS mode with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal host/kdc1.example.com with kvno 3, encryption type Triple DES cbc mode with HMAC/shal added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal host/kdc1.example.com with kvno 3, encryption type ArcFour with HMAC/md5 added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal host/kdc1.example.com with kvno 3, encryption type DES cbc mode with RSA-MDS added to keytab WRFILE:/etc/krb5/krb5.keytab.
```

kdadmin:

d. **Quit kdadmin.**

```
kadmin: quit
```

15 **(Optional) Synchronize the master KDCs clock by using NTP or another clock synchronization mechanism.**

Installing and using the Network Time Protocol (NTP) is not required. However, every clock must be within the default time that is defined in the libdefaults section of the krb5.conf file for authentication to succeed. See “Synchronizing Clocks Between KDCs and Kerberos Clients” on page 405 for information about NTP.

16 **Configure Slave KDCs.**

To provide redundancy, make sure to install at least one slave KDC. See “How to Manually Configure a Slave KDC” on page 371 for specific instructions.

▼ **How to Automatically Configure a Slave KDC**

In the Oracle Solaris release, a slave KDC can be automatically configured by using the following procedure.

**Before You Begin**

You must assume the root role. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.
Create the KDC.

Run the kdcmgr utility to create the KDC. You need to provide both the master key password and the password for the administrative principal.

```
kdc2# kdcmgr -a kws/admin -r EXAMPLE.COM create -m kdc1 slave
```

Starting server setup

```
Setting up /etc/krb5/kdc.conf
```

```
Setting up /etc/krb5/krb5.conf
Obtaining TGT for kws/admin ...
Password for kws/admin@EXAMPLE.COM: <Type the password>
```

```
Setting up /etc/krb5/kadm5.acl.
```

```
Setting up /etc/krb5/kpropd.acl.
```

Waiting for database from master...
Waiting for database from master...
Waiting for database from master...

```
kdb5_util: Cannot find/read stored master key while reading master key
kdb5_util: Warning: proceeding without master key
Enter KDC database master key: <Type the password>
```

Setup COMPLETE.

```
kdc2#
```

How to Interactively Configure a Slave KDC

Use the following procedure to interactively configure a slave KDC.

Before You Begin

You must assume the root role. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

Create the KDC.

Run the kdcmgr utility to create the KDC. You need to provide both the master key password and the password for the administrative principal.

```
kdc1# kdcmgr create slave
```

Starting server setup

```
Enter the Kerberos realm: EXAMPLE.COM
What is the master KDC’s host name?: kdc1
```
Setting up /etc/krb5/kdc.conf

Setting up /etc/krb5/krb5.conf
Obtaining TGT for kws/admin ...
Password for kws/admin@EXAMPLE.COM: <Type the password>

Setting up /etc/krb5/kadm5.acl.

Setting up /etc/krb5/kpropd.acl.

Waiting for database from master...
Waiting for database from master...
Waiting for database from master...

kdb5_util: Cannot find/read stored master key while reading master key
kdb5_util: Warning: proceeding without master key
Enter KDC database master key: <Type the password>

-----------------------------------------
Setup COMPLETE.

kdc2#

▼ How to Manually Configure a Slave KDC

In this procedure, a new slave KDC named kdc2 is configured. Also, incremental propagation is configured. This procedure uses the following configuration parameters:

- Realm name = EXAMPLE.COM
- DNS domain name = example.com
- Master KDC = kdc1.example.com
- Slave KDC = kdc2.example.com
- admin principal = kws/admin

Before You Begin

The master KDC must be configured. For specific instructions if this slave is to be swappable, see “Swapping a Master KDC and a Slave KDC” on page 407.

You must assume the root role on the KDC server. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.
1 On the master KDC, start kadmin.
You must log in with one of the admin principal names that you created when you configured the master KDC.

```
kdcl # /usr/sbin/kadmin -p kws/admin
Enter password:  <Type kws/admin password>
kadmin:
```

a. On the master KDC, add slave host principals to the database, if not already done.
For the slave to function, it must have a host principal. Note that when the principal instance is a host name, the FQDN must be specified in lowercase letters, regardless of the case of the domain name in the name service.

```
kadmin: addprinc -randkey host/kdc2.example.com
Principal "host/kdc2.example.com@EXAMPLE.COM" created.
kadmin:
```

b. On the master KDC, create the kiprop principal.
The kiprop principal is used to authorize incremental propagation from the master KDC.

```
kadmin: addprinc -randkey kiprop/kdc2.example.com
Principal "kiprop/kdc2.example.com@EXAMPLE.COM" created.
kadmin:
```

c. Quit kadmin.

```
kadmin: quit
```

2 On the master KDC, edit the Kerberos configuration file (krb5.conf).
You need to add an entry for each slave. See the krb5.conf(4) man page for a full description of this file.

```
kdc1 # cat /etc/krb5/krb5.conf
  .
  [realms]
    EXAMPLE.COM = {
      kdc = kdc1.example.com
      kdc = kdc2.example.com
      admin_server = kdc1.example.com
    }
```

3 On the master KDC, add a kiprop entry to kadm5.acl.
This entry allows the master KDC to receive requests for incremental propagation for the kdc2 server.

```
kdc1 # cat /etc/krb5/kadm5.acl
*/admin@EXAMPLE.COM *
kiprop/kdc2.example.com@EXAMPLE.COM p
```

4 On the master KDC, restart kadmind to use the new entries in the kadm5.acl file.

```
kdc1 # svcadm restart network/security/kadmin
```
5 On all slave KDCs, copy the KDC administration files from the master KDC server.

This step needs to be followed on all slave KDCs, because the master KDC server has updated information that each KDC server needs. You can use ftp or a similar transfer mechanism to grab copies of the following files from the master KDC:

- `/etc/krb5/krb5.conf`
- `/etc/krb5/kdc.conf`

6 On all slave KDCs, add an entry for the master KDC and each slave KDC into the database propagation configuration file, `kpropd.acl`.

This information needs to be updated on all slave KDC servers.

```
kdc2 # cat /etc/krb5/kpropd.acl
host/kdc1.example.com@EXAMPLE.COM
host/kdc2.example.com@EXAMPLE.COM
```

7 On all slave KDCs, make sure that the Kerberos access control list file, `kadm5.acl`, is not populated.

An unmodified `kadm5.acl` file would look like:

```
kdc2 # cat /etc/krb5/kadm5.acl
*/admin@___default_realm___ *
```

If the file has kprop entries, remove them.

8 On the new slave, change an entry in `kdc.conf`.

Replace the `sunw_dbprop_master_uologsize` entry with an entry defining `sunw_dbprop_slave_poll`. The entry sets the poll time to two minutes.

```
kdc1 # cat /etc/krb5/kdc.conf
[kdcdefaults]
　kdc_ports = 88,750

[realms]
　EXAMPLE.COM = {
　　profile = /etc/krb5/krb5.conf
　　database_name = /var/krb5/principal
　　acl_file = /etc/krb5/kadm5.acl
　　kadmind_port = 749
　　max_life = 8h 0m 0s
　　max_renewable_life = 7d 0h 0m 0s
　　sunw_dbprop_enable = true
　　sunw_dbprop_slave_poll = 2m
　}
```
On the new slave, start the `kadmin` command.

You must log in with one of the `admin` principal names that you created when you configured the master KDC.

```
kdc2 # /usr/sbin/kadmin -p kws/admin
```

Enter password: <Type kws/admin password>

```
kadmin:
```

a. Add the slave's host principal to the slave's keytab file by using `kadmin`.

This entry allows `kprop` and other Kerberized applications to function. Note that when the principal instance is a host name, the FQDN must be specified in lowercase letters, regardless of the case of the domain name in the name service.

```
kadmin: ktadd host/kdc2.example.com
Entry for principal host/kdc2.example.com with kvno 3, encryption type AES-256 CTS mode with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal host/kdc2.example.com with kvno 3, encryption type AES-128 CTS mode with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal host/kdc2.example.com with kvno 3, encryption type Triple DES cbc mode with HMAC/sha1 added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal host/kdc2.example.com with kvno 3, encryption type ArcFour with HMAC/md5 added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal host/kdc2.example.com with kvno 3, encryption type DES cbc mode with RSA-MD5 added to keytab WRFILE:/etc/krb5/krb5.keytab.
```

```
kadmin:
```

b. Add the `kiprop` principal to the slave KDC's keytab file.

Adding the `kiprop` principal to the `krb5.keytab` file allows the `kpropd` command to authenticate itself when incremental propagation is started.

```
kadmin: ktadd kiprop/kdc2.example.com
Entry for principal kiprop/kdc2.example.com with kvno 3, encryption type AES-256 CTS mode with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal kiprop/kdc2.example.com with kvno 3, encryption type AES-128 CTS mode with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal kiprop/kdc2.example.com with kvno 3, encryption type Triple DES cbc mode with HMAC/sha1 added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal kiprop/kdc2.example.com with kvno 3, encryption type ArcFour with HMAC/md5 added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal kiprop/kdc2.example.com with kvno 3, encryption type DES cbc mode with RSA-MD5 added to keytab WRFILE:/etc/krb5/krb5.keytab.
```

```
kadmin:
```

c. Quit `kadmin`.

```
kadmin: quit
```

10 On the new slave, start the Kerberos propagation daemon.

```
kdc2 # svcadm enable network/security/krb5_prop
```

11 On the new slave, create a stash file by using `kdb5_util`.

```
kdc2 # /usr/sbin/kdb5_util stash
kdb5_util: Cannot find/read stored master key while reading master key
kdb5_util: Warning: proceeding without master key
```
Enter KDC database master key:  
(Type the key)

12  (Optional) On the new slave KDC, synchronize the master KDCs clock by using NTP or another clock synchronization mechanism.

Installing and using the Network Time Protocol (NTP) is not required. However, every clock must be within the default time that is defined in the `libdefaults` section of the `krb5.conf` file for authentication to succeed. See “Synchronizing Clocks Between KDCs and Kerberos Clients” on page 405 for information about NTP.

13  On the new slave, start the KDC daemon (krb5kdc).

```
kdc2 # svcadm enable network/security/krb5kdc
```

**How to Refresh the Ticket-Granting Service Keys on a Master Server**

When the ticket-granting service (TGS) principal only has a DES key, which is the case for KDC servers created prior to the Solaris 10 release, the key restricts the encryption type of the ticket-granting ticket (TGT) session key to DES. If a KDC is updated to a release that supports additional, stronger encryption types, the administrator can expect that stronger encryption will be used for all session keys generated by the KDC. However, if the existing TGS principal does not have its keys refreshed to include the new encryption types, then the TGT session key will be continue to be limited to DES. The following procedure refreshes the key so that additional encryption types can be used.

- Refresh the TGS service principal key.

```
kdc1 % /usr/sbin/kadmin -p kws/admin
Enter password:  <Type kws/admin password>
kadmin: cpw -randkey krbtgt/EXAMPLE.COM@EXAMPLE.COM
```

**Example 21–2** Refreshing the Principal Keys from a Master Server

If you are logged on to the KDC master as root, you can refresh the TGS service principal with the following command:

```
kdc1 # kadmin.local -q 'cpw -randkey krbtgt/EXAMPLE.COM@EXAMPLE.COM'
```
Configuring Cross-Realm Authentication

You have several ways of linking realms together so that users in one realm can be authenticated in another realm. Cross-realm authentication is accomplished by establishing a secret key that is shared between the two realms. The relationship of the realms can be either hierarchal or directional (see “Realm Hierarchy” on page 347).

▼ How to Establish Hierarchical Cross-Realm Authentication

The example in this procedure uses two realms, ENG.EAST.EXAMPLE.COM and EAST.EXAMPLE.COM. Cross-realm authentication will be established in both directions. This procedure must be completed on the master KDC in both realms.

Before You Begin

The master KDC for each realm must be configured. To fully test the authentication process, several Kerberos clients must be configured.

You must assume the root role on both KDC servers. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1 Create ticket-granting ticket service principals for the two realms.

You must log in with one of the admin principal names that was created when you configured the master KDC.

```
#/usr/sbin/kadmin -p kws/admin
Enter password: <Type kws/admin password>
```

```
# kadmin: addprinc krbtgt/ENG.EAST.EXAMPLE.COM@EAST.EXAMPLE.COM
```

```
Enter password for principal krbtgt/ENG.EAST.EXAMPLE.COM@EAST.EXAMPLE.COM: <Type password>
```

```
# kadmin: addprinc krbtgt/EAST.EXAMPLE.COM@ENG.EAST.EXAMPLE.COM
```

```
Enter password for principal krbtgt/EAST.EXAMPLE.COM@ENG.EAST.EXAMPLE.COM: <Type password>
```

```
# kadmin: quit
```

Note – The password that is specified for each service principal must be identical in both KDCs. Thus, the password for the service principal krbtgt/ENG.EAST.EXAMPLE.COM@EAST.EXAMPLE.COM must be the same in both realms.

2 Add entries to the Kerberos configuration file (krb5.conf) to define domain names for every realm.

```
#/usr/sbin/kadmin -p kws/admin
Enter password: <Type kws/admin password>
```

```
# cat /etc/krb5/krb5.conf
[libdefaults]
.
[domain_realm]
  .eng.east.example.com = ENG.EAST.EXAMPLE.COM
  .east.example.com = EAST.EXAMPLE.COM
```

Oracle Solaris 11.1 Administration: Security Services • February 2014
In this example, domain names for the ENG.EAST.EXAMPLE.COM and EAST.EXAMPLE.COM realms are defined. It is important to include the subdomain first, because the file is searched top down.

3  **Copy the Kerberos configuration file to all clients in this realm.**

For cross-realm authentication to work, all systems (including slave KDCs and other servers) must have the new version of the Kerberos configuration file (/etc/krb5/krb5.conf) installed.

4  **Repeat all of these steps in the second realm.**

### How to Establish Direct Cross-Realm Authentication

The example in this procedure uses two realms, ENG.EAST.EXAMPLE.COM and SALES.WEST.EXAMPLE.COM. Cross-realm authentication will be established in both directions. This procedure must be completed on the master KDC in both realms.

#### Before You Begin

The master KDC for each realm must be configured. To fully test the authentication process, several Kerberos clients must be configured.

You must assume the root role on both KDC servers. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1  **Create ticket-granting ticket service principals for the two realms.**

You must log in with one of the admin principal names that was created when you configured the master KDC.

```
# /usr/sbin/kadmin -p kws/admin
Enter password: <Type kws/admin password>
kadmin: addprinc krbtgt/ENG.EAST.EXAMPLE.COM@SALES.WEST.EXAMPLE.COM
Enter password for principal
  krbtgt/ENG.EAST.EXAMPLE.COM@SALES.WEST.EXAMPLE.COM: <Type the password>
kadmin: addprinc krbtgt/SALES.WEST.EXAMPLE.COM@ENG.EAST.EXAMPLE.COM
Enter password for principal
  krbtgt/SALES.WEST.EXAMPLE.COM@ENG.EAST.EXAMPLE.COM: <Type the password>
kadmin: quit
```

**Note** – The password that is specified for each service principal must be identical in both KDCs. Thus, the password for the service principal krbtgt/ENG.EAST.EXAMPLE.COM@SALES.WEST.EXAMPLE.COM must be the same in both realms.
2 Add entries in the Kerberos configuration file to define the direct path to the remote realm.
This example shows the clients in the ENG.EAST.EXAMPLE.COM realm. You would need to swap
the realm names to get the appropriate definitions in the SALES.WEST.EXAMPLE.COM realm.

```
# cat /etc/krb5/krb5.conf
[libdefaults]
.
[capaths]
  ENG.EAST.EXAMPLE.COM = {
    SALES.WEST.EXAMPLE.COM = .
  }

  SALES.WEST.EXAMPLE.COM = {
    ENG.EAST.EXAMPLE.COM = .
  }
```

3 Copy the Kerberos configuration file to all clients in the current realm.
For cross-realm authentication to work, all systems (including slave KDCs and other servers)
must have the new version of the Kerberos configuration file (/etc/krb5/krb5.conf) installed.

4 Repeat all of these steps for the second realm.

### Configuring Kerberos Network Application Servers

Network application servers are hosts that provide access using one or more of the following
network applications: ftp, rcp, rlogin, rsh, ssh, and telnet. Only a few steps are required to
enable the Kerberos version of these commands on a server.

▼ How to Configure a Kerberos Network Application Server

This procedure uses the following configuration parameters:

- Application server = boston
- admin principal = kws/admin
- DNS domain name = example.com
- Realm name = EXAMPLE.COM

**Before You Begin**

This procedure requires that the master KDC has been configured. To fully test the process,
several Kerberos clients must be configured.

You must assume the root role on the application server. For more information, see “How to
Use Your Assigned Administrative Rights” on page 157.
1 **(Optional) Install the NTP client or another clock synchronization mechanism.**

See “Synchronizing Clocks Between KDCs and Kerberos Clients” on page 405 for information about NTP.

2 **Add principals for the new server and update the server’s keytab file.**

The following command reports the existence of the host principal:

```
boston # klist -k |grep host
```

```
4 host/boston.example.com@EXAMPLE.COM
4 host/boston.example.com@EXAMPLE.COM
4 host/boston.example.com@EXAMPLE.COM
4 host/boston.example.com@EXAMPLE.COM
```

If the command does not return a principal, then create new principals using the following steps.

How to use the GUI Kerberos Administration Tool to add a principal is explained in “How to Create a New Kerberos Principal” on page 463. The example in the following steps shows how to add the required principals using the command line. You must log in with one of the admin principal names that you created when configuring the master KDC.

```
boston # /usr/sbin/kadmin -p kws/admin
Enter password:  <Type kws/admin password>
```

```
kadmin:
```

**a. Create the server’s host principal.**

The host principal is used in the following ways:

- To authenticate traffic when using the remote commands, such as rsh and ssh.
- By pam_krb5 to prevent KDC spoofing attacks by using the host principal to verify that a user’s Kerberos credential was obtained from a trusted KDC.
- To allow the root user to automatically acquire a Kerberos credential without requiring that a root principal exist. This can be useful when doing a manual NFS mount where the share requires a Kerberos credential.

This principal is required if traffic using the remote application is to be authenticated using the Kerberos service. If the server has multiple hostnames associated with it, then create a principal for each hostname using the FQDN form of the hostname.

```
kadmin: addprinc -randkey host/boston.example.com
Principal "host/boston.example.com" created.
```

**b. Add the server’s host principal to the server’s keytab file.**

If the kadmin command is not running, restart it with a command similar to the following:

```
/usr/sbin/kadmin -p kws/admin
```

```
kadmin:
```
If the server has multiple host names associated with it, then add a principal to the keytab for each hostname.

```
kadmin: ktadd host/boston.example.com
Entry for principal host/boston.example.com with kvno 3, encryption type AES-256 CTS mode with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal host/boston.example.com with kvno 3, encryption type AES-128 CTS mode with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal host/boston.example.com with kvno 3, encryption type Triple DES cbc mode with HMAC/sha1 added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal host/boston.example.com with kvno 3, encryption type ArcFour with HMAC/md5 added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal host/boston.example.com with kvno 3, encryption type DES cbc mode with RSA-MD5 added to keytab WRFILE:/etc/krb5/krb5.keytab.
```

```
kadmin: quit
```

▼ How to Use the Generic Security Service With Kerberos When Running FTP

The generic security service (GSS) can be used to applications to easily use Kerberos for authentication, integrity, and privacy. The following steps show how to enable the GSS service for ProFTPD.

**Before You Begin**

You must assume the root role on the FTP server. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1. **Add principals for the FTP server and create the FTP server’s keytab file.**

   These steps might not be needed if the changes were made earlier.

   a. **Start the kadmin command.**
   
   ```
   ftpserver1 # /usr/sbin/kadmin -p kws/admin
   Enter password: <Type kws/admin password>
   kadmin:
   ```

   b. **Add the ftp service principal for the FTP server.**
   
   ```
   kadmin: ank -randkey ftp/ftpserver1.example.com
   ```

   c. **Add the ftp service principal to a new keytab file.**
   
   Creating a new keytab file, allows this information to be available to the ftp service, without exposing all of the information it the keytab file for the server.
   
   ```
   kadmin: ktadd -k /etc/krb5/ftp.keytab ftp/ftpserver1.example.com
   ```
2 **Change ownership of the new keytab file.**

   ftpserver1 # chown ftp:ftp /etc/krb5/ftp.keytab

3 **Enable GSS for the FTP server.**

   Make the following changes to the `/etc/proftpd.conf` file.

   ```
   # cat /etc/proftpd.conf
   LoadModule mod_gss.c
   GSSEngine on
   GSSKeytab /etc/krb5/ftp.keytab
   ```

4 **Restart the FTP server.**

   ```
   # svcadm restart network/ftp
   ```

---

**Configuring Kerberos NFS Servers**

NFS services use UNIX user IDs (UIDs) to identify a user and cannot directly use GSS credentials. To translate the credential to a UID, a credential table that maps user credentials to UNIX UIDs might need to be created. See "Mapping GSS Credentials to UNIX Credentials" on page 349 for more information on the default credential mapping. The procedures in this section focus on the tasks that are necessary to configure a Kerberos NFS server, to administer the credential table, and to initiate Kerberos security modes for NFS-mounted file systems. The following task map describes the tasks that are covered in this section.

**TABLE 21-2  Configuring Kerberos NFS Servers (Task Map)**

<table>
<thead>
<tr>
<th>Task</th>
<th>Description</th>
<th>For Instructions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configure a Kerberos NFS server.</td>
<td>Enables a server to share a file system that requires Kerberos authentication.</td>
<td>&quot;How to Configure Kerberos NFS Servers&quot; on page 382</td>
</tr>
<tr>
<td>Create a credential table.</td>
<td>Generates a credential table which can be used to provide mapping from GSS credentials to UNIX user UIDs, if the default mapping is not sufficient.</td>
<td>&quot;How to Create a Credential Table&quot; on page 383</td>
</tr>
<tr>
<td>Change the credential table that maps user credentials to UNIX UIDs.</td>
<td>Updates information in the credential table.</td>
<td>&quot;How to Add a Single Entry to the Credential Table&quot; on page 384</td>
</tr>
<tr>
<td>Create credential mappings between two like realms.</td>
<td>Provides instructions on how to map UIDs from one realm to another if the realms share a password file.</td>
<td>&quot;How to Provide Credential Mapping Between Realms&quot; on page 384</td>
</tr>
<tr>
<td>Share a file system with Kerberos authentication.</td>
<td>Shares a file system with security modes so that Kerberos authentication is required.</td>
<td>&quot;How to Set Up a Secure NFS Environment With Multiple Kerberos Security Modes&quot; on page 385</td>
</tr>
</tbody>
</table>
How to Configure Kerberos NFS Servers

In this procedure, the following configuration parameters are used:

- Realm name = EXAMPLE.COM
- DNS domain name = example.com
- NFS server = denver.example.com
- admin principal = kws/admin

Before You Begin

You must assume the root role on the NFS server. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1. Complete the prerequisites for configuring a Kerberos NFS server.
   The master KDC must be configured. To fully test the process, you need several clients.

2. (Optional) Install the NTP client or another clock synchronization mechanism.
   Installing and using the Network Time Protocol (NTP) is not required. However, every clock must be synchronized with the time on the KDC server within a maximum difference defined by the clockskew relation in the krb5.conf file for authentication to succeed. See “Synchronizing Clocks Between KDCs and Kerberos Clients” on page 405 for information about NTP.

3. Configure the NFS server as a Kerberos client.
   Follow the instructions in “Configuring Kerberos Clients” on page 387.

4. Start kadmin.
   You can use the Graphical Kerberos Administration Tool to add a principal, as explained in “How to Create a New Kerberos Principal” on page 463. To do so, you must log in with one of the admin principal names that you created when you configured the master KDC. However, the following example shows how to add the required principals by using the command line.

   denver # /usr/sbin/kadmin -p kws/admin
   Enter password: <Type kws/admin password>
   kadmin:

   a. Create the server’s NFS service principal.
      Note that when the principal instance is a host name, the FQDN must be specified in lowercase letters, regardless of the case of the domain name in the name service.
      Repeat this step for each unique interface on the system that might be used to access NFS data. If a host has multiple interfaces with unique names, each unique name must have its own NFS service principal.

      kadmin: addprinc -randkey nfs/denver.example.com
      Principal "nfs/denver.example.com" created.
      kadmin:
b. Add the server’s NFS service principal to the server’s keytab file.

Repeat this step for each unique service principal created in Step a.

kadmin: ktadd nfs/denver.example.com
Entry for principal nfs/denver.example.com with kvno 3, encryption type AES-256 CTS mode with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal nfs/denver.example.com with kvno 3, encryption type AES-128 CTS mode with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal nfs/denver.example.com with kvno 3, encryption type Triple DES cbc mode with HMAC/sha1 added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal nfs/denver.example.com with kvno 3, encryption type ArcFour with HMAC/md5 added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal nfs/denver.example.com with kvno 3, encryption type DES cbc mode with RSA-MD5 added to keytab WRFILE:/etc/krb5/krb5.keytab.

kadmin: quit

5 (Optional) Create special GSS credential maps, if needed.

Normally, the Kerberos service generates appropriate maps between the GSS credentials and the UNIX UIDs. The default mapping is described in “Mapping GSS Credentials to UNIX Credentials” on page 349. If the default mapping is not sufficient, see “How to Create a Credential Table” on page 383 for more information.

6 Share the NFS file system with Kerberos security modes.

See “How to Set Up a Secure NFS Environment With Multiple Kerberos Security Modes” on page 385 for more information.

▼ How to Create a Credential Table

The gsscred credential table is used by an NFS server to map Kerberos credentials to a UID. By default, the primary part of the principal name is matched to a UNIX login name. For NFS clients to mount file systems from an NFS server with Kerberos authentication, this table must be created if the default mapping is not sufficient.

Before You Begin
You must assume the root role on the NFS server. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1 Edit /etc/gss/gsscred.conf and change the security mechanism.

Change the mechanism to files.

2 Create the credential table by using the gsscred command.

# gsscred -m kerberos_v5 -a
The gsscred command gathers information from all sources that are listed with the passwd entry in the svc:/system/name-service/switch:default service. You might need to temporarily remove the files entry, if you do not want the local password entries included in the credential table. See the gsscred(1M) man page for more information.

▼ How to Add a Single Entry to the Credential Table

Before You Begin
This procedure requires that the gsscred table has already been created on the NFS server. See “How to Create a Credential Table” on page 383 for instructions.

You must assume the root role on the NFS server. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

● Add an entry to the credential table by using the gsscred command.

```
# gsscred -m mech [ -n name [ -u uid ] ] -a
```

- **mech** Defines the security mechanism to be used.
- **name** Defines the principal name for the user, as defined in the KDC.
- **uid** Defines the UID for the user, as defined in the password database.
- **-a** Adds the UID to principal name mapping.

Example 21–3 Adding a Multiple Component Principal to the Credential Table

In the following example, an entry is added for a principal named sandy/admin, which is mapped to UID 3736.

```
# gsscred -m kerberos_v5 -n sandy/admin -u 3736 -a
```

Example 21–4 Adding a Principal in a Different Domain to the Credential Table

In the following example, an entry is added for a principal named sandy/admin@EXAMPLE.COM, which is mapped to UID 3736.

```
# gsscred -m kerberos_v5 -n sandy/admin@EXAMPLE.COM -u 3736 -a
```

▼ How to Provide Credential Mapping Between Realms

This procedure provides appropriate credential mapping between realms that use the same password file. In this example, the realms CORP.EXAMPLE.COM and SALES.EXAMPLE.COM use the same password file. The credentials for bob@CORP.EXAMPLE.COM and bob@SALES.EXAMPLE.COM are mapped to the same UID.
You must assume the root role on the client system. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

On the client system, add entries to the `krb5.conf` file.

```bash
# cat /etc/krb5/krb5.conf
[libdefaults]
    default_realm = CORP.EXAMPLE.COM

[realms]
    CORP.EXAMPLE.COM = {
        .
        auth_to_local_realm = SALES.EXAMPLE.COM
    }
```

Mapping Credentials Between Realms Using the Same Password File

This example provides appropriate credential mapping between realms that use the same password file. In this example, the realms `CORP.EXAMPLE.COM` and `SALES.EXAMPLE.COM` use the same password file. The credentials for `bob@CORP.EXAMPLE.COM` and `bob@SALES.EXAMPLE.COM` are mapped to the same UID. On the client system, add entries to the `krb5.conf` file.

```bash
# cat /etc/krb5/krb5.conf
[libdefaults]
    default_realm = CORP.EXAMPLE.COM

[realms]
    CORP.EXAMPLE.COM = {
        .
        auth_to_local_realm = SALES.EXAMPLE.COM
    }
```

Troubleshooting

See “Observing Mapping From GSS Credentials to UNIX Credentials” on page 450 to help with the process of troubleshooting credential mapping problems.

### How to Set Up a Secure NFS Environment With Multiple Kerberos Security Modes

This procedure enables a NFS server to provide secure NFS access using different security modes or flavors. When a client negotiates a security flavor with the NFS server, the first flavor that is offered by the server that the client has access to is used. This flavor is used for all subsequent client requests of the file system shared by the NFS server.
Before You Begin

You must assume the root role on the NFS server. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1 Verify that there is an NFS service principal in the keytab file.

The `klist` command reports if there is a keytab file and displays the principals. If the results show that no keytab file exists or that no NFS service principal exists, you need to verify the completion of all the steps in “How to Configure Kerberos NFS Servers” on page 382.

```
# klist -k
Keytab name: FILE:/etc/krb5/krb5.keytab
KVNO Principal
-----------------------------------
3    nfs/denver.example.com@EXAMPLE.COM
3    nfs/denver.example.com@EXAMPLE.COM
3    nfs/denver.example.com@EXAMPLE.COM
3    nfs/denver.example.com@EXAMPLE.COM
```

2 Enable Kerberos security modes in the `/etc/nfssec.conf` file.

Edit the `/etc/nfssec.conf` file and remove the “#” that is placed in front of the Kerberos security modes.

```
# cat /etc/nfssec.conf
#
# Uncomment the following lines to use Kerberos V5 with NFS
#
krb5     390003 kerberos_v5 default - # RPCSEC_GSS
krb5i    390004 kerberos_v5 default integrity # RPCSEC_GSS
krb5p    390005 kerberos_v5 default privacy # RPCSEC_GSS
```

3 Share the file systems with the appropriate security modes.

```
share -F nfs -o sec=mode file-system
```

`mode` specifies the security modes to be used when sharing the file system. When using multiple security modes, the first mode in the list is used as the default.

`file-system` defines the path to the file system to be shared.

All clients that attempt to access files from the named file system require Kerberos authentication. To access files, the user principal on the NFS client should be authenticated.

4 (Optional) If the automounter is being used, edit the `auto_master` database to select a security mode other than the default.

You need not follow this procedure if you are not using the automounter to access the file system or if the default selection for the security mode is acceptable.

```
file-system auto_home -nosuid,sec=mode
```
5  (Optional) Manually issue the `mount` command to access the file system by using a non-default mode.

Alternatively, you could use the `mount` command to specify the security mode, but this alternative does not take advantage of the automounter.

```
# mount -F nfs -o sec=mode file-system
```

**Example 21–6**  Sharing a File System With One Kerberos Security Mode

In this example, Kerberos authentication must succeed before any files can be accessed through the NFS service.

```
# share -F nfs -o sec=krb5 /export/home
```

**Example 21–7**  Sharing a File System With Multiple Kerberos Security Modes

In this example, all three Kerberos security modes have been selected. Which mode is used is negotiated between the client and the NFS server. If the first mode in the command fails, then the next is tried. See the `nfssec(5)` man page for more information.

```
# share -F nfs -o sec=krb5:krb5i:krb5p /export/home
```

## Configuring Kerberos Clients

Kerberos clients include any host, that is not a KDC server, on the network that needs to use Kerberos services. This section provides procedures for installing a Kerberos client, as well as specific information about using `root` authentication to mount NFS file systems.

### Configuring Kerberos Clients (Task Map)

The following task map includes all of the procedures associated with setting up Kerberos clients. Each row includes a task identifier, a description of why you would want to do that task, followed by a link to the task.

<table>
<thead>
<tr>
<th>Task</th>
<th>Description</th>
<th>For Instructions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Establish a Kerberos client installation profile.</td>
<td>Generates a client installation profile that can be used to automatically install a Kerberos client.</td>
<td>&quot;How to Create a Kerberos Client Installation Profile&quot; on page 388</td>
</tr>
<tr>
<td>Configure a Kerberos client.</td>
<td>Manually installs a Kerberos client. Use this procedure if each client installation requires unique installation parameters.</td>
<td>&quot;How to Manually Configure a Kerberos Client&quot; on page 394</td>
</tr>
</tbody>
</table>
### Task Description For Instructions

<table>
<thead>
<tr>
<th>Task</th>
<th>Description</th>
<th>For Instructions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Automatically installs a Kerberos client. Use this procedure if the installation parameters for each client are the same.</td>
<td>Automatically installs a Kerberos client of an Active Directory server.</td>
<td>“How to Automatically Configure a Kerberos Client” on page 388</td>
</tr>
<tr>
<td>Interactively installs a Kerberos client. Use this procedure if only a few of the installation parameters need to change.</td>
<td></td>
<td>“How to Interactively Configure a Kerberos Client” on page 390</td>
</tr>
<tr>
<td>Automatically installs a Kerberos client of an Active Directory server.</td>
<td></td>
<td>“How to Configure a Kerberos Client for an Active Directory Server” on page 393</td>
</tr>
<tr>
<td>Allow a client to access an NFS filesystem as the root user</td>
<td>Creates a root principal on the client, so that the client can mount an NFS filesystem shared with root access. Also, allows for the client to set up non-interactive root access to the NFS filesystem, so that cron jobs can run.</td>
<td>“How to Access a Kerberos Protected NFS File System as the root User” on page 400</td>
</tr>
<tr>
<td>Disable verification of the KDC that issued a client Ticket Granting Ticket (TGT).</td>
<td>Allows clients that do not have a host principal stored in the local Kerberos keytab file to skip the security check that verifies that the KDC that issued the TGT is the same server that issued the host principal.</td>
<td>“How to Disable Verification of the Ticket-Granting Ticket” on page 399</td>
</tr>
</tbody>
</table>

### How to Create a Kerberos Client Installation Profile

This procedure creates a `kclient` profile that can be used when you install a Kerberos client. By using the `kclient` profile, you reduce the likelihood of typing errors. Also, using the profile reduces user intervention as compared to the interactive process.

1. **Become superuser.**

2. **Create a `kclient` installation profile.**
   A sample `kclient` profile could look similar to the following:
   ```
   client# cat /net/denver.example.com/export/install/profile
   REALM EXAMPLE.COM
   KDC kdc1.example.com
   ADMIN clntconfig
   FILEPATH /net/denver.example.com/export/install/krb5.conf
   NFS 1
   DNSLOOKUP none
   ```

### How to Automatically Configure a Kerberos Client

**Before You Begin** This procedure uses an installation profile. See “How to Create a Kerberos Client Installation Profile” on page 388.
To complete this task, you must assume the root role. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

- **Run the kclient installation script.**
  
  You need to provide the password for the clntconfig principal to complete the process.

  ```
  client# /usr/sbin/kclient -p /net/denver.example.com/export/install/profile
  ```

  Starting client setup

  ---------------------------------------------------

  kdc1.example.com

  Setting up /etc/krb5/krb5.conf.

  Obtaining TGT for clntconfig/admin ...

  Password for clntconfig/admin@EXAMPLE.COM: <Type the password>

  nfs/client.example.com entry ADDED to KDC database.

  nfs/client.example.com entry ADDED to keytab.

  host/client.example.com entry ADDED to KDC database.

  host/client.example.com entry ADDED to keytab.

  Copied /net/denver.example.com/export/install/krb5.conf.

  ---------------------------------------------------

  Setup COMPLETE.

  client#

  ![Example21–8](image)

  **Example 21–8** Automatically Configuring a Kerberos Client With Command-Line Overrides

  The following example overrides the DNSARG and the KDC parameters that are set in the installation profile.

  ```
  # /usr/sbin/kclient -p /net/denver.example.com/export/install/profile\n-\n-d dns_fallback -k kdc2.example.com
  ```

  Starting client setup

  ---------------------------------------------------

  kdc1.example.com

  Setting up /etc/krb5/krb5.conf.

  Obtaining TGT for clntconfig/admin ...

  Password for clntconfig/admin@EXAMPLE.COM: <Type the password>

  nfs/client.example.com entry ADDED to KDC database.

  nfs/client.example.com entry ADDED to keytab.
host/client.example.com entry ADDED to KDC database.
host/client.example.com entry ADDED to keytab.

Copied /net/denver.example.com/export/install/krb5.conf.

---------------------------------------------------
Setup COMPLETE.

client#

▼ How to Interactively Configure a Kerberos Client

This procedure uses the kclient installation utility without a installation profile. In the Oracle Solaris 11 release, the kclient utility improved ease of use and the ability to work with Active Directory servers. See "How to Configure a Kerberos Client for an Active Directory Server" on page 393 for more information. See Example 21–10 for an example of running kclient on a previous release.

Before You Begin To complete this task, you must assume the root role. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

Run the kclient installation script.
You need to provide the following information:

■ Kerberos realm name
■ KDC master host name
■ KDC slave host names
■ Domains to map to the local realm
■ PAM service names and options to use for Kerberos authentication

a. Indicate if the KDC server is not running an Oracle Solaris release.
If this system is a client of a KDC server that is not running an Oracle Solaris release, you need to define the type of server that is running the KDC. The available servers are: Microsoft Active Directory, MIT KDC server, Heimdal KDC server, and Shishi KDC server.

b. Select if DNS should be used for Kerberos lookups.
If you use DNS for Kerberos lookups, you need to enter the DNS lookup option that you want to use. Valid options are dns_lookup_kdc, dns_lookup.realm, and dns_fallback. See the krb5.conf(4) man page for more information about these values.

c. Define the name of the Kerberos realm and the master KDC hostname.
This information is added to the /etc/krb5/krb5.conf configuration file.
d. **Select if slave KDCs exist.**
   If there are slave KDCs in the realm, then you need to enter the slave KDC host names. This information is used to create additional KDC entries in the client’s configuration file.

e. **Indicate if service or host keys are required.**
   Normally, service or host keys are not required unless the client system is hosting Kerberized services.

f. **Specify if the client is a member of a cluster.**
   If the client is a member of a cluster, then you need to provide the logical name of the cluster. The logical host name is used when creating service keys, which is required when hosting Kerberos services from clusters.

g. **Identify any domains or hosts to map to the current realm.**
   This mapping allows other domains to belong in the default realm of the client.

h. **Specify if the client will use Kerberized NFS.**
   NFS service keys need to be created if the client will host NFS services using Kerberos.

i. **Indicate if a new PAM policy needs to be created.**
   This allows you to set which PAM services use Kerberos for authentication. You need to enter the service name and a flag indicating how Kerberos authentication is to be used. The valid flag options are:
   - first – use Kerberos authentication first, and only use UNIX if Kerberos authentication fails
   - only – use Kerberos authentication only
   - optional – use Kerberos authentication optionally

j. **Select if the master /etc/krb5/krb5.conf file should be copied.**
   This option allows for specific configuration information to be used when the arguments to kclient are not sufficient.

---

**Example 21–9**  **Running the kclient Installation Utility**

```bash
client# /usr/sbin/kclient
Starting client setup
------------------------------
Is this a client of a non-Solaris KDC ? [y/n]: n
No action performed.
Do you want to use DNS for kerberos lookups ? [y/n]: n
No action performed.
```
Configuring Kerberos Clients

Enter the Kerberos realm: EXAMPLE.COM
Specify the KDC hostname for the above realm: kdc1.example.com

Note, this system and the KDC's time must be within 5 minutes of each other for Kerberos to function. Both systems should run some form of time synchronization system like Network Time Protocol (NTP).

Do you have any slave KDC(s) ? [y/n]: y
Enter a comma-separated list of slave KDC host names: kdc2.example.com

Will this client need service keys ? [y/n]: n
No action performed.

Is this client a member of a cluster that uses a logical host name ? [y/n]: n
No action performed.

Do you have multiple domains/host to map to realm ? [y/n]: y
Enter a comma-separated list of domain/hosts to map to the default realm: engineering.example.com, example.com

Setting up /etc/krb5/krb5.conf.

Do you plan on doing Kerberized nfs ? [y/n]: y
Do you want to update /etc/pam.conf ? [y/n]: y
Enter a comma-separated list of PAM service names in the following format:
  service:{first|only|optional}: xscreensaver:first
Configuring /etc/pam.conf.

Do you want to copy over the master krb5.conf file ? [y/n]: n
No action performed.

--------------------------------------------------------------------------
Setup COMPLETE.

Example 21–10  Running the kclient Installation Utility in the Oracle Solaris 10 Release

The following output shows the results of running the kclient command.

client# /usr/sbin/kclient
Starting client setup

Do you want to use DNS for kerberos lookups ? [y/n]: n
No action performed.

Enter the Kerberos realm: EXAMPLE.COM
Specify the KDC hostname for the above realm: kdc1.example.com

Setting up /etc/krb5/krb5.conf.

Enter the krb5 administrative principal to be used: clntconfig/admin
Obtaining TGT for clntconfig/admin ...
Password for clntconfig/admin@EXAMPLE.COM: <Type the password>
Do you plan on doing Kerberized nfs ? [y/n]: n

host/client.example.com entry ADDED to KDC database.
host/client.example.com entry ADDED to keytab.

Do you want to copy over the master krb5.conf file ? [y/n]: y
Enter the pathname of the file to be copied: 
/net/denver.example.com/export/install/krb5.conf
Copied /net/denver.example.com/export/install/krb5.conf.

Setup COMPLETE!
#

▼ How to Configure a Kerberos Client for an Active Directory Server

This procedure uses the kclient installation utility without an installation profile.

**Before You Begin**

You must assume the root role. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1. (Optional) Enable DNS resource record creation for the client.
   ```bash
client# sharectl set -p ddns_enable=true smb
   ```

2. Run the kclient utility.
   The -T option selects a KDC server type. In this case an Active Directory server is selected.
   ```bash
client# kclient -T ms_ad
   ```
   By default, you will need to provide the password for the Administrator principal.

**Example 21–11 Confi guring a Kerberos Client for an Active Directory Server Using kclient**

The following output shows the results of running the kclient command using the ms_ad (Microsoft Active Directory) server type argument. The client will be joined to the Active Directory domain called EXAMPLE.COM.

```bash
client# /usr/sbin/kclient -T ms_ad
Starting client setup

Attempting to join 'CLIENT' to the 'EXAMPLE.COM' domain.
Password for Administrator@EXAMPLE.COM:  <Type the password>
Forest name found: example.com
Looking for local KDCs, DCs and global catalog servers (SVR RRs).

Setting up /etc/krb5/krb5.conf

Creating the machine account in AD via LDAP.
```
How to Manually Configure a Kerberos Client

In this procedure, the following configuration parameters are used:

- Realm name = EXAMPLE.COM
- DNS domain name = example.com
- Master KDC = kdc1.example.com
- Slave KDC = kdc2.example.com
- NFS server = denver.example.com
- Client = client.example.com
- admin principal = kws/admin
- User principal = mre

Note – Adjust the URL to point to the section, as described in “Online Help URL in the Graphical Kerberos Administration Tool” on page 353.

Before You Begin
You must assume the root role. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1 Edit the Kerberos configuration file (krb5.conf).

To change the file from the Kerberos default version, you need to change the realm names and the server names. You also need to identify the path to the help files for gkadmin.

```bash
kdc1 # cat /etc/krb5/krb5.conf
[libdefaults]
default_realm = EXAMPLE.COM

[realms]
EXAMPLE.COM = {
kdc = kdc1.example.com
kdc = kdc2.example.com
admin_server = kdc1.example.com
}

[domain_realm]
.example.com = EXAMPLE.COM
#
# if the domain name and realm name are equivalent,
```
# this entry is not needed
#
[logging]
default = FILE:/var/krb5/kdc.log
kdc = FILE:/var/krb5/kdc.log

[appdefaults]
gkadmin = {
}

Note – If you want to restrict the encryption types, you can set the `default_tkt_enctypes` or `default_tgs_enctypes` lines. Refer to “Using Kerberos Encryption Types” on page 519 for a description of the issues involved with restricting the encryption types.

2 (Optional) Change the process that is used to locate the KDCs.
By default, the Kerberos realm to KDC mapping is determined in the following order:

- The definition in the `realms` section in `krb5.conf`
- By looking up SRV records in DNS.

You can change this behavior by adding `dns_lookup_kdc` or `dns_fallback` to the `libdefaults` section of the `krb5.conf` file. See the `krb5.conf(4)` man page for more information. Note that referrals are always tried first.

3 (Optional) Change the process used to determine the realm for a host.
By default the host to realm mapping is determined in the following order:

- If the KDC supports referrals, then the KDC may inform the client which realm the host belongs to.
- By the definition of `domain_realm` in the `krb5.conf` file.
- The DNS domain name of the host.
- The default realm.

You can change this behavior by adding `dns_lookup_kdc` or `dns_fallback` to the `libdefaults` section of the `krb5.conf` file. See the `krb5.conf(4)` man page for more information. Note that referrals will always be tried first.

4 (Optional) Synchronize the client’s clock with the master KDC’s clock by using NTP or another clock synchronization mechanism.
Installing and using the Network Time Protocol (NTP) is not required. However, every clock must be synchronized with the time on the KDC server within a maximum difference defined in the `clockskew` relation in the `krb5.conf` file for authentication to succeed. See “Synchronizing Clocks Between KDCs and Kerberos Clients” on page 405 for information about NTP.
5 Start kadmin.

You can use the Graphical Kerberos Administration Tool to add a principal, as explained in “How to Create a New Kerberos Principal” on page 463. To do so, you must log in with one of the admin principal names that you created when you configured the master KDC. However, the following example shows how to add the required principals by using the command line.

denver # /usr/sbin/kadmin -p kws/admin
Enter password: 
</Type kws/admin password>

kadmin:

a. (Optional) Create a user principal if a user principal does not already exist.

You need to create a user principal only if the user associated with this host does not already have a principal assigned to him or her.

kadmin: addprinc mre
Enter password for principal mre@EXAMPLE.COM: 
</Type the password>
Re-enter password for principal mre@EXAMPLE.COM: 
</Type it again>

kadmin:

b. (Optional) Create a root principal and add the principal to the server's keytab file.

This step is required so that the client can have root access to file systems mounted using the NFS service. This step is also required if non-interactive root access is needed, such as running cron jobs as root.

If the client does not require root access to a remote file system which is mounted using the NFS service, then you can skip this step. The root principal should be a two component principal with the second component the host name of the Kerberos client system to avoid the creation of a realm wide root principal. Note that when the principal instance is a host name, the FQDN must be specified in lowercase letters, regardless of the case of the domain name in the name service.

kadmin: addprinc -randkey root/client.example.com
Principal "root/client.example.com" created.

kadmin: ktadd root/client.example.com
Entry for principal root/client.example.com with kvno 3, encryption type AES-256 CTS mode with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal root/client.example.com with kvno 3, encryption type AES-128 CTS mode with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal root/client.example.com with kvno 3, encryption type Triple DES cbc mode with HMAC/sha1 added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal root/client.example.com with kvno 3, encryption type ArcFour with HMAC/md5 added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal root/client.example.com with kvno 3, encryption type DES cbc mode with RSA-MD5 added to keytab WRFILE:/etc/krb5/krb5.keytab.

kadmin:

c. Create a host principal and add the principal to the server's keytab file.

The host principal is used by remote access services to provide authentication. The principal allows root to acquire a credential, if there is not one already in the keytab file.

kadmin: addprinc -randkey host/denver.example.com
Principal "host/denver.example.com@EXAMPLE.COM" created.
kadmin: ktadd host/denver.example.com
Entry for principal host/denver.example.com with kvno 3, encryption type AES-256 CTS mode with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal host/denver.example.com with kvno 3, encryption type AES-128 CTS mode with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal host/denver.example.com with kvno 3, encryption type Triple DES cbc mode with HMAC/sha1 added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal host/denver.example.com with kvno 3, encryption type ArcFour with HMAC/md5 added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal host/denver.example.com with kvno 3, encryption type DES cbc mode with RSA-MD5 added to keytab WRFILE:/etc/krb5/krb5.keytab.

kadmin:

d. (Optional) Add the server's NFS service principal to the server's keytab file.
This step is only required if the client needs to access NFS file systems using Kerberos authentication.

kadmin: ktadd nfs/denver.example.com
Entry for principal nfs/denver.example.com with kvno 3, encryption type AES-256 CTS mode with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal nfs/denver.example.com with kvno 3, encryption type AES-128 CTS mode with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal nfs/denver.example.com with kvno 3, encryption type Triple DES cbc mode with HMAC/sha1 added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal nfs/denver.example.com with kvno 3, encryption type ArcFour with HMAC/md5 added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal nfs/denver.example.com with kvno 3, encryption type DES cbc mode with RSA-MD5 added to keytab WRFILE:/etc/krb5/krb5.keytab.

kadmin:

e. Quit kadmin.

kadmin: quit

6 (Optional) Enable Kerberos with NFS.

a. Enable Kerberos security modes in the /etc/nfssec.conf file.

Edit the /etc/nfssec.conf file and remove the "#" that is placed in front of the Kerberos security modes.

```
# cat /etc/nfssec.conf
.
.
# Uncomment the following lines to use Kerberos V5 with NFS
#
krb5  390003  kerberos_v5  default -  # RPCSEC_GSS
krb5i 390004  kerberos_v5  default integrity # RPCSEC_GSS
krb5p 390005  kerberos_v5  default privacy  # RPCSEC_GSS
```

b. Enable DNS.

If the svc:/network/dns/client:default service is not enabled, enable it. See the resolv.conf(4) man page for more information.
c. Restart the gssd service.
   
   # svcadm restart network/rpc/gss

7 If you want the client to automatically renew the TGT or to warn users about Kerberos ticket expiration, create an entry in the /etc/krb5/warn.conf file.

See the warn.conf(4) man page for more information.

Example 21–12 Setting Up a Kerberos Client Using a Non-Solaris KDC

A Kerberos client can be set up to work with a non-Solaris KDC. In this case, a line must be included in the /etc/krb5/krb5.conf file in the realms section. This line changes the protocol that is used when the client is communicating with the Kerberos password-changing server. The format of this line follows.

[realms]

EXAMPLE.COM = {
  kdc = kdc1.example.com
  kdc = kdc2.example.com
  admin_server = kdc1.example.com
  kpasswd_protocol = SET_CHANGE
}

Example 21–13 DNS TXT Records for the Mapping of Host and Domain Name to Kerberos Realm

@ IN SOA kdc1.example.com root.kdc1.example.com (1989020501 ;serial
10800 ;refresh
3600 ;retry
3600000 ;expire
86400 ) ;minimum

IN NS kdc1.example.com.
kdc1 IN A 192.146.86.20
kdc2 IN A 192.146.86.21

._kerberos.example.com. IN TXT "EXAMPLE.COM"
._kerberos.kdc1.example.com. IN TXT "EXAMPLE.COM"
._kerberos.kdc2.example.com. IN TXT "EXAMPLE.COM"

Example 21–14 DNS SRV Records for Kerberos Server Locations

This example defines the records for the location of the KDCs, the admin server, and the kpasswd server, respectively.

@ IN SOA kdc1.example.com root.kdc1.example.com (1989020501 ;serial
10800 ;refresh
3600 ;retry
3600000 ;expire
86400 ) ;minimum
Example 21–15  Configuring a Solaris Client to Work with a Multiple Master KDC

The Microsoft Active Directory Kerberos service, provides a KDC that runs on multiple masters. In order for a Solaris Client to be able to update information either the admin_server or the kpasswd_server declaration in /etc/krb5/krb5.conf needs to be updated to list all of the servers. This example shows how to allow the client to update information on the KDC shared between kdc1 and kdc2.

```
[realms]
EXAMPLE.COM = {
    kdc = kdc1.example.com
    kdc = kdc2.example.com
    admin_server = kdc1.example.com
    admin_server = kdc2.example.com
}
```

How to Disable Verification of the Ticket-Granting Ticket

This procedure disables the security check that checks that the KDC of the host principal stored in the local /etc/krb5/krb5.keytab file is the same KDC that issued the ticket-granting ticket (TGT). This check prevents DNS spoofing attacks. However, for some client configurations, the host principal may not be available, so this check would need to be disabled to allow the client to function. These are the configurations that require that this check is disabled:

- The client IP address is dynamically assigned. For instance, a DHCP client.
- The client is not configured to host any services, so no host principal was created.
- The host key is not stored on the client.

Before You Begin  You must assume the root role. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.
Change the krb5.conf file.

If the verify_ap_req_nofail option is set to false, the TGT verification process is not enabled. See the krb5.conf(4) man page for more information about this option.

```
cat /etc/krb5/krb5.conf
```

For example:
```
[libdefaults]
default_realm = EXAMPLE.COM
verify_ap_req_nofail = false
```

Note – The verify_ap_req_nofail option can be entered in either the [libdefaults] or the [realms] section of the krb5.conf file. If the option is in the [libdefaults] section, the setting is used for all realms. If the option is in the [realms] section, the setting only applies to the defined realm.

How to Access a Kerberos Protected NFS File System as the root User

This procedure allows a client to access a NFS file system that requires Kerberos authentication with the root ID privilege. In particular, when the NFS file system is shared with options like: -o sec=krb5,root=client1.sun.com.

Start kadmin.

You can use the GUI Kerberos Administration Tool to add a principal, as explained in “How to Create a New Kerberos Principal” on page 463. To do so, you must log in with one of the admin principal names that you created when you configured the master KDC. However, the following example shows how to add the required principals by using the command line.

```
denver # /usr/sbin/kadmin -p kws/admin
Enter password:   <Type kws/admin password>
kadmin:
```

a. Create a root principal for the NFS client.

This principal is used to provide root equivalent access to NFS mounted file systems that require Kerberos authentication. The root principal should be a two component principal with the second component the host name of the Kerberos client system to avoid the creation of a realm wide root principal. Note that when the principal instance is a host name, the FQDN must be specified in lowercase letters, regardless of the case of the domain name in the name service.

```
addprinc -randkey root/client.example.com
```

Principal "root/client.example.com" created.

kadmin:
b. Add the root principal to the server's keytab file.

This step is required if you added a root principal so that the client can have root access to file systems mounted using the NFS service. This step is also required if non-interactive root access is needed, such as running cron jobs as root.

```
kadmin: ktadd root/client.example.com
Entry for principal root/client.example.com with kvno 3, encryption type AES-256 CTS mode with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal root/client.example.com with kvno 3, encryption type AES-128 CTS mode with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal root/client.example.com with kvno 3, encryption type Triple DES cbc mode with HMAC/sha1 added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal root/client.example.com with kvno 3, encryption type ArcFour with HMAC/md5 added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal root/client.example.com with kvno 3, encryption type DES cbc mode with RSA-MD5 added to keytab WRFILE:/etc/krb5/krb5.keytab.
```

c. Quit kadmin.
```
kadmin: quit
```

▼ How to Configure Automatic Migration of Users in a Kerberos Realm

Users, who do not have a Kerberos principal, can be automatically migrated to an existing Kerberos realm. The migration is achieved by using the PAM framework for the service in use by stacking the `pam_krb5_migrate` module in the service's authentication stack in the PAM configuration files.

In this example, the `gdm` and other PAM service names are configured to use the automatic migration. The following configuration parameters are used:

- Realm name = EXAMPLE.COM
- Master KDC = kdc1.example.com
- Machine hosting the migration service = server1.example.com
- Migration service principal = host/server1.example.com

**Before You Begin**

Set up server1 as a Kerberos client of the realm EXAMPLE.COM. See "Configuring Kerberos Clients" on page 387 for more information.

You must assume the root role. For more information, see "How to Use Your Assigned Administrative Rights" on page 157.
1. **Check to see if a host service principal for server1 exists.**

   The host service principal in the keytab file of server1 is used to authenticate the server to the master KDC.

   ```
   server1 # klist -k
   Keytab name: FILE:/etc/krb5/krb5.keytab
   KVNO Principal
   ---- ------------------------------------------------
   3 host/server1.example.com@EXAMPLE.COM
   3 host/server1.example.com@EXAMPLE.COM
   3 host/server1.example.com@EXAMPLE.COM
   3 host/server1.example.com@EXAMPLE.COM
   ```

2. **Make changes to the PAM configuration file.**

   a. **Update entries for the gdm service.**

   ```
   # cat /etc/pam.d/gdm
   ...
   auth  definitive  pam_user_policy.so.1
   auth  requisite   pam_authtok_get.so.1
   auth  required    pam_dhkeys.so.1
   auth  sufficient  pam_krb5.so.1
   auth  requisite   pam_unix_auth.so.1
   auth  required    pam_unix_cred.so.1
   auth  optional    pam_krb5_migrate.so.1
   ```

   b. **(Optional) Force an immediate password change, if needed.**

   The newly created Kerberos accounts can have their password expiration time set to the current time (now), in order to force an immediate Kerberos password change. To set the expiration time to now, add the `expire_pw` option to the lines that use the `pam_krb5_migrate` module. See the `pam_krb5_migrate(5)` man page for more information.

   ```
   # cat /etc/pam.d/gdm
   ...
   auth  optional  pam_krb5_migrate.so.1 expire_pw
   ```

   c. **Add the pam_krb5 module to the account stack.**

   This addition allows for password expiration in Kerberos to block access.

   ```
   # cat /etc/pam.d/other
   ...
   # Default definition for Account management
   # Used when service name is not explicitly mentioned for account management
   #
   account  requisite  pam_roles.so.1
   account  definitive  pam_user_policy.so.1
   account  required   pam_krb5.so.1
   account  required   pam_unix.account.so.1
   account  required   pam_tso1_account.so.1
   #
   ```
d. Add the `pam_krb5` module to the password stack.
   This addition allows for passwords to be updated when the password expire.

   ```bash
   # cat /etc/pam.d/other
   #
   # Default definition for Password management
   #
   password include    pam_authtok_common
   password sufficient pam_krb5.so.1
   password required   pam_authtok_store.so.1
   ```

3 On the master KDC, update the access control file.
   The following entries grant migrate and inquire privileges to the `host/server1.example.com` service principal for all users, excepting the `root` user. It is important that users who should not be migrated are listed in the `kadm5.acl` file using the `U` privilege. These entries need to be before the `permit all` or `ui` entry. See the `kadm5.acl(4)` man page for more information.

   ```bash
   kdc1 # cat /etc/krb5/kadm5.acl
   host/server1.example.com@EXAMPLE.COM U root
   host/server1.example.com@EXAMPLE.COM ui *
   */admin@EXAMPLE.COM *
   ```

4 On the master KDC, restart the Kerberos administration daemon.
   This step allows the `kadmind` daemon to use the new `kadm5.acl` entries.

   ```bash
   kdc1 # svcadm restart network/security/kadmin
   ```

5 On the master KDC, add entries to the `pam.conf` file.
   The following entries enable the `kadmind` daemon to use the `k5migrate` PAM service, to validate UNIX user password for accounts that require migration.

   ```bash
   # grep k5migrate /etc/pam.conf
   k5migrate   auth    required    pam_unix_auth.so.1
   k5migrate   account required    pam_unix_account.so.1
   ```
How to Configure Account Lockout

a. Start kadmin.

   % /usr/sbin/kadmin -p kws/admin
   Enter password:  <Type kws/admin password>
   kadmin:

   a. Create a policy with account lockout parameters.

   In the following example, the add_policy subcommand command is used to create a policy
   named default. Three authentication failures, during a span of 300 seconds will trigger an
   account lockout of 900 seconds.

   kadmin: add_policy -maxfailure 3 -failurecountinterval "300 seconds"
   -lockoutduration "900 seconds" default

   b. Quit kadmin.

   kadmin: quit

Example 21–16  Unlocking a Locked Out Principal

In the following example, a user principal is unlocked:

   /usr/sbin/kadmin -p kws/admin
   Enter password:  <Type kws/admin password>
   kadmin: modify_principal -unlock principal

How to Automatically Renew All Ticket-Granting Tickets (TGTs)

Before You Begin

You must assume the root role. For more information, see “How to Use Your Assigned
Administrative Rights” on page 157.

1  Edit the warn.conf configuration file.

   Add an entry like the following to renew the TGT 30 minutes before the ticket expires and log
   the message on the users terminal when the renewal succeeds or fails:

   # cat /etc/krb5/warn.conf
   * renew:log terminal 30m

2  Restart the service.

   # svcadm restart network/security/ktkt_warn
Example 21–17  Configuring TGT Expiration Messages on a Server

The following example shows several ways to configure the renewal and message system for TGTs.

```
# cat /etc/krb5/warn.conf
#
# renew the TGT 30 minutes before expiration and send message to users terminal
#
gtb@EXAMPLE.COM renew:log terminal 30m
#
# send a warning message to a specific email address 20 minutes before TGT expiration
#
mre@EXAMPLE.COM mail 20m mre@example2.com
#
# renew the TGT 20 minutes before expiration and send an email message on failure
#
bricker@EXAMPLE.COM renew:log-failure mail 20m &
#
# catch-all: any principal not matched above will get an email warning
# * mail 20m &
```

Example 21–18  Configuring TGT Expiration Messages for a User

In addition to configuring an entry for each user at the system level, each user can configure an individual warnr configuration file, which is named `/var/user/$USER/krb-warn.conf`. For example:

```
% cat /var/user/gtb/krb-warn.conf
gtb@EXAMPLE.COM renew:log mail 30m &
```

Synchronizing Clocks Between KDCs and Kerberos Clients

All hosts that participate in the Kerberos authentication system must have their internal clocks synchronized within a specified maximum amount of time (known as clock skew). This requirement provides another Kerberos security check. If the clock skew is exceeded between any of the participating hosts, client requests are rejected.

The clock skew also determines how long application servers must keep track of all Kerberos protocol messages, in order to recognize and reject replayed requests. So, the longer the clock skew value, the more information that application servers have to collect.

The default value for the maximum clock skew is 300 seconds (five minutes). You can change this default in the `libdefaults` section of the `krb5.conf` file.

Note – For security reasons, do not increase the clock skew beyond 300 seconds.
Because maintaining synchronized clocks between the KDCs and Kerberos clients is important, you should use the Network Time Protocol (NTP) software to synchronize them. NTP public domain software from the University of Delaware is included in the Oracle Solaris software.

**Note** – Another way to synchronize clocks is to use the `rdate` command and `cron` jobs, a process that can be less involved than using NTP. However, this section focuses on using NTP. And, if you use the network to synchronize the clocks, the clock synchronization protocol must itself be secure.

NTP enables you to manage precise time or network clock synchronization, or both, in a network environment. NTP is basically a server-client implementation. You pick one system to be the master clock (the NTP server). Then, you set up all your other systems (the NTP clients) to synchronize their clocks with the master clock.

To synchronize the clocks, NTP uses the `xntpd` daemon, which sets and maintains a UNIX system time-of-day in agreement with Internet standard time servers. The following shows an example of this server-client NTP implementation.

**FIGURE 21–1  Synchronizing Clocks by Using NTP**
Ensuring that the KDCs and Kerberos clients maintain synchronized clocks involves implementing the following steps:

1. Setting up an NTP server on your network. This server can be any system, except the master KDC. See “Managing Network Time Protocol (Tasks)” in Introduction to Oracle Solaris 11 Network Services to find the NTP server task.

2. As you configure the KDCs and Kerberos clients on the network, setting them up to be NTP clients of the NTP server. See “Managing Network Time Protocol (Tasks)” in Introduction to Oracle Solaris 11 Network Services to find the NTP client task.

Swapping a Master KDC and a Slave KDC

You should use the procedures in this section to make the swap of a master KDC with a slave KDC easier. You should swap the master KDC with a slave KDC only if the master KDC server fails for some reason, or if the master KDC needs to be re-installed (for example, because new hardware is installed).

How to Configure a Swappable Slave KDC

Perform this procedure on the slave KDC server that you want to have available to become the master KDC. This procedure assumes that you are using incremental propagation.

Before You Begin
You must assume the root role. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1. Use alias names for the master KDC and the swappable slave KDC during the KDC installation.
   When you define the host names for the KDCs, make sure that each system has an alias included in DNS. Also, use the alias names when you define the hosts in the /etc/krb5/krb5.conf file.

2. Follow the steps to install a slave KDC.
   Prior to any swap, this server should function as any other slave KDC in the realm. See “How to Manually Configure a Slave KDC” on page 371 for instructions.

3. Move the master KDC commands.
   To prevent the master KDC commands from being run from this slave KDC, move the kprop, kadmind, and kadmin.local commands to a reserved place.

   kdc4 # mv /usr/lib/krb5/kprop /usr/lib/krb5/kprop.save
   kdc4 # mv /usr/lib/krb5/kadmind /usr/lib/krb5/kadmind.save
   kdc4 # mv /usr/sbin/kadmin.local /usr/sbin/kadmin.local.save
How to Swap a Master KDC and a Slave KDC

In this procedure, the master KDC server that is being swapped out is named kdc1. The slave KDC that will become the new master KDC is named kdc4. This procedure assumes that you are using incremental propagation.

Before You Begin

This procedure requires that the slave KDC server has been set up as a swappable slave. For more information, see "How to Configure a Swappable Slave KDC" on page 407.

You must assume the root role. For more information, see "How to Use Your Assigned Administrative Rights" on page 157.

1 On the new master KDC, start kadmin.

   kdc4 # /usr/sbin/kadmin -p kws/admin
   Enter password: <Type kws/admin password>
   kadmin:

   a. Create new principals for the kadmind service.

      The following example shows the first addprinc command on two lines, but it should be
typed on one line.

      kadmin: addprinc -randkey -allow_tgs_req +password_changing_service -clearpolicy \
               changepw/kdc4.example.com
      Principal "changepw/kdc4.example.com@ENG.SUN.COM" created.
      kadmin: addprinc -randkey -allow_tgs_req -clearpolicy kadmin/kdc4.example.com
      Principal "kadmin/kdc4.example.com@EXAMPLE.COM" created.
      kadmin:

   b. Quit kadmin.

      kadmin: quit

2 On the new master KDC, force synchronization.

   The following steps force a full KDC update on the slave server.

   kdc4 # svcadm disable network/security/krb5kdc
   kdc4 # rm /var/krb5/principal.ulog

3 On the new master KDC, verify that the update is complete.

   kdc4 # /usr/sbin/kproplog -h

4 On the new master KDC, restart the KDC service.

   kdc4 # svcadm enable -r network/security/krb5kdc

5 On the new master KDC, clear the update log.

   These steps reinitialize the update log for the new master KDC server.

   kdc4 # svcadm disable network/security/krb5kdc
   kdc4 # rm /var/krb5/principal.ulog
6 On the old master KDC, kill the kadmind and krb5kdc processes.
When you kill the kadmind process, you prevent any changes from being made to the KDC database.

kdc1 # svcadm disable network/security/kadmind
kdc1 # svcadm disable network/security/krb5kdc

7 On the old master KDC, specify the poll time for requesting propagations.
Comment out the sunw_dbprop_master_uologsize entry in /etc/krb5/kdc.conf and add an entry defining sunw_dbprop_slave_poll. The entry sets the poll time to two minutes.

kdc1 # cat /etc/krb5/kdc.conf
[kdcdetails]
  kdc_ports = 88,750
[realms]
  EXAMPLE.COM = {  
    profile = /etc/krb5/krb5.conf  
    database_name = /var/krb5/principal  
    acl_file = /etc/krb5/kadm5.acl  
    kadmind_port = 749  
    max_life = 8h 0m 0s  
    max_renewable_life = 7d 0h 0m 0s  
    sunw_dbprop_enable = true  
    # sunw_dbprop_master_uologsize = 1000  
    sunw_dbprop_slave_poll = 2m
  }

8 On the old master KDC, move the master KDC commands and the kadm5.acl file.
To prevent the master KDC commands from being run, move the kprop, kadmind, and kadmin.local commands to a reserved place.

kdc1 # mv /usr/lib/krb5/kprop /usr/lib/krb5/kprop.save
kdc1 # mv /usr/lib/krb5/kadmind /usr/lib/krb5/kadmind.save
kdc1 # mv /usr/sbin/kadmin.local /usr/sbin/kadmin.local.save
kdc1 # mv /etc/krb5/kadm5.acl /etc/krb5/kadm5.acl.save

9 On the DNS server, change the alias names for the master KDC.
To change the servers, edit the example.com zone file and change the entry for masterkdc.

masterkdc IN CNAME kdc4

10 On the DNS server, restart the Internet domain name server.
Run the following command to reload the new alias information:

# svcadm refresh network/dns/server

11 On the new master KDC, move the master KDC commands and the slave kpropd.acl file.
kdc4 # mv /usr/lib/krb5/kprop.save /usr/lib/krb5/kprop
kdc4 # mv /usr/lib/krb5/kadmind.save /usr/lib/krb5/kadmind
kdc4 # mv /usr/sbin/kadmin.local.save /usr/sbin/kadmin.local
kdc4 # mv /etc/krb5/kpropd.acl /etc/krb5/kpropd.acl.save

Swapping a Master KDC and a Slave KDC

Chapter 21 • Configuring the Kerberos Service (Tasks) 409
On the new master KDC, create the Kerberos access control list file (kadm5.acl).

Once populated, the /etc/krb5/kadm5.acl file should contain all principal names that are allowed to administer the KDC. The file should also list all of the slaves that make requests for incremental propagation. See the kadm5.acl(4) man page for more information.

```
kadm4 # cat /etc/krb5/kadm5.acl
kws/admin@EXAMPLE.COM * 
kiprop/kdc1.example.com@EXAMPLE.COM p
```

On the new master KDC, specify the update log size in the kdc.conf file.

Comment out the sunw_dbprop_slave_poll entry and add an entry defining sunw_dbprop_master_ulogsize. The entry sets the log size to 1000 entries.

```
kadm1 # cat /etc/krb5/kdc.conf
[kdcdfaults]
 kdc_ports = 88,750

[realms] EXAMPLE.COM= {
  profile = /etc/krb5/krb5.conf
  database_name = /var/krb5/principal
  acl_file = /etc/krb5/kadm5.acl
  kadmind_port = 749
  max_life = 8h 0m 0s
  max_renewable_life = 7d 0h 0m 0s
  sunw_dbprop_enable = true
  #
  sunw_dbprop_slave_poll = 2m
  sunw_dbprop_master_ulogsize = 1000
}
```

On the new master KDC, start kadmind and krb5kdc.

```
kadm4 # svcadm enable -r network/security/krb5kdc
kadm4 # svcadm enable -r network/security/kadmin
```

On the old master KDC, add the kprop service principal.

Adding the kprop principal to the krb5.keytab file allows the kpropd daemon to authenticate itself for the incremental propagation service.

```
kadm1 /usr/sbin/kadmin -p kws/admin
Authenticating as principal kws/admin@EXAMPLE.COM with password.
Enter password: <Type kws/admin password>
nadmin: ktdadd kprop/kdc1.example.com
```

Entry for principal kprop/kdc1.example.com with kvno 3, encryption type AES-256 CTS mode with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5/krb5.keytab.

Entry for principal kprop/kdc1.example.com with kvno 3, encryption type AES-128 CTS mode with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5/krb5.keytab.

Entry for principal kprop/kdc1.example.com with kvno 3, encryption type Triple DES cbc mode with HMAC/shal added to keytab WRFILE:/etc/krb5/krb5.keytab.

Entry for principal kprop/kdc1.example.com with kvno 3, encryption type ArcFour with HMAC/md5 added to keytab WRFILE:/etc/krb5/krb5.keytab.

Entry for principal kprop/kdc1.example.com with kvno 3, encryption type DES cbc mode with RSA-MD5 added to keytab WRFILE:/etc/krb5/krb5.keytab.

kadm1: quit
16 On the old master KDC, add an entry for each KDC listed in krb5.conf to the propagation configuration file, kpropd.acl.

-kdc1 # cat /etc/krb5/kpropd.acl
host/kdc1.example.com@EXAMPLE.COM
host/kdc2.example.com@EXAMPLE.COM
host/kdc3.example.com@EXAMPLE.COM
host/kdc4.example.com@EXAMPLE.COM

17 On the old master KDC, start kpropd and krb5kdc.

-kdc1 # svcadm enable -r network/security/krb5_prop
-kdc1 # svcadm enable -r network/security/krb5kdc

Administering the Kerberos Database

The Kerberos database is the backbone of Kerberos and must be maintained properly. This section provides some procedures on how to administer the Kerberos database, such as backing up and restoring the database, setting up incremental or parallel propagation, and administering the stash file. The steps to initially set up the database are in “How to Manually Configure a Master KDC” on page 359.

Backing Up and Propagating the Kerberos Database

Propagating the Kerberos database from the master KDC to the slave KDCs is one of the most important configuration tasks. If propagation doesn’t happen often enough, the master KDC and the slave KDCs will lose synchronization. So, if the master KDC goes down, the slave KDCs will not have the most recent database information. Also, if a slave KDC has been configured as a master KDC for purposes of load balancing, the clients that use that slave KDC as a master KDC will not have the latest information. Therefore, you must make sure that propagation occurs often enough or else configure the servers for incremental propagation, based on how often you change the Kerberos database. Incremental propagation is preferred over manual propagation because there is more administrative overhead when you manually propagate the database. Also, there are inefficiencies when you do full propagation of the database.

When you configure the master KDC, you set up the kprop_script command in a cron job to automatically back up the Kerberos database to the /var/krb5/slave_datrans dump file and propagate it to the slave KDCs. But, as with any file, the Kerberos database can become corrupted. If data corruption occurs on a slave KDC, you might never notice, because the next automatic propagation of the database installs a fresh copy. However, if corruption occurs on the master KDC, the corrupted database is propagated to all of the slave KDCs during the next propagation. And, the corrupted backup overwrites the previous uncorrupted backup file on the master KDC.

Because there is no “safe” backup copy in this scenario, you should also set up a cron job to periodically copy the slave_datrans dump file to another location or to create another
separate backup copy by using the dump command of kdb5_util. Then, if your database becomes corrupted, you can restore the most recent backup on the master KDC by using the load command of kdb5_util.

Another important note: Because the database dump file contains principal keys, you need to protect the file from being accessed by unauthorized users. By default, the database dump file has read and write permissions only as root. To protect against unauthorized access, use only the kprop command to propagate the database dump file, which encrypts the data that is being transferred. Also, kprop propagates the data only to the slave KDCs, which minimizes the chance of accidentally sending the database dump file to unauthorized hosts.

Caution – If the Kerberos database is updated after it has been propagated and if the database subsequently is corrupted before the next propagation, the KDC slaves will not contain the updates. The updates will be lost. For this reason, if you add significant updates to the Kerberos database before a regularly scheduled propagation, you should manually propagate the database to avoid data loss.

The kpropd.acl File

The kpropd.acl file on a slave KDC provides a list of host principal names, one name per line, that specifies the systems from which the KDC can receive an updated database through propagation. If the master KDC is used to propagate all the slave KDCs, the kpropd.acl file on each slave needs to contain only the host principal name of the master KDC.

However, the Kerberos installation and subsequent configuration steps in this book instruct you to add the same kpropd.acl file to the master KDC and the slave KDCs. This file contains all the KDC host principal names. This configuration enables you to propagate from any KDC, in case the propagating KDCs become temporarily unavailable. And, by keeping an identical copy on all KDCs, you make the configuration easy to maintain.

The kprop_script Command

The kprop_script command uses the kprop command to propagate the Kerberos database to other KDCs. If the kprop_script command is run on a slave KDC, it propagates the slave KDC’s copy of the Kerberos database to other KDCs. The kprop_script accepts a list of host names for arguments, separated by spaces, which denote the KDCs to propagate.

When kprop_script is run, it creates a backup of the Kerberos database to the /var/krb5/slave_datatrans file and copies the file to the specified KDCs. The Kerberos database is locked until the propagation is finished.
How to Back Up the Kerberos Database

Before You Begin
You must assume the root role. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

- Back up the Kerberos database by using the dump command of the kdb5_util command.
  
  
  # /usr/sbin/kdb5_util dump [-verbose] [-d dbname] [filename [principals...]]

  
  - *verbose*: Prints the name of each principal and policy that is being backed up.

  - *dbname*: Defines the name of the database to back up. Note that you can specify an absolute path for the file. If the -d option is not specified, the default database name is /var/krb5/principal.

  - *filename*: Defines the file that is used to back up the database. You can specify an absolute path for the file. If you don’t specify a file, the database is dumped to standard output.

  - *principals*: Defines a list of one or more principals (separated by a space) to back up. You must use fully qualified principal names. If you don’t specify any principals, the entire database is backed up.

Example 21–19  Backing Up the Kerberos Database

In the following example, the Kerberos database is backed up to a file called dumpfile. Because the -verbose option is specified, each principal is printed as it is backed up.

  
  # kdb5_util dump -verbose dumpfile
  kadmin/kdcl.eng.example.com@ENG.EXAMPLE.COM
  krbtgt/ENG.EXAMPLE.COM@ENG.EXAMPLE.COM
  kadmin/history@ENG.EXAMPLE.COM
  pak/admin@ENG.EXAMPLE.COM
  pak@ENG.EXAMPLE.COM
  changepw/kdcl.eng.example.com@ENG.EXAMPLE.COM

In the following example, the pak and pak/admin principals from the Kerberos database are backed up.

  
  # kdb5_util dump -verbose dumpfile pak/admin@ENG.EXAMPLE.COM pak@ENG.EXAMPLE.COM
  pak/admin@ENG.EXAMPLE.COM
  pak@ENG.EXAMPLE.COM

How to Restore the Kerberos Database

Before You Begin
On the KDC master, you must assume the root role. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.
1. **On the master, stop the KDC daemons.**
   
   kdc1 # svcadm disable network/security/krb5kdc  
   kdc1 # svcadm disable network/security/kadmin

2. **Restore the Kerberos database by using the load command of the kdb_util command.**
   
   # /usr/sbin/kdb5_util load [-verbose] [-d dbname] [-update] [filename]

   - **-verbose** Prints the name of each principal and policy that is being restored.
   - **dbname** Defines the name of the database to restore. Note you can specify an absolute path for the file. If the -d option is not specified, the default database name is /var/krb5/principal.
   - **-update** Updates the existing database. Otherwise, a new database is created or the existing database is overwritten.
   - **filename** Defines the file from which to restore the database. You can specify an absolute path for the file.

3. **Start the KDC daemons.**
   
   kdc1 # svcadm enable -r network/security/krb5kdc  
   kdc1 # svcadm enable -r network/security/kadmin

**Example 21–20**  
**Restoring the Kerberos Database**

In the following example, the database called database1 is restored into the current directory from the dumpfile file. Because the -update option isn’t specified, a new database is created by the restore.

# kdb5_util load -d database1 dumpfile

**How to Convert a Kerberos Database After a Server Upgrade**

If your KDC database was created on a server running the Solaris 8 or Solaris 9 release, converting the database enables you to take advantage of the improved database format.

**Before You Begin**  
Make sure that the database is using an older format.

On the KDC master, you must assume the root role. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.
1 On the master, stop the KDC daemons.
   kdc1 # svcadm disable network/security/krb5kdc
   kdc1 # svcadm disable network/security/kadmin

2 Create a directory to store a temporary copy of the database.
   kdc1 # mkdir /var/krb5/tmp
   kdc1 # chmod 700 /var/krb5/tmp

3 Dump the KDC database.
   kdc1 # kdb5_util dump /var/krb5/tmp/prdb.txt

4 Save copies of the current database files.
   kdc1 # cd /var/krb5
   kdc1 # mv princ* tmp/

5 Load the database.
   kdc1 # kdb5_util load /var/krb5/tmp/prdb.txt

6 Start the KDC daemons.
   kdc1 # svcadm enable -r network/security/krb5kdc
   kdc1 # svcadm enable -r network/security/kadmin

▼ How to Reconfigure a Master KDC to Use Incremental Propagation

The steps in this procedure can be used to reconfigure an existing master KDC to use incremental propagation. In this procedure, the following configuration parameters are used:

- Realm name = EXAMPLE.COM
- DNS domain name = example.com
- Master KDC = kdc1.example.com
- Slave KDC = kdc2.example.com
- admin principal = kws/admin

You must assume the root role. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1 Add entries to kdc.conf.

You need to enable incremental propagation and select the number of updates the KDC master keeps in the log. See the kdc.conf(4) man page for more information.

   kdc1 # cat /etc/krb5/kdc.conf
   [kdcdefaults]
   kdc_ports = 88,750
[realms]
EXAMPLE.COM = {
    profile = /etc/krb5/krb5.conf
    database_name = /var/krb5/principal
    acl_file = /etc/krb5/kadm5.acl
    kadmind_port = 749
    max_life = 8h 0m 0s
    max_renewable_life = 7d 0h 0m 0s
    sunw_dbprop_enable = true
    sunw_dbprop_master_ulogfile = 1000
}

2 Create the kiprop principal.
The kiprop principal is used to authenticate the master KDC server and to authorize updates from the master KDC.

    kdc1 # /usr/sbin/kadmin -p kws/admin
    Enter password:  <Type kws/admin password>
    kadmin: addprinc -randkey kiprop/kdc1.example.com
    Principal "kiprop/kdc1.example.com@EXAMPLE.COM" created.
    kadmin: addprinc -randkey kiprop/kdc2.example.com
    Principal "kiprop/kdc2.example.com@EXAMPLE.COM" created.
    kadmin:

3 On the master KDC, add a kiprop entry to kadm5.acl
This entry allows the master KDC to receive requests for incremental propagation from the kdc2 server.

    kdc1 # cat /etc/krb5/kadm5.acl
    */admin@EXAMPLE.COM *
    kiprop/kdc2.example.com@EXAMPLE.COM p

4 Comment out the kprop line in the root crontab file.
This step prevents the master KDC from propagating its copy of the KDC database.

    kdc1 # crontab -e
    #ident "@(#)root 1.20 01/11/06 SMI"
    #
    # The root crontab should be used to perform accounting data collection.
    #
    # The rtc command is run to adjust the real time clock if and when
    # daylight savings time changes.
    #
    10 3 * * * /usr/sbin/logadm
    15 3 * * 0 /usr/lib/fs/nfs/nfsfind
    1 2 * * * [-x /usr/sbin/rtc ] && /usr/sbin/rtc -c > /dev/null 2>&1
    30 3 * * * [-x /usr/lib/gss/gsscred_clean ] && /usr/lib/gss/gsscred_clean
    #10 3 * * * /usr/lib/krb5kprop_script kdc2.example.com@EXAMPLE.COM #SUNWkr5ma

5 Restart kadmind.

    kdc1 # svcadm restart network/security/kadmin
6 Reconfigure all slave KDC servers that use incremental propagation.
See "How to Reconfigure a Slave KDC to Use Incremental Propagation" on page 417 for complete instructions.

How to Reconfigure a Slave KDC to Use Incremental Propagation

Before You Begin
You must assume the root role. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1 Add entries to kdc.conf.
The first new entry enables incremental propagation. The second new entry sets the poll time to two minutes.

```
kdc2 # cat /etc/krb5/kdc.conf
[kdcdefaults]
    kdc_ports = 88,750

[realms]
    EXAMPLE.COM = {
        profile = /etc/krb5/krb5.conf
        database_name = /var/krb5/principal
        acl_file = /etc/krb5/kadm5.acl
        kadmind_port = 749
        max_life = 8h 0m 0s
        max_renewable_life = 7d 0h 0m 0s
        sunw_dbprop_enable = true
        sunw_dbprop_slave_poll = 2m
    }
```

2 Add the kprop principal to the krb5.keytab file.

```
kdc2 # /usr/sbin/kadmin -p kws/admin
Enter password: <Type kws/admin password>
kadmin: ktadd kprop/kdc2.example.com
Entry for principal kprop/kdc2.example.com with kvno 3, encryption type AES-256 CTS mode with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal kprop/kdc2.example.com with kvno 3, encryption type AES-128 CTS mode with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal kprop/kdc2.example.com with kvno 3, encryption type Triple DES cbc mode with HMAC/sha1 added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal kprop/kdc2.example.com with kvno 3, encryption type ArcFour with HMAC/md5 added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal kprop/kdc2.example.com with kvno 3, encryption type DES cbc mode with RSA-MD5 added to keytab WRFILE:/etc/krb5/krb5.keytab.
kadmin: quit
```
3  Restart kpropd.
   kdc2 # svcadm restart network/security/krb5_prop

How to Configure a Slave KDC to Use Full Propagation

This procedure shows how to reconfigure a slave KDC server that is running the Solaris 10 release to use full propagation. Normally, the procedure would only be used if the master KDC server is running either the Solaris 9 release or an earlier release. In this case, the master KDC server cannot support incremental propagation, so the slave must be configured to allow propagation to work.

In this procedure, a slave KDC named kdc3 is configured. This procedure uses the following configuration parameters:

- Realm name = EXAMPLE.COM
- DNS domain name = example.com
- Master KDC = kdc1.example.com
- Slave KDC = kdc2.example.com and kdc3.example.com
- admin principal = kws/admin

Note – Adjust the URL to point to the section, as described in “Online Help URL in the Graphical Kerberos Administration Tool” on page 353.

Before You Begin

The master KDC must be configured. For specific instructions if this slave is to be swappable, see “Swapping a Master KDC and a Slave KDC” on page 407.

On the master KDC, you must assume the root role. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.
1 On the master KDC, start kadmin.

You must log in with one of the admin principal names that you created when you configured the master KDC.

```
kdc1 # /usr/sbin/kadmin -p kws/admin
Enter password: <Type kws/admin password>
```

```
kadmin:
```

a. On the master KDC, add slave host principals to the database, if not already done.

For the slave to function, it must have a host principal. Note that when the principal instance is a host name, the FQDN must be specified in lowercase letters, regardless of the case of the domain name in the name service.

```
kadmin: addprinc -randkey host/kdc3.example.com
Principal "host/kdc3@EXAMPLE.COM" created.
kadmin:
```

b. Quit kadmin.

```
kadmin: quit
```

2 On the master KDC, edit the Kerberos configuration file (krb5.conf).

You need to add an entry for each slave. See the `krb5.conf(4)` man page for a full description of this file.

```
kdc1 # cat /etc/krb5/krb5.conf
```

```
[realms]

EXAMPLE.COM = {
  kdc = kdc1.example.com
  kdc = kdc2.example.com
  kdc = kdc3.example.com
  admin_server = kdc1.example.com
}
```

3 On the master KDC, add an entry for the master KDC and each slave KDC into the kpropd.acl file.

See the `kprop(1M)` man page for a full description of this file.

```
kdc1 # cat /etc/krb5/kpropd.acl
host/kdc1.example.com@EXAMPLE.COM
host/kdc2.example.com@EXAMPLE.COM
host/kdc3.example.com@EXAMPLE.COM
```

4 On all slave KDCs, copy the KDC administration files from the master KDC server.
This step needs to be followed on all slave KDCs, because the master KDC server has updated information that each KDC server needs. You can use ftp or a similar transfer mechanism to grab copies of the following files from the master KDC:

- /etc/krb5/krb5.conf
- /etc/krb5/kdc.conf
- /etc/krb5/kpropd.acl

5 On all slave KDCs, make sure that the Kerberos access control list file, kadm5.acl, is not populated.

An unmodified kadm5.acl file would look like:

```
kdc2 # cat /etc/krb5/kadm5.acl
*/admin@___default_realm___ *
```

If the file has kprop entries, remove them.

6 On the new slave, start the kadmin command.

You must log in with one of the admin principal names that you created when you configured the master KDC.

```
kdc2 # /usr/sbin/kadmin -p kws/admin
```

```
Enter password: <Type kws/admin password>
kadmin:
```

a. Add the slave's host principal to the slave's keytab file by using kadmin.

This entry allows kprop and other Kerberized applications to function. Note that when the principal instance is a host name, the FQDN must be specified in lowercase letters, regardless of the case of the domain name in the name service.

```
kadmin: ktadd host/kdc3.example.com
```

```
Entry for principal host/kdc3.example.com with kvno 3, encryption type AES-256 CTS mode with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5/krb5.keytab.
```

```
Entry for principal host/kdc3.example.com with kvno 3, encryption type AES-128 CTS mode with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5/krb5.keytab.
```

```
Entry for principal host/kdc3.example.com with kvno 3, encryption type Triple DES cbc mode with HMAC/sha1 added to keytab WRFILE:/etc/krb5/krb5.keytab.
```

```
Entry for principal host/kdc3.example.com with kvno 3, encryption type ArcFour with HMAC/md5 added to keytab WRFILE:/etc/krb5/krb5.keytab.
```

```
Entry for principal host/kdc3.example.com with kvno 3, encryption type DES cbc mode with RSA-MD5 added to keytab WRFILE:/etc/krb5/krb5.keytab.
```

```
kadmin: quit
```

b. Quit kadmin.

```
kadmin: quit
```
On the master KDC, add the slave KDC name to the `cron` job, which automatically runs the backups, by running `crontab -e`.

Add the name of each slave KDC server at the end of the `kprop_script` line.

```
1 03 *** /usr/lib/krb5/kprop_script kdc2.example.com kdc3.example.com
```

You might also want to change the time of the backups. This entry starts the backup process every day at 3:10 AM.

On the new slave, start the Kerberos propagation daemon.

```
kdc3 # svcadm enable network/security/krb5_prop
```

On the master KDC, backup and propagate the database by using `kprop_script`.

If a backup copy of the database is already available, it is not necessary to complete another backup. See “How to Manually Propagate the Kerberos Database to the Slave KDCs” on page 423 for further instructions.

```
kdc1 # /usr/lib/krb5/kprop_script kdc3.example.com
```

Database propagation to kdc3.example.com: SUCCEEDED

On the new slave, create a stash file by using `kdb5_util`.

```
kdc3 # /usr/sbin/kdb5_util stash
```

```
kdb5 util: Cannot find/read stored master key while reading master key
kdb5 util: Warning: proceeding without master key
```

Enter KDC database master key:  

```
<Type the key>
```

(Optional) On the new slave KDC, synchronize the master KDCs clock by using NTP or another clock synchronization mechanism.

Installing and using the Network Time Protocol (NTP) is not required. However, every clock must be within the default time that is defined in the `libdefaults` section of the `krb5.conf` file for authentication to succeed. See “Synchronizing Clocks Between KDCs and Kerberos Clients” on page 405 for information about NTP.

On the new slave, start the KDC daemon (`krb5kdc`).

```
kdc3 # svcadm enable network/security/krb5kdc
```

How to Verify That the KDC Servers Are Synchronized

If incremental propagation has been configured, this procedure ensures that the information on the slave KDC has been updated.

Before You Begin

You must assume the `root` role. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.
1. On the KDC master server, run the \texttt{kproplog} command.
   \begin{verbatim}
   kdc1 # /usr/sbin/kproplog -h
   \end{verbatim}

2. On a KDC slave server, run the \texttt{kproplog} command.
   \begin{verbatim}
   kdc2 # /usr/sbin/kproplog -h
   \end{verbatim}

3. Check that the last serial # and the last timestamp values match.

\textbf{Example 21–21 Verifying That the KDC Servers Are Synchronized}

The following is a sample of results from running the \texttt{kproplog} command on the master KDC server.

\begin{verbatim}
# Kerberos update log (/var/krb5/principal.ulog)
Update log dump:
  Log version #: 1
  Log state: Stable
  Entry block size: 2048
  Number of entries: 2500
  First serial #: 137966
  Last serial #: 140465
  First time stamp: Fri Nov 28 00:59:27 2004
  Last time stamp: Fri Nov 28 01:06:13 2004
\end{verbatim}

The following is a sample of results from running the \texttt{kproplog} command on a slave KDC server.

\begin{verbatim}
# Kerberos update log (/var/krb5/principal.ulog)
Update log dump:
  Log version #: 1
  Log state: Stable
  Entry block size: 2048
  Number of entries: 0
  First serial #: None
  Last serial #: 140465
  First time stamp: None
  Last time stamp: Fri Nov 28 01:06:13 2004
\end{verbatim}

Notice that the values for the last serial number and the last timestamp are identical, which indicates that the slave is synchronized with the master KDC server.

In the slave KDC server output, notice that no update entries exist in the slave KDC server’s update log. No entries exist because the slave KDC server does not keep a set of updates, unlike the master KDC server. Also, the KDC slave server does not include information on the first serial number or the first timestamp because this is not relevant information.
How to Manually Propagate the Kerberos Database to the Slave KDCs

This procedure shows you how to propagate the Kerberos database by using the kprop command. Use this procedure if you need to synchronize a slave KDC with the master KDC outside the periodic cron job. Unlike the kprop_script, you can use kprop to propagate just the current database backup without first making a new backup of the Kerberos database.

Note – Do not use this procedure if you are using incremental propagation.

Before You Begin
You must assume the root role. For more information, see "How to Use Your Assigned Administrative Rights" on page 157.

1 (Optional) Back up the database by using the kdb5_util command.
   # /usr/sbin/kdb5_util dump /var/krb5/slave_datatrans

2 Propagate the database to a slave KDC by using the kprop command.
   # /usr/lib/krb5/kprop -f /var/krb5/slave_datatrans slave-KDC

Example 21–22 Manually Propagating the Kerberos Database to the Slave KDCs Using kprop_script
If you want to back up the database and propagate it to a slave KDC outside the periodic cron job, you can also use the kprop_script command as follows:

   # /usr/lib/krb5/kprop_script slave-KDC

Setting Up Parallel Propagation
In most cases, the master KDC is used exclusively to propagate its Kerberos database to the slave KDCs. However, if your site has many slave KDCs, you might consider load-sharing the propagation process, known as parallel propagation.

Note – Do not use this procedure if you are using incremental propagation.

Parallel propagation allows specific slave KDCs to share the propagation duties with the master KDC. This sharing of duties enables the propagation to be done faster and to lighten the work for the master KDC.
For example, say your site has one master KDC and six slave KDCs (shown in Figure 21–2), where slave-1 through slave-3 consist of one logical grouping and slave-4 through slave-6 consist of another logical grouping. To set up parallel propagation, you could have the master KDC propagate the database to slave-1 and slave-4. In turn, those KDC slaves could propagate the database to the KDC slaves in their group.

**Figure 21–2  Example of Parallel Propagation Configuration**

```
master

slave-1  Propagation Slaves  slave-4

slave-2  slave-3           slave-5  slave-6
```

**Configuration Steps for Setting Up Parallel Propagation**

The following is not a detailed step-by-step procedure, but a high-level list of configuration steps to enable parallel propagation. These steps involve the following:

1. On the master KDC, changing the kprop_script entry in its cron job to include arguments for only the KDC slaves that will perform the succeeding propagation (the propagation slaves).

2. On each propagation slave, adding a kprop_script entry to its cron job, which must include arguments for the slaves to propagate. To successfully propagate in parallel, the cron job should be set up to run after the propagation slave is itself propagated with the new Kerberos database.

   **Note** – How long it will take for a propagation slave to be propagated depends on factors such as network bandwidth and the size of the Kerberos database.

3. On each slave KDC, setting up the appropriate permissions to be propagated. This step is done by adding the host principal name of its propagating KDC to its kpropd.acl file.

**Example 21–23  Setting Up Parallel Propagation**

Using the example in Figure 21–2, the master KDC’s kprop_script entry would look similar to the following:
EXAMPLE 21–23  Setting Up Parallel Propagation  (Continued)

03*** /usr/lib/krb5/kprop_script slave-1.example.com slave-4.example.com

The slave-1’s kprop_script entry would look similar to the following:

04*** /usr/lib/krb5/kprop_script slave-2.example.com slave-3.example.com

Note that the propagation on the slave starts an hour after it is propagated by the master.

The kpropd.acl file on the propagation slaves would contain the following entry:

host/master.example.com@EXAMPLE.COM

The kpropd.acl file on the KDC slaves being propagated by slave-1 would contain the following entry:

host/slave-1.example.com@EXAMPLE.COM

Administering the Stash File

The stash file contains the master key for the Kerberos database, which is automatically created when you create a Kerberos database. If the stash file gets corrupted, you can use the stash command of the kdb5_util utility to replace the corrupted file. The only time you should need to remove a stash file is after removing the Kerberos database with the destroy command of kdb5_util. Because the stash file is not automatically removed with the database, you have to remove the stash file to finish the cleanup.

▼ How to Remove a Stash File

Before You Begin

You must assume the root role. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

- Remove the stash file.

  # rm stash-file

  Where stash-file is the path to the stash file. By default, the stash file is located at /var/krb5/.k5.realm.

  Note – If you need to re-create the stash file, you can use the -f option of the kdb5_util command.
How to Employ a New Master Key

Before You Begin

You must assume the root role. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1 Create a new master key.

This command adds a new, randomly generated master key. The -s option requests that the new master key be stored in the default keytab.

```
# kdb5_util add_mkey -s
```

Creating new master key for master key principal 'K/M@EXAMPLE.COM'
You will be prompted for a new database Master Password.
It is important that you NOT FORGET this password.
Enter KDC database master key:  
Re-enter KDC database master key to verify:  

2 Verify that the new master key exists.

```
# kdb5_util list_mkeys
```

Master keys for Principal: K/M@EXAMPLE.COM
KNVO: 2, Enctype: AES-128 CTS mode with 96-bit SHA-1 HMAC, No activate time set
KNVO: 1, Enctype: DES cbc mode with RSA-MD5, Active on: Wed Dec 31 18:00:00 CST 2001 *

The asterisk in this output identifies the currently active master key.

3 Set a time for the newly created master key to become active.

```
# date
Fri Jul 1 17:57:00 CDT 2011
# kdb5_util use_mkey 2 'now+2days'
# kdb5_util list_mkeys
```

Master keys for Principal: K/M@EXAMPLE.COM
KNVO: 2, Enctype: AES-128 CTS mode with 96-bit SHA-1 HMAC, Active on: Sun Jul 03 17:57:15 CDT 2011
KNVO: 1, Enctype: DES cbc mode with RSA-MD5, Active on: Wed Dec 31 18:00:00 CST 2001 *

In this example, the date is set to two days in the future to allow time for the new master key to propagate to all of the KDCs. Adjust the date as appropriate for your environment.

4 (Optional) After creating a new principal, verify that the new master key is being used.

```
# kadmin.local -q 'getprinc jimf' |egrep 'Principal|MKey'
```

Authenticating as principal root/admin@EXAMPLE.COM with password.
Principal: jimf@EXAMPLE.COM
MKey: vno 2

In this example, MKey: vno 2 indicates that the principal's secret key is protected by newly created master key, 2.
5 **Re-encrypt the user principal secret keys with the new master key.**

If you add a pattern argument to the end of the command, the principals that match the pattern will be updated. Add the `-n` option to this command syntax to identify which principals will be updated.

```bash
# kdb5_util update_princ_encryption -f -v
Principals whose keys WOULD BE re-encrypted to master key vno 2:
updating: host/kdc1.example.com@EXAMPLE.COM
updating: kadmin/changepw@EXAMPLE.COM
updating: kadmin/history@EXAMPLE.COM
updating: kdc/admin@EXAMPLE.COM
updating: host/kdc2.example.com@EXAMPLE.COM
6 principals processed: 5 updated, 1 already current
```

6 **Purge the old master key.**

After a master key is no longer used to protect any principal secret keys, it can be purged from the master key principal. This command will not purge the key if the key is still being used by any principals. Add the `-n` option to this command to verify that the correct master key will be purged.

```bash
# kdb5_util purge_mkeys -f -v
Purging the following master key(s) from K/M@EXAMPLE.COM:
KNVO: 1
1 key(s) purged.
```

7 **Verify that the old master key has been purged.**

```bash
# kdb5_util list_mkeys
Master keys for Principal: K/M@EXAMPLE.COM
KNVO: 2, Enctype: AES-128 CTS mode with 96-bit SHA-1 HMAC, Active on: Sun Jul 03 17:57:15 CDT 2011 *
```

8 **Update the stash file.**

```bash
# kdb5_util stash
Using existing stashed keys to update stash file.
```

9 **Verify that the stash file has been updated.**

```bash
# klist -kt /var/krb5/.k5.EXAMPLE.COM
Keytab name: FILE:.k5.EXAMPLE.COM
KVNO Timestamp Principal
-----------------------------------------------
2 05/07/2011 15:08 K/M@EXAMPLE.COM
```

**Managing a KDC on an LDAP Directory Server**

Most of the KDC administration tasks using an LDAP Directory Server are the same as those for the DB2 server. There are some new tasks that are specific to working with LDAP.
TABLE 21–3 Configuring KDC Servers to Use LDAP (Task Map)

<table>
<thead>
<tr>
<th>Task</th>
<th>Description</th>
<th>For Instructions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configure a master KDC.</td>
<td>Configures and builds the master KDC server and database for a realm using a manual process and using LDAP for the KDC.</td>
<td>“How to Configure a KDC to Use an LDAP Data Server” on page 363</td>
</tr>
<tr>
<td>Mix Kerberos principal attributes with non-Kerberos object class types.</td>
<td>Allows information stored with the Kerberos records to be shared with other LDAP databases.</td>
<td>“How to Mix Kerberos Principal Attributes in a Non-Kerberos Object Class Type” on page 428</td>
</tr>
<tr>
<td>Destroy a realm.</td>
<td>Removes all of the data associated with a realm.</td>
<td>“How to Destroy a Realm on an LDAP Directory Server” on page 429</td>
</tr>
</tbody>
</table>

▼ How to Mix Kerberos Principal Attributes in a Non-Kerberos Object Class Type

This procedure allows for Kerberos principal attributes to be associated with non-Kerberos object class types. In this procedure the krbprincipalaux, krbTicketPolicyAux and krbPrincipalName attributes are associated with the people object class.

In this procedure, the following configuration parameters are used:

- Directory Server = dsserver.example.com
- user principal = willf@EXAMPLE.COM

Before You Begin You must assume the root role. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1 Prepare each entry in the people object class.
   Repeat this step for each entry.
   ```
   cat << EOF | ldapmodify -h dsserver.example.com -D "cn=directory manager"  
   dn: uid=willf,ou=people,dc=example,dc=com  
   changetype: modify  
   objectClass: krbprincipalaux  
   objectClass: krbTicketPolicyAux  
   krbPrincipalName: willf@EXAMPLE.COM  
   EOF
   ```

2 Add a subtree attribute to the realm container.
   This step allows for searching of principal entries in the ou=people,dc=example,dc=com container, as well as in the default EXAMPLE.COM container.
   ```
   # kdb5_ldap_util -D "cn=directory manager" modify \
   -subtrees 'ou=people,dc=example,dc=com' -r EXAMPLE.COM
   ```
3  (Optional) If the KDC records are stored in DB2, migrate DB2 entries.
   a. Dump the DB2 entries.
      # kdb5_util dump > dumpfile
   b. Load the database into the LDAP server.
      # kdb5_util load -update dumpfile

4  (Optional) Add the principal attributes to the KDC.
   # kadmin.local -q 'addprinc willf'

▼ How to Destroy a Realm on an LDAP Directory Server

This procedure can be used if a different LDAP Directory Server has been configured to handle a realm.

Before You Begin
You must assume the root role. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

- Destroy the realm.
  # kdb5_ldap_util -D "cn=directory manager" destroy

Increasing Security on Kerberos Servers

Follow these steps to increase security on Kerberos application servers and on KDC servers.

TABLE 21–4  Increasing Security on Kerberos Servers (Task Map)

<table>
<thead>
<tr>
<th>Task</th>
<th>Description</th>
<th>For Instructions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Restrict access to the KDC servers.</td>
<td>Increases the security of the KDC servers and their data.</td>
<td>“How to Restrict Access to KDC Servers” on page 429</td>
</tr>
<tr>
<td>Increase password security by using a dictionary file.</td>
<td>Increases the security of any new passwords by checking the new password against a dictionary.</td>
<td>“How to Use a Dictionary File to Increase Password Security” on page 430</td>
</tr>
</tbody>
</table>

▼ How to Restrict Access to KDC Servers

Both master KDC servers and slave KDC servers have copies of the KDC database stored locally. Restricting access to these servers so that the databases are secure is important to the overall security of the Kerberos installation.
Increasing Security on Kerberos Servers

Before You Begin
You must assume the root role. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1. **Restrict access to the hardware that supports the KDC.**
   To restrict physical access, make sure that the KDC server and its monitor are located in a secure facility. Users should not be able to access this server in any way.

2. **Store KDC database backups on local disks or on the KDC slaves.**
   Make tape backups of your KDC only if the tapes are stored securely. Follow the same practice for copies of keytab files. It would be best to store these files on a local file system that is not shared with other systems. The storage file system can be on either the master KDC server or any of the slave KDCs.

### How to Use a Dictionary File to Increase Password Security

A dictionary file can be used by the Kerberos service to prevent words in the dictionary from being used as passwords when creating new credentials. Preventing the use of dictionary terms as passwords makes it harder for someone else to guess any password. By default the /var/krb5/kadm5.dict file is used, but it is empty.

**Before You Begin**
You must assume the root role. For more information, see "How to Use Your Assigned Administrative Rights" on page 157.

1. **Edit the KDC configuration file (kdc.conf).**
   You need add a line to instruct the service to use a dictionary file. In this example, the dictionary that is included with the spell utility is used. See the kdc.conf(4) man page for a full description of the configuration file.

   ```bash
   kdc1 # cat /etc/krb5/kdc.conf
   [kdcdefaults]
   kdc_ports = 88,750

   [realms]
   EXAMPLE.COM = {
     profile = /etc/krb5/krb5.conf
     database_name = /var/krb5/principal
     acl_file = /etc/krb5/kadm5.acl
     kadmind_port = 749
     max_life = 8h 0m 0s
     max_renewable_life = 7d 0h 0m 0s
     sunw dbprop_enable = true
     sunw dbprop_master ulogsize = 1000
     dict_file = /usr/share/lib/dict/words
   }
   ```

Oracle Solaris 11.1 Administration: Security Services • February 2014
2 Restart the Kerberos daemons.
   kdc1 # svcadm restart -r network/security/krb5kdc
   kdc1 # svcadm restart -r network/security/kadmin
This chapter provides resolutions for error messages that you might receive when you use the Kerberos service. This chapter also provides some troubleshooting tips for various problems. This is a list of the error message and troubleshooting information in this chapter.

- “SEAM Tool Error Messages” on page 433
- “Common Kerberos Error Messages (A-M)” on page 434
- “Common Kerberos Error Messages (N-Z)” on page 443
- “Problems With the Format of the krb5.conf File” on page 448
- “Problems Propagating the Kerberos Database” on page 448
- “Problems Mounting a Kerberized NFS File System” on page 449
- “Problems Authenticating as the root User” on page 449
- “Observing Mapping From GSS Credentials to UNIX Credentials” on page 450

Kerberos Error Messages

This section provides information about Kerberos error messages, including why each error occurs and a way to fix it.

SEAM Tool Error Messages

Unable to view the list of principals or policies; use the Name field.

**Cause:** The admin principal that you logged in with does not have the list privilege (l) in the Kerberos ACL file (kadm5.acl). So, you cannot view the principal list or policy list.

**Solution:** You must type the principal and policy names in the Name field to work on them, or you need to log in with a principal that has the appropriate privileges.
JNI: Java array creation failed
JNI: Java class lookup failed
JNI: Java field lookup failed
JNI: Java method lookup failed
JNI: Java object lookup failed
JNI: Java object field lookup failed
JNI: Java string access failed
JNI: Java string creation failed

**Cause:** A serious problem exists with the Java Native Interface that is used by the SEAM Tool (gkadmin).

**Solution:** Exit gkadmin and restart it. If the problem persists, please report a bug.

---

**Common Kerberos Error Messages (A-M)**

This section provides an alphabetical list (A-M) of common error messages for the Kerberos commands, Kerberos daemons, PAM framework, GSS interface, the NFS service, and the Kerberos library.

**All authentication systems disabled; connection refused**

**Cause:** This version of rlogind does not support any authentication mechanism.

**Solution:** Make sure that rlogind is invoked with the -k option.

**Another authentication mechanism must be used to access this host**

**Cause:** Authentication could not be done.

**Solution:** Make sure that the client is using Kerberos V5 mechanism for authentication.

**Authentication negotiation has failed, which is required for encryption. Good bye.**

**Cause:** Authentication could not be negotiated with the server.

**Solution:** Start authentication debugging by invoking the telnet command with the toggle authdebug command and look at the debug messages for further clues. Also, make sure that you have valid credentials.

**Bad krb5 admin server hostname while initializing kadmin interface**

**Cause:** An invalid host name is configured for admin_server in the krb5.conf file.

**Solution:** Make sure that the correct host name for the master KDC is specified on the admin_server line in the krb5.conf file.
Bad lifetime value  
   **Cause:** The lifetime value provided is not valid or incorrectly formatted.  
   **Solution:** Make sure that the value provided is consistent with the Time Formats section in the `kinit(1)` man page.

Bad start time value  
   **Cause:** The start time value provided is not valid or incorrectly formatted.  
   **Solution:** Make sure that the value provided is consistent with the Time Formats section in the `kinit(1)` man page.

Cannot contact any KDC for requested realm  
   **Cause:** No KDC responded in the requested realm.  
   **Solution:** Make sure that at least one KDC (either the master or a slave) is reachable or that the `krb5kdc` daemon is running on the KDCs. Check the `/etc/krb5/krb5.conf` file for the list of configured KDCs (kdc = kdc-name).

Cannot determine realm for host: host is `hostname`  
   **Cause:** Kerberos cannot determine the realm name for the host.  
   **Solution:** Make sure that there is a default realm name, or that the domain name mappings are set up in the Kerberos configuration file (krb5.conf).

Cannot find a kadmin KDC entry in krb5.conf(4) or DNS Service Location records for realm `realmname`  
Cannot find a kpassword KDC entry in krb5.conf(4) or DNS Service Location records for realm `realmname`  
Cannot find a master KDC entry in krb5.conf(4) or DNS Service Location records for realm `realmname`  
Cannot find any KDC entries in krb5.conf(4) or DNS Service Location records for realm `realmname`  
   **Cause:** Either the krb5.conf file or the DNS server record are incorrectly configured.  
   **Solution:** Make sure that the Kerberos configuration file (/etc/krb5/krb5.conf) or that the DNS server records for the KDC are configured properly.

Cannot find address for `hostname`: `error-string`  
   **Cause:** No address was found in the DNS records for the given hostname.  
   **Solution:** Fix the host record in DNS or correct the error in the DNS lookup process.

Cannot find KDC for requested realm  
   **Cause:** No KDC was found in the requested realm.
**Solution:** Make sure that the Kerberos configuration file (`krb5.conf`) specifies a KDC in the realm section.

**Cannot initialize realm realm-name**

**Cause:** The KDC might not have a stash file.

**Solution:** Make sure that the KDC has a stash file. If not, create a stash file by using the `kdb5_util` command, and try restarting the `krb5kdc` command.

**Cannot resolve KDC for requested realm**

**Cause:** Kerberos cannot determine any KDC for the realm.

**Solution:** Make sure that the Kerberos configuration file (`krb5.conf`) specifies a KDC in the realm section.

**Cannot resolve network address for KDCs 'hostname' discovered via DNS Service Location records for realm 'realm-name'**

**Cannot resolve network address for KDCs 'hostname' specified in krb5.conf(4) for realm 'realm-name'**

**Cause:** Either the `krb5.conf` file or the DNS server record is incorrectly configured.

**Solution:** Make sure that the Kerberos configuration file (`/etc/krb5/krb5.conf`) and the DNS server records for the KDC are configured properly.

**Cannot reuse password**

**Cause:** The password that you specified has been used before by this principal.

**Solution:** Choose a password that has not been chosen before, at least not within the number of passwords that are kept in the KDC database for each principal. This policy is enforced by the principal’s policy.

**Can’t get forwarded credentials**

**Cause:** Credential forwarding could not be established.

**Solution:** Make sure that the principal has forwardable credentials.

**Can’t open/find Kerberos configuration file**

**Cause:** The Kerberos configuration file (`krb5.conf`) was unavailable.

**Solution:** Make sure that the `krb5.conf` file is available in the correct location and has the correct permissions. This file should be writable by `root` and readable by everyone else.

**Client 'principal' not found in Kerberos database**

**Cause:** The principal is missing from the Kerberos database.

**Solution:** Add the client principal to the Kerberos database.
Client 'principal’ pre-authentication failed  
**Cause:** Authentication failed for the principal.  
**Solution:** Make sure that the user is using the correct password.

Client did not supply required checksum--connection rejected  
**Cause:** Authentication with checksum was not negotiated with the client. The client might be using an old Kerberos V5 protocol that does not support initial connection support.  
**Solution:** Make sure that the client is using a Kerberos V5 protocol that supports initial connection support.

Client/server realm mismatch in initial ticket request: 'client-principal' requesting ticket 'service-principal'  
**Cause:** A realm mismatch between the client and server occurred in the initial ticket request.  
**Solution:** Make sure that the server you are communicating with is in the same realm as the client, or that the realm configurations are correct.

Client or server has a null key  
**Cause:** The principal has a null key.  
**Solution:** Modify the principal to have a non-null key by using the `cpw` command of `kadmin`.

Clock skew too great: 'client’ requesting ticket 'service-principal’ from KDC 'KDC-hostname' (KDC-time). Skew is value  
Clock skew too great: 'client’ AP request with ticket for 'service-principal’. Skew is value (allowable value)  
**Cause:** The difference between the time reported on the client and the KDC server or application server is too large.  
**Solution:** Configure the Network Time Protocol (NTP) to keep the clocks synchronized. See "Synchronizing Clocks Between KDCs and Kerberos Clients” on page 405 for more information.

Communication failure with server while initializing kadmin interface  
**Cause:** The host that was specified for the admin server, also called the master KDC, did not have the `kadmind` daemon running.  
**Solution:** Make sure that you specified the correct host name for the master KDC. If you specified the correct host name, make sure that `kadmind` is running on the master KDC that you specified.

Credentials cache file permissions incorrect  
**Cause:** You do not have the appropriate read or write permissions on the credentials cache (/tmp/krb5cc_uid).
Solution: Make sure that you have read and write permissions on the credentials cache.

Credentials cache I/O operation failed XXX
Cause: Kerberos had a problem writing to the system's credentials cache (/tmp/krb5cc_uid).
Solution: Make sure that the credentials cache has not been removed, and that there is space left on the device by using the df command.

Decrypt integrity check failed
Cause: You might have an invalid ticket.
Solution: Verify both of these conditions:
- Make sure that your credentials are valid. Destroy your tickets with kdestroy, and create new tickets with kinit.
- Make sure that the target host has a keytab file with the correct version of the service key. Use kadmin to view the key version number of the service principal (for example, host/FQDN-hostname) in the Kerberos database. Also, use klist -k on the target host to make sure that it has the same key version number.

Decrypt integrity check failed for client 'principal' and server 'hostname'
Cause: You might have an invalid ticket.
Solution: Make sure that your credentials are valid. Destroy your tickets with the kdestroy command, and create new tickets with the kinit command.

Encryption could not be enabled. Goodbye.
Cause: Encryption could not be negotiated with the server.
Solution: Start authentication debugging by invoking the telnet command with the toggle encdebug command and look at the debug messages for further clues.

Failed to find realm for principal in keytab
Cause: The realm name included in the principal does not match the realm name in the principal stored in the keytab file.
Solution: Make sure that the principals are using the correct realm.

failed to obtain credentials cache
Cause: During kadmin initialization, a failure occurred when kadmin tried to obtain credentials for the admin principal.
Solution: Make sure that you used the correct principal and password when you executed kadmin.
Field is too long for this implementation
  
  **Cause:** The message size that was being sent by a Kerberized application was too long. This error could be generated if the transport protocol is UDP, which has a default maximum message size 65535 bytes. In addition, there are limits on individual fields within a protocol message that is sent by the Kerberos service.

  **Solution:** Verify that you have not restricted the transport to UDP in the KDC server’s `/etc/krb5/kdc.conf` file.

GSS-API (or Kerberos) error

  **Cause:** This message is a generic GSS-API or Kerberos error message and can be caused by several different problems.

  **Solution:** Check the `/var/krb5/kdc.log` file to find the more specific error message that was logged when this error occurred.

Hostname cannot be canonicalized for `hostname`: `error-string`

  **Cause:** The Kerberos client cannot find the fully qualified host name for the server.

  **Solution:** Make sure that the server host name is defined in DNS and that the hostname-to-address and address-to-hostname mappings are consistent.

Illegal cross-realm ticket

  **Cause:** The ticket sent did not have the correct cross-realms. The realms might not have the correct trust relationships set up.

  **Solution:** Make sure that the realms you are using have the correct trust relationships.

Improper format of Kerberos configuration file

  **Cause:** The Kerberos configuration file has invalid entries.

  **Solution:** Make sure that all the relations in the `krb5.conf` file are followed by the “=” sign and a value. Also, verify that the brackets are present in pairs for each subsection.

Inappropriate type of checksum in message

  **Cause:** The message contained an invalid checksum type.

  **Solution:** Check which valid checksum types are specified in the `krb5.conf` and `kdc.conf` files.

Incorrect net address

  **Cause:** There was a mismatch in the network address. The network address in the ticket that was being forwarded was different from the network address where the ticket was processed. This message might occur when tickets are being forwarded.
Kerberos Error Messages

Solution: Make sure that the network addresses are correct. Destroy your tickets with kdestroy, and create new tickets with kinit.

Invalid credential was supplied
Service key not available
  Cause: The service ticket in the credentials cache may be incorrect.
  Solution: Destroy current credential cache and rerun kinit before trying to use this service.

Invalid flag for file lock mode
  Cause: An internal Kerberos error occurred.
  Solution: Please report a bug.

Invalid message type specified for encoding
  Cause: Kerberos could not recognize the message type that was sent by the Kerberized application.
  Solution: If you are using a Kerberized application that was developed by your site or a vendor, make sure that it is using Kerberos correctly.

Invalid number of character classes
  Cause: The password that you specified for the principal does not contain enough password classes, as enforced by the principal's policy.
  Solution: Make sure that you specify a password with the minimum number of password classes that the policy requires.

KADM err: Memory allocation failure
  Cause: There is insufficient memory to run kadmin.
  Solution: Free up memory and try running kadmin again.

kadmin: Bad encryption type while changing host/FQDN's key
  Cause: More default encryption types are included in the base release in newer releases. Clients can request encryption types that might not be supported by a KDC running an older version of the software.
  Solution: Several solutions exist to fix this problem. The easiest one to implement is listed first:

1. Add the SUNWcry and SUNWcryr packages to the KDC server. This increases the number of encryption types supported by the KDC.
2. Set permitted_enctypes in krb5.conf on the client to not include the aes256 encryption type. This step will need to be done on each new client.
KDC can’t fulfill requested option

**Cause:** The KDC did not allow the requested option. A possible problem might be that postdating or forwardable options were being requested, and the KDC did not allow them. Another problem might be that you requested the renewal of a TGT, but you didn’t have a renewable TGT.

**Solution:** Determine if you are either requesting an option that the KDC does not allow or a type of ticket that is not available.

KDC policy rejects request

**Cause:** The KDC policy did not allow the request. For example, the request to the KDC did not have an IP address in its request. Or forwarding was requested, but the KDC did not allow it.

**Solution:** Make sure that you are using `kinit` with the correct options. If necessary, modify the policy that is associated with the principal or change the principal’s attributes to allow the request. You can modify the policy or principal by using `kadmin`.

KDC reply did not match expectation: KDC not found. Probably got an unexpected realm referral

**Cause:** The KDC reply did not contain the expected principal name, or other values in the response were incorrect.

**Solution:** Make sure that the KDC you are communicating with complies with RFC4120, that the request you are sending is a Kerberos V5 request, and that the KDC is available.

kdestroy: Could not obtain principal name from cache

**Cause:** The credentials cache is missing or corrupted.

**Solution:** Check that the cache location provided is correct. Remove and obtain a new TGT by using `kinit`, if necessary.

kdestroy: No credentials cache file found while destroying cache

**Cause:** The credentials cache (/tmp/krb5c_uid) is missing or corrupted.

**Solution:** Check that the cache location provided is correct. Remove and obtain a new TGT using `kinit`, if necessary.

kdestroy: TGT expire warning NOT deleted

**Cause:** The credentials cache is missing or corrupted.

**Solution:** Check that the cache location provided is correct. Remove and obtain a new TGT using `kinit`, if necessary.

Kerberos authentication failed

**Cause:** The Kerberos password is either incorrect or the password might not be synchronized with the UNIX password.
Solution: If the password are not synchronized, then you must specify a different password to complete Kerberos authentication. It is possible that the user has forgotten their original password.

Kerberos V5 refuses authentication
Solution: Start authentication debugging by invoking the telnet command with the toggle authdebug command and look at the debug messages for further clues. Also, make sure that you have valid credentials.

Key table entry not found
Solution: Add the appropriate service principal to the server's keytab file so that it can provide the Kerberized service.

Key table file 'filename' not found
Solution: Create the key table file.

Key version number is not available for principal principal
Solution: Check the version of the keys on the application server using the klist -k command.

Key version number for principal in key table is incorrect
Solution: If a service's key has been changed (for example, by using kadmin), you need to extract the new key and store it in the host's keytab file where the service is running.

Alternately, you might be using an old service ticket that has an older key. You might want to run the kdestroy command and then the kinit command again.

kinit: gethostname failed
Solution: Make sure that the host is configured correctly.

login: load_modules: can not open module /usr/lib/security/pam_krb5.so.1
Solution: Either the Kerberos PAM module is missing or it is not a valid executable binary.
**Solution:** Make sure that the Kerberos PAM module is in the /usr/lib/security directory and that it is a valid executable binary. Also, make sure that the /etc/pam.conf file contains the correct path to pam_krb5.so.1.

Looping detected getting initial creds: 'client-principal' requesting ticket 'service-principal'. Max loops is value. Make sure a KDC is available.

**Cause:** Kerberos made several attempts to get the initial tickets but failed.

**Solution:** Make sure that at least one KDC is responding to authentication requests.

Master key does not match database

**Cause:** The loaded database dump was not created from a database that contains the master key. The master key is located in /var/krb5/.k5.REALM.

**Solution:** Make sure that the master key in the loaded database dump matches the master key that is located in /var/krb5/.k5.REALM.

Matching credential not found

**Cause:** The matching credential for your request was not found. Your request requires credentials that are unavailable in the credentials cache.

**Solution:** Destroy your tickets with kdestroy, and create new tickets with kinit.

Message out of order

**Cause:** Messages that were sent using sequential-order privacy arrived out of order. Some messages might have been lost in transit.

**Solution:** You should reinitialize the Kerberos session.

Message stream modified

**Cause:** There was a mismatch between the computed checksum and the message checksum. The message might have been modified while in transit, which can indicate a security leak.

**Solution:** Make sure that the messages are being sent across the network correctly. Because this message can also indicate the possible tampering of messages while they are being sent, destroy your tickets using kdestroy and reinitialize the Kerberos services that you are using.

---

**Common Kerberos Error Messages (N-Z)**

This section provides an alphabetical list (N-Z) of common error messages for the Kerberos commands, Kerberos daemons, PAM framework, GSS interface, the NFS service, and the Kerberos library.

**No credentials cache file found**

**Cause:** Kerberos could not find the credentials cache (/tmp/krb5cc_uid).
Solution: Make sure that the credential file exists and is readable. If it isn’t, try performing kinit again.

No credentials were supplied, or the credentials were unavailable or inaccessible

No credential cache found
   Cause: The user’s credential cache is incorrect or does not exist.
   Solution: The user should run kinit before trying to start the service.

No credentials were supplied, or the credentials were unavailable or inaccessible

No principal in keytab (‘filename’) matches desired name principal
   Cause: An error occurred during an attempt to authenticate the server.
   Solution: Make sure that the host or service principal is in the server’s keytab file.

Operation requires “privilege” privilege
   Cause: The admin principal that was being used does not have the appropriate privilege configured in the kadm5.acl file.
   Solution: Use a principal that has the appropriate privileges. Or, configure the principal that was being used to have the appropriate privileges by modifying the kadm5.acl file. Usually, a principal with /admin as part of its name has the appropriate privileges.

PAM-KRB5 (auth): krb5_verify_init_creds failed: Key table entry not found
   Cause: The remote application tried to read the host’s service principal in the local /etc/krb5/krb5.keytab file, but one does not exist.
   Solution: Add the host’s service principal to the host’s keytab file.

Password is in the password dictionary
   Cause: The password that you specified is in a password dictionary that is being used. Your password is not a good choice for a password.
   Solution: Choose a password that has a mix of password classes.

Permission denied in replay cache code
   Cause: The system’s replay cache could not be opened. Your server might have been first run under a user ID different than your current user ID.
   Solution: Make sure that the replay cache has the appropriate permissions. The replay cache is stored on the host where the Kerberized server application is running. The replay cache file is called /var/krb5/rcache/rc_service_name_uid for non-root users. For root users the replay cache file is called /var/krb5/rcache/root/rc_service_name.
Protocol version mismatch

**Cause:** Most likely, a Kerberos V4 request was sent to the KDC. The Kerberos service supports only the Kerberos V5 protocol.

**Solution:** Make sure that your applications are using the Kerberos V5 protocol.

Request is a replay

**Cause:** The request has already been sent to this server and processed. The tickets might have been stolen, and someone else is trying to reuse the tickets.

**Solution:** Wait for a few minutes, and reissue the request.

Requested principal and ticket don’t match: Requested principal is `service-principal` and TGT principal is `TGT-principal`

**Cause:** The service principal that you are connecting to and the service ticket that you have do not match.

**Solution:** Make sure that DNS is functioning properly. If you are using another vendor’s software, make sure that the software is using principal names correctly.

Requested protocol version not supported

**Cause:** Most likely, a Kerberos V4 request was sent to the KDC. The Kerberos service supports only the Kerberos V5 protocol.

**Solution:** Make sure that your applications are using the Kerberos V5 protocol.

Service key service-principal not available

**Cause:** The named service principal is not in the keytab file on the application server.

**Solution:** Make sure that the service principal matches or is included in the keytab file on the application server.

Server refused to negotiate authentication, which is required for encryption. Good bye.

**Cause:** The remote application is not capable or has been configured not to accept Kerberos authentication from the client.

**Solution:** Provide a remote application that can negotiate authentication or configure the application to use the appropriate flags to turn on authentication.

Server refused to negotiate encryption. Good bye.

**Cause:** Encryption could not be negotiated with the server.

**Solution:** Start authentication debugging by invoking the `telnet` command with the `toggle encdebug` command and look at the debug messages for further clues.
Server rejected authentication (during sendauth exchange)

**Cause:** The server that you are trying to communicate with rejected the authentication. Most often, this error occurs during Kerberos database propagation. Some common causes might be problems with the `kpropd.acl` file, DNS, or the keytab file.

**Solution:** If you get this error when you are running applications other than `kprop`, investigate whether the server's keytab file is correct.

Server **service-principal** not found in Kerberos database

**Cause:** The service principal is not correct or is missing from the principal database.

**Solution:** Make sure that the service principal is correct and that it is in the database.

Target name principal 'principal' does not match **service-principal**

**Cause:** The service principal that is being used does not match the service principal that the application server is using.

**Solution:** On the application server, make sure that the service principal is included in the keytab file. For the client, make sure that the correct service principal is being used.

The ticket isn't for us

**Ticket/authenticator don’t match**

**Cause:** There was a mismatch between the ticket and the authenticator. The principal name in the request might not have matched the service principal’s name. Either because the ticket was being sent with an FQDN name of the principal while the service expected a non-FQDN name, or a non-FQDN name was sent when the service expected an FQDN name.

**Solution:** If you get this error when you are running applications other than `kprop`, investigate whether the server's keytab file is correct.

Ticket expired

**Cause:** Your ticket times have expired.

**Solution:** Destroy your tickets with `kdestroy`, and create new tickets with `kinit`.

Ticket is ineligible for postdating

**Cause:** The principal does not allow its tickets to be postdated.

**Solution:** Modify the principal with `kadmin` to allow postdating.

Ticket not yet valid: 'client-principal' requesting ticket 'service-principal' from 'kdc-hostname' (time). TGT start time is time.

**Cause:** The postdated ticket is not yet valid.

**Solution:** Create a new ticket with the correct date, or wait until the current ticket is valid.
Truncated input file detected
   **Cause:** The database dump file that was being used in the operation is not a complete dump file.

   **Solution:** Create the dump file again, or use a different database dump file.

Unable to securely authenticate user ... exit
   **Cause:** Authentication could not be negotiated with the server.

   **Solution:** Start authentication debugging by invoking the `telnet` command with the `toggle authdebug` command and look at the debug messages for further clues. Also, make sure that you have valid credentials.

Unknown encryption type: `name`
   **Cause:** The encryption type that is included with the credential cannot be used.

   **Solution:** Determine which encryption types the client is using with the `klist -e` command. Make sure that the application server supports at least one of the encryption types.

Wrong principal in request
   **Cause:** There was an invalid principal name in the ticket. This error might indicate a DNS or FQDN problem.

   **Solution:** Make sure that the principal of the service matches the principal in the ticket.

**Kerberos Troubleshooting**

This section provides troubleshooting information for the Kerberos software.

### How to Identify Problems With Key Version Numbers

Sometimes, the key version number (KVNO) used by the KDC and the service principal keys stored in `/etc/krb5/krb5.keytab` for services hosted on the system do not match. The KVNO can get out of synchronization when a new set of keys are created on the KDC without updating the keytable file with the new keys. This problem can be diagnosed by using the following procedure.

1. **List the keytab entries.**

   Note that the KVNO for each principal is included in the list.

   ```bash
   # klist -k
   Keytab name: FILE:/etc/krb5/krb5.keytab
   KVNO Principal -----------------------------------------------
   ```

   ```bash
   ```
Acquire an initial credential by using the host key.

```
# kinit -k
```

Determine the KVNO that is used by the KDC.

```
# kvno nfs/denver.example.com
nfs/denver.example.com@EXAMPLE.COM: kvno = 3
```

Note that the KVNO listed here is 3 instead of 2.

**Problems With the Format of the `krb5.conf` File**

If the `krb5.conf` file is not formatted properly, then the following error message might be displayed in a terminal window or recorded in the log file:

*Improper format of Kerberos configuration file while initializing krb5 library*

If there is a problem with the format of the `krb5.conf` file, then the associated services could be vulnerable to attack. You must fix the problem before you allow Kerberos features to be used.

**Problems Propagating the Kerberos Database**

If propagating the Kerberos database fails, try `/usr/bin/rlogin -x` between the slave KDC and master KDC, and from the master KDC to the slave KDC server.

If the KDCs have been set up to restrict access, `rlogin` is disabled and cannot be used to troubleshoot this problem. To enable `rlogin` on a KDC, you must enable the `eklogin` service.

```
# svcadm enable svc:/network/login:eklogin
```

After you finish troubleshooting the problem, you need to disable the `eklogin` service.

If `rlogin` does not work, problems are likely because of the keytab files on the KDCs. If `rlogin` does work, the problem is not in the keytab file or the name service, because `rlogin` and the propagation software use the same `host/host-name` principal. In this case, make sure that the `kpropd.acl` file is correct.
Problems Mounting a Kerberized NFS File System

- If mounting a Kerberized NFS file system fails, make sure that the `/var/rcache/root` file exists on the NFS server. If the file system is not owned by `root`, remove it and try the mount again.
- If you have a problem accessing a Kerberized NFS file system, make sure that the `gssd` service is enabled on your system and the NFS server.
- If you see either the `invalid argument` or `bad directory` error message when you are trying to access a Kerberized NFS file system, the problem might be that you are not using a fully qualified DNS name when you are trying to mount the NFS file system. The host that is being mounted is not the same as the host name part of the service principal in the server’s keytab file.

This problem might also occur if your server has multiple Ethernet interfaces, and you have set up DNS to use a “name per interface” scheme instead of a “multiple address records per host” scheme. For the Kerberos service, you should set up multiple address records per host as follows:\footnote{Ken Hornstein, “Kerberos FAQ,” [http://www.cmf.nrl.navy.mil/CCS/people/kenh/kerberos-faq.html#kerbdns](http://www.cmf.nrl.navy.mil/CCS/people/kenh/kerberos-faq.html#kerbdns), accessed 10 March 2010.}

```plaintext
my.host.name. A 1.2.3.4
    A 1.2.4.4
    A 1.2.5.4
my-en0.host.name. A 1.2.3.4
my-en1.host.name. A 1.2.4.4
my-en2.host.name. A 1.2.5.4
4.3.2.1 Pointer my.host.name.
4.4.2.1 Pointer my.host.name.
4.5.2.1 Pointer my.host.name.
```

In this example, the setup allows one reference to the different interfaces and a single service principal instead of three service principals in the server’s keytab file.

Problems Authenticating as the root User

If authentication fails when you try to become superuser on your system and you have already added the `root` principal to your host’s keytab file, there are two potential problems to check. First, make sure that the root principal in the keytab file has a fully qualified host name as its instance. If it does, check the `/etc/resolv.conf` file to make sure that the system is correctly set up as a DNS client.
Observing Mapping From GSS Credentials to UNIX Credentials

To be able to monitor the credential mappings, first uncomment this line from the /etc/gss/gsscred.conf file.

SYSLOG_UID_MAPPING=yes

Next instruct the gssd service to get information from the /etc/gss/gsscred.conf file.

# pkill -HUP gssd

Now you should be able to monitor the credential mappings as gssd requests them. The mappings are recorded by syslogd, if the syslog.conf file is configured for the auth system facility with the debug severity level.

Using DTrace With the Kerberos Service

The Kerberos mechanism supports various DTrace probes for decoding various protocol messages. The probes include KRB_AP_*, KRB_KDC_*, KRB_CRED, KRB_ERROR, KRB_PRIV, KRB_SAFE, and general messaging information. This has the distinct advantage over other protocol inspectors, by allowing the privileged user to easily look at unencrypted Kerberos and application data.

The following example shows what pre-authentication is chosen by the client. The first step is to create a DTrace script, like the following:

```
cat krbtrace.d
kerberos$target:::krb_message-recv
{  printf("<- krb message recved: %s\n", args[0]->krb_message_type);
    printf("<- krb message remote addr: %s\n", args[1]->kconn_remote);
    printf("<- krb message ports: local %d remote %d\n",
           args[1]->kconn_localport, args[1]->kconn_remoteport);
    printf("<- krb message protocol: %s transport: %s\n",
           args[1]->kconn_protocol, args[1]->kconn_type);
}

kerberos$target:::krb_message-send
{  printf("-> krb message sent: %s\n", args[0]->krb_message_type);
    printf("-> krb message remote addr: %s\n", args[1]->kconn_remote);
    printf("-> krb message ports: local %d remote %d\n",
           args[1]->kconn_localport, args[1]->kconn_remoteport);
    printf("-> krb message protocol: %s transport: %s\n",
           args[1]->kconn_protocol, args[1]->kconn_type);
    printf("\n");
}
```
Next, execute the `krbtrace.d` script as a privileged user on the Kerberos system by typing the following command:

```
# LD_BIND_NOW=1 dtrace -qs krbtrace.d -c "kinit -k"
```

The pre-authentication types are displayed in the output. For more information about the various pre-authentication types see RFC 4120.
Administering Kerberos Principals and Policies (Tasks)

This chapter provides procedures for administering principals and the policies that are associated with them. This chapter also shows how to administer a host’s keytab file.

This chapter should be used by anyone who needs to administer principals and policies. Before you use this chapter, you should be familiar with principals and policies, including any planning considerations. Refer to Chapter 19, “Introduction to the Kerberos Service,” and Chapter 20, “Planning for the Kerberos Service,” respectively.

This is a list of the information in this chapter.

- “Ways to Administer Kerberos Principals and Policies” on page 453
- “SEAM Tool” on page 454
- “Administering Kerberos Principals” on page 458
- “Administering Kerberos Policies” on page 471
- “SEAM Tool Reference” on page 479
- “Administering Keytab Files” on page 483

Ways to Administer Kerberos Principals and Policies

The Kerberos database on the master KDC contains all of your realm’s Kerberos principals, their passwords, policies, and other administrative information. To create and delete principals, and to modify their attributes, you can use either the kadmin or gkadmin command.

The kadmin command provides an interactive command-line interface that enables you to maintain Kerberos principals, policies, and keytab files. There are two versions of the kadmin command:

- kadmin – Uses Kerberos authentication to operate securely from anywhere on the network
- kadmin.local – Must be run directly on the master KDC
Other than kadmin using Kerberos to authenticate the user, the capabilities of the two versions are identical. The local version is necessary to enable you to set up enough of the database so that you can use the remote version.

Also, the Oracle Solaris release provides the SEAM Tool, gkadmin, which is an interactive graphical user interface (GUI) that provides essentially the same capabilities as the kadmin command. See “SEAM Tool” on page 454 for more information.

SEAM Tool

The SEAM Tool (gkadmin) is an interactive graphical user interface (GUI) that enables you to maintain Kerberos principals and policies. This tool provides much the same capabilities as the kadmin command. However, this tool does not support the management of keytab files. You must use the kadmin command to administer keytab files, which is described in “Administering Keytab Files” on page 483.

Similar to the kadmin command, the SEAM Tool uses Kerberos authentication and encrypted RPC to operate securely from anywhere on the network. The SEAM Tool enables you to do the following:

- Create new principals that are based on default values or existing principals.
- Create new policies that are based on existing policies.
- Add comments for principals.
- Set up default values for creating new principals.
- Log in as another principal without exiting the tool.
- Print or save principal lists and policy lists.
- View and search principal lists and policy lists.

The SEAM Tool also provides context-sensitive help and general online help.

The following task maps provide pointers to the various tasks that you can do with the SEAM Tool:

- “Administering Kerberos Principals (Task Map)” on page 458
- “Administering Kerberos Policies (Task Map)” on page 471

Also, go to “SEAM Tool Panel Descriptions” on page 479 for descriptions of all the principal attributes and policy attributes that you can either specify or view in the SEAM Tool.

Command-Line Equivalents of the SEAM Tool

This section lists the kadmin commands that provide the same capabilities as the SEAM Tool. These commands can be used without running an X Window system. Even though most procedures in this chapter use the SEAM Tool, many procedures also provide corresponding examples that use the command-line equivalents.
### Command-Line Equivalents of the SEAM Tool

<table>
<thead>
<tr>
<th>SEAM Tool Procedure</th>
<th>Equivalent <code>kadmin</code> Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>View the list of principals.</td>
<td><code>list_principals</code> or <code>get_principals</code></td>
</tr>
<tr>
<td>View a principal’s attributes.</td>
<td><code>get_principal</code></td>
</tr>
<tr>
<td>Create a new principal.</td>
<td><code>add_principal</code></td>
</tr>
<tr>
<td>Duplicate a principal.</td>
<td>No command-line equivalent</td>
</tr>
<tr>
<td>Modify a principal.</td>
<td><code>modify_principal</code> or <code>change_password</code></td>
</tr>
<tr>
<td>Delete a principal.</td>
<td><code>delete_principal</code></td>
</tr>
<tr>
<td>Set up defaults for creating new principals.</td>
<td>No command-line equivalent</td>
</tr>
<tr>
<td>View the list of policies.</td>
<td><code>list_policies</code> or <code>get_policies</code></td>
</tr>
<tr>
<td>View a policy’s attributes.</td>
<td><code>get_policy</code></td>
</tr>
<tr>
<td>Create a new policy.</td>
<td><code>add_policy</code></td>
</tr>
<tr>
<td>Duplicate a policy.</td>
<td>No command-line equivalent</td>
</tr>
<tr>
<td>Modify a policy.</td>
<td><code>modify_policy</code></td>
</tr>
<tr>
<td>Delete a policy.</td>
<td><code>delete_policy</code></td>
</tr>
</tbody>
</table>

### The Only File Modified by the SEAM Tool

The only file that the SEAM Tool modifies is the `$HOME/.gkadmin` file. This file contains the default values for creating new principals. You can update this file by choosing Properties from the Edit menu.

### Print and Online Help Features of the SEAM Tool

The SEAM Tool provides both print features and online help features. From the Print menu, you can send the following to a printer or a file:

- List of available principals on the specified master KDC
- List of available policies on the specified master KDC
- The currently selected principal or the loaded principal
- The currently selected policy or the loaded policy

From the Help menu, you can access context-sensitive help and general help. When you choose Context-Sensitive Help from the Help menu, the Context-Sensitive Help window is displayed and the tool is switched to help mode. In help mode, when you click on any fields, labels, or buttons on the window, help on that item is displayed in the Help window. To switch back to the tool’s normal mode, click Dismiss in the Help window.
You can also choose Help Contents, which opens an HTML browser that provides pointers to the general overview and task information that is provided in this chapter.

**Working With Large Lists in the SEAM Tool**

As your site starts to accumulate a large number of principals and policies, the time it takes the SEAM Tool to load and display the principal and policy lists will become increasingly longer. Thus, your productivity with the tool will decrease. There are several ways to work around this problem.

First, you can completely eliminate the time to load the lists by not having the SEAM Tool load the lists. You can set this option by choosing Properties from the Edit menu, and unchecking the Show Lists field. Of course, when the tool doesn’t load the lists, it can’t display the lists, and you can no longer use the list panels to select principals or policies. Instead, you must type a principal or policy name in the new Name field that is provided, then select the operation that you want to perform on it. In effect, typing a name is equivalent to selecting an item from the list.

Another way to work with large lists is to cache them. In fact, caching the lists for a limited time is set as the default behavior for the SEAM Tool. The SEAM Tool must still initially load the lists into the cache. But after that, the tool can use the cache rather than retrieve the lists again. This option eliminates the need to keep loading the lists from the server, which is what takes so long.

You can set list caching by choosing Properties from the Edit menu. There are two cache settings. You can choose to cache the list forever, or you can specify a time limit when the tool must reload the lists from the server into the cache.

Caching the lists still enables you to use the list panels to select principals and policies, so it doesn’t affect how you use the SEAM Tool as the first option does. Also, even though caching doesn’t enable you to see the changes of other users, you can still see the latest list information based on your changes, because your changes update the lists both on the server and in the cache. And, if you want to update the cache to see other changes and get the lastest copy of the lists, you can use the Refresh menu whenever you want to refresh the cache from the server.
How to Start the SEAM Tool

1 Start the SEAM Tool by using the gkadmin command.

```
$ /usr/sbin/gkadmin
```

The SEAM Administration Login window is displayed.

2 If you don’t want to use the default values, specify new default values.

The window automatically fills in with default values. The default principal name is determined by taking your current identity from the USER environment variable and appending /admin to it (username/admin). The default Realm and Master KDC fields are selected from the /etc/krb5/krb5.conf file. If you ever want to retrieve the default values, click Start Over.

Note – The administration operations that each Principal Name can perform are dictated by the Kerberos ACL file, /etc/krb5/kadm5.acl. For information about limited privileges, see “Using the SEAM Tool With Limited Kerberos Administration Privileges” on page 482.

3 Type a password for the specified principal name.

4 Click OK.

A window showing all of the principals is displayed.
Administering Kerberos Principals

This section provides the step-by-step instructions used to administer principals with the SEAM Tool. This section also provides examples of command-line equivalents, when available.

### Administering Kerberos Principals (Task Map)

<table>
<thead>
<tr>
<th>Task</th>
<th>Description</th>
<th>For Instructions</th>
</tr>
</thead>
<tbody>
<tr>
<td>View the list of principals.</td>
<td>View the list of principals by clicking the Principals tab.</td>
<td>&quot;How to View the List of Kerberos Principals&quot; on page 459</td>
</tr>
<tr>
<td>View a principal’s attributes.</td>
<td>View a principal’s attributes by selecting the Principal in the Principal List, then clicking the Modify button.</td>
<td>&quot;How to View a Kerberos Principal’s Attributes&quot; on page 461</td>
</tr>
<tr>
<td>Create a new principal.</td>
<td>Create a new principal by clicking the Create New button in the Principal List panel.</td>
<td>&quot;How to Create a New Kerberos Principal&quot; on page 463</td>
</tr>
<tr>
<td>Duplicate a principal.</td>
<td>Duplicate a principal by selecting the principal to duplicate in the Principal List, then clicking the Duplicate button.</td>
<td>&quot;How to Duplicate a Kerberos Principal&quot; on page 466</td>
</tr>
<tr>
<td>Modify a principal.</td>
<td>Modify a principal by selecting the principal to modify in the Principal List, then clicking the Modify button. Note that you cannot modify a principal’s name. To rename a principal, you must duplicate the principal, specify a new name for it, save it, and then delete the old principal.</td>
<td>&quot;How to Modify a Kerberos Principal&quot; on page 466</td>
</tr>
<tr>
<td>Delete a principal.</td>
<td>Delete a principal by selecting the principal to delete in the Principal List, then clicking the Delete button.</td>
<td>&quot;How to Delete a Kerberos Principal&quot; on page 467</td>
</tr>
<tr>
<td>Set up defaults for creating new principals.</td>
<td>Set up defaults for creating new principals by choosing Properties from the Edit menu.</td>
<td>&quot;How to Set Up Defaults for Creating New Kerberos Principals&quot; on page 468</td>
</tr>
<tr>
<td>Modify the Kerberos administration privileges (kadm5.acl file).</td>
<td>Command-line only. The Kerberos administration privileges determine what operations a principal can perform on the Kerberos database, such as add and modify. You need to edit the /etc/krb5/kadm5.acl file to modify the Kerberos administration privileges for each principal.</td>
<td>&quot;How to Modify the Kerberos Administration Privileges&quot; on page 469</td>
</tr>
</tbody>
</table>
Automating the Creation of New Kerberos Principals

Even though the SEAM Tool provides ease-of-use, it doesn't provide a way to automate the creation of new principals. Automation is especially useful if you need to add 10 or even 100 new principals in a short time. However, by using the `kadmin.local` command in a Bourne shell script, you can do just that.

The following shell script line is an example of how to automate the creation of new principals:

```bash
awk '{ print "ank +needchange -pw", $2, $1 }' < /tmp/princnames | time /usr/sbin/kadmin.local> /dev/null
```

This example is split over two lines for readability. The script reads in a file called `princnames` that contains principal names and their passwords, and adds them to the Kerberos database. You would have to create the `princnames` file, which contains a principal name and its password on each line, separated by one or more spaces. The `+needchange` option configures the principal so that the user is prompted for a new password during login with the principal for the first time. This practice helps to ensure that the passwords in the `princnames` file are not a security risk.

You can build more elaborate scripts. For example, your script could use the information in the name service to obtain the list of user names for the principal names. What you do and how you do it is determined by your site's needs and your scripting expertise.

How to View the List of Kerberos Principals

An example of the command-line equivalent follows this procedure.

1. **If necessary, start the SEAM Tool.**
   See “How to Start the SEAM Tool” on page 457 for more information.

   ```bash
   $ /usr/sbin/gkadmin
   ```
2 Click the Principals tab.

The list of principals is displayed.

3 Display a specific principal or a sublist of principals.

Type a filter string in the Filter field, and press Return. If the filter succeeds, the list of principals that match the filter is displayed.

The filter string must consist of one or more characters. Because the filter mechanism is case sensitive, you need to use the appropriate uppercase and lowercase letters for the filter. For example, if you type the filter string ge, the filter mechanism displays only the principals with the ge string in them (for example, george or edge).
If you want to display the entire list of principals, click Clear Filter.

Example 23–1  Viewing the List of Kerberos Principals (Command Line)

In the following example, the list_principals command of kadmin is used to list all the principals that match kadmin*. Wildcards can be used with the list_principals command.

```
Kadmin: list_principals kadmin*
Kadmin/changepw@EXAMPLE.COM
Kadmin/kdcl.example.con@EXAMPLE.COM
Kadmin/history@EXAMPLE.COM
Kadmin: quit
```

▼ How to View a Kerberos Principal's Attributes

An example of the command-line equivalent follows this procedure.

1  **If necessary, start the SEAM Tool.**
   See “How to Start the SEAM Tool” on page 457 for more information.

   ```
   $ /usr/sbin/gkadmin
   ```

2  **Click the Principals tab.**

3  **Select the principal in the list that you want to view, then click Modify.**
   The Principal Basics panel that contains some of the principal's attributes is displayed.

4  **Continue to click Next to view all the principal's attributes.**
   Three windows contain attribute information. Choose Context-Sensitive Help from the Help menu to get information about the various attributes in each window. Or, for all the principal attribute descriptions, go to “SEAM Tool Panel Descriptions” on page 479.

5  **When you are finished viewing, click Cancel.**

Example 23–2  Viewing a Kerberos Principal's Attributes

The following example shows the first window when you are viewing the jdb/admin principal.
Viewing a Kerberos Principal's Attributes (Command Line)

In the following example, the `get_principal` command of `kadmin` is used to view the attributes of the `jdb/admin` principal.

```
kadmin: getprinc jdb/admin
Principal: jdb/admin@EXAMPLE.COM
Expiration date: [never]
Last password change: [never]
Password expiration date: Wed Apr 14 11:53:10 PDT 2011
Maximum ticket life: 1 day 16:00:00
Maximum renewable life: 1 day 16:00:00
```
How to Create a New Kerberos Principal

An example of the command-line equivalent follows this procedure.

1 If necessary, start the SEAM Tool.
   See “How to Start the SEAM Tool” on page 457 for more information.

   Note – If you are creating a new principal that might need a new policy, you should create the new policy before you create the new principal. Go to “How to Create a New Kerberos Policy” on page 475.

   $ /usr/sbin/gkadmin

2 Click the Principals tab.

3 Click New.
   The Principal Basics panel that contains some attributes for a principal is displayed.

4 Specify a principal name and a password.
   Both the principal name and the password are mandatory.
5 Specify the encryption types for the principal.
Click on the box to the right of the encryption key types field to open a new window that displays all of the encryption key types available. Click OK after selecting the required encryption types.

![Encryption Types List Helper](image)

6 Specify the policy for the principal.

7 Specify values for the principal’s attributes, and continue to click Next to specify more attributes.
Three windows contain attribute information. Choose Context-Sensitive Help from the Help menu to get information about the various attributes in each window. Or, for all the principal attribute descriptions, go to "SEAM Tool Panel Descriptions" on page 479.

8 Click Save to save the principal, or click Done on the last panel.

9 If needed, set up Kerberos administration privileges for the new principal in the `/etc/krb5/kadm5.acl` file.
See “How to Modify the Kerberos Administration Privileges” on page 469 for more details.

---

**Example 23–4 Creating a New Kerberos Principal**

The following example shows the Principal Basics panel when a new principal called pak is created. The policy is set to testuser.
Creating a New Kerberos Principal (Command Line)

In the following example, the `add_principal` command of `kadmin` is used to create a new principal called `pak`. The principal's policy is set to `testuser`.

```
kadmin: add_principal -policy testuser pak
Enter password for principal "pak@EXAMPLE.COM": <Type the password>
Re-enter password for principal "pak@EXAMPLE.COM": <Type the password again>
Principal "pak@EXAMPLE.COM" created.
kadmin: quit
```
How to Duplicate a Kerberos Principal

This procedure explains how to use all or some of the attributes of an existing principal to create a new principal. No command-line equivalent exists for this procedure.

1. If necessary, start the SEAM Tool.
   See "How to Start the SEAM Tool" on page 457 for more information.
   $ /usr/sbin/gkadmin

2. Click the Principals tab.

3. Select the principal in the list that you want to duplicate, then click Duplicate.
   The Principal Basics panel is displayed. All the attributes of the selected principal are duplicated, except for the Principal Name and Password fields, which are empty.

4. Specify a principal name and a password.
   Both the principal name and the password are mandatory. To make an exact duplicate of the principal you selected, click Save and skip to Step 7.

5. Specify different values for the principal’s attributes, and continue to click Next to specify more attributes.
   Three windows contain attribute information. Choose Context-Sensitive Help from the Help menu to get information about the various attributes in each window. Or, for all the principal attribute descriptions, go to "SEAM Tool Panel Descriptions" on page 479.

6. Click Save to save the principal, or click Done on the last panel.

7. If needed, set up Kerberos administration privileges for the principal in /etc/krb5/kadm5.acl file.
   See “How to Modify the Kerberos Administration Privileges” on page 469 for more details.

How to Modify a Kerberos Principal

An example of the command-line equivalent follows this procedure.

1. If necessary, start the SEAM Tool.
   See “How to Start the SEAM Tool” on page 457 for more information.
   $ /usr/sbin/gkadmin

2. Click the Principals tab.
Select the principal in the list that you want to modify, then click Modify.
The Principal Basics panel that contains some of the attributes for the principal is displayed.

Modify the principal’s attributes, and continue to click Next to modify more attributes.
Three windows contain attribute information. Choose Context-Sensitive Help from the Help menu to get information about the various attributes in each window. Or, for all the principal attribute descriptions, go to "SEAM Tool Panel Descriptions" on page 479.

Note – You cannot modify a principal’s name. To rename a principal, you must duplicate the principal, specify a new name for it, save it, and then delete the old principal.

Click Save to save the principal, or click Done on the last panel.

Modify the Kerberos administration privileges for the principal in the /etc/krb5/kadm5.acl file.
See "How to Modify the Kerberos Administration Privileges" on page 469 for more details.

Example 23–6 Modifying a Kerberos Principal's Password (Command Line)
In the following example, the change_password command of kadmin is used to modify the password for the jdb principal. The change_password command does not let you change the password to a password that is in the principal's password history.

kadmin: change_password jdb
Enter password for principal "jdb": <Type the new password>
Re-enter password for principal "jdb": <Type the password again>
Password for "jdb@EXAMPLE.COM" changed.
kadmin: quit

To modify other attributes for a principal, you must use the modify_principal command of kadmin.

How to Delete a Kerberos Principal
An example of the command-line equivalent follows this procedure.

If necessary, start the SEAM Tool.
See “How to Start the SEAM Tool” on page 457 for more information.

Click the Principals tab.

Select the principal in the list that you want to delete, then click Delete.
After you confirm the deletion, the principal is deleted.
4 Remove the principal from the Kerberos access control list (ACL) file, /etc/krb5/kadm5.acl. See "How to Modify the Kerberos Administration Privileges" on page 469 for more details.

Example 23–7 Deleting a Kerberos Principal (Command Line)

In the following example, the delete_principal command of kadmin is used to delete the jdb principal.

```
kadmin: delete_principal pak
Are you sure you want to delete the principal "pak@EXAMPLE.COM"? (yes/no): yes
Principal "pak@EXAMPLE.COM" deleted.
Make sure that you have removed this principal from all ACLs before reusing.
kadmin: quit
```

How to Set Up Defaults for Creating New Kerberos Principals

No command-line equivalent exists for this procedure.

1 If necessary, start the SEAM Tool.
See "How to Start the SEAM Tool" on page 457 for more information.

$ /usr/sbin/gkadmin
2 Choose Properties from the Edit Menu.  
The Properties window is displayed.

3 Select the defaults that you want to use when you create new principals.  
Choose Context-Sensitive Help from the Help menu for information about the various attributes in each window.

4 Click Save.

How to Modify the Kerberos Administration Privileges

Even though your site probably has many user principals, you usually want only a few users to be able to administer the Kerberos database. Privileges to administer the Kerberos database are determined by the Kerberos access control list (ACL) file, \texttt{kadm5.acl}. The \texttt{kadm5.acl} file
enables you to allow or disallow privileges for individual principals. Or, you can use the('*') wildcard in the principal name to specify privileges for groups of principals.

**Before You Begin**

You must assume the root role. For more information, see "How to Use Your Assigned Administrative Rights" on page 157.

- Edit the `/etc/krb5/kadm5.acl` file.

  An entry in the `kadm5.acl` file must have the following format:

  ```
  principal  privileges  [principal-target]
  ```

  **principal**

  Specifies the principal to which the privileges are granted. Any part of the principal name can include the '*' wildcard, which is useful for providing the same privileges for a group of principals. For example, if you want to specify all principals with the `admin` instance, you would use `*/admin@realm`.

  Note that a common use of an `admin` instance is to grant separate privileges (such as administration access to the Kerberos database) to a separate Kerberos principal. For example, the user `jdb` might have a principal for his administrative use, called `jdb/admin`. This way, the user `jdb` obtains `jdb/admin` tickets only when he or she actually needs to use those privileges.

  **privileges**

  Specifies which operations can or cannot be performed by the principal. This field consists of a string of one or more of the following list of characters or their uppercase counterparts. If the character is uppercase (or not specified), then the operation is disallowed. If the character is lowercase, then the operation is permitted.

  - `a` [Dis]allows the addition of principals or policies.
  - `d` [Dis]allows the deletion of principals or policies.
  - `m` [Dis]allows the modification of principals or policies.
  - `c` [Dis]allows the changing of passwords for principals.
  - `i` [Dis]allows inquiries to the Kerberos database.
  - `l` [Dis]allows the listing of principals or policies in the Kerberos database.
  - `x` or `*` Allows all privileges (admcil).

  **principal-target**

  When a principal is specified in this field, the `privileges` apply to the `principal` only when the `principal` operates on the `principal-target`. Any part of the principal name can include the '*' wildcard, which is useful to group principals.

**Example 23–8** Modifying the Kerberos Administration Privileges

The following entry in the `kadm5.acl` file gives any principal in the `EXAMPLE.COM` realm with the `admin` instance all the privileges on the Kerberos database:
The following entry in the kadm5.acl file gives the jdb@EXAMPLE.COM principal the privileges to add, list, and inquire about any principal that has the root instance.

jdb@EXAMPLE.COM ali */root@EXAMPLE.COM

Administering Kerberos Policies

This section provides step-by-step instructions used to administer policies with the SEAM Tool. This section also provides examples of command-line equivalents, when available.

Administering Kerberos Policies (Task Map)

<table>
<thead>
<tr>
<th>Task</th>
<th>Description</th>
<th>For Instructions</th>
</tr>
</thead>
<tbody>
<tr>
<td>View the list of policies.</td>
<td>View the list of policies by clicking the Policies tab.</td>
<td>“How to View the List of Kerberos Policies” on page 471</td>
</tr>
<tr>
<td>View a policy’s attributes.</td>
<td>View a policy’s attributes by selecting the policy in the Policy List, then clicking the Modify button.</td>
<td>“How to View a Kerberos Policy’s Attributes” on page 473</td>
</tr>
<tr>
<td>Create a new policy.</td>
<td>Create a new policy by clicking the Create New button in the Policy List panel.</td>
<td>“How to Create a New Kerberos Policy” on page 475</td>
</tr>
<tr>
<td>Duplicate a policy.</td>
<td>Duplicate a policy by selecting the policy to duplicate in the Policy List, then clicking the Duplicate button.</td>
<td>“How to Duplicate a Kerberos Policy” on page 477</td>
</tr>
<tr>
<td>Modify a policy.</td>
<td>Modify a policy by selecting the policy to modify in the Policy List, then clicking the Modify button. Note that you cannot modify a policy’s name. To rename a policy, you must duplicate the policy, specify a new name for it, save it, and then delete the old policy.</td>
<td>“How to Modify a Kerberos Policy” on page 477</td>
</tr>
<tr>
<td>Delete a policy.</td>
<td>Delete a policy by selecting the policy to delete in the Policy List, then clicking the Delete button.</td>
<td>“How to Delete a Kerberos Policy” on page 478</td>
</tr>
</tbody>
</table>

▼ How to View the List of Kerberos Policies

An example of the command-line equivalent follows this procedure.

1 If necessary, start the SEAM Tool.
See “How to Start the SEAM Tool” on page 457 for more information.

$ /usr/sbin/gkadmin
2 **Click the Policies tab.**

The list of policies is displayed.

3 **Display a specific policy or a sublist of policies.**

Type a filter string in the Filter field, and press Return. If the filter succeeds, the list of policies that match the filter is displayed.

The filter string must consist of one or more characters. Because the filter mechanism is case sensitive, you need to use the appropriate uppercase and lowercase letters for the filter. For example, if you type the filter string `ge`, the filter mechanism displays only the policies with the `ge` string in them (for example, `george` or `edge`).
If you want to display the entire list of policies, click Clear Filter.

**Example 23–9** Viewing the List of Kerberos Policies (Command Line)

In the following example, the `list_policies` command of `kadmin` is used to list all the policies that match `*user*`. Wildcards can be used with the `list_policies` command.

```
kadmin: list_policies *user*
testuser
enguser
kadmin: quit
```

▼ **How to View a Kerberos Policy's Attributes**

An example of the command-line equivalent follows this procedure.

1. **If necessary, start the SEAM Tool.**
   
   See "How to Start the SEAM Tool" on page 457 for more information.
   
   ```
   $ /usr/sbin/gkadmin
   ```

2. **Click the Policies tab.**

3. **Select the policy in the list that you want to view, then click Modify.**
   
   The Policy Details panel is displayed.

4. **When you are finished viewing, click Cancel.**

**Example 23–10** Viewing a Kerberos Policy's Attributes

The following example shows the Policy Details panel when you are viewing the `test` policy.
Example 23–11  Viewing a Kerberos Policy's Attributes (Command Line)

In the following example, the get_policy command of kadmin is used to view the attributes of the enguser policy.

```plaintext
kadmin: get_policy enguser
Policy: enguser
Maximum password life: 2592000
Minimum password life: 0
Minimum password length: 8
Minimum number of password character classes: 2
Number of old keys kept: 3
Reference count: 0
kadmin: quit
```
The Reference count is the number of principals that use this policy.

▼ How to Create a New Kerberos Policy

An example of the command-line equivalent follows this procedure.

1 **If necessary, start the SEAM Tool.**
   See “How to Start the SEAM Tool” on page 457 for more information.
   $ /usr/sbin/gkadmin

2 **Click the Policies tab.**

3 **Click New.**
   The Policy Details panel is displayed.

4 **Specify a name for the policy in the Policy Name field.**
   The policy name is mandatory.

5 **Specify values for the policy's attributes.**
   Choose Context-Sensitive Help from the Help menu for information about the various attributes in this window. Or, go to Table 23–5 for all the policy attribute descriptions.

6 **Click Save to save the policy, or click Done.**

**Example 23–12 Creating a New Kerberos Policy**

In the following example, a new policy called build11 is created. The Minimum Password Classes is set to 3.
Creating a New Kerberos Policy (Command Line)

In the following example, the add_policy command of kadmin is used to create the build11 policy. This policy requires at least 3 character classes in a password.

```
$ kadmin
kadmin: add_policy -minclasses 3 build11
kadmin: quit
```
How to Duplicate a Kerberos Policy

This procedure explains how to use all or some of the attributes of an existing policy to create a new policy. No command-line equivalent exists for this procedure.

1. If necessary, start the SEAM Tool.
   See “How to Start the SEAM Tool” on page 457 for more information.
   $ /usr/sbin/gkadmin

2. Click the Policies tab.

3. Select the policy in the list that you want to duplicate, then click Duplicate.
   The Policy Details panel is displayed. All the attributes of the selected policy are duplicated, except for the Policy Name field, which is empty.

4. Specify a name for the duplicated policy in the Policy Name field.
   The policy name is mandatory. To make an exact duplicate of the policy you selected, skip to Step 6.

5. Specify different values for the policy’s attributes.
   Choose Context-Sensitive Help from the Help menu for information about the various attributes in this window. Or, go to Table 23–5 for all the policy attribute descriptions.

6. Click Save to save the policy, or click Done.

How to Modify a Kerberos Policy

An example of the command-line equivalent follows this procedure.

1. If necessary, start the SEAM Tool.
   See “How to Start the SEAM Tool” on page 457 for details.
   $ /usr/sbin/gkadmin

2. Click the Policies tab.

3. Select the policy in the list that you want to modify, then click Modify.
   The Policy Details panel is displayed.

4. Modify the policy’s attributes.
   Choose Context-Sensitive Help from the Help menu for information about the various attributes in this window. Or, go to Table 23–5 for all the policy attribute descriptions.
Note—You cannot modify a policy’s name. To rename a policy, you must duplicate the policy, specify a new name for it, save it, and then delete the old policy.

5 Click Save to save the policy, or click Done.

Example 23–14 Modifying a Kerberos Policy (Command Line)

In the following example, the `modify_policy` command of `kadmin` is used to modify the minimum length of a password to five characters for the `build11` policy.

```
$ kadmin
kadmin: modify_policy -minlength 5 build11
kadmin: quit
```

▼ How to Delete a Kerberos Policy

An example of the command-line equivalent follows this procedure.

Note—Before you delete a policy, you must cancel the policy from all principals that are currently using it. To do so, you need to modify the principals’ Policy attribute. The policy cannot be deleted if any principal is using it.

1 If necessary, start the SEAM Tool.
   See “How to Start the SEAM Tool” on page 457 for more information.
   $ /usr/sbin/gkadmin

2 Click the Policies tab.

3 Select the policy in the list that you want to delete, then click Delete.
   After you confirm the deletion, the policy is deleted.

Example 23–15 Deleting a Kerberos Policy (Command Line)

In the following example, the `delete_policy` command of the `kadmin` command is used to delete the `build11` policy.

```
kadmin: delete_policy build11
Are you sure you want to delete the policy "build11"? (yes/no): yes
kadmin: quit
```

Before you delete a policy, you must cancel the policy from all principals that are currently using it. To do so, you need to use the `modify_principal -policy` command of `kadmin` on the affected principals. The `delete_policy` command fails if the policy is in use by a principal.
SEAM Tool Reference

This section provides descriptions of each panel in the SEAM Tool. Also, information about using limited privileges with SEAM Tool are provided.

SEAM Tool Panel Descriptions

This section provides descriptions for each principal and policy attribute that you can either specify or view in the SEAM Tool. The attributes are organized by the panel in which they are displayed.

TABLE 23–2  Attributes for the Principal Basics Panel of the SEAM Tool

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Principal Name</td>
<td>The name of the principal (which is the primary/instance part of a fully qualified principal name). A principal is a unique identity to which the KDC can assign tickets. If you are modifying a principal, you cannot edit its name.</td>
</tr>
<tr>
<td>Password</td>
<td>The password for the principal. You can use the Generate Random Password button to create a random password for the principal.</td>
</tr>
<tr>
<td>Policy</td>
<td>A menu of available policies for the principal.</td>
</tr>
<tr>
<td>Account Expires</td>
<td>The date and time on which the principal’s account expires. When the account expires, the principal can no longer get a ticket-granting ticket (TGT) and might be unable to log in.</td>
</tr>
<tr>
<td>Last Principal Change</td>
<td>The date on which information for the principal was last modified. (Read only)</td>
</tr>
<tr>
<td>Last Changed By</td>
<td>The name of the principal that last modified the account for this principal. (Read only)</td>
</tr>
<tr>
<td>Comments</td>
<td>Comments that are related to the principal (for example, ”Temporary Account”).</td>
</tr>
</tbody>
</table>

TABLE 23–3  Attributes for the Principal Details Panel of the SEAM Tool

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Last Success</td>
<td>The date and time when the principal last logged in successfully. (Read only)</td>
</tr>
<tr>
<td>Last Failure</td>
<td>The date and time when the last login failure for the principal occurred. (Read only)</td>
</tr>
<tr>
<td>Failure Count</td>
<td>The number of times a login failure has occurred for the principal. (Read only)</td>
</tr>
<tr>
<td>Last Password Change</td>
<td>The date and time when the principal’s password was last changed. (Read only)</td>
</tr>
<tr>
<td>Password Expires</td>
<td>The date and time when the principal’s current password expires.</td>
</tr>
<tr>
<td>Key Version</td>
<td>The key version number for the principal. This attribute is normally changed only when a password has been compromised.</td>
</tr>
</tbody>
</table>
### TABLE 23–3  Attributes for the Principal Details Panel of the SEAM Tool  
(Continued)

<table>
<thead>
<tr>
<th>Attribute (Radio Buttons)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum Lifetime (seconds)</td>
<td>The maximum length of time for which a ticket can be granted for the principal (without renewal).</td>
</tr>
<tr>
<td>Maximum Renewal (seconds)</td>
<td>The maximum length of time for which an existing ticket can be renewed for the principal.</td>
</tr>
</tbody>
</table>

### TABLE 23–4  Attributes of the Principal Flags Panel of the SEAM Tool

<table>
<thead>
<tr>
<th>Attribute (Radio Buttons)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Disable Account</td>
<td>When checked, prevents the principal from logging in. This attribute provides an easy way to temporarily freeze a principal account.</td>
</tr>
<tr>
<td>Require Password Change</td>
<td>When checked, expires the principal’s current password, which forces the user to use the kpasswd command to create a new password. This attribute is useful if a security breach occurs, and you need to make sure that old passwords are replaced.</td>
</tr>
<tr>
<td>Allow Postdated Tickets</td>
<td>When checked, allows the principal to obtain postdated tickets. For example, you might need to use postdated tickets for cron jobs that must run after hours, but you cannot obtain tickets in advance because of short ticket lifetimes.</td>
</tr>
<tr>
<td>Allow Forwardable Tickets</td>
<td>When checked, allows the principal to obtain forwardable tickets. Forwardable tickets are tickets that are forwarded to the remote host to provide a single-sign-on session. For example, if you are using forwardable tickets and you authenticate yourself through ftp or rsh, then other services, such as NFS services, are available without your being prompted for another password.</td>
</tr>
<tr>
<td>Allow Renewable Tickets</td>
<td>When checked, allows the principal to obtain renewable tickets. A principal can automatically extend the expiration date or time of a ticket that is renewable (rather than having to get a new ticket after the first ticket expires). Currently, the NFS service is the ticket service that can renew tickets.</td>
</tr>
<tr>
<td>Allow Proxiable Tickets</td>
<td>When checked, allows the principal to obtain proxiable tickets. A proxiable ticket is a ticket that can be used by a service on behalf of a client to perform an operation for the client. With a proxiable ticket, a service can take on the identity of a client and obtain a ticket for another service. However, the service cannot obtain a ticket-granting ticket (TGT).</td>
</tr>
<tr>
<td>Allow Service Tickets</td>
<td>When checked, allows service tickets to be issued for the principal. You should not allow service tickets to be issued for the kadmin/hostname and changepw/hostname principals. This practice ensures that only these principals can update the KDC database.</td>
</tr>
</tbody>
</table>
### TABLE 23-4 Attributes of the Principal Flags Panel of the SEAM Tool  
*(Continued)*

<table>
<thead>
<tr>
<th>Attribute (Radio Buttons)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Allow TGT-Based Authentication</td>
<td>When checked, allows the service principal to provide services to another principal. More specifically, this attribute allows the KDC to issue a service ticket for the service principal. This attribute is valid only for service principals. When unchecked, service tickets cannot be issued for the service principal.</td>
</tr>
<tr>
<td>Allow Duplicate Authentication</td>
<td>When checked, allows the user principal to obtain service tickets for other user principals. This attribute is valid only for user principals. When unchecked, the user principal can still obtain service tickets for service principals, but not for other user principals.</td>
</tr>
<tr>
<td>Required Preauthentication</td>
<td>When checked, the KDC will not send a requested ticket-granting ticket (TGT) to the principal until the KDC can authenticate (through software) that the principal is really the principal that is requesting the TGT. This preauthentication is usually done through an extra password, for example, from a DES card. When unchecked, the KDC does not need to preauthenticate the principal before the KDC sends a requested TGT to the principal.</td>
</tr>
<tr>
<td>Required Hardware Authentication</td>
<td>When checked, the KDC will not send a requested ticket-granting ticket (TGT) to the principal until the KDC can authenticate (through hardware) that the principal is really the principal that is requesting the TGT. Hardware preauthentication can occur, for example, on a Java ring reader. When unchecked, the KDC does not need to preauthenticate the principal before the KDC sends a requested TGT to the principal.</td>
</tr>
</tbody>
</table>

### TABLE 23-5 Attributes for the Policy Basics Pane of the SEAM Tool

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Policy Name</td>
<td>The name of the policy. A policy is a set of rules that govern a principal’s password and tickets. If you are modifying a policy, you cannot edit its name.</td>
</tr>
<tr>
<td>Minimum Password Length</td>
<td>The minimum length for the principal’s password.</td>
</tr>
<tr>
<td>Minimum Password Classes</td>
<td>The minimum number of different character types that are required in the principal’s password. For example, a minimum classes value of 2 means that the password must have at least two different character types, such as letters and numbers (hi2mom). A value of 3 means that the password must have at least three different character types, such as letters, numbers, and punctuation (hi2mom!). And so on. A value of 1 sets no restriction on the number of password character types.</td>
</tr>
<tr>
<td>Saved Password History</td>
<td>The number of previous passwords that have been used by the principal, and a list of the previous passwords that cannot be reused.</td>
</tr>
<tr>
<td>Minimum Password Lifetime</td>
<td>The minimum length of time that the password must be used before it can be changed.</td>
</tr>
</tbody>
</table>
TABLE 23–5  Attributes for the Policy Basics Pane of the SEAM Tool  (Continued)

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum Password Lifetime</td>
<td>The maximum length of time that the password can be used before it must be changed.</td>
</tr>
<tr>
<td>(seconds)</td>
<td></td>
</tr>
<tr>
<td>Principals Using This Policy</td>
<td>The number of principals to which this policy currently applies. (Read only)</td>
</tr>
</tbody>
</table>

Using the SEAM Tool With Limited Kerberos Administration Privileges

All capabilities of the SEAM Tool are available if your admin principal has all the privileges to administer the Kerberos database. However, you might have limited privileges, such as only being allowed to view the list of principals or to change a principal’s password. With limited Kerberos administration privileges, you can still use the SEAM Tool. However, various parts of the SEAM Tool change based on the Kerberos administration privileges that you do not have. Table 23–6 shows how the SEAM Tool changes based on your Kerberos administration privileges.

The most visual change to the SEAM Tool occurs when you don’t have the list privilege. Without the list privilege, the List panels do not display the list of principals and polices for you to manipulate. Instead, you must use the Name field in the List panels to specify a principal or a policy that you want to manipulate.

If you log in to the SEAM Tool, and you do not have sufficient privileges to perform tasks with it, the following message displays and you are sent back to the SEAM Administration Login window:

Insufficient privileges to use gkadmin: ADMCIL. Please try using another principal.

To change the privileges for a principal so that it can administer the Kerberos database, go to “How to Modify the Kerberos Administration Privileges” on page 469.

<table>
<thead>
<tr>
<th>Disallowed Privilege</th>
<th>How the SEAM Tool Changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>a (add)</td>
<td>The Create New and Duplicate buttons are unavailable in the Principal List and Policy List panels. Without the add privilege, you cannot create new principals or policies, or duplicate them.</td>
</tr>
<tr>
<td>d (delete)</td>
<td>The Delete button is unavailable in the Principal List and Policy List panels. Without the delete privilege, you cannot delete principals or policies.</td>
</tr>
</tbody>
</table>
TABLE 23–6 Using the SEAM Tool With Limited Kerberos Administration Privileges (Continued)

<table>
<thead>
<tr>
<th>Disallowed Privilege</th>
<th>How the SEAM Tool Changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>m (modify)</td>
<td>The Modify button is unavailable in the Principal List and Policy List panels. Without the modify privilege, you cannot modify principals or policies. Also, with the Modify button unavailable, you cannot modify a principal’s password, even if you have the change password privilege.</td>
</tr>
<tr>
<td>c (change password)</td>
<td>The Password field in the Principal Basics panel is read only and cannot be changed. Without the change password privilege, you cannot modify a principal’s password. Note that even if you have the change password privilege, you must also have the modify privilege to change a principal’s password.</td>
</tr>
<tr>
<td>i (inquiry to database)</td>
<td>The Modify and Duplicate buttons are unavailable in the Principal List and Policy List panels. Without the inquiry privilege, you cannot modify or duplicate a principal or a policy. Also, with the Modify button unavailable, you cannot modify a principal’s password, even if you have the change password privilege.</td>
</tr>
<tr>
<td>l (list)</td>
<td>The list of principals and policies in the List panels are unavailable. Without the list privilege, you must use the Name field in the List panels to specify the principal or the policy that you want to manipulate.</td>
</tr>
</tbody>
</table>

**Administering Keytab Files**

Every host that provides a service must have a local file, called a *keytab* (short for “key table”). The keytab contains the principal for the appropriate service, called a *service key*. A service key is used by a service to authenticate itself to the KDC and is known only by Kerberos and the service itself. For example, if you have a Kerberized NFS server, that server must have a keytab file that contains its *nfs* service principal.

To add a service key to a keytab file, you add the appropriate service principal to a host’s keytab file by using the `ktadd` command of `kadmin`. Because you are adding a service principal to a keytab file, the principal must already exist in the Kerberos database so that `kadmin` can verify its existence. On application servers that provide Kerberized services, the keytab file is located at `/etc/krb5/krb5.keytab`, by default.

A keytab is analogous to a user’s password. Just as it is important for users to protect their passwords, it is equally important for application servers to protect their keytab files. You should always store keytab files on a local disk, and make them readable only by the *root* user. Also, you should never send a keytab file over an unsecured network.

There is also a special instance in which to add a *root* principal to a host’s keytab file. If you want a user on the Kerberos client to mount Kerberized NFS file systems that require...
root-equivalent access, you must add the client's root principal to the client's keytab file. Otherwise, users must use the kinit command as root to obtain credentials for the client's root principal whenever they want to mount a Kerberized NFS file system with root access, even when they are using the automounter.

Another command that you can use to administer keytab files is the ktutil command. This interactive command enables you to manage a local host's keytab file without having Kerberos administration privileges, because ktutil doesn't interact with the Kerberos database as kadmin does. So, after a principal is added to a keytab file, you can use ktutil to view the keylist in a keytab file or to temporarily disable authentication for a service.

**Note** – When you change a principal in a keytab file using the ktadd command in kadmin, a new key is generated and added to the keytab file.

### Administering Keytab Files (Task Map)

<table>
<thead>
<tr>
<th>Task</th>
<th>Description</th>
<th>For Instructions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add a service principal to a keytab file.</td>
<td>Use the ktadd command of kadmin to add a service principal to a keytab file.</td>
<td>&quot;How to Add a Kerberos Service Principal to a Keytab File&quot; on page 484</td>
</tr>
<tr>
<td>Remove a service principal from a keytab file.</td>
<td>Use the ktrm command of kadmin to remove a service from a keytab file.</td>
<td>&quot;How to Remove a Service Principal From a Keytab File&quot; on page 485</td>
</tr>
<tr>
<td>Display the keylist (list of principals) in a keytab file.</td>
<td>Use the ktutil command to display the keylist in a keytab file.</td>
<td>&quot;How to Display the Keylist (Principals) in a Keytab File&quot; on page 486</td>
</tr>
<tr>
<td>Temporarily disable authentication for a service on a host.</td>
<td>This procedure is a quick way to temporarily disable authentication for a service on a host without requiring kadmin privileges. Before you use ktutil to delete the service principal from the server's keytab file, copy the original keytab file to a temporary location. When you want to enable the service again, copy the original keytab file back to its proper location.</td>
<td>&quot;How to Temporarily Disable Authentication for a Service on a Host&quot; on page 487</td>
</tr>
</tbody>
</table>

▼ **How to Add a Kerberos Service Principal to a Keytab File**

1. **Make sure that the principal already exists in the Kerberos database.**

   See "How to View the List of Kerberos Principals" on page 459 for more information.
Assume the root role on the host that needs a principal added to its keytab file.

Start the kadmin command.

```
#/usr/sbin/kadmin
```

Add a principal to a keytab file by using the ktadd command.

```
kadmin: ktadd [-e enctype] [-k keytab] [-q] [principal | -glob principal-exp]
```

- `-e enctype` Overrides the list of encryption types defined in the krb5.conf file.
- `-k keytab` Specifies the keytab file. By default, /etc/krb5/krb5.keytab is used.
- `-q` Displays less verbose information.
- `principal` Specifies the principal to be added to the keytab file. You can add the following service principals: `host`, `root`, `nfs`, and `ftp`.
- `-glob principal-exp` Specifies the principal expressions. All principals that match the `principal-exp` are added to the keytab file. The rules for principal expression are the same as for the list_principals command of kadmin.

Quit the kadmin command.

```
kadmin: quit
```

Example 23–16

Adding a Service Principal to a Keytab File

In the following example, denver's `host` principal is added to denver's keytab file, so that the KDC can authenticate denver's network services.

```
denver # /usr/sbin/kadmin
kadmin: ktadd host/denver.example.com
Entry for principal host/denver.example.com with kvno 3, encryption type AES-256 CTS mode with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal host/denver.example.com with kvno 3, encryption type AES-128 CTS mode with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal host/denver.example.com with kvno 3, encryption type Triple DES cbc mode with HMAC/sha1 added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal host/denver.example.com with kvno 3, encryption type ArcFour with HMAC/md5 added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal host/denver.example.com with kvno 3, encryption type DES cbc mode with RSA-MD5 added to keytab WRFILE:/etc/krb5/krb5.keytab.
kadmin: quit
```

How to Remove a Service Principal From a Keytab File

Assume the root role on the host with a service principal that must be removed from its keytab file.
2 Start the kadmin command.
   # /usr/sbin/kadmin

3 (Optional) To display the current list of principals (keys) in the keytab file, use the ktutil command.
   See "How to Display the Keylist (Principals) in a Keytab File" on page 486 for detailed instructions.

4 Remove a principal from the keytab file by using the ktremove command.
   kadmin: ktremove [-k keytab] [-q] principal [kvno | all | old]
   -k keytab Specifies the keytab file. By default, /etc/krb5/krb5.keytab is used.
   -q Displays less verbose information.
   principal Specifies the principal to be removed from the keytab file.
   kvno Removes all entries for the specified principal whose key version number matches kvno.
   all Removes all entries for the specified principal.
   old Removes all entries for the specified principal, except those principals with the highest key version number.

5 Quit the kadmin command.
   kadmin: quit

Example 23–17 Removing a Service Principal From a Keytab File

In the following example, denver's host principal is removed from denver's keytab file.

denver # /usr/sbin/kadmin
kadmin: ktremove host/denver.example.com@EXAMPLE.COM
kadmin: Entry for principal host/denver.example.com@EXAMPLE.COM with kvno 3
removed from keytab WRFILE:/etc/krb5/krb5.keytab.
kadmin: quit

▼ How to Display the Keylist (Principals) in a Keytab File

1 Assume the root role on the host with the keytab file.

Note – Although you can create keytab files that are owned by other users, using the default location for the keytab file requires root ownership.
2 Start the ktutil command.
   
   # /usr/bin/ktutil

3 Read the keytab file into the keylist buffer by using the read_kt command.
   
   ktutil: read_kt keytab

4 Display the keylist buffer by using the list command.
   
   ktutil: list
   The current keylist buffer is displayed.

5 Quit the ktutil command.
   
   ktutil: quit

**Example 23–18** Displaying the Keylist (Principals) in a Keytab File

The following example displays the keylist in the /etc/krb5/krb5.keytab file on the denver host.

```
denver # /usr/bin/ktutil
   ktutil: read_kt /etc/krb5/krb5.keytab
   ktutil: list
   slot KVNO Principal
   ---- ---- ---------------------------------------
   1 5 host/denver@EXAMPLE.COM
   ktutil: quit
```

**▼ How to Temporarily Disable Authentication for a Service on a Host**

At times, you might need to temporarily disable the authentication mechanism for a service, such as rlogin or ftp, on a network application server. For example, you might want to stop users from logging in to a system while you are performing maintenance procedures. The ktutil command enables you to accomplish this task by removing the service principal from the server’s keytab file, without requiring kadmin privileges. To enable authentication again, you just need to copy the original keytab file that you saved back to its original location.

**Note** – By default, most services are set up to require authentication. If a service is not set up to require authentication, then the service still works, even if you disable authentication for the service.

1 Assume the root role on the host with the keytab file.
**Note**—Although you can create keytab files that are owned by other users, using the default location for the keytab file requires root ownership.

1. Save the current keytab file to a temporary file.
2. Start the ktutil command.
   ```
   # /usr/bin/ktutil
   ```
3. Read the keytab file into the keylist buffer by using the read_kt command.
   ```
   ktutil: read_kt keytab
   ```
4. Display the keylist buffer by using the list command.
   ```
   ktutil: list
   ```
   The current keylist buffer is displayed. Note the slot number for the service that you want to disable.
5. To temporarily disable a host's service, remove the specific service principal from the keylist buffer by using the delete_entry command.
   ```
   ktutil: delete_entry slot-number
   ```
   Where `slot-number` specifies the slot number of the service principal to be deleted, which is displayed by the list command.
6. Write the keylist buffer to a new keytab file by using the write_kt command.
   ```
   ktutil: write_kt new-keytab
   ```
7. Quit the ktutil command.
   ```
   ktutil: quit
   ```
8. Move the new keytab file.
   ```
   # mv new-keytab keytab
   ```
9. When you want to re-enable the service, copy the temporary (original) keytab file back to its original location.

**Example 23–19** Temporarily Disabling a Service on a Host

In the following example, the host service on the denver host is temporarily disabled. To re-enable the host service on denver, you would copy the krb5.keytab.temp file to the /etc/krb5/krb5.keytab file.
denver # cp /etc/krb5/krb5.keytab /etc/krb5/krb5.keytab.temp
denver # /usr/bin/ktutil
   ktutil: read kt /etc/krb5/krb5.keytab
   ktutil: list
slot KVNO Principal
---- ---- ---------------------------------------
 1 8 root/denver@EXAMPLE.COM
 2 5 host/denver@EXAMPLE.COM
   ktutil: delete_entry 2
   ktutil: list
slot KVNO Principal
---- ---- ---------------------------------------
 1 8 root/denver@EXAMPLE.COM
   ktutil: write kt /etc/krb5/new.krb5.keytab
   ktutil: quit
denver # cp /etc/krb5/new.krb5.keytab /etc/krb5/krb5.keytab
Using Kerberos Applications (Tasks)

This chapter is intended for anyone on a system with the Kerberos service configured on it. This chapter explains how to use the “Kerberized” commands and services that are provided. You should already be familiar with these commands (in their non-Kerberized versions) before you read about them here.

Because this chapter is intended for the general reader, it includes information on tickets: obtaining, viewing, and destroying them. This chapter also includes information on choosing or changing a Kerberos password.

This is a list of the information in this chapter:

- “Kerberos Ticket Management” on page 491
- “Kerberos Password Management” on page 495
- “Kerberos User Commands” on page 500

For an overview of the Oracle Solaris Kerberos product, see Chapter 19, “Introduction to the Kerberos Service.”

Kerberos Ticket Management

This section explains how to obtain, view, and destroy tickets. For an introduction to tickets, see “How the Kerberos Service Works” on page 334.

Do You Need to Worry About Tickets?

With any of the Oracle Solaris releases installed, Kerberos is built into the login command. However, to obtain tickets automatically, you must configure the PAM service for the applicable login services. For more information, see the pam_krb5(5) man page. The Kerberized commands rsh, rcp, telnet, and rlogin are usually set up to forward copies of your tickets to the other machines, so you don’t have to explicitly ask for tickets to get access to those
machines. Your configuration might not include this automatic forwarding, since by default system is not configured to have credentials forwarded. See “Overview of Kerberized Commands” on page 500 and “Forwarding Kerberos Tickets” on page 503 for more information on forwarding tickets.

For information on ticket lifetimes, see “Ticket Lifetimes” on page 513.

Creating a Kerberos Ticket

Normally, if PAM is configured properly, a ticket is created automatically when you log in, and you need not do anything special to obtain a ticket. However, you might need to create a ticket if your ticket expires. Also, you might need to use a different principal besides your default principal, for example, if you use `rlogin -l` to log in to a machine as someone else.

To create a ticket, use the `kinit` command.

```
% /usr/bin/kinit
```

The `kinit` command prompts you for your password. For the full syntax of the `kinit` command, see the `kinit(1)` man page.

**EXAMPLE 24-1  Creating a Kerberos Ticket**

This example shows a user, `jennifer`, creating a ticket on her own system.

```
% kinit
Password for jennifer@ENG.EXAMPLE.COM: <Type password>
```

Here, the user `david` creates a ticket that is valid for three hours with the `-l` option.

```
% kinit -l 3h david@EXAMPLE.ORG
Password for david@EXAMPLE.ORG: <Type password>
```

This example shows the user `david` creating a forwardable ticket (with the `-f` option) for himself. With this forwardable ticket, he can, for example, log in to a second system, and then `telnet` to a third system.

```
% kinit -f david@EXAMPLE.ORG
Password for david@EXAMPLE.ORG: <Type password>
```

For more information on how forwarding tickets works, see “Forwarding Kerberos Tickets” on page 503 and “Types of Tickets” on page 512.
Viewing Kerberos Tickets

Not all tickets are alike. One ticket might, for example, be forwardable. Another ticket might be postdated. While a third ticket might be both forwardable and postdated. You can see which tickets you have, and what their attributes are, by using the klist command with the \(-f\) option:

```
% /usr/bin/klist -f
```

The following symbols indicate the attributes that are associated with each ticket, as displayed by klist:

A  Preauthenticated
D  Postdatable
D  Postdated
F  Forwardable
f  Forwarded
I  Initial
i  Invalid
P  Proxiable
p  Proxy
R  Renewable

"Types of Tickets" on page 512 describes the various attributes that a ticket can have.

EXAMPLE 24–2  Viewing Kerberos Tickets

This example shows that the user jennifer has an initial ticket, which is forwardable (F) and postdated (d), but not yet validated (i).

```
% /usr/bin/klist -f
Ticket cache: /tmp/krb5cc_74287
Default principal: jennifer@EXAMPLE.COM
Valid starting   Expires       Service principal
  09 Mar 04 15:09:51 09 Mar 04 21:09:51  nfs/EXAMPLE.COM@EXAMPLE.COM
  renew until 10 Mar 04 15:12:51, Flags: Fdi
```

The following example shows that the user david has two tickets that were forwarded (f) to his host from another host. The tickets are also forwardable (F).

```
% klist -f
Ticket cache: /tmp/krb5cc_74287
Default principal: david@EXAMPLE.COM
```
EXAMPLE 24–2 Viewing Kerberos Tickets (Continued)

Valid starting Expires Service principal
07 Mar 04 06:09:51 09 Mar 04 23:33:51 host/EXAMPLE.COM@EXAMPLE.COM
   renew until 10 Mar 04 17:09:51, Flags: ff

Valid starting Expires Service principal
08 Mar 04 08:09:51 09 Mar 04 12:54:51 nfs/EXAMPLE.COM@EXAMPLE.COM
   renew until 10 Mar 04 15:22:51, Flags: ff

The following example shows how to display the encryption types of the session key and the ticket by using the -e option. The -a option is used to map the host address to a host name if the name service can do the conversion.

% klist -fea
Ticket cache: /tmp/krb5cc_74287
Default principal: david@EXAMPLE.COM

Valid starting Expires Service principal
07 Mar 04 06:09:51 09 Mar 04 23:33:51 krbtgt/EXAMPLE.COM@EXAMPLE.COM
   renew until 10 Mar 04 17:09:51, Flags: FRIA
   Etype(skey, tkt): DES cbc mode with RSA-MD5, DES cbc mode with CRC-32
   Addresses: client.example.com

Destroying Kerberos Tickets

If you want to destroy all Kerberos tickets acquired during your current session, use the kdestroy command. The command destroys your credential cache, which destroys all your credentials and tickets. While this is not usually necessary, running kdestroy reduces the chance of the credential cache being compromised during times that you are not logged in.

To destroy your tickets, use the kdestroy command.

% /usr/bin/kdestroy

The kdestroy command destroys all your tickets. You cannot use this command to selectively destroy a particular ticket.

If you are going to be away from your system and are concerned about an intruder using your permissions, you should use either kdestroy or a screen saver that locks the screen.
Kerberos Password Management

With the Kerberos service configured, you now have two passwords: your regular Solaris password and a Kerberos password. You can make both passwords the same, or they can be different.

Advice on Choosing a Password

Your password can include almost any character that you can type. The main exceptions are the Control keys and the Return key. A good password is a password that you can remember readily, but no one else can easily guess. Examples of bad passwords include the following:

- Words that can be found in a dictionary
- Any common or popular name
- The name of a famous person or character
- Your name or user name in any form (for example: your name spelled backward, repeated twice, and so forth)
- A spouse’s name, child’s name, or pet’s name
- Your birth date or a relative’s birth date
- Your social security number, driver’s license number, passport number, or other similar identifying number
- Any sample password that appears in this manual or any other manual

A good password is at least eight characters long. Moreover, a password should include a mix of characters, such as uppercase and lowercase letters, numbers, and punctuation marks. Examples of passwords that would be good if they didn’t appear in this manual include the following:

- Acronyms, such as “I2LMHinSF” (which is recalled as “I too left my heart in San Francisco”)
- Easy-to-pronounce nonsense words, such as “WumpaBun” or “WangDangdoodle!”
- Deliberately misspelled phrases, such as “6o’cluck” or “RriotGrrrlsRrrule!”

Caution – Don’t use these examples. Passwords that appear in manuals are the first passwords that an intruder will try.
Changing Your Password

If PAM is properly configured, you can change your Kerberos password in two ways:

- With the usual UNIX `passwd` command. With the Kerberos service configured, the `passwd` command also automatically prompts for a new Kerberos password.

  The advantage of using `passwd` instead of `kpasswd` is that you can set both UNIX and Kerberos passwords at the same time. However, you generally do not have to change both passwords with `passwd`. Often, you can change only your UNIX password and leave the Kerberos password untouched, or vice-versa.

  **Note** – The behavior of `passwd` depends on how the PAM module is configured. You might be required to change both passwords in some configurations. For some sites, the UNIX password must be changed, while other sites require the Kerberos password to change.

- With the `kpasswd` command. `kpasswd` is very similar to `passwd`. One difference is that `kpasswd` changes only Kerberos passwords. You must use `passwd` if you want to change your UNIX password.

  Another difference is that `kpasswd` can change a password for a Kerberos principal that is not a valid UNIX user. For example, `david/admin` is a Kerberos principal, but not an actual UNIX user, so you must use `kpasswd` instead of `passwd`.

After you change your password, it takes some time for the change to propagate through a system (especially over a large network). Depending on how your system is set up, this delay might take anywhere from a few minutes to an hour or more. If you need to get new Kerberos tickets shortly after you change your password, try the new password first. If the new password doesn't work, try again using the old password.

Kerberos V5 protocol enables system administrators to set criteria about allowable passwords for each user. Such criteria is defined by the policy set for each user (or by a default policy). See “Administering Kerberos Policies” on page 471 for more on policies.

For example, suppose that user `jennifer`'s policy (call it `jenpol`) mandates that passwords be at least eight letters long and include a mix of at least two types of characters. `kpasswd` will therefore reject an attempt to use “sloth” as a password.

```bash
% kpasswd
kpasswd: Changing password for jennifer@ENG.EXAMPLE.COM.
Old password:  <Jennifer types her existing password>
kpasswd: jennifer@ENG.EXAMPLE.COM's password is controlled by the policy jenpol
which requires a minimum of 8 characters from at least 2 classes (the five classes are lowercase, uppercase, numbers, punctuation, and all other characters).
```
New password:  <Jennifer types 'sloth'>
New password (again):  <Jennifer re-types 'sloth'>
kpasswd: New password is too short.
Please choose a password which is at least 4 characters long.

Here, jennifer uses “slothrop49” as a password. “slothrop49” meets the criteria, because it is over eight letters long and contains two different types of characters (numbers and lowercase letters).

% kpasswd
kpasswd: Changing password for jennifer@ENG.EXAMPLE.COM.
Old password:  <Jennifer types her existing password>
kpasswd: jennifer@ENG.EXAMPLE.COM’s password is controlled by the policy jenpol which requires a minimum of 8 characters from at least 2 classes (the five classes are lowercase, uppercase, numbers, punctuation, and all other characters).
New password:  <Jennifer types 'slothrop49'>
New password (again):  <Jennifer re-types 'slothrop49'>
Kerberos password changed.

EXAMPLE 24-3  Changing Your Password

In the following example, user david changes both his UNIX password and Kerberos password with passwd.

% passwd
passwd: Changing password for david
Enter login password:  <Type the current UNIX password>
New password:  <Type the new UNIX password>
Re-enter password:  <Confirm the new UNIX password>
Old KRB5 password:  <Type the current Kerberos password>
New KRB5 password:  <Type the new Kerberos password>
Re-enter new KRB5 password:  <Confirm the new Kerberos password>

Note that passwd asks for both the UNIX password and the Kerberos password. This behavior is established by the default configuration. In that case, user david must use kpasswd to set his Kerberos password to something else, as shown next.

This example shows user david changing only his Kerberos password with kpasswd.

% kpasswd
kpasswd: Changing password for david@ENG.EXAMPLE.COM.
Old password:  <Type the current Kerberos password>
New password:  <Type the new Kerberos password>
New password (again):  <Confirm the new Kerberos password>
Kerberos password changed.
EXAMPLE 24–3    Changing Your Password    (Continued)

In this example, user david changes the password for the Kerberos principal david/admin (which is not a valid UNIX user). He must use kpasswd.

% kpasswd david/admin
kpasswd: Changing password for david/admin.
Old password:  <Type the current Kerberos password>
New password:  <Type the new Kerberos password>
New password (again):  <Type the new Kerberos password>
Kerberos password changed.

Granting Access to Your Account

If you need to give someone access to log in to your account (as you), you can do so through Kerberos, without revealing your password, by putting a .k5login file in your home directory. A .k5login file is a list of one or more Kerberos principals corresponding to each person for whom you want to grant access. Each principal must be on a separate line.

Suppose that the user david keeps a .k5login file in his home directory that looks like the following:

Jennifer@ENG.EXAMPLE.COM
joe@EXAMPLE.ORG

This file allows the users jennifer and joe to assume david’s identity, provided that they already have Kerberos tickets in their respective realms. For example, jennifer can remotely log in to david’s machine (boston), as him, without having to give his password.
In the case where David’s home directory is NFS-mounted, using Kerberos V5 protocols, from another (third) machine, Jennifer must have a forwardable ticket in order to access his home directory. See “Creating a Kerberos Ticket” on page 492 for an example of using a forwardable ticket.

If you will be logging into other machines across a network, you’ll want to include your own Kerberos principal in .k5login files on those machines.

Using a .k5login file is much safer than giving out your password for these reasons:

- You can take access away any time by removing the principal from your .k5login file.
- Although users principals named in the .k5login file in your home directory have full access to your account on that machine (or sets of machines, if the .k5login file is shared, for example, over NFS). However, any Kerberized services will authorize access based on that user’s identity, not yours. So Jennifer can log in to Joe’s machine and perform tasks there. However, if she uses a Kerberized program such as ftp or rlogin, she does so as herself.
- Kerberos keeps a log of who obtains tickets, so a system administrator can find out, if necessary, who is capable of using your user identity at a particular time.

One common way to use the .k5login file is to put it in root’s home directory, giving root access for that machine to the Kerberos principals listed. This configuration allows system administrators to become root locally, or to log in remotely as root, without having to give out the root password, and without requiring anyone to type the root password over the network.

**EXAMPLE 24-4** Using the .k5login File to Grant Access to Your Account

Suppose Jennifer decides to log in to the machine boston.example.com as root. Because she has an entry for her principal name in the .k5login file in root’s home directory on boston.example.com, she again does not have to type in her password.
Kerberos User Commands

Kerberos V5 product is a single-sign-on system, which means that you only have to type your password once. The Kerberos V5 programs do the authenticating (and optional encrypting) for you, because Kerberos has been built into each of a suite of existing, familiar network programs. The Kerberos V5 applications are versions of existing UNIX network programs with Kerberos features added.

For example, when you use a Kerberized program to connect to a remote host, the program, the KDC, and the remote host perform a set of rapid negotiations. When these negotiations are completed, your program has proven your identity on your behalf to the remote host, and the remote host has granted you access.

Note that Kerberized commands try to authenticate with Kerberos first. If Kerberos authentication fails, an error occurs or UNIX authentication is attempted, depending on what options were used with the command. Refer to the Kerberos Security section in each Kerberos command man page for more detailed information.

Overview of Kerberized Commands

The Kerberized network services are programs that connect to another machine somewhere on the Internet. These programs are the following:

- ftp
- rcp
- rlogin
- rsh
- ssh
- telnet

These programs have features that transparently use your Kerberos tickets for negotiating authentication and optional encryption with the remote host. In most cases, you'll notice only that you no longer have to type your password to use them, because Kerberos will provide proof of your identity for you.
The Kerberos V5 network programs include options that enable you to do the following:

- Forward your tickets to the another host (if you initially obtained forwardable tickets).
- Encrypt data transmitted between you and the remote host.

**Note** – This section assumes you are already familiar with the non-Kerberos versions of these programs, and highlights the Kerberos functionality added by the Kerberos V5 package. For detailed descriptions of the commands described here, see their respective man pages.

The following Kerberos options have been added to `ftp`, `rcp`, `rlogin`, `rsh`, and `telnet`:

- `-a`  Attempts automatic login using your existing tickets. Uses the username as returned by `getlogin()`, unless the name is different from the current user ID. See the `telnet(1)` man page for details.

- `-f`  Forwards a *non-forwardable* ticket to a remote host. This option is mutually exclusive with the `-F` option. They cannot be used together in the same command.

  You'll want to forward a ticket if you have reason to believe you'll need to authenticate yourself to other Kerberos-based services on a third host. For example, you might want to remotely log in to another machine and then remotely log in from it to a third machine.

  You should definitely use a forwardable ticket if your home directory on the remote host is NFS-mounted using the Kerberos V5 mechanism. Otherwise, you won't be able to access your home directory. That is, suppose you initially log in to System 1. From System 1, you remotely log in to your home machine, System 2, which mounts your home directory from System 3. Unless you've used the `-f` or `-F` option with `rlogin`, you won't be able to get to your home directory because your ticket can't be forwarded to System 3.

  By default, `kinit` obtains forwardable ticket-granting tickets (TGTs). However, your configuration might differ in this respect.

  For more information on forwarding tickets, see "Forwarding Kerberos Tickets" on page 503.

- `-F`  Forwards a *reforwardable* copy of your TGT to a remote system. It is similar to `-f`, but it allows for access to a further (say, fourth or fifth) machine. The `-F` option can therefore be regarded as being a superset of the `-f` option. The `-F` option is mutually exclusive with the `-f` option. They cannot be used together in the same command.
For more information on forwarding tickets, see “Forwarding Kerberos Tickets” on page 503.

- **k realm** Requests tickets for the remote host in the specified realm, instead of determining the realm itself using the krb5.conf file.

- **K** Uses your tickets to authenticate to the remote host, but does not automatically log in.

- **m mechanism** Specifies the GSS-API security mechanism to use, as listed in the /etc/gss/mech file. Defaults to kerberos_v5.

- **x** Encrypts this session.

- **X auth-type** Disables the auth-type type of authentication.

The following table shows which commands have specific options. An “X” indicates that the command has that option.

<table>
<thead>
<tr>
<th>TABLE 24-1 Kerberos Options for Network Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Command</strong></td>
</tr>
<tr>
<td>----------</td>
</tr>
<tr>
<td>-a &amp; -f</td>
</tr>
<tr>
<td>-F</td>
</tr>
<tr>
<td>-k &amp; -K</td>
</tr>
<tr>
<td>-m</td>
</tr>
<tr>
<td>-x</td>
</tr>
<tr>
<td>-X</td>
</tr>
</tbody>
</table>

Additionally, `ftp` allows the protection level for a session to be set at its prompt:

- **clear** Sets the protection level to "clear" (no protection). This protection level is the default.
- **private** Sets the protection level to "private." Data transmissions are confidentiality-protected and integrity-protected by encryption. The privacy service might not be available to all Kerberos users, however.
- **safe** Sets the protection level to "safe." Data transmissions are integrity-protected by cryptographic checksum.
You can also set the protection level at the `ftp` prompt by typing `protect` followed by any of the protection levels shown above (clear, private, or safe).

**Forwarding Kerberos Tickets**

As described in “Overview of Kerberized Commands” on page 500, some commands allow you to forward tickets with either the `-f` or `-F` option. Forwarding tickets allows you to “chain” your network transactions. You can, for example, remotely log in to one machine and then remotely log in from it to another machine. The `-f` option allows you to forward a ticket, while the `-F` option allows you to reforward a forwarded ticket.

In the following figure, the user david obtains a non-forwardable ticket-granting ticket (TGT) with `kinit`. The ticket is non-forwardable because he did not specify the `-f` option. In scenario 1, he is able to remotely log in to machine B, but he can go no further. In scenario 2, the `rlogin` `-f` command fails because he is attempting to forward a ticket that is non-forwardable.

In actuality, Kerberos configuration files are set up so that `kinit` obtains forwardable tickets by default. However, your configuration might differ. For the sake of explanation, assume that `kinit` does not obtain forwardable TGTs unless it is invoked with `kinit -f`. Notice, by the way, that `kinit` does not have a `-F` option. TGTs are either forwardable or not.

In the following figure, the user david obtains forwardable TGTs with `kinit -f`. In scenario 3, he is able to reach machine C because he uses a forwardable ticket with `rlogin`. In scenario 4, the second `rlogin` fails because the ticket is not reforwardable. By using the `-F` option instead, as in scenario 5, the second `rlogin` succeeds and the ticket can be reforwarded on to machine D.
Using Kerberized Commands (Examples)

The following examples show how the options to the Kerberized commands work.

EXAMPLE 24–5 Using the -a, -f, and -x Options With telnet

In this example, the user david has already logged in, and wants to telnet to the machine denver.example.com. He uses the -f option to forward his existing tickets, the -x option to encrypt the session, and the -a option to perform the login automatically. Because he does not plan to use the services of a third host, he can use -f instead of -F.

```
% telnet -a -f -x denver.example.com
Trying 128.0.0.5...
Connected to denver.example.com. Escape character is ‘^]’.
[ Kerberos V5 accepts you as "david@eng.example.com"]
[ Kerberos V5 accepted forwarded credentials ]
SunOS 5.9: Tue May 21 00:31:42 EDT 2004 Welcome to SunOS
```

Notice that david’s machine used Kerberos to authenticate him to denver.example.com, and logged him in automatically as himself. He had an encrypted session, a copy of his tickets already waiting for him, and he never had to type his password. If he had used a non-Kerberos version of telnet, he would have been prompted for his password, and it would have been sent over the network unencrypted. If an intruder had been watching network traffic at the time, the intruder would have known david’s password.
EXAMPLE 24–5 Using the -a, -f, and -x Options With telnet  (Continued)

If you forward your Kerberos tickets, telnet (as well as the other commands discussed here) destroys them when it exits.

EXAMPLE 24–6 Using rlogin With the -F Option

Here, the user jennifer wants to log in to her own machine, boston.example.com. She forwards her existing tickets with the -F option, and encrypts the session with the -x option. She chooses -F rather than -f because after she is logged in to boston, she might want to perform other network transactions requiring tickets to be reforwarded. Also, because she is forwarding her existing tickets, she does not have to type her password.

% rlogin boston.example.com -F -x
This rlogin session is using encryption for all transmissions.  
Last login Mon May 19 15:19:49 from daffodil  
SunOS Release 5.9 (GENERIC) #2 Tue Nov 14 18:09:3 EST 2003  
%

EXAMPLE 24–7 Setting the Protection Level in ftp

Suppose that joe wants to use ftp to get his mail from the directory ~joe/MAIL from the machine denver.example.com, encrypting the session. The exchange would look like the following:

% ftp -f denver.example.com
Connected to denver.example.com
220 denver.example.org FTP server (Version 6.0) ready.
334 Using authentication type GSSAPI; ADAT must follow
GSSAPI accepted as authentication type
GSSAPI authentication succeeded Name (daffodil.example.org:joe)
232 GSSAPI user joe@MELPOMENE.EXAMPLE.COM is authorized as joe
230 User joe logged in.
Remote system type is UNIX.
Using BINARY mode to transfer files.
ftp> protect private
200 Protection level set to Private
ftp> cd ~joe/MAIL
250 CWD command successful.
ftp> get RMAIL
227 Entering Passive Mode (128,0,0,5,16,49)
150 Opening BINARY mode data connection for RMAIL (158336 bytes).
226 Transfer complete. 158336 bytes received in 1.9 seconds (1.4e+02 Kbytes/s)
ftp> quit
%

To encrypt the session, joe sets the protection level to private.
This chapter lists many of the files, commands, and daemons that are part of the Kerberos product. In addition, this chapter provides detailed information about how Kerberos authentication works.

This is a list of the reference information in this chapter.

- "Kerberos Files" on page 507
- "Kerberos Commands" on page 509
- "Kerberos Daemons" on page 510
- "Kerberos Terminology" on page 510
- "How the Kerberos Authentication System Works" on page 516
- "Gaining Access to a Service Using Kerberos" on page 516
- "Using Kerberos Encryption Types" on page 519
- "Using the gsscred Table" on page 521
- "Notable Differences Between Oracle Solaris Kerberos and MIT Kerberos" on page 521

Kerberos Files

This section lists some of the files that are used by the Kerberos service.

<table>
<thead>
<tr>
<th>File Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>~/.gkadmin</td>
<td>Default values for creating new principals in the SEAM Tool</td>
</tr>
<tr>
<td>~/.k5login</td>
<td>List of principals that grant access to a Kerberos account</td>
</tr>
<tr>
<td>/etc/krb5/kadm5.acl</td>
<td>Kerberos access control list file, which includes principal names of KDC administrators and their Kerberos administration privileges</td>
</tr>
<tr>
<td>File Name</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>---------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><code>/etc/krb5/kadm5.keytab</code></td>
<td>Obsolete: This file was removed in the Oracle Solaris 11 release.</td>
</tr>
<tr>
<td><code>/etc/krb5/kdc.conf</code></td>
<td>KDC configuration file</td>
</tr>
<tr>
<td><code>/etc/krb5/kpropd.acl</code></td>
<td>Kerberos database propagation configuration file</td>
</tr>
<tr>
<td><code>/etc/krb5/krb5.conf</code></td>
<td>Kerberos realm configuration file</td>
</tr>
<tr>
<td><code>/etc/krb5/krb5.keytab</code></td>
<td>Keytab file for network application servers</td>
</tr>
<tr>
<td><code>/etc/krb5/warn.conf</code></td>
<td>Kerberos ticket expiration warning and automatic renewal configuration file</td>
</tr>
<tr>
<td><code>/etc/pam.conf</code></td>
<td>PAM configuration file</td>
</tr>
<tr>
<td><code>/tmp/krb5cc_uid</code></td>
<td>Default credentials cache, where <code>uid</code> is the decimal UID of the user</td>
</tr>
<tr>
<td><code>/tmp/ovsec_adm.xxxxxx</code></td>
<td>Temporary credentials cache for the lifetime of the password changing operation, where <code>xxxxxx</code> is a random string</td>
</tr>
<tr>
<td><code>/var/krb5/.k5.REALM</code></td>
<td>KDC stash file, which contains a copy of the KDC master key</td>
</tr>
<tr>
<td><code>/var/krb5/kadmin.log</code></td>
<td>Log file for kadmind</td>
</tr>
<tr>
<td><code>/var/krb5/kdc.log</code></td>
<td>Log file for the KDC</td>
</tr>
<tr>
<td><code>/var/krb5/principal</code></td>
<td>Kerberos principal database</td>
</tr>
<tr>
<td><code>/var/krb5/principal.kadm5</code></td>
<td>Kerberos administrative database, which contains policy information</td>
</tr>
<tr>
<td><code>/var/krb5/principal.kadm5.lock</code></td>
<td>Kerberos administrative database lock file</td>
</tr>
<tr>
<td><code>/var/krb5/principal.ok</code></td>
<td>Kerberos principal database initialization file that is created when the Kerberos database is initialized successfully</td>
</tr>
<tr>
<td><code>/var/krb5/principal.ulog</code></td>
<td>Kerberos update log, which contains updates for incremental propagation</td>
</tr>
<tr>
<td><code>/var/krb5/slave_datatrans</code></td>
<td>Backup file of the KDC that the <code>kprop_script</code> script uses for propagation</td>
</tr>
<tr>
<td><code>/var/krb5/slave_datatrans_slave</code></td>
<td>Temporary dump file that is created when full updates are made to the specified <code>slave</code></td>
</tr>
<tr>
<td><code>/var/user/$USER/krb-warn.conf</code></td>
<td>Per-user ticket expiration warning and automatic renewal configuration file</td>
</tr>
</tbody>
</table>
### Kerberos Commands

This section lists some commands that are included in the Kerberos product.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>/usr/bin/ftp</td>
<td>File Transfer Protocol program</td>
</tr>
<tr>
<td>/usr/bin/kdestroy</td>
<td>Destroys Kerberos tickets</td>
</tr>
<tr>
<td>/usr/bin/kinit</td>
<td>Obtains and caches Kerberos ticket-granting tickets</td>
</tr>
<tr>
<td>/usr/bin/klist</td>
<td>Displays current Kerberos tickets</td>
</tr>
<tr>
<td>/usr/bin/kpasswd</td>
<td>Changes a Kerberos password</td>
</tr>
<tr>
<td>/usr/bin/ktutil</td>
<td>Manages Kerberos keytab files</td>
</tr>
<tr>
<td>/usr/bin/kvno</td>
<td>Lists key version numbers for Kerberos principals</td>
</tr>
<tr>
<td>/usr/bin/rcp</td>
<td>Remote file copy program</td>
</tr>
<tr>
<td>/usr/bin/rlogin</td>
<td>Remote login program</td>
</tr>
<tr>
<td>/usr/bin/rsh</td>
<td>Remote shell program</td>
</tr>
<tr>
<td>/usr/bin/scp</td>
<td>Secure remote file copy program</td>
</tr>
<tr>
<td>/usr/bin/sftp</td>
<td>Secure file transfer program</td>
</tr>
<tr>
<td>/usr/bin/ssh</td>
<td>Secure remote login program</td>
</tr>
<tr>
<td>/usr/bin/telnet</td>
<td>Kerberized telnet program</td>
</tr>
<tr>
<td>/usr/lib/krb5/kprop</td>
<td>Kerberos database propagation program</td>
</tr>
<tr>
<td>/usr/sbin/gkadmin</td>
<td>Kerberos database administration GUI program, which is used to manage principals and policies</td>
</tr>
<tr>
<td>/usr/sbin/gsscred</td>
<td>Manage gsscred table entries</td>
</tr>
<tr>
<td>/usr/sbin/kadmin</td>
<td>Remote Kerberos database administration program (run with Kerberos authentication), which is used to manage principals, policies, and keytab files</td>
</tr>
<tr>
<td>/usr/sbin/kadmin.local</td>
<td>Local Kerberos database administration program (run without Kerberos authentication and must be run on master KDC), which is used to manage principals, policies, and keytab files</td>
</tr>
<tr>
<td>/usr/sbin/kclient</td>
<td>Kerberos client installation script which is used with or without an installation profile</td>
</tr>
<tr>
<td>/usr/sbin/kdb5_ldap_util</td>
<td>Creates LDAP containers for Kerberos databases</td>
</tr>
</tbody>
</table>
Kerberos Commands

(Continued)

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>/usr/sbin/kdb5_util</td>
<td>Creates Kerberos databases and stash files</td>
</tr>
<tr>
<td>/usr/sbin/kgcmgr</td>
<td>Configures Kerberos master and slave KDCs</td>
</tr>
<tr>
<td>/usr/sbin/kproplog</td>
<td>Lists a summary of update entries in the update log</td>
</tr>
</tbody>
</table>

Kerberos Daemons

The following table lists the daemons that the Kerberos product uses.

<table>
<thead>
<tr>
<th>Daemon</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>/usr/lib/inet/proftpd</td>
<td>Secure File Transfer Protocol daemon</td>
</tr>
<tr>
<td>/usr/lib/ssh/sshd</td>
<td>Secure shell daemon</td>
</tr>
<tr>
<td>/usr/lib/krb5/kadmind</td>
<td>Kerberos database administration daemon</td>
</tr>
<tr>
<td>/usr/lib/krb5/kpropd</td>
<td>Kerberos database propagation daemon</td>
</tr>
<tr>
<td>/usr/lib/krb5/krb5kdc</td>
<td>Kerberos ticket processing daemon</td>
</tr>
<tr>
<td>/usr/lib/krb5/ktkt_warnd</td>
<td>Kerberos ticket expiration warning and automatic renewal daemon</td>
</tr>
<tr>
<td>/usr/sbin/in.rlogind</td>
<td>Remote login daemon</td>
</tr>
<tr>
<td>/usr/sbin/in.rshd</td>
<td>Remote shell daemon</td>
</tr>
<tr>
<td>/usr/sbin/in.telnetd</td>
<td>telnet daemon</td>
</tr>
</tbody>
</table>

Kerberos Terminology

The following section presents Kerberos terms and their definitions. These terms are used throughout the Kerberos documentation. To grasp Kerberos concepts, an understanding of these terms is essential.

Kerberos-Specific Terminology

You need to understand the terms in this section in order to administer KDCs.

The Key Distribution Center or KDC is the component of Kerberos that is responsible for issuing credentials. These credentials are created by using information that is stored in the KDC
database. Each realm needs at least two KDCs, a master and at least one slave. All KDCs generate credentials, but only the master KDC handles any changes to the KDC database.

A *stash file* contains the master key for the KDC. This key is used when a server is rebooted to automatically authenticate the KDC before starting the *kadmind* and *krb5kdc* commands. Because this file includes the master key, the file and any backups of the file should be kept secure. The file is created with read-only permissions for *root*. To keep the file secure, do not change the permissions. If the file is compromised, then the key could be used to access or modify the KDC database.

**Authentication-Specific Terminology**

You need to know the terms in this section to understand the authentication process. Programmers and system administrators should be familiar with these terms.

A *client* is the software that runs on a user's workstation. The Kerberos software that runs on the client makes many requests during this process. So, differentiating the actions of this software from the user is important.

The terms *server* and *service* are often used interchangeably. To clarify, the term *server* is used to define the physical system that Kerberos software is running on. The term *service* corresponds to a particular function that is being supported on a server (for example, *ftp* or *nfs*). Documentation often mentions servers as part of a service, but this definition clouds the meaning of the terms. Therefore, the term *server* refers to the physical system. The term *service* refers to the software.

The Kerberos product uses two types of keys. One type of key is a password derived key. The password derived key is given to each user principal and is known only to the user and to the KDC. The other type of key used by the Kerberos product is a random key that is not associated with a password and so is not suitable for use by user principals. Random keys are typically used for service principals that have entries in a keytab and session keys generated by the KDC. Service principals can use random keys since the service can access the key in the keytab which allows it to run non-interactively. Session keys are generated by the KDC (and shared between the client and service) to provide secure transactions between a client and a service.

A *ticket* is an information packet that is used to securely pass the identity of a user to a server or service. A ticket is valid for only a single client and a particular service on a specific server. A ticket contains:

- Principal name of the service
- Principal name of the user
- IP address of the user's host
- Timestamp
- Value which defines the lifetime of the ticket
- Copy of the session key
All of this data is encrypted in the server's service key. Note, the KDC issues the ticket embedded in a credential described below. After a ticket has been issued, it can be reused until the ticket expires.

A credential is a packet of information that includes a ticket and a matching session key. The credential is encrypted with the requesting principal's key. Typically, the KDC generates a credential in response to a ticket request from a client.

An authenticator is information used by the server to authenticate the client user principal. An authenticator includes the principal name of the user, a timestamp, and other data. Unlike a ticket, an authenticator can be used once only, usually when access to a service is requested. An authenticator is encrypted by using the session key shared by the client and server. Typically, the client creates the authenticator and sends it with the server's or service's ticket in order to authenticate to the server or service.

**Types of Tickets**

Tickets have properties that govern how they can be used. These properties are assigned to the ticket when it is created, although you can modify a ticket's properties later. For example, a ticket can change from being forwardable to being forwarded. You can view ticket properties with the klist command. See "Viewing Kerberos Tickets" on page 493.

Tickets can be described by one or more of the following terms:

- **Forwardable/forwarded** A forwardable ticket can be sent from one host to another host, obviating the need for a client to reauthenticate itself. For example, if the user david obtains a forwardable ticket while on user jennifer's machine, he can log in to his own machine without having to get a new ticket (and thus authenticate himself again). See Example 24–1 for an example of a forwardable ticket.

- **Initial** An initial ticket is a ticket that is issued directly, not based on a ticket-granting ticket. Some services, such as applications that change passwords, can require tickets to be marked initial in order to assure themselves that the client can demonstrate a knowledge of its secret key. An initial ticket indicates that the client has recently authenticated itself, instead of relying on a ticket-granting ticket, which might have been around for a long time.

- **Invalid** An invalid ticket is a postdated ticket that has not yet become usable. An invalid ticket will be rejected by an application server until it becomes validated. To be validated, a ticket must be presented to the KDC by the client in a ticket-granting service request, with the VALIDATE flag set, after its start time has passed.
Postdatable/postdated

A postdated ticket is a ticket that does not become valid until some specified time after its creation. Such a ticket is useful, for example, for batch jobs that are intended to be run late at night, because the ticket, if stolen, cannot be used until the batch job is to be run. When a postdated ticket is issued, it is issued as invalid and remains that way until its start time has passed, and the client requests validation by the KDC. A postdated ticket is normally valid until the expiration time of the ticket-granting ticket. However, if the ticket is marked renewable, its lifetime is normally set to be equal to the duration of the full life of the ticket-granting ticket.

Proxiable/proxy

At times, it is necessary for a principal to allow a service to perform an operation on its behalf. The principal name of the proxy must be specified when the ticket is created. The Oracle Solaris release does not support proxiable or proxy tickets.

A proxiable ticket is similar to a forwardable ticket, except that it is valid only for a single service, whereas a forwardable ticket grants the service the complete use of the client’s identity. A forwardable ticket can therefore be thought of as a sort of super-proxy.

Renewable

Because it is a security risk to have tickets with very long lives, tickets can be designated as renewable. A renewable ticket has two expiration times: the time at which the current instance of the ticket expires, and the maximum lifetime for any ticket, which is one week. If a client wants to continue to use a ticket, the client renews it before the first expiration occurs. For example, a ticket can be valid for one hour, with all tickets having a maximum lifetime of 10 hours. If the client that is holding the ticket wants to keep it for more than an hour, the client must renew it within that hour. When a ticket reaches the maximum ticket lifetime (10 hours), it automatically expires and cannot be renewed.

For information on how to view the attributes of tickets, see “Viewing Kerberos Tickets” on page 493.

Ticket Lifetimes

Any time a principal obtains a ticket, including a ticket-granting ticket (TGT), the ticket’s lifetime is set as the smallest of the following lifetime values:

- The lifetime value that is specified by the -l option of kinit, if kinit is used to get the ticket. By default, kinit used the maximum lifetime value.
- The maximum lifetime value (max_life) that is specified in the kdc.conf file.
- The maximum lifetime value that is specified in the Kerberos database for the service principal that provides the ticket. In the case of `kinit`, the service principal is `krbtgt/realm`.

- The maximum lifetime value that is specified in the Kerberos database for the user principal that requests the ticket.

Figure 25–1 shows how a TGT’s lifetime is determined and where the four lifetime values come from. Even though this figure shows how a TGT’s lifetime is determined, basically the same thing happens when any principal obtains a ticket. The only differences are that `kinit` doesn’t provide a lifetime value, and the service principal that provides the ticket provides a maximum lifetime value (instead of the `krbtgt/realm` principal).

**FIGURE 25–1  How a TGT’s Lifetime is Determined**

The renewable ticket lifetime is also determined from the minimum of four values, but renewable lifetime values are used instead, as follows:

- The renewable lifetime value that is specified by the `-r` option of `kinit`, if `kinit` is used to obtain or renew the ticket.

- The maximum renewable lifetime value (`max_renewable_life`) that is specified in the `kdc.conf` file.

- The maximum lifetime renewable value that is specified in the Kerberos database for the service principal that provides the ticket. In the case of `kinit`, the service principal is `krbtgt/realm`. 
The maximum lifetime renewable value that is specified in the Kerberos database for the user principal that requests the ticket.

**Kerberos Principal Names**

Each ticket is identified by a principal name. The principal name can identify a user or a service. Here are examples of several principal names.

<table>
<thead>
<tr>
<th>Principal Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>changepw/kdc1.example.com@EXAMPLE.COM</td>
<td>A principal for the master KDC server that allows access to the KDC when you are changing passwords.</td>
</tr>
<tr>
<td>clntconfig/admin@EXAMPLE.COM</td>
<td>A principal that is used by the kclient installation utility.</td>
</tr>
<tr>
<td>ftp/boston.example.com@EXAMPLE.COM</td>
<td>A principal used by the ftp service. This principal can be used instead of a host principal.</td>
</tr>
<tr>
<td>host/boston.example.com@EXAMPLE.COM</td>
<td>A principal that is used by the Kerberized applications (klist and kprop, for example) and services (such as ftp and telnet). This principal is called a host or service principal. The principal is used to authenticate NFS mounts. This principal is also used by a client to verify that the TGT that is issued to the client is from the correct KDC.</td>
</tr>
<tr>
<td>K/M@EXAMPLE.COM</td>
<td>The master key name principal. One master key name principal is associated with each master KDC.</td>
</tr>
<tr>
<td>kadmin/history@EXAMPLE.COM</td>
<td>A principal that includes a key used to keep password histories for other principals. Each master KDC has one of these principals.</td>
</tr>
<tr>
<td>kadmin/kdc1.example.com@EXAMPLE.COM</td>
<td>A principal for the master KDC server that allows access to the KDC by using kadmin.</td>
</tr>
<tr>
<td>kadmin/changepw.example.com@EXAMPLE.COM</td>
<td>A principal that is used to accept password change requests from clients that are not running an Oracle Solaris release.</td>
</tr>
<tr>
<td>krbtgt/EXAMPLE.COM@EXAMPLE.COM</td>
<td>This principal is used when you generate a ticket-granting ticket.</td>
</tr>
<tr>
<td>krbtgt/EAST.EXAMPLE.COM@WEST.EXAMPLE.COM</td>
<td>This principal is an example of a cross-realm ticket-granting ticket.</td>
</tr>
<tr>
<td>nfs/boston.example.com@EXAMPLE.COM</td>
<td>A principal that is used by the NFS service. This principal can be used instead of a host principal.</td>
</tr>
<tr>
<td>root/boston.example.com@EXAMPLE.COM</td>
<td>A principal that is associated with the root account on a client. This principal is called a root principal and provides root access to NFS mounted file systems.</td>
</tr>
<tr>
<td><a href="mailto:username@EXAMPLE.COM">username@EXAMPLE.COM</a></td>
<td>A principal for a user.</td>
</tr>
<tr>
<td>username/admin@EXAMPLE.COM</td>
<td>An admin principal that can be used to administer the KDC database.</td>
</tr>
</tbody>
</table>
How the Kerberos Authentication System Works

Applications allow you to log in to a remote system if you can provide a ticket that proves your identity, and a matching session key. The session key contains information that is specific to the user and the service that is being accessed. A ticket and session key are created by the KDC for all users when they first log in. The ticket and the matching session key form a credential. While using multiple networking services, a user can gather many credentials. The user needs to have a credential for each service that runs on a particular server. For example, access to the ftp service on a server named boston requires one credential. Access to the ftp service on another server requires its own credential.

The process of creating and storing the credentials is transparent. Credentials are created by the KDC that sends the credential to the requester. When received, the credential is stored in a credential cache.

How the Kerberos Service Interacts With DNS and the nsswitch Service

The Kerberos service is compiled to use DNS to resolve host names. The nsswitch service is not checked at all when host name resolution is done.

Gaining Access to a Service Using Kerberos

To access a specific service on a specific server, the user must obtain two credentials. The first credential is for the ticket-granting ticket (known as the TGT). Once the ticket-granting service has decrypted this credential, the service creates a second credential for the server that the user is requesting access to. This second credential can then be used to request access to the service on the server. After the server has successfully decrypted the second credential, then the user is given access. The following sections describe this process in more detail.

Obtaining a Credential for the Ticket-Granting Service

1. To start the authentication process, the client sends a request to the authentication server for a specific user principal. This request is sent without encryption. No secure information is included in the request, so it is not necessary to use encryption.

2. When the request is received by the authentication service, the principal name of the user is looked up in the KDC database. If a principal matches the entry in the database, the authentication service obtains the private key for that principal. The authentication service then generates a session key to be used by the client and the ticket-granting service (call it Session key 1) and a ticket for the ticket-granting service (Ticket 1). This ticket is also known
as the ticket-granting ticket (TGT). Both the session key and the ticket are encrypted by using the user’s private key, and the information is sent back to the client.

3. The client uses this information to decrypt Session Key 1 and Ticket 1, by using the private key for the user principal. Because the private key should only be known by the user and the KDC database, the information in the packet should be safe. The client stores the information in the credentials cache.

During this process, a user is normally prompted for a password. If the password the user specifies is the same as the password that was used to build the private key stored in the KDC database, then the client can successfully decrypt the information that is sent by the authentication service. Now the client has a credential to be used with the ticket-granting service. The client is ready to request a credential for a server.

**FIGURE 25–2** Obtaining a Credential for the Ticket-Granting Service

1. Client requests credential for server.
2. Authentication service sends back credential, which includes Session Key 1 and TGT.
3. Client decrypts credential with entered password.

**TGT** = Ticket-granting ticket
**KDC** = Key Distribution Center

---

**Obtaining a Credential for a Server**

1. To request access to a specific server, a client must first have obtained a credential for that server from the authentication service. See “Obtaining a Credential for the Ticket-Granting Service” on page 516. The client then sends a request to the ticket-granting service, which includes the service principal name, Ticket 1, and an authenticator that was encrypted with Session Key 1. Ticket 1 was originally encrypted by the authentication service by using the service key of the ticket-granting service.
2. Because the service key of the ticket-granting service is known to the ticket-granting service, Ticket 1 can be decrypted. The information in Ticket 1 includes Session Key 1, so the ticket-granting service can decrypt the authenticator. At this point, the user principal is authenticated with the ticket-granting service.

3. Once the authentication is successful, the ticket-granting service generates a session key for the user principal and the server (Session Key 2), and a ticket for the server (Ticket 2). Session Key 2 and Ticket 2 are then encrypted by using Session Key 1. Because Session Key 1 is known only to the client and the ticket-granting service, this information is secure and can be safely sent over the network.

4. When the client receives this information packet, the client decrypts the information by using Session Key 1, which it had stored in the credential cache. The client has obtained a credential to be used with the server. Now the client is ready to request access to a particular service on that server.

**Obtaining Access to a Specific Service**

1. To request access to a specific service, the client must first have obtained a credential for the ticket-granting service from the authentication server, and a server credential from the ticket-granting service. See “Obtaining a Credential for the Ticket-Granting Service” on
The client can then send a request to the server including Ticket 2 and another authenticator. The authenticator is encrypted by using Session Key 2.

2. Ticket 2 was encrypted by the ticket-granting service with the service key for the service. Because the service key is known by the service principal, the service can decrypt Ticket 2 and get Session Key 2. Session Key 2 can then be used to decrypt the authenticator. If the authenticator is successfully decrypted, the client is given access to the service.

**FIGURE 25–4  Obtaining Access to a Specific Service**

Using Kerberos Encryption Types

Encryption types identify which cryptographic algorithms and mode to use when cryptographic operations are performed. The aes, des3-cbc-sha1 and rc4-hmac encryption types enable the creation of keys that can be used for higher strength cryptographic operations. These higher strength operations enhance the overall security of the Kerberos service.

When a client requests a ticket from the KDC, the KDC must use keys whose encryption type is compatible with both the client and the server. While the Kerberos protocol allows the client to request that the KDC use particular encryption types for the client's part of the ticket reply, the protocol does not allow the server to specify encryption types to the KDC.

**Note** – If you have a master KDC installed that is not running the Solaris 10 release, the slave KDCs must be upgraded to the Solaris 10 release before you upgrade the master KDC. A Solaris 10 master KDC will use the new encryption types, which an older slave will not be able to handle.

The following lists some of the issues that must be considered before you change the encryption types.
The KDC assumes that the first key/encryption-type associated with the server principal entry in the principal database is supported by the server.

On the KDC, you should make sure that the keys generated for the principal are compatible with the systems on which the principal will be authenticated. By default, the kadmin command creates keys for all supported encryption types. If the systems that the principal is used on do not support this default set of encryption types, then you should restrict the encryption types when creating a principal. You can restrict the encryption types through use of the -e flag in kadmin addprinc or by setting the supported_enctypes parameter in the kdc.conf file to this subset. The supported_enctypes parameter should be used when most of the systems in a Kerberos realm support a subset of the default set of encryption types. Setting supported_enctypes specifies the default set of encryption types kadmin addprinc uses when it creates a principal for a particular realm. As a general rule, it is best to control the encryption types used by Kerberos using one of these two methods.

When determining the encryption types a system supports, consider both the version of Kerberos running on the system as well as the cryptographic algorithms supported by the server application for which a server principal is being created. For example, when creating an nfs/hostname service principal, you should restrict the encryption types to the types supported by the NFS server on that host. Note that in the Solaris 10 release, all supported Kerberos encryption types are also supported by the NFS server.

The master_key_enctype parameter in the kdc.conf file can be used to control the encryption type of the master key that encrypts the entries in the principal database. Do not use this parameter if the KDC principal database has already been created. The master_key_enctype parameter can be used at database creation time to change the default master key encryption type from des-cbc-crc to a stronger encryption type. Make sure that all slave KDCs support the chosen encryption type and that they have an identical master_key_enctype entry in their kdc.conf when configuring the slave KDCs. Also, make sure that the master_key_enctype is set to one of the encryption types in supported_enctypes, if supported_enctypes is set in kdc.conf. If either of these issues are not handled properly, then the master KDC might not be able to work with the slave KDCs.

On the client, you can control which encryption types the client requests when getting tickets from the KDC through a couple of parameters in krb5.conf. The default_tkt_enctypes parameter specifies the encryption types the client is willing to use when the client requests a ticket-granting ticket (TGT) from the KDC. The TGT is used by the client to acquire other server tickets in a more efficient manner. The effect of setting default_tkt_enctypes is to give the client some control over the encryption types used to protect the communication between the client and KDC when the client requests a server ticket using the TGT (this is called a TGS request). Note, that the encryption types specified in default_tkt_enctypes must match at least one of the principal key encryption types in the principal database stored on the KDC. Otherwise, the TGT request will fail. In most situations, it is best not to set default_tkt_enctypes because this parameter can be a source of interoperability problems. By default, the client code requests that all supported encryption types and the KDC choose the encryption types based on the keys the KDC finds in the principal database.
The `default_tgs_enctypes` parameter restricts the encryption types the client requests in its TGS requests, which are used to acquire server tickets. This parameter also restricts the encryption types the KDC uses when creating the session key that the client and server share. For example, if a client wants to only use 3DES encryption when doing secure NFS, you should set `default_tgs_enctypes = des3-cbc-sha1`. Make sure that the client and server principals have a `des-3-cbc-sha1` key in the principal database. As with `default_tkt_enctype`, it is probably best in most cases not to set this because it can cause interoperability problems if the credentials are not setup properly both on the KDC and the server.

On the server, you can control the encryption types accepted by the server with the `permitted_enctypes` in `kdc.conf`. In addition, you can specify the encryption types used when creating keytab entries. Again, it is generally best not to use either of these methods to control encryption types and instead let the KDC determine the encryption types to use because the KDC does not communicate with the server application to determine which key or encryption type to use.

### Using the `gsscred` Table

The `gsscred` table is used by an NFS server when the server is trying to identify a Kerberos user, if the default mappings are not sufficient. The NFS service uses UNIX IDs to identify users. These IDs are not part of a user principal or a credential. The `gsscred` table provides additional mapping from GSS credentials to UNIX UIDs (from the password file). The table must be created and administered after the KDC database is populated. See "Mapping GSS Credentials to UNIX Credentials" on page 349 for more information.

When a client request comes in, the NFS service tries to map the credential name to a UNIX ID. If the mapping fails, the `gsscred` table is checked.

### Notable Differences Between Oracle Solaris Kerberos and MIT Kerberos

The Solaris 10 version of the Kerberos service is based on MIT Kerberos version 1.2.1. The following lists the enhancements included in the Solaris 10 release that are not included in the MIT 1.2.1 version:

- Kerberos support of Oracle Solaris remote applications
- Incremental propagation for the KDC database
- Client configuration script
- Localized error messages
- Oracle Solaris audit record support
- Thread safe use of Kerberos using GSS-API
- Use of the Cryptographic Framework for cryptography
This version also includes some post MIT 1.2.1 bug fixes. In particular, 1.2.5 btree bug fixes and 1.3 TCP support have been added.
PART VII

Auditing in Oracle Solaris

This section provides information about the configuration, management, and use of the auditing subsystem.

- Chapter 26, "Auditing (Overview)"
- Chapter 27, "Planning for Auditing"
- Chapter 28, "Managing Auditing (Tasks)"
- Chapter 29, "Auditing (Reference)"
The auditing subsystem of Oracle Solaris keeps a record of how the system is being used. The audit service includes tools to assist with the analysis of the auditing data.

This chapter introduces how auditing works in Oracle Solaris:

- “What Is Auditing?” on page 525
- “Audit Terminology and Concepts” on page 526
- “How Is Auditing Related to Security?” on page 534
- “How Does Auditing Work?” on page 535
- “How Is Auditing Configured?” on page 536
- “Auditing on a System With Oracle Solaris Zones” on page 537
- “About the Audit Service in This Release” on page 538

For planning suggestions, see Chapter 27, “Planning for Auditing.” For procedures to configure auditing at your site, see Chapter 28, “Managing Auditing (Tasks).” For reference information, see Chapter 29, “Auditing (Reference).”

**What Is Auditing?**

Auditing is the collecting of data about the use of system resources. The audit data provides a record of security-related system events. This data can then be used to assign responsibility for actions that take place on a host. Successful auditing starts with two security features: identification and authentication. At each login, after a user supplies a user name and PAM authentication succeeds, a unique and immutable audit user ID is generated and associated with the user, and a unique audit session ID is generated and associated with the user’s process. The audit session ID is inherited by every process that is started during that login session. When a user switches to another user, all user actions are tracked with the same audit user ID. For more details about switching identity, see the su(1M) man page. Note that by default, certain actions such as booting and shutting down the system are always audited.
The audit service makes the following possible:

- Monitoring security-relevant events that take place on the host
- Recording the events in a network-wide audit trail
- Detecting misuse or unauthorized activity
- Reviewing patterns of access and the access histories of individuals and objects
- Discovering attempts to bypass the protection mechanisms
- Discovering extended use of privilege that occurs when a user changes identity

## Audit Terminology and Concepts

The following terms are used to describe the audit service. Some definitions include pointers to more complete descriptions.

- **audit class**: A grouping of audit events. Audit classes provide a way to select a group of events to be audited.
  
  For more information, see "Audit Classes and Preselection" on page 529, and the `audit_flags(5)`, `audit_class(4)`, and `audit_event(4)` man pages.

- **audit file system**: A repository of audit files in binary format.
  
  For more information, see “Audit Logs” on page 531 and the `audit.log(4)` man page.

- **audit event**: A security-related system action that is auditable. For ease of selection, events are grouped into audit classes.
  
  For more information, see “Audit Events” on page 528 and the `audit_event(4)` man page.

- **audit flag**: An audit class that is supplied as an argument to a command or keyword. A flag can be prefixed by a plus sign or minus sign to indicate that the class is audited for success (+) or failure (-). A preceding caret (^) indicates that a success is not to be audited (^+) or a failure is not to be audited (^-).
  
  For more information, see the `audit_flags(5)` man page and “Audit Class Syntax” on page 615.

- **audit plugin**: A module that transfers the audit records in the queue to a specified location. The `audit_binfile` plugin creates binary audit files. Binary files comprise the audit trail, which is stored on audit file systems. The
audit_remote plugin sends binary audit records to a remote repository. The audit_syslog plugin summarizes selected audit records in the syslog logs.

For more information, see "Audit Plugin Modules" on page 531 and the module man pages, audit_binfile(5), audit_remote(5), and audit_syslog(5).

audit policy A set of auditing options that you can enable or disable at your site. These options include whether to record certain kinds of audit data. The options also include whether to suspend auditable actions when the audit queue is full.

For more information, see "Understanding Audit Policy" on page 546 and the auditconfig(1M) man page.

audit record Audit data that is collected in the audit queue. An audit record describes a single audit event. Each audit record is composed of audit tokens.

For more information, see "Audit Records and Audit Tokens" on page 530 and the audit.log(4) man page.

audit token A field of an audit record or event. Each audit token describes an attribute of an audit event, such as a user, a program, or other object.

For more information, see "Audit Token Formats" on page 622 and the audit.log(4) man page.

audit trail A collection of one or more audit files that store the audit data from all audited systems that use the default plugin, audit_binfile.

For more information, see "Audit Trail" on page 620.

local auditing The collecting of audit records that are generated on the local system. The records can be generated in the global zone or in non-global zones, or both.

For more information, see "Audit Plugin Modules" on page 531.

post-selection The choice of which audit events to examine in the audit trail. The default active plugin, audit_binfile, creates the audit trail. A post-selection tool, the auditreduce command, selects records from the audit trail.

For more information, see the auditreduce(1M) and praudit(1M) man pages.

preselection The choice of which audit classes to monitor. The audit events of preselected audit classes are collected in the audit queue. Audit classes that are not preselected are not audited, so their events do not appear in the queue.
For more information, see “Audit Classes and Preselection” on page 529 and the audit_flags(5) and auditconfig(1M) man pages.

public object

A file that is owned by the root user and readable by the world. For example, files in the /etc directory and the /usr/bin directory are public objects. Public objects are not audited for read-only events. For example, even if the file_read (fr) audit class is preselected, the reading of public objects is not audited. You can override the default by changing the public audit policy option.

remote auditing

The Audit Remote Server (ARS) that receives and stores audit records from a system that is being audited and is configured with an active audit_remote plugin. To distinguish an audited system from an ARS, the audited system can be referred to as the “locally audited system”.

For more information, see the -setremote options on the auditconfig(1M) man page and “Audit Remote Server” on page 617.

Audit Events

Audit events represent auditable actions on a system. Audit events are listed in the /etc/security/audit_event file. Each audit event is connected to a system call or user command, and is assigned to one or more audit classes. For a description of the format of the audit_event file, see the audit_event(4) man page.

For example, the AUE_EXECVE audit event audits the execve() system call. The command auditrecord -e execve displays this entry:

```
execve  
  system call execve  See execve(2)  
  event ID 23 AUE EXECVE  
  class ps,ex  (0x0000000040100000)  
  header path  
  [attribute] omitted on error  
  [exec_arguments] output if argv policy is set  
  [exec_environment] output if arge policy is set  
  subject [use_of_privilege]  
  return  
```

When you preselect either the audit class ps or the audit class ex, then every execve() system call is recorded in the audit queue.
Auditing handles *attributable* and *non-attributable* events. Audit policy divides events into *synchronous* and *asynchronous* events, as follows:

- **Attributable events** – Events that can be attributed to a user. The `execve()` system call can be attributed to a user, so the call is considered an attributable event. All attributable events are synchronous events.

- **Non-attributable events** – Events that occur at the kernel-interrupt level or before a user is authenticated. The `na` audit class handles audit events that are non-attributable. For example, booting the system is a non-attributable event. Most non-attributable events are asynchronous events. However, non-attributable events that have associated processes, such as failed login, are synchronous events.

- **Synchronous events** – Events that are associated with a process in the system. Synchronous events are the majority of system events.

- **Asynchronous events** – Events that are not associated with any process, so no process is available to be blocked and later woken up. Initial system boot and PROM enter and exit events are examples of asynchronous events.

In addition to the audit events that are defined by the audit service, third-party applications can generate audit events. Audit event numbers from 32768 to 65535 are available for third-party applications. Vendors need to contact their Oracle Solaris representative to reserve event numbers and obtain access to the audit interfaces.

### Audit Classes and Preselection

Each audit event belongs to an *audit class* or classes. Audit classes are convenient containers for large numbers of audit events. When you *preselect* a class to be audited, all the events in that class are recorded in the audit queue. For example, when you preselect the `ps` audit class, `execve()`, `fork()`, and other system calls are recorded.

You can preselect for events on a system and for events initiated by a particular user.

- **System-wide preselection** – Specify the system-wide defaults for auditing by using the `-setflags` and `-setnaflags` options to the `auditconfig` command.

**Note** – If the `perzone` policy is set, default audit classes can be specified in every zone. For `perzone` auditing, the defaults are zone-wide, not system-wide.

- **User-specific preselection** – Specify differences from the system-wide auditing defaults for individual users by configuring the audit flags for the user. The `useradd`, `roleadd`, `usermod`, and `rolemod` commands place the `audit_flags` security attribute in the `user_attr` database. The `profiles` command places audit flags for rights profiles in the `prof_attr` database.
The audit preselection mask determines which classes of events are audited for a user. For a description of the user preselection mask, see “Process Audit Characteristics” on page 619. For which configured audit flags are used, see “Order of Search for Assigned Security Attributes” on page 197.

Audit classes are defined in the /etc/security/audit_class file. Each entry contains the audit mask for the class, the name for the class, and a descriptive name for the class. For example, the lo and ps class definitions appear in the audit_class file as follows:

0x0000000000001000:lo:login or logout
0x0000000000100000:ps:process start/stop

The audit classes include the two global classes: all and no. The audit classes are described in the audit_class(4) man page. For the list of classes, read the /etc/security/audit_class file.

The mapping of audit events to classes is configurable. You can remove events from a class, add events to a class, and create a new class to contain selected events. For the procedure, see “How to Change an Audit Event’s Class Membership” on page 564. To view the events that are mapped to a class, use the auditrecord -c class command.

### Audit Records and Audit Tokens

Each audit record records the occurrence of a single audited event. The record includes information such as who did the action, which files were affected, what action was attempted, and where and when the action occurred. The following example shows a login audit record with three tokens, header, subject, and return:

```
header,69,2,login - local,,example_system,2010-10-10 10:10:10.020 -07:00
subject,jdoe,jdoe,staff,jdoe,staff,1218,4076076536,69 2 example_system
return,success,0
```

The type of information that is saved for each audit event is defined by a set of audit tokens. Each time an audit record is created for an event, the record contains some or all of the tokens that are defined for the event. The nature of the event determines which tokens are recorded. In the preceding example, each line begins with the name of the audit token. The content of the audit token follows the token name. Together, the header, subject, and return audit tokens comprise the login - local audit record. To display the tokens that comprise an audit record, use the auditrecord -e event command.

For a detailed description of the structure of each audit token with an example of praudit output, see “Audit Token Formats” on page 622. For a description of the binary stream of audit tokens, see the audit.log(4) man page.
Audit Plugin Modules

The audit plugin modules direct the audit records from the audit queue to a file or repository. At least one plugin must be active. By default, the audit_binfile plugin is active. You configure plugins with the auditconfig -set plugin plugin-name command.

The audit service provides the following plugins:

- **audit_binfile plugin** – Handles delivery of the audit queue to the binary audit files. For more information, see the audit.log(4) man page.
- **audit_remote plugin** – Handles secure delivery of binary audit records from the audit queue to a configured remote server. The audit_remote plugin uses the libgss() library to authenticate the server. The transmission is protected for privacy and integrity.
- **audit_syslog plugin** – Handles delivery of selected records from the audit queue to the syslog logs.

To configure a plugin, see the auditconfig(1M) man page. For examples of plugin configuration, see the tasks in “Configuring Audit Logs (Tasks)” on page 565. For information about the plugins, see the audit_binfile(5), audit_remote(5), and audit_syslog(5) man pages.

Audit Logs

Audit records are collected in audit logs. The audit service provides three output modes for audit records.

- Logs that are called audit files store audit records in binary format. The set of audit files from a system or site provides a complete audit record. The complete audit record is called the audit trail. These logs are created by the audit_binfile plugin, and can be reviewed by the praudit and auditreduce post-selection commands.
- The audit_remote plugin streams audit records to a remote repository. The repository is responsible for maintaining an audit trail and supplying post-selection tools.
- The syslog utility collects and stores text summaries of the audit record. A syslog record is not complete. The following example shows a syslog entry for a login audit record:

  Oct 10 10:10:20 example_system auditd: [ID 6472 audit.notice] \ 
  login - login ok session 4076172534 by root as root:other

A site can configure auditing to collect audit records in all formats. You can configure the systems at your site to use binary mode locally, to send binary files to a remote repository, and to use syslog mode. The following table compares binary audit records with syslog audit records.
### TABLE 26–1  Comparison of Binary, Remote, and syslog Audit Records

<table>
<thead>
<tr>
<th>Feature</th>
<th>Binary and Remote Records</th>
<th>syslog Records</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Protocol</strong></td>
<td>Binary – Writes to the file system</td>
<td>Uses UDP for remote logging</td>
</tr>
<tr>
<td></td>
<td>Remote – Streams to a remote repository</td>
<td></td>
</tr>
<tr>
<td><strong>Data type</strong></td>
<td>Binary</td>
<td>Text</td>
</tr>
<tr>
<td><strong>Record length</strong></td>
<td>No limit</td>
<td>Up to 1024 characters per audit record</td>
</tr>
<tr>
<td><strong>Location</strong></td>
<td>Binary – Stored in a zpool on the system</td>
<td>Stored in a location that is specified in the syslog.conf file</td>
</tr>
<tr>
<td></td>
<td>Remote – Remote repository</td>
<td></td>
</tr>
<tr>
<td><strong>How to configure</strong></td>
<td>Binary – Set the p_dir attribute on the audit_binfile plugin</td>
<td>Make the audit_syslog plugin active and configure the syslog.conf file</td>
</tr>
<tr>
<td></td>
<td>Remote – Set the p_hosts attribute on the audit_remote plugin and make the plugin active</td>
<td></td>
</tr>
<tr>
<td><strong>How to read</strong></td>
<td>Binary – Typically, in batch mode, browser output in XML</td>
<td>In real time or searched by scripts that you have created for syslog</td>
</tr>
<tr>
<td></td>
<td>Remote – Repository dictates the procedure</td>
<td>Plain text output</td>
</tr>
<tr>
<td><strong>Completeness</strong></td>
<td>Guaranteed to be complete and to appear in the correct order</td>
<td>Not guaranteed to be complete</td>
</tr>
<tr>
<td><strong>Time stamp</strong></td>
<td>Coordinated Universal Time (UTC)</td>
<td>Time on the system that is being audited</td>
</tr>
</tbody>
</table>

Binary records provide the greatest security and coverage. Binary output meets the requirements of security certifications, such as the [Common Criteria](http://www.commoncriteriaportal.org/) audit requirements.

The audit_binfile plugin writes the records to a file system that you protect from snooping. On a single system, all binary records are collected and displayed in order. The UTC time stamp on binary logs enables accurate comparison when systems on one audit trail are distributed across time zones. The praudit -x command enables you to view the records in a browser in XML. You can also use scripts to parse the XML output.

The audit_remote plugin writes the records to a remote repository. The repository handles storage and post-selection.

In contrast, the syslog records might provide greater convenience and flexibility. For example, you can collect the syslog data from a variety of sources. Also, when you monitor audit.notice events in the syslog.conf file, the syslog utility logs an audit record summary with the current time stamp. You can use the same management and analysis tools that you
have developed for syslog messages from a variety of sources, including workstations, servers, firewalls, and routers. The records can be viewed in real time, and can be stored on a remote system.

By using syslog.conf to store audit records remotely, you protect log data from alteration or deletion by an attacker. On the other hand, when audit records are stored remotely, the records are susceptible to network attacks such as denial of service and spoofed source addresses. Also, UDP can drop packets or can deliver packets out of order. The limit on syslog entries is 1024 characters, so some audit records could be truncated in the log. On a single system, not all audit records are collected. The records might not display in order. Because each audit record is stamped with the local system’s date and time, you cannot rely on the time stamp to construct an audit trail for several systems.

For more information about plugins and audit logs, refer to the following:
- audit_binfile(5) man page
- audit_syslog(5) man page
- audit.log(4) man page
- “How to Assign Audit Space for the Audit Trail” on page 569
- “How to Configure syslog Audit Logs” on page 577

Storing and Managing the Audit Trail

When the audit_binfile plugin is active, an audit file system holds audit files in binary format. A typical installation uses the /var/audit file system and can use additional file systems. The contents of all audit file systems comprise the audit trail. Audit records are stored in these file systems in the following order:

- **Primary audit file system**— The /var/audit file system, the default file system for audit files for a system
- **Secondary audit file systems**— File systems where the audit files for a system are placed at administrator discretion

The file systems are specified as arguments to the p_dir attribute of the audit_binfile plugin. A file system is not used until a file system that is earlier in the list is full. For an example with a list of file system entries, see “How to Create ZFS File Systems for Audit Files” on page 566.

Placing the audit files in the default audit root directory assists the audit reviewer when reviewing the audit trail. The auditreduce command uses the audit root directory to find all files in the audit trail. The default audit root directory is /var/audit. The -M option to the auditreduce command can be used to specify the audit files from a specific machine, and the -S option can be used to specify a different audit file system. For more information, see the auditreduce(1M) man page.
Ensuring Reliable Time Stamps

When you merge audit logs from several systems, the date and time on those systems must be accurate. Similarly, when you send audit logs to a remote system, the recording system and the repository system must have accurate clocks. The Network Time Protocol (NTP) keeps system clocks accurate and coordinated. For more information, see Chapter 3, “Time-Related Services,” in Introduction to Oracle Solaris 11 Network Services and the `xntpd(1M)` manpage.

Managing a Remote Repository

After the `audit_remote` plugin is configured, a remote repository receives the audit records. The Audit Remote Server (ARS) provides a receiver for audit records. The audit records stream to the ARS over a protected connection and can be stored similarly to how they are stored locally. To configure an ARS, see “How to Configure a Remote Repository for Audit Files” on page 573. For a description of the ARS, see “Audit Remote Server” on page 617 and the `ars(5)` manpage.

How Is Auditing Related to Security?

Auditing helps to detect potential security breaches by revealing suspicious or abnormal patterns of system usage. Auditing also provides a means to trace suspect actions back to a particular user, thus serving as a deterrent. Users who know that their activities are being audited are less likely to attempt malicious activities.

To protect a computer system, especially a system on a network, requires mechanisms that control activities before system processes or user processes begin. Security requires tools that monitor activities as the activities occur. Security also requires reports of activities after the activities have happened.

Best practice requires that audit parameters be set before users log in or system processes begin, because most audit activity involves monitoring current events and reporting the events that meet the specified parameters. How the audit service monitors and reports these events is discussed in detail in Chapter 27, “Planning for Auditing,” and Chapter 28, “Managing Auditing (Tasks).”
Auditing cannot prevent hackers from unauthorized entry. However, the audit service can report, for example, that a specific user performed specific actions at a specific time and date. The audit report can identify the user by entry path and user name. Such information can be reported immediately to your terminal and to a file for later analysis. Thus, the audit service provides data that helps you determine the following:

- How system security was compromised
- What loopholes need to be closed to ensure the desired level of security

**How Does Auditing Work?**

Auditing generates audit records when specified events occur. Most commonly, events that generate audit records include the following:

- System startup and system shutdown
- Login and logout
- Process creation or process destruction, or thread creation or thread destruction
- Opening, closing, creating, destroying, or renaming of objects
- Use of privilege capabilities or role-based access control (RBAC)
- Identification actions and authentication actions
- Permission changes by a process or user
- Administrative actions, such as installing a package
- Site-specific applications

Audit records are generated from three sources:

- By an application
- As a result of an asynchronous audit event
- As a result of a process system call

After the relevant event information has been captured, the information is formatted into an audit record. Contained in each audit record is information that identifies the event, what caused the event, the time of the event, and other relevant information. This record is then placed in an audit queue and sent to the active plugins for storage. At least one plugin must be active, although all plugins can be active. Plugins are described in "How Is Auditing Configured?" on page 536 and "Audit Plugin Modules" on page 531.
How Is Auditing Configured?

During system configuration, you preselect which classes of audit records to monitor. You can also fine-tune the degree of auditing that is done for individual users. The following figure shows details of the flow of auditing in Oracle Solaris.

After audit data is collected in the kernel, plugins distribute the data to the appropriate locations.

- The `audit_binfile` plugin places binary audit records in the `/var/audit` file system. By default, the `audit_binfile` plugin is active. Post-selection tools enable you to examine interesting parts of the audit trail.
  
  Audit files can be stored in one or more ZFS pools. These pools can be on different systems and on different but linked networks. The collection of audit files that are linked together is considered an audit trail.

- The `audit_remote` plugin sends binary audit records across a protected link to a remote repository.

- The `audit_syslog` plugin sends text summaries of audit records to the `syslog` utility.
Systems that install non-global zones can audit all zones identically from the global zone. These systems can also be configured to collect different records in the non-global zones. For more information, see “Auditing and Oracle Solaris Zones” on page 614.

Auditing on a System With Oracle Solaris Zones

A zone is a virtualized operating system environment that is created within a single instance of the Oracle Solaris OS. The audit service audits the entire system, including activities in zones. A system that has installed non-global zones can run a single audit service to audit all zones identically. Or, it can run one audit service per zone, including the global zone.

Sites that satisfy the following conditions can run a single audit service:

- The site requires a single-image audit trail.
- The non-global zones are used as application containers. The zones are part of one administrative domain. That is, no non-global zone has customized naming service files.
  
  If all the zones on a system are within one administrative domain, the zonename audit policy can be used to distinguish audit events that are configured in different zones.
- Administrators want low audit overhead. The global zone administrator audits all zones identically. Also, the global zone’s audit daemon serves all zones on the system.

Sites that satisfy the following conditions can run one audit service per zone:

- The site does not require a single-image audit trail.
- The non-global zones have customized naming service files. These separate administrative domains typically function as servers.
- Individual zone administrators want to control auditing in the zones that they administer.
  
  In per-zone auditing, zone administrators can decide to enable or to disable auditing for the zone that they administer.

The advantages of per-zone auditing are a customized audit trail for each zone, and the ability to disable auditing on a zone by zone basis. These advantages can be offset by the administrative overhead. Each zone administrator must administer auditing. Each zone runs its own audit daemon, and has its own audit queue and audit logs. These audit logs must be managed.
About the Audit Service in This Release

The following features have been introduced to auditing in this release of Oracle Solaris:

- A remote repository that receives audit records can be configured. The connection between an audited system and its remote repository is protected by the GSS-API mechanism. Any number of audited systems can connect to the same remote repository.
- A new audit class is defined. The cusa audit class holds events that affect users and roles.
- The audit configuration files, audit_class, audit_event, and audit_warn, have two package attributes set. The preserve=renamew new attribute enables the files to be modified, and the modifications are retained across package updates and package fixes. The overlay=allow attribute enables the files to be replaced by files in packages that a customer creates.
Planning Auditing (Tasks)

You want to be selective about what kinds of activities are audited. At the same time, you want to collect useful audit information. You also need to carefully plan who to audit and what to audit. If you are using the default audit_bin file plugin, audit files can quickly grow to fill the available space, so you must allocate enough disk space.

The following task map points to the major tasks that are required for planning disk space and which events to record.

<table>
<thead>
<tr>
<th>Task</th>
<th>For Instructions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Determine auditing strategy for non-global zones</td>
<td>“How to Plan Auditing in Zones” on page 540</td>
</tr>
<tr>
<td>Determine who and what to audit</td>
<td>“How to Plan Who and What to Audit” on page 541</td>
</tr>
<tr>
<td>Plan storage space for the audit trail</td>
<td>“How to Plan Disk Space for Audit Records” on page 544</td>
</tr>
<tr>
<td>Plan transmission of the audit trail to a remote server</td>
<td>“How to Prepare to Stream Audit Records to Remote Storage” on page 545</td>
</tr>
</tbody>
</table>
How to Plan Auditing in Zones

If your system contains non-global zones, the zones can be audited as the global zone is audited, or the audit service for each non-global zone can be configured, enabled, and disabled separately. For example, you could audit only the non-global zones, and not audit the global zone.

For a discussion of the trade-offs, see "Auditing on a System With Oracle Solaris Zones" on page 537.

Choose one of the following options:

- **OPTION 1 - Configure a single audit service for all zones.**
  Auditing all zones identically can create a single-image audit trail. A single-image audit trail occurs when you are using the `audit_binfile` or the `audit Remote` plugin, and all zones on a system are part of one administrative domain. The audit records can then be easily compared because the records in every zone are preselected with identical settings.

  This configuration treats all zones as part of one system. The global zone runs the only audit service on a system and collects audit records for every zone. You customize the `audit_class` and `audit_event` files only in the global zone, then copy these files to every non-global zone.

  a. **Use the same naming service for every zone.**

     **Note** – If naming service files are customized in non-global zones, and per zone policy is not set, then careful use of the audit tools is required to select usable records. A user ID in one zone can refer to a different user from the same ID in a different zone.

  b. **Enable the audit records include the name of the zone.**

     To put the zone name as part of the audit record, set the `zonename` policy in the global zone. The `audit reduce` command can then select audit events by zone from the audit trail. For an example, see the `audit reduce(1M)` man page.

     To plan a single-image audit trail, refer to "How to Plan Who and What to Audit" on page 541. Start with the first step. The global zone administrator must also set aside storage, as described in "How to Plan Disk Space for Audit Records" on page 544.

- **OPTION 2 - Configure one audit service per zone.**

  Choose to configure per-zone auditing if different zones use different naming service databases, or if zone administrators want to control auditing in their zones.
Note – To audit non-global zones, the per zone policy must be set, but the audit service does not have to be enabled in the global zone. Non-global zone auditing is configured, and its audit service is enabled and disabled separately from the global zone.

- When you configure per-zone auditing, you set the per zone audit policy in the global zone. If per-zone auditing is set before a non-global zone is first booted, auditing begins at the zone’s first boot. To set audit policy, see “How to Configure Per-Zone Auditing” on page 581.
- Each zone administrator configures auditing for the zone.
  A non-global zone administrator can set all policy options except per zone and ahlt.
- Each zone administrator can enable or disable auditing in the zone.
- To generate records that can be traced to their originating zone during review, set the zonename audit policy.

To plan per-zone auditing, see “How to Plan Who and What to Audit” on page 541. You can skip the first step. If the audit_binfile plugin is active, each zone administrator must also set aside storage for every zone, as described in “How to Plan Disk Space for Audit Records” on page 544.

▼ How to Plan Who and What to Audit

Before You Begin

If you are implementing non-global zones, review “How to Plan Auditing in Zones” on page 540 before using this procedure.

1 Determine if you want a single-system image audit trail.

Note – This step applies only to the audit_binfile plugin.

Systems within a single administrative domain can create a single-system image audit trail. If your systems use different naming services, start with Step 2. Then, complete the rest of the planning steps for every system.

To create a single-system image audit trail for a site, every system in the installation should be configured as follows:

- Use the same naming service for all systems.
  For correct interpretation of the audit records, the passwd, group, and hosts files must be consistent.
- Configure the audit service identically on all systems. For information about displaying and modifying the service settings, see the auditconfig(1M) man page.
- Use the same audit_warn, audit_event, and audit_class files for all systems.
2 Determine the audit policy.

By default, only the cnt policy is enabled.

Use the auditconfig -ls policy command to see a description of available policy options.

- For the effects of the policy options, see “Understanding Audit Policy” on page 546.
- For the effect of the cnt policy, see “Audit Policies for Asynchronous and Synchronous Events” on page 618.
- To set audit policy, see “How to Change Audit Policy” on page 558.

3 Determine if you want to modify event-to-class mappings.

In almost all situations, the default mapping is sufficient. However, if you add new classes, change class definitions, or determine that a record of a specific system call is not useful, you might want to modify event-to-class mappings.

For an example, see “How to Change an Audit Event’s Class Membership” on page 564.

4 Determine which audit classes to preselect.

The best time to add audit classes or to change the default classes is before users log in to the system.

The audit classes that you preselect with the -setflags and -setnaflags options to the auditconfig command apply to all users and processes. You can preselect a class for success, for failure, or for both.

For the list of audit classes, read the /etc/security/audit_class file.

5 Determine user modifications to the system-wide preselections.

If you decide that some users should be audited differently from the system, you can modify the audit_flags security attribute for individual users or for a rights profile. The user preselection mask is modified for users whose audit flags are explicitly set, or who are assigned a rights profile with explicit audit flags.

For the procedure, see “How to Configure a User’s Audit Characteristics” on page 555. For which audit flag values are in effect, see “Order of Search for Assigned Security Attributes” on page 197.

6 Decide how to manage the audit_warn email alias.

The audit_warn script is run whenever the audit system detects a situation that requires administrative attention. By default, the audit_warn script sends email to an audit_warn alias and sends a message to the console.

To set up the alias, see “How to Configure the audit_warn Email Alias” on page 562.

7 Decide in which format and where to collect audit records.
You have three choices.

- By default, store binary audit records locally. The default storage directory is `/var/audit`. To further configure the `audit_binfile` plugin, see “How to Create ZFS File Systems for Audit Files” on page 566.

- Stream binary audit records to a remote protected repository by using the `audit_remote` plugin. You must have a receiver for the records. For the requirements, see “Managing a Remote Repository” on page 534. For the procedure, see “How to Send Audit Files to a Remote Repository” on page 572.

- Send audit record summaries to `syslog` by using the `audit_syslog` plugin. For the procedure, see “How to Configure syslog Audit Logs” on page 577.

For a comparison of binary and `syslog` formats, see “Audit Logs” on page 531.

8 **Determine when to warn the administrator about shrinking disk space.**

**Note** – This step applies only to the `audit_binfile` plugin.

When disk space on an audit file system drops below the minimum free space percentage, or soft limit, the audit service switches to the next available audit directory. The service then sends a warning that the soft limit has been exceeded.

To set a minimum free space percentage, see Example 28–17.

9 **Decide what action to take when all the audit directories are full.**

**Note** – This step applies only to the `audit_binfile` plugin.

In the default configuration, the `audit_binfile` plugin is active, and the `cnt` policy is set. In this configuration, when the kernel audit queue is full, the system continues to work. The system counts the audit records that are dropped, but does not record the events. For greater security, you can disable the `cnt` policy, and enable the `ahlt` policy. The `ahlt` policy stops the system when an asynchronous event cannot be placed in the audit queue.

For a discussion of these policy options, see “Audit Policies for Asynchronous and Synchronous Events” on page 618. To configure these policy options, see Example 28–6.

However, if the `audit_binfile` queue is full, and the queue for another active plugin is not full, then the kernel queue will continue to send records to the plugin that is not full. When the `audit_binfile` queue can again accept records, the audit service will resume sending records to it.
Note – The cnt or ahlt policy is not triggered if the queue for at least one plugin is accepting audit records.

▼ How to Plan Disk Space for Audit Records

The audit_binfile plugin creates an audit trail. The audit trail requires dedicated file space. This space must be available and secure. The system uses the /var/audit file system for initial storage. You can configure additional audit file systems for audit files. The following procedure covers the issues that you must resolve when you plan for audit trail storage.

Before You Begin

If you are implementing non-global zones, complete “How to Plan Auditing in Zones” on page 540 before using this procedure.

You are using the audit_binfile plugin.

1 Determine how much auditing your site needs.

Balance your site’s security needs against the availability of disk space for the audit trail.

For guidance on how to reduce space requirements while still maintaining site security, as well as how to design audit storage, see “Controlling Auditing Costs” on page 548 and “Auditing Efficiently” on page 550.

For practical steps, see “How to Lessen the Volume of Audit Records That Are Produced” on page 598, “How to Compress Audit Files on a Dedicated File System” on page 606, and Example 28–30.

2 Determine which systems are to be audited and configure their audit file systems.

Create a list of all the file systems that you plan to use. For configuration guidelines, see “Storing and Managing the Audit Trail” on page 533 and the audit reduce(1M) man page. To specify the audit file systems, see “How to Assign Audit Space for the Audit Trail” on page 569.

3 Synchronize the clocks on all systems.

For more information, see “Ensuring Reliable Time Stamps” on page 534.
How to Prepare to Stream Audit Records to Remote Storage

Note – If you have a Kerberos realm configured with an identified Audit Remote Server (ARS) and all audited systems within the realm, you can skip this procedure. The steps to configure the ARS and the audited systems are covered in "How to Configure a Remote Repository for Audit Files" on page 573 and "How to Send Audit Files to a Remote Repository" on page 572.

The audit_remote plugin sends the binary audit trail to an ARS in the same format as the audit_binfile plugin writes to the local audit files. The audit_remote plugin uses the libgss library to authenticate the ARS, and a GSS-API mechanism to protect the transmission with privacy and integrity. For reference, see "What Is the Kerberos Service?" on page 333 and "Kerberos Components" on page 341.

The only currently supported GSS-API mechanism is kerberosv5. For more information, see the mech(4) man page.

Before You Begin
You plan to use the audit_remote plugin.

1 Create a Kerberos realm for your audited systems.

   a. Install the master KDC package.
      
      You can use the system that will serve as the ARS, or you can use a nearby system. The ARS sends a significant amount of authentication traffic to the master KDC.
      
      If the Kerberos packages are already on the system, the output appears similar to the following:
      
      # pkg search -l kerberos-5
      INDEX ACTION VALUE PACKAGE
      pkg.summary set Kerberos version 5 support pkg:/service/security/kerberos-5@vm
      pkg.summary set Kerberos V5 Master KDC pkg:/system/security/kerberos-5@vm
      
      The first command checks if the Kerberos V5 Master KDC package is installed. The second command installs the package.
      
      # pkg info system/security/kerberos-5
      pkg: info: no packages matching these patterns are installed on the system.
      
      # pkg install pkg:/system/security/kerberos-5
      
      On the master KDC, you use the Kerberos kdcmgr and kadmin commands to manage the realm. For reference, see the kdcmgr(1M) and kadmin(1M) man pages.

2 On every audited system that will send audit records to the ARS, install the master KDC package.

   # pkg install pkg:/system/security/kerberos-5
This package includes the `kclient` command. On these systems, you run the `kclient` command to connect with the KDC. For reference, see the `kclient(1M)` man page.

3 Synchronize the clocks in the KDC realm.

If the clock skew is too big between the audited systems and the ARS, the attempt at connection will fail. After a connection is established, the local time on the ARS determines the names of the stored audit files, as described in "Conventions for Binary Audit File Names" on page 620.

For more information about the clocks, see "Ensuring Reliable Time Stamps" on page 534.

**Understanding Audit Policy**

Audit policy determines the characteristics of the audit records for the local system. You use the `auditconfig` command to set these policies. For more information, see the `auditconfig(1M)` man page.

Most audit policy options are disabled by default to minimize storage requirements and system processing demands. These options are properties of the audit service and determine the policies that are in effect at system boot. For more information, see the `auditconfig(1M)` man page.

Use the following table to determine if the needs of your site justify the additional overhead that results from enabling one or more audit policy options.

<table>
<thead>
<tr>
<th>Policy Name</th>
<th>Description</th>
<th>Policy Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>aht</td>
<td>This policy applies to asynchronous events only. When disabled, this policy allows the event to complete without an audit record being generated. When enabled, this policy stops the system when the audit queue is full. Administrative intervention is required to clean up the audit queue, make space available for audit records, and reboot. This policy can only be enabled in the global zone. The policy affects all zones.</td>
<td>The disabled option makes sense when system availability is more important than security. The enabled option makes sense in an environment where security is paramount. For a fuller discussion, see &quot;Audit Policies for Asynchronous and Synchronous Events&quot; on page 618.</td>
</tr>
<tr>
<td>arge</td>
<td>When disabled, this policy omits environment variables of an executed program from the execve audit record. When enabled, this policy adds the environment variables of an executed program to the execve audit record. The resulting audit records contain much more detail than when this policy is disabled.</td>
<td>The disabled option collects much less information than the enabled option. For a comparison, see “How to Audit All Commands by Users” on page 600. The enabled option makes sense when you are auditing a few users. The option is also useful when you have suspicions about the environment variables that are being used in programs in the ex audit class.</td>
</tr>
<tr>
<td>Policy Name</td>
<td>Description</td>
<td>Policy Considerations</td>
</tr>
<tr>
<td>-------------</td>
<td>-------------</td>
<td>-----------------------</td>
</tr>
<tr>
<td>argv</td>
<td>When disabled, this policy omits the arguments of an executed program from the execve audit record. When enabled, this policy adds the arguments of an executed program to the execve audit record. The resulting audit records contain much more detail than when this policy is disabled.</td>
<td>The disabled option collects much less information than the enabled option. For a comparison, see “How to Audit All Commands by Users” on page 600. The enabled option makes sense when you are auditing a few users. The option is also useful when you have reason to believe that unusual programs in the ex audit class are being run.</td>
</tr>
<tr>
<td>cnt</td>
<td>When disabled, this policy blocks a user or application from running. The blocking happens when audit records cannot be added to the audit trail because the audit queue is full. When enabled, this policy allows the event to complete without an audit record being generated. The policy maintains a count of audit records that are dropped.</td>
<td>The disabled option makes sense in an environment where security is paramount. The enabled option makes sense when system availability is more important than security. For a fuller discussion, see “Audit Policies for Asynchronous and Synchronous Events” on page 618.</td>
</tr>
<tr>
<td>group</td>
<td>When disabled, this policy does not add a groups list to audit records. When enabled, this policy adds a groups list to every audit record as a special token.</td>
<td>The disabled option usually satisfies requirements for site security. The enabled option makes sense when you need to audit which supplemental groups the subject belongs to.</td>
</tr>
<tr>
<td>path</td>
<td>When disabled, this policy records in an audit record at most one path that is used during a system call. When enabled, this policy records every path that is used in conjunction with an audit event to every audit record.</td>
<td>The disabled option places at most one path in an audit record. The enabled option enters each file name or path that is used during a system call in the audit record as a path token.</td>
</tr>
<tr>
<td>perzone</td>
<td>When disabled, this policy maintains a single audit configuration for a system. One audit service runs in the global zone. Audit events in specific zones can be located in the audit record if the zonename audit token was preselected. When enabled, this policy maintains a separate audit configuration, audit queue, and audit logs for each zone. An audit service runs in each zone. This policy can be enabled in the global zone only.</td>
<td>The disabled option is useful when you have no special reason to maintain a separate audit log, queue, and daemon for each zone. The enabled option is useful when you cannot monitor your system effectively by simply examining audit records with the zonename audit token.</td>
</tr>
<tr>
<td>public</td>
<td>When disabled, this policy does not add read-only events of public objects to the audit trail when the reading of files is preselected. Audit classes that contain read-only events include fr, fa, and cl. When enabled, this policy records every read-only audit event of public objects if an appropriate audit class is preselected.</td>
<td>The disabled option usually satisfies requirements for site security. The enabled option is rarely useful.</td>
</tr>
</tbody>
</table>
TABLE 27–1  Effects of Audit Policy Options  (Continued)

<table>
<thead>
<tr>
<th>Policy Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>seq</td>
<td>When disabled, this policy does not add a sequence number to every audit record. When enabled, this policy adds a sequence number to every audit record. The sequence token holds the sequence number.</td>
</tr>
<tr>
<td>trail</td>
<td>When disabled, this policy does not add a trailer token to audit records. When enabled, this policy adds a trailer token to every audit record.</td>
</tr>
<tr>
<td>zonename</td>
<td>When disabled, this policy does not include a zonename token in audit records. When enabled, this policy includes a zonename token in every audit record.</td>
</tr>
</tbody>
</table>

Policy Considerations

- The disabled option is sufficient when auditing is running smoothly.
- The enabled option makes sense when the cnt policy is enabled. The seq policy enables you to determine when data was discarded. Alternatively, you can use the auditstat command to view dropped records.
- The disabled option creates a smaller audit record. The enabled option clearly marks the end of each audit record with a trailer token. The trailer token is often used with the sequence token. The trailer token aids in the recovery of damaged audit trails.
- The disabled option is useful when you do not need to track audit behavior per zone. The enabled option is useful when you want to isolate and compare audit behavior across zones by post-selecting records according to zone.

Controlling Auditing Costs

Because auditing consumes system resources, you must control the degree of detail that is recorded. When you decide what to audit, consider the following costs of auditing:

- Cost of increased processing time
- Cost of analysis of audit data

If you are using the default plugin, audit_binfile, you must also consider the storage cost of audit data.

Cost of Increased Processing Time of Audit Data

The cost of increased processing time is the least significant of the costs of auditing. The first reason is that auditing generally does not occur during computation-intensive tasks, such as image processing, complex calculations, and so forth. If you are using the audit_binfile plugin, another reason is that audit administrators can move the post-selection tasks from the audited system to systems that are dedicated to analyzing audit data. Finally, unless kernel events are preselected, the audit service has no measurable impact on system performance.
Cost of Analysis of Audit Data

The cost of analysis is roughly proportional to the amount of audit data that is collected. The cost of analysis includes the time that is required to merge and review audit records.

For records that are collected by the audit_binfile plugin, cost also includes the time that is required to archive the records and their supporting name service databases, and to keep the records in a safe place. Supporting databases include groups, hosts, and passwd.

The fewer records that you generate, the less time that is required to analyze the audit trail. The sections “Cost of Storage of Audit Data” on page 549 and “Auditing Efficiently” on page 550 describe ways to audit efficiently. Efficient auditing reduces the amount of audit data, while still providing enough coverage to achieve your site’s security goals.

Cost of Storage of Audit Data

If you are using the audit_binfile plugin, storage cost is the most significant cost of auditing. The amount of audit data depends on the following:

- Number of users
- Number of systems
- Amount of use
- Degree of traceability and accountability that is required

Because these factors vary from site to site, no formula can predetermine the amount of disk space to set aside for audit data storage. Use the following information as a guide:

- Understand the audit classes
  Before you configure auditing, you should understand the types of events that the classes contain. You can change the audit event-class mappings to optimize audit record collection.

- Preselect audit classes judiciously to reduce the volume of records that are generated.
  Full auditing, that is, with the all class, fills disk space quickly. Even a simple task such as compiling a program could generate a large audit file. A program of modest size could generate thousands of audit records in less than a minute.
  For example, by omitting the file_read audit class, fr, you can significantly reduce audit volume. By choosing to audit for failed operations only, you can at times reduce audit volume. For example, by auditing for failed file_read operations, -fr, you can generate far fewer records than by auditing for all file_read events.

- If you are using the audit_binfile plugin, efficient audit file management is also important. For example, you can compress a ZFS file system that is dedicated to audit files.

- Develop a philosophy of auditing for your site.
  Base your philosophy on sensible measures. Such measures include the amount of traceability that your site requires, and the types of users that you administer.
Auditing Efficiently

The following techniques can help you achieve your organization's security goals while auditing more efficiently.

- For as many audit classes as possible, only preselect those classes for users and roles, not system-wide.
- Randomly audit only a certain percentage of users at any one time.
- If the audit_binfile plugin is active, reduce the disk storage requirements for audit files by filtering, merging, and compressing the files. Develop procedures for archiving the files, for transferring the files to removable media, and for storing the files offline.
- Monitor the audit data in real time for unusual behaviors.
  - audit_syslog plugin – You can extend management and analysis tools that you have already developed to handle the audit records in syslog files.
  - audit_binfile plugin – You can set up procedures to monitor the audit trail for certain activities. You can write a script to trigger an automatic increase in the auditing of certain users or certain systems in response to detection of unusual events.

For example, you could write a script that does the following:
1. Monitors the creation of audit files on the audited systems.
2. Processes the audit files with the tail command.
   The piping of the output from the tail -of command through the praudit command can yield a stream of audit records as the records are generated. For more information, see the tail(1) man page.
3. Analyzes this stream for unusual message types or other indicators, and delivers the analysis to the auditor.
   Or, the script can be used to trigger automatic responses.
4. Constantly monitors the audit file systems for the appearance of new not_terminated audit files.
5. Terminates outstanding tail processes when their files are no longer being written to.
Managing Auditing (Tasks)

This chapter provides procedures to help you configure and manage auditing on an Oracle Solaris system. This chapter also includes instructions for administering the audit trail and troubleshooting the audit service. The chapter covers the following tasks:

- “Configuring the Audit Service (Tasks)” on page 551
- “Configuring Audit Logs (Tasks)” on page 565
- “Configuring the Audit Service in Zones (Tasks)” on page 579
- “Enabling and Disabling the Audit Service (Tasks)” on page 582
- “Managing Audit Records on Local Systems (Tasks)” on page 585
- “Troubleshooting the Audit Service (Tasks)” on page 595

For an overview of the audit service, see Chapter 26, “Auditing (Overview).” For planning suggestions, see Chapter 27, “Planning for Auditing.” For reference information, see Chapter 29, “Auditing (Reference).”

Configuring the Audit Service (Tasks)

Before you enable auditing on your network, you can modify the defaults to satisfy your site auditing requirements. Best practice is to customize your audit configuration as much as possible before the first users log in.

If you have implemented zones, you can choose to audit all zones from the global zone or to audit non-global zones individually. For an overview, see “Auditing and Oracle Solaris Zones” on page 614. For planning, see “How to Plan Auditing in Zones” on page 540. For procedures, see “Configuring the Audit Service in Zones (Tasks)” on page 579.

Configuring the Audit Service (Task Map)

The following task map points to the procedures for configuring auditing. All tasks are optional.
Configuring the Audit Service (Tasks)

<table>
<thead>
<tr>
<th>Task</th>
<th>Description</th>
<th>For Instructions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Display auditing defaults.</td>
<td>Before configuring auditing, displays the default policy, queue controls, flags, and plugin usage.</td>
<td>&quot;How to Display Audit Service Defaults&quot; on page 552</td>
</tr>
<tr>
<td>Select which events are audited.</td>
<td>Preselects system-wide audit classes. If an event is attributable, then all users are audited for this event.</td>
<td>&quot;How to Preselect Audit Classes&quot; on page 554</td>
</tr>
<tr>
<td>Select which events are audited for specific users.</td>
<td>Sets user-specific differences from the system-wide audit classes.</td>
<td>&quot;How to Configure a User's Audit Characteristics&quot; on page 555</td>
</tr>
<tr>
<td>Specify audit policy.</td>
<td>Defines additional audit data that your site requires.</td>
<td>&quot;How to Change Audit Policy&quot; on page 558</td>
</tr>
<tr>
<td>Specify queue controls.</td>
<td>Modifies the default buffer size, audit records in the queue, and interval between writing audit records to the buffer.</td>
<td>&quot;How to Change Audit Queue Controls&quot; on page 560</td>
</tr>
<tr>
<td>Create the audit_warn email alias.</td>
<td>Defines who receives email warnings when the audit service needs attention.</td>
<td>&quot;How to Configure the audit_warn Email Alias&quot; on page 562</td>
</tr>
<tr>
<td>Configure audits.</td>
<td>Configures the location of audit records for each plugin.</td>
<td>&quot;Configuring Audit Logs (Tasks)&quot; on page 565</td>
</tr>
<tr>
<td>Add audit classes.</td>
<td>Reduces the number of audit records by creating a new audit class to hold critical events.</td>
<td>&quot;How to Add an Audit Class&quot; on page 563</td>
</tr>
<tr>
<td>Change event-to-class mappings.</td>
<td>Reduces the number of audit records by changing the event-class mapping.</td>
<td>&quot;How to Change an Audit Event's Class Membership&quot; on page 564</td>
</tr>
</tbody>
</table>

▼ How to Display Audit Service Defaults

The commands in this procedure display the current audit configuration. The output in this procedure is taken from an unconfigured system.

Before You Begin

You must become an administrator who is assigned the Audit Configuration or Audit Control rights profile. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1 Display the preselected classes for attributable events.

```sh
# auditconfig -getflags
active user default audit flags = lo(0x1000,0x1000)
configured user default audit flags = lo(0x1000,0x1000)
```

`lo` is the flag for the login/logout audit class. The format of the mask output is `(success,failure)`. 
Note – To see which events are assigned to a class, and therefore which events are being recorded, run the auditrecord -c class command.

2 Display the preselected classes for non-attributable events.

```
# auditconfig -getnaflags
active non-attributable audit flags = lo(0x1000,0x1000)
configured non-attributable audit flags = lo(0x1000,0x1000)
```

3 Display the audit policy.

```
$ auditconfig -getpolicy
configured audit policies = cnt
active audit policies = cnt
```

The active policy is the current policy, but the policy value is not being stored by the audit service. The configured policy is stored by the audit service, so the policy is restored when you restart the audit service.

4 Display information about the audit plugins.

```
$ auditconfig -getplugin
Plugin: audit_binfile
   Attributes: p_dir=/var/audit;p_fsize=0;p_minfree=1;

Plugin: audit_syslog (inactive)
   Attributes: p_flags=

Plugin: audit_remote (inactive)
   Attributes: p_hosts=;p_retries=3;p_timeout=5;
```

The audit_binfile plugin is active by default.

5 Display the audit queue controls.

```
$ auditconfig -getqctrl
   no configured audit queue hiwater mark
   no configured audit queue lowater mark
   no configured audit queue buffer size
   no configured audit queue delay
   active audit queue hiwater mark (records) = 100
   active audit queue lowater mark (records) = 10
   active audit queue buffer size (bytes) = 8192
   active audit queue delay (ticks) = 20
```

The active queue control is the queue control that is currently used by the kernel. The string no configured indicates that the system is using the default values.

6 Display the audit classes that are preselected for existing users.

Find the users, then display each user’s audit_flags attribute value.

```
# who
adoe pts/1  Oct 10 10:20 (:0.0)
adoe pts/2  Oct 10 10:20 (:0.0)
jdoe pts/5  Oct 12 12:20 (:0.0)
```
By default, users are audited for the system-wide settings only.

For a description of the `userattr` command, see the `userattr(1)` man page. For a description of the `audit_flags` keyword, see the `user_attr(4)` man page.

## How to Preselect Audit Classes

Preselect audit classes that contain the events that you want to monitor. Events that are not in preselected classes are not recorded.

### Before You Begin

You must become an administrator who is assigned the Audit Configuration rights profile. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1. **Determine the current preselected classes.**
   ```
   # auditconfig -getflags
   ...
   # auditconfig -getnaflags
   ,,,
   ```
   For an explanation of the output, see “How to Display Audit Service Defaults” on page 552.

2. **Preselect the attributable classes.**
   ```
   # auditconfig -setflags lo,ps,fw
   user default audit flags = ps,lo,fw(0x101002,0x101002)
   ```
   This command audits the events in the login/logout, process start/stop, and file write classes for success and for failure.

   **Note** – The `auditconfig -setflags` command replaces the current preselection, so you must specify all classes that you want to preselect.

3. **Preselect the non-attributable classes.**
   The na class contains PROM, boot, and non-attributable mounts, among other events.
   ```
   # auditconfig -setnaflags lo,na
   non-attributable audit flags = lo,na(0x1400,0x1400)
   ```
   lo and na are the only useful arguments to the `-setnaflags` option.
Note - The auditconfig –setnaflags command replaces the current preselection, so you must specify all classes that you want to preselect.

How to Configure a User's Audit Characteristics

By preselecting classes on a per user basis rather than on a per system basis, you can sometimes reduce the impact of auditing on system performance. Also, you might want to audit specific users slightly differently from the system.

Audit class preselections for each user are specified by the audit_flags security attribute. These user-specific values, plus the preselected classes for the system, determine the user's audit mask, as described in "Process Audit Characteristics" on page 619.

Before You Begin

You must assume the root role. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

● Set the audit flags in the user_attr or in the prof_attr database.

For example, you can create a rights profile that defines the rights of a subset of your users. Users who are assigned that rights profile are audited identically.

■ To set audit flags for a user, use the usermod command.

```bash
# usermod -K audit_flags=fw:no jdoe
```

The format of the audit_flags keyword is always-audit:never-audit.

always-audit  Lists the audit classes that are audited for this user. Modifications to the system-wide classes are prefixed by a caret (^). Classes that are added to the system-wide classes are not prefixed by a caret.

never-audit   Lists the audit classes that are never audited for the user, even if these audit events are audited system-wide. Modifications to the system-wide classes are prefixed by a caret (^).

To specify multiple audit classes, separate the classes with commas. For more information, see the audit_flags(5) man page.

■ To set audit flags for a rights profile, use the profiles command.

```bash
# profiles -p "System Administrator"
profiles:System Administrator> set name="Audited System Administrator"
profiles:Audited System Administrator> set always_audit=fw,as
profiles:Audited System Administrator> end
profiles:Audited System Administrator> exit
```
When you assign the Audited System Administrator rights profile to a user or a role, that user or role is audited for those flags, subject to search order as described in “Order of Search for Assigned Security Attributes” on page 197.

Example 28–1  Changing Which Events Are Audited for One User

In this example, the audit preselection mask for all users is the following:

```
# auditconfig -getflags
active user default audit flags = ss,lo(0x11000,0x11000)
configured user default audit flags = ss,lo(0x11000,0x11000)
```

No user except the administrator is logged in.

To lessen the impact of the AUE_PFEXEC audit event on system resources, the administrator does not audit this event at the system level. Rather, the administrator preselects the pf class for a user, jdoe. The pf class is created in Example 28–10.

```
# usermod -K audit_flags=pf:no jdoe
```

The userattr command shows the addition.

```
# userattr audit_flags jdoe
pf:no
```

When the user jdoe logs in, jdoe's audit preselection mask is a combination of the audit_flags values with the system default values. 289 is the PID of jdoe's login shell.

```
# auditconfig -getpinfo 289
audit id = jdoe(1234)
process preselection mask = ss,pf,lo(0x01000000008011000,0x01000000008011000)
terminal id (maj,min,host) = 242,511,example1(192.168.160.171)
audit session id = 103203403
```

Example 28–2  Modifying Audit Preselection Exception for One User

In this example, the audit preselection mask for all users is the following:

```
# auditconfig -getflags
active user default audit flags = ss,lo(0x11000,0x11000)
configured user default audit flags = ss,lo(0x11000,0x11000)
```

No users except the administrator are logged in.

The administrator decides not to collect failed ss events for the jdoe user.

```
# usermod -K audit_flags=-ss:no jdoe
```

The userattr command shows the exception.

556  Oracle Solaris 11.1 Administration: Security Services  •  February 2014
When the user jdoe logs in, jdoe’s audit preselection mask is a combination of the audit_flags values with the system default values. 289 is the PID of jdoe’s login shell.

```
# auditconfig -getinfo 289
```

```
audit id = jdoe(1234)
process preselection mask = +ss,lo(0x11000,0x1000)
terminal id (maj,min,host) = 242,511,example1(192.168.160.171)
audit session id = 103203403
```

**Example 28–3 Auditing Selected Users, No System-Wide Auditing**

In this example, the login and role activities of four selected users are audited on the system. No audit classes are preselected for the system.

First, the administrator removes all system-wide flags.

```
# auditconfig -setflags no
```

Then, the administrator preselects two audit classes for the four users. The pf class is created in Example 28–10.

```
# usermod -K audit_flags=lo,pf:no jdoe
# usermod -K audit_flags=lo,pf:no kdoe
# usermod -K audit_flags=lo,pf:no pdoe
# usermod -K audit_flags=lo,pf:no zdoe
```

Then, the administrator preselects the pf class for the root role.

```
# userattr audit_flags root
# rolemod -K audit_flags=lo,pf:no root
# userattr audit_flags root
lo,pf:no
```

To continue to record unwarranted intrusion, the administrator does not change the auditing of non-attributable logins.

```
# auditconfig -getnaflags
active non-attributable audit flags = lo(0x1000,0x1000)
configured non-attributable audit flags = lo(0x1000,0x1000)
```

**Example 28–4 Removing a User’s Audit Flags**

In the following example, the administrator removes all user-specific audit flags. Existing processes of users who are currently logged in continue to be audited.

The administrator runs the usermod command with the audit_flags keyword set to no value.
Creating a Rights Profile for a Group of Users

The administrator wants all administrative rights profiles at the site to explicitly audit the pf class. For every rights profile that is going to be assigned, the administrator creates a site-specific version in LDAP that includes audit flags.

First, the administrator clones an existing rights profile, then changes the name and adds audit flags.

```bash
# profiles -p "Network Wifi Management" -S ldap
profiles: Network Wifi Management> set name="Wifi Management"
profiles: Wifi Management> set desc="Audited wifi management"
profiles: Wifi Management> set audit_always=pf
profiles: Wifi Management> exit
```

After repeating this procedure for every rights profile that is going to be used, the administrator lists the information in the Wifi Management profile.

```bash
# profiles -p "Wifi Management" -S ldap info
name=Wifi Management
desc=Audited wifi management
auths=solaris.network.wifi.config
help=RtNetWifiMngmnt.html
always_audit=pf
```

How to Change Audit Policy

You might change default audit policy to record detailed information about audited commands, to add a zone name to every record, or to satisfy other site security requirements.

**Before You Begin**  
You must become an administrator who is assigned the Audit Configuration rights profile. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1. **View the current audit policy.**
   ```bash
   $ auditconfig -getpolicy
   ...
   ```
For an explanation of the output, see “How to Display Audit Service Defaults” on page 552.

2 View the available policy options.

$ auditconfig -lspolicy

<table>
<thead>
<tr>
<th>policy string</th>
<th>description:</th>
</tr>
</thead>
<tbody>
<tr>
<td>ahlt</td>
<td>halt machine if it can not record an async event</td>
</tr>
<tr>
<td>all</td>
<td>all policies for the zone</td>
</tr>
<tr>
<td>arge</td>
<td>include exec environment args in audit recs</td>
</tr>
<tr>
<td>argv</td>
<td>include exec command line args in audit recs</td>
</tr>
<tr>
<td>cnt</td>
<td>when no more space, drop recs and keep a cnt</td>
</tr>
<tr>
<td>group</td>
<td>include supplementary groups in audit recs</td>
</tr>
<tr>
<td>none</td>
<td>no policies</td>
</tr>
<tr>
<td>path</td>
<td>allow multiple paths per event</td>
</tr>
<tr>
<td>perzone</td>
<td>use a separate queue and auditd per zone</td>
</tr>
<tr>
<td>public</td>
<td>audit public files</td>
</tr>
<tr>
<td>seq</td>
<td>include a sequence number in audit recs</td>
</tr>
<tr>
<td>trail</td>
<td>include trailer token in audit recs</td>
</tr>
<tr>
<td>windata_down</td>
<td>include downgraded window information in audit recs</td>
</tr>
<tr>
<td>windata_up</td>
<td>include upgraded window information in audit recs</td>
</tr>
<tr>
<td>zonename</td>
<td>include zonename token in audit recs</td>
</tr>
</tbody>
</table>

Note – The perzone and ahlt policy options can only be set in the global zone. For the trade-offs to using a particular policy option, see “Understanding Audit Policy” on page 546.

3 Enable or disable selected audit policy options.

# auditconfig [ -t ] -setpolicy [prefix]policy[ ,policy...]

- t Optional. Creates a temporary, or active, policy. You might set a temporary policy for debugging or testing purposes.

A temporary policy is in effect until the audit service is refreshed, or until the policy is modified by the auditconfig -setpolicy command.

prefix A prefix value of + adds the list of policies to the current policy. A prefix value of - removes the list of policies from the current policy. Without a prefix, audit policy is reset. This option enables you to retain current audit policies.

policy Selects the policy to be enabled or to be disabled.

Example 28–6 Setting the ahlt Audit Policy Option

In this example, strict site security requires the ahlt policy.

# auditconfig -setpolicy -cnt
# auditconfig -setpolicy +ahlt

The plus sign (+) before the ahlt policy adds the policy to current policy settings. Without the plus sign, the ahlt policy replaces all current audit policies.
Example 28–7  Setting a Temporary Audit Policy

In this example, the ahlt audit policy is configured. For debugging, the administrator adds the trail audit policy to the active policy (+trail) temporarily (-t). The trail policy aids in the recovery of damaged audit trails.

```
$ auditconfig -setpolicy ahlt
$ auditconfig -getpolicy
configured audit policies = ahlt
active audit policies = ahlt
$ auditconfig -t -setpolicy +trail
configured audit policies = ahlt
active audit policies = ahlt, trail
```

The administrator disables the trail policy when the debugging is completed.

```
$ auditconfig -setpolicy -trail
$ auditconfig -getpolicy
configured audit policies = ahlt
active audit policies = ahlt
```

Refreshing the audit service by running the audit -s command also removes this temporary policy, plus any other temporary values in the audit service. For examples of other temporary values, see "How to Change Audit Queue Controls" on page 560.

Example 28–8  Setting the perzone Audit Policy

In this example, the perzone audit policy is added to the existing policy in the global zone. The perzone policy setting is stored as a permanent property, so perzone policy is in effect during the session and when the audit service is restarted. For the zones, the policy is available at the next zone boot.

```
$ auditconfig -getpolicy
configured audit policies = cnt
active audit policies = cnt
$ auditconfig -setpolicy +perzone
$ auditconfig -getpolicy
configured audit policies = perzone,cnt
active audit policies = perzone,cnt
```

▼ How to Change Audit Queue Controls

The audit service provides default values for audit queue parameters. You can inspect, change, and temporarily change these values with the auditconfig command.

Before You Begin  You must become an administrator who is assigned the Audit Configuration rights profile. For more information, see "How to Use Your Assigned Administrative Rights" on page 157.
1 View the current values of the audit queue controls.
phonemew / $ auditconfig -getqctrl
... 
For an explanation of the output, see “How to Display Audit Service Defaults” on page 552.

2 Modify selected audit queue controls.
For examples and a description of the audit queue controls, see the auditconfig(1M) man page.
- To modify some or all audit queue controls, use the -setqctrl option.

# auditconfig [ -t ] -setqctrl hiwater lowater bufsz interval

For example, set the interval value to 10 without setting the other controls.

# auditconfig -setqctrl 0 0 0 10
- To modify a specific audit queue control, specify its option. The -setqdelay option is the equivalent of -setqctrl 0 0 0 interval, as in # auditconfig -setqdelay 10.

# auditconfig [ -t ] -setqhiwater value
# auditconfig [ -t ] -setqlowater value
# auditconfig [ -t ] -setqbufsz value
# auditconfig [ -t ] -setqdelay value

Example 28–9 Resetting an Audit Queue Control to the Default

The administrator sets all audit queue controls, then changes the lowater value in the repository back to the default.

# auditconfig -setqctrl 200 5 10216 10
# auditconfig -setqctrl 200 0 10216 10
configured audit queue hiwater mark (records) = 200
no configured audit queue lowater mark
configured audit queue buffer size (bytes) = 10216
configured audit queue delay (ticks) = 10
active audit queue hiwater mark (records) = 200
active audit queue lowater mark (records) = 5
active audit queue buffer size (bytes) = 10216
active audit queue delay (ticks) = 10

Later, the administrator sets the lowater value to the default for the current session.

# auditconfig -setqlowater 10
# auditconfig -getqlowater
configured audit queue lowater mark (records) = 10
active audit queue lowater mark (records) = 10

1 2 3 4 5
How to Configure the audit_warn Email Alias

The /etc/security/audit_warn script generates mail to notify the administrator of audit incidents that might need attention. You can customize the script and you can send the mail to an account other than root.

If the perzone policy is set, the non-global zone administrator must configure the audit_warn email alias in the non-global zone.

Before You Begin
You must become an administrator who is assigned the solaris.admin.edit/etc/security/audit_warn authorization. By default, only the root role has this authorization. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

Configure the audit_warn email alias.
Choose one of the following options:

- **OPTION 1** – Replace the audit_warn email alias with another email account in the audit_warn script.

  Change the audit_warn email alias in the ADDRESS line of the script to another address:

  ```
  #ADDRESS=audit_warn # standard alias for audit alerts
  ADDRESS=audadmin # role alias for audit alerts
  ```

  **Note** – For information about the effects of modifying an audit configuration file, see “Audit Configuration Files and Packaging” on page 615.

- **OPTION 2** – Redirect the audit_warn email to another mail account.

  Add the audit_warn email alias to the appropriate mail aliases file. You could add the alias to the local /etc/mail/aliases file or to the mail_aliases database in the name space. The /etc/mail/aliases entry would resemble the following if the root and audadmin email accounts were added as members of the audit_warn email alias:

  ```
  audit_warn: root,audadmin
  ```

  Then, run the newaliases command to rebuild the random access database for the aliases file.

  ```
  # newaliases
  /etc/mail/aliases: 14 aliases, longest 10 bytes, 156 bytes total
  ```
How to Add an Audit Class

When you create your own audit class, you can place into it just those audit events that you want to audit for your site. This strategy can reduce the number of records that are collected and reduce noise in your audit trail.

When you add the class on one system, copy the change to all systems that are being audited. Best practice is to create audit classes before the first users log in.

Note – For information about the effects of modifying an audit configuration file, see “Audit Configuration Files and Packaging” on page 615.

Before You Begin

Choose free bits for your unique entry. Verify which bits are available for customer use in the /etc/security/audit_class file.

You must become an administrator who is assigned the solaris.admin.edit/etc/security/audit_class authorization. By default, only the root role has this authorization. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1  (Optional) Save a backup copy of the audit_class file.
   # cp /etc/security/audit_class /etc/security/audit_class.orig

2  Add new entries to the audit_class file.
   Each entry has the following format:
   0x64bitnumber:flag:description
   For a description of the fields, see the audit_class(4) man page. For the list of existing classes, read the /etc/security/audit_class file.

Tip – The audit configuration files from Oracle Solaris enable you to create your own package that contains these files and replace the Oracle Solaris packages with your site-customized files. When you set the preserve attribute to true in your package, the pkg subcommands, such as verify, fix, revert, and so on, will run relative to your packages. For more information, see the pkg(1) and pkg(5) man pages.

Example 28–10  Creating a New Audit Class

This example creates a class to hold administrative commands that are executed in a role. The added entry to the audit_class file is as follows:

0x0100000000000000:pf:profile command
The entry creates the new pf audit class. Example 28–11 populates the new audit class.

**Troubleshooting**
If you have customized the audit_class file, make sure that any audit flags that are assigned directly to users or rights profiles are consistent with the new audit classes. Errors occur when an audit_flags value is not a subset of the audit_class file.

▼ **How to Change an Audit Event's Class Membership**
You might want to change an audit event's class membership to reduce the size of an existing audit class, or to place the event in a class of its own.

**Caution** – Never comment out events in the audit_event file. This file is used by the praudit command to read binary audit files. Archived audit files might contain events that are listed in the file.

When you reconfigure audit event-class mappings on one system, copy the change to all systems that are being audited. Best practice is to change event-class mappings before the first users log in.

**Note** – For information about the effects of modifying an audit configuration file, see “Audit Configuration Files and Packaging” on page 615.

**Before You Begin**
You must become an administrator who is assigned the solaris.admin.edit/etc/security/audit_event authorization. By default, only the root role has this authorization. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1. **(Optional) Save a backup copy of the audit_event file.**
   
   # cp /etc/security/audit_event /etc/security/audit_event.orig

2. **Change the class to which particular events belong by changing the class-list of the events.**
   Each entry has the following format:
   
   number:name:description:class-list
   
   - **number** is the audit event ID.
   - **name** is the name of the audit event.
   - **description** typically, the system call or executable that triggers the creation of an audit record.
Tip – The audit configuration files from Oracle Solaris allow you to create your own package that contains these files, and replace the Oracle Solaris packages with your site-customized files. When you set the preserve attribute to true in your package, the pkg subcommands, such as verify, fix, revert, and so on will run relative to your packages. For more information, see the pkg(1) and pkg(5) man pages.

Example 28–11 Mapping Existing Audit Events to a New Class

This example maps an existing audit event to the new class that was created in Example 28–10. By default, the AUE_PFEXEC audit event is mapped to several audit classes. By creating the new class, the administrator can audit AUE_PFEXEC events without auditing the events in the other classes.

```
# grep pf /etc/security/audit_class
0x0100000000000000:pf:profile command
# grep AUE_PFEXEC /etc/security/audit_event
116:AUE_PFEXEC:execve(2) with pfexec enabled:ps,ex,ua,as,cusa
# pfedit /etc/security/audit_event
#116:AUE_PFEXEC:execve(2) with pfexec enabled:ps,ex,ua,as,cusa
116:AUE_PFEXEC:execve(2) with pfexec enabled:pf
# auditconfig -setflags lo,pf
user default audit flags = pf,lo(0x0100000000001000,0x0100000000001000)
```

Configuring Audit Logs (Tasks)

Two audit plugins, audit_binfile and audit_syslog, can create local audit logs. The following tasks help you configure these logs.

Configuring Audit Logs (Task Map)

The following task map points to the procedures for configuring audit logs for the various plugins. Configuring logs for the audit_binfile plugin is optional. The logs for other plugins must be configured by an administrator.

<table>
<thead>
<tr>
<th>Task</th>
<th>Description</th>
<th>For Instructions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add local storage for the audit_binfile plugin.</td>
<td>Creates additional disk space for the audit files and protects them with file permissions.</td>
<td>&quot;How to Create ZFS File Systems for Audit Files&quot; on page 566</td>
</tr>
</tbody>
</table>
How to Create ZFS File Systems for Audit Files

The following procedure shows how to create a ZFS pool for audit files, as well as the corresponding file systems and mount point. By default, the /var/audit file system holds audit files for the audit_binfile plugin.

Before You Begin

You must become an administrator who is assigned the ZFS File System Management and ZFS Storage Management rights profiles. The latter profile enables you to create storage pools. For more information, see "How to Use Your Assigned Administrative Rights" on page 157.

1. **Determine the amount of disk space that is required.**
   Assign at least 200 MB of disk space per host. However, how much auditing you require dictates the disk space requirements. So, your disk space requirements might be far greater than this figure.

   *Note* – The default class preselection creates files in /var/audit that grow by about 80 bytes for every recorded instance of an event in the 1o class, such as a login, logout, or role assumption.

2. **Create a mirrored ZFS storage pool.**
   The zpool create command creates a storage pool, that is, a container for the ZFS file systems. For more information, see Chapter 1, "Oracle Solaris ZFS File System (Introduction)," in Oracle Solaris 11.1 Administration: ZFS File Systems.

   ```bash
   # zpool create audit-pool mirror disk1 disk2
   # zpool create audtp pool c3t1d0 c3t2d0
   ```
3 **Create a ZFS file system and mount point for the audit files.**

You create the file system and mount point with one command. At creation, the file system is mounted. For example, the following illustration shows audit trail storage that is stored by host name.

```
# zfs create -o mountpoint=/mountpoint audit-pool/mountpoint
```

For example, create the /audit mount point for the auditf file system.

```
# zfs create -o mountpoint=/audit auditp/auditf
```

Create a ZFS file system for the audit files.

```
# zfs create -p auditp/auditf/system
```

For example, create an unencrypted ZFS file system for the sys1 system.

```
# zfs create -p auditp/auditf/sys1
```

5 **(Optional) Create additional file systems for audit files.**

One reason to create additional file systems is to prevent audit overflow. You can set a ZFS quota per file system, as shown in Step 8. The audit_warn email alias notifies you when each quota is reached. To free space, you can move the closed audit files to a remote server.

```
# zfs create -p auditp/auditf/sys1.1
# zfs create -p auditp/auditf/sys1.2
```
6 Protect the parent audit file system.
The following ZFS properties are set to off for all file systems in the pool:

```bash
# zfs set devices=off auditp/auditf
# zfs set exec=off auditp/auditf
# zfs set setuid=off auditp/auditf
```

7 Compress the audit files in the pool.
Typically, compression is set in ZFS at the file system level. However, because all the file systems in this pool contain audit files, compression is set at the top-level dataset for the pool.

```bash
# zfs set compression=on auditp
```

See also “Interactions Between ZFS Compression, Deduplication, and Encryption Properties” in Oracle Solaris 11.1 Administration: ZFS File Systems.

8 Set quotas.
You can set quotas at the parent file system, the descendant file systems, or both. If you set a quota on the parent audit file system, quotas on the descendant file systems impose an additional limit.

a. Set a quota on the parent audit file system.
In the following example, when both disks in the auditp pool reach the quota, the audit_warn script notifies the audit administrator.

```bash
# zfs set quota=510G auditp/auditf
```

b. Set a quota on the descendant audit file systems.
In the following example, when the quota for the auditp/auditf/system file system is reached, the audit_warn script notifies the audit administrator.

```bash
# zfs set quota=170G auditp/auditf/sys1
# zfs set quota=170G auditp/auditf/sys1.1
# zfs set quota=165G auditp/auditf/sys1.2
```

9 For a large pool, limit the size of the audit files.
By default, an audit file can grow to the size of the pool. For manageability, limit the size of the audit files. See Example 28–14.

**Example 28–12 Creating an Encrypted File System for Audit Files**

To comply with site security requirements, the administrator creates the audit file system with encryption turned on. Then, the administrator sets the mount point.

```bash
# zfs create -o encryption=on auditp/auditf
Enter passphrase for auditp/auditf: /** Type 8-character minimum passphrase**/
Enter again: /** Confirm passphrase **/
# zfs set -o mountpoint=/audit auditp/auditf
```
The file system is unreachable without the passphrase, so the administrator saves it in a secure location.

When the administrator creates additional file systems under the `audit` file system, these descendant file systems are also encrypted.

**Example 28–13  Setting a Quota on the /var/audit Directory**

In this example, the administrator sets a quota on the default audit file system. When this quota is reached, the `audit_warn` script warns the audit administrator.

```
# zfs set quota=252G rpool/var/audit
```

## How to Assign Audit Space for the Audit Trail

In this procedure, you use attributes to the `audit_binfile` plugin to assign additional disk space to the audit trail.

**Before You Begin**

You must become an administrator who is assigned the Audit Configuration rights profile. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1. **Determine the attributes to the `audit_binfile` plugin.**
   
   Read the OBJECT ATTRIBUTES section of the `audit_binfile(5)` man page.

   ```
   # man audit_binfile
   ...
   OBJECT ATTRIBUTES
   The p_dir attribute specifies where the audit files will be created.
   The directories are listed in the order in which they are to be used.

   The p_minfree attribute defines the percentage of free space that the audit system requires before the audit daemon invokes the audit_warn script.

   The p_fsize attribute defines the maximum size that an audit file can become before it is automatically closed and a new audit file is opened. ... The format of the p_fsize value can be specified as an exact value in bytes or in a human-readable form with a suffix of B, K, M, G, T, P, E, Z (for bytes, kilobytes, megabytes, gigabytes, terabytes, petabytes, exabytes, or zettabytes, respectively). Suffixes of KB, MB, GB, TB, PB, EB, and ZB are also accepted.
   ```

2. **To add directories to the audit trail, specify the p_dir attribute.**
   
   The default file system is `/var/audit`.

   ```
   # auditconfig -setplugin audit_binfile p_dir=/audit/sys1.1,/var/audit
   ```
The preceding command sets the /audit/sys1.1 file system as the primary directory for audit files and the default /var/audit file system as the secondary directory. In this scenario, /var/audit is the directory of last resort. For this configuration to succeed, the /audit/sys1.1 file system must exist.

You created a similar file system in "How to Create ZFS File Systems for Audit Files" on page 566.

3 Refresh the audit service.

The auditconfig -setplugin command sets the configured value. This value is a property of the audit service, so is restored when the service is refreshed or restarted. The configured value becomes active when the audit service is refreshed or restarted. For information about configured and active values, see the auditconfig(1M) man page.

# audit -s

Example 28–14 Limiting File Size for the audit_binfile Plugin

In the following example, the size of a binary audit file is set to a specific size. The size is specified in megabytes.

# auditconfig -setplugin audit_binfile p_fsize=4M
# auditconfig -getplugin audit_binfile
Plugin: audit_binfile
 Attributes: p_dir=/var/audit;p_fsize=4M;p_minfree=1;

By default, an audit file can grow without limit. To create smaller audit files, the administrator specifies a file size limit of 4MB. The audit service creates a new file when the size limit is reached. The file size limit goes into effect after the administrator refreshes the audit service.

# audit -s

Example 28–15 Specifying Several Changes to an Audit Plugin

In the following example, the administrator on a system with high throughput and a large ZFS pool changes the queue size, the binary file size, and the soft limit warning for the audit_binfile plugin. The administrator allows audit files to grow to 4 GB, is warned when 2 percent of the ZFS pool remains, and doubles the allowed queue size. The default queue size is the high water mark for the kernel audit queue, 100, as in active audit queue hiwater mark (records) = 100.

# auditconfig -getplugin audit_binfile
Plugin: audit_binfile
 Attributes: p_dir=/var/audit;p_fsize=2G;p_minfree=1;
# auditconfig -setplugin audit_binfile "p_minfree=2;p_fsize=4G" 200
# auditconfig -getplugin audit_binfile
Plugin: audit_binfile
 Attributes: p_dir=/var/audit;p_fsize=4G;p_minfree=2;
 Queue size: 200
The changed specifications go into effect after the administrator refreshes the audit service.

```
# audit -s
```

**Example 28–16  Removing Queue Size for an Audit Plugin**

In the following example, the queue size for the audit_binfile plugin is removed.

```
# auditconfig -getplugin audit_binfile
Plugin: audit_binfile
 Attributes: p_dir=/var/audit; p_fsize=4G; p_minfree=2;
 Queue size: 200
# auditconfig -setplugin audit_binfile "" 0
# auditconfig -getplugin audit_binfile
Plugin: audit_binfile
 Attributes: p_dir=/var/audit; p_fsize=4G; p_minfree=2;
```

The empty quotation marks ("") retain the current attribute values. The final 0 sets the queue size for the plugin to the default.

The change in qsize specification for the plugin goes into effect after the administrator refreshes the audit service.

```
# audit -s
```

**Example 28–17  Setting a Soft Limit for Warnings**

In this example, the minimum free-space level for all audit file systems is set so that a warning is issued when two percent of the file system is still available.

```
# auditconfig -setplugin audit_binfile p_minfree=2
```

The default percentage is one (1). For a large ZFS pool, choose a reasonably low percentage. For example, 10 percent of a 16 TB pool is around 16 GB, which would warn the audit administrator when plenty of disk space remains. A value of 2 sends the audit_warn message when about two GB of disk space remains.

The audit_warn email alias receives the warning. To set up the alias, see “How to Configure the audit_warn Email Alias” on page 562.

For a large pool, the administrator also limits the file size to 3 GB.

```
# auditconfig -setplugin audit_binfile p_fsize=3G
```

The p_minfree and p_fsize specifications for the plugin go into effect after the administrator refreshes the audit service.

```
# audit -s
```
How to Send Audit Files to a Remote Repository

In this procedure, you use attributes of the audit_remote plugin to send the audit trail to a remote audit repository. To configure a remote repository on an Oracle Solaris system, see “How to Configure a Remote Repository for Audit Files” on page 573.

Before You Begin

You must have a receiver of audit files at your remote repository. You must become an administrator who is assigned the Audit Configuration rights profile. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1 Determine the attributes of the audit_remote plugin.

Read the OBJECT ATTRIBUTES section of the audit_remote manpage.

```
# man audit_remote
...
OBJECT ATTRIBUTES
    The p_hosts attribute specifies the remote servers.
    You can also specify the port number and the GSS-API mechanism.

    The p_retries attribute specifies the number of retries for connecting and sending data. The default is 3.

    The p_timeout attribute specifies the number of seconds in which a connection times out.

The default port is the solaris_audit IANA-assigned port, 16162/tcp. The default mechanism is kerberos_v5. The timeout default is 5 seconds. You can also specify a queue size for the plugin.
```

2 To specify the remote receiving system, use the p_hosts attribute.

In this example, the receiving system uses a different port.

```
# auditconfig -setplugin audit_remote p_hosts=ars.example.com:16088:kerberos_v5
```

3 To change other attributes of the plugin, specify them.

For example, the following command specifies values for all optional attributes:

```
# auditconfig -setplugin audit_remote "p_retries=;p_timeout=3" 300
```

4 Verify the values, then activate the plugin.

For example, the following commands specify and verify the values of the plugin:

```
# auditconfig -getplugin audit_remote
Plugin: audit_remote (inactive)
    Attributes: p_hosts=ars.example.com:16088:kerberos_v5;p_retries=5;p_timeout=3;
    Queue size: 300
# auditconfig -setplugin audit_remote active
```
5 **Refresh the audit service.**

The audit service reads the audit plugin change upon refresh.

```
# audit -s
```

**Example 28–18 Tuning the Audit Queue Buffer Size**

In this example, the audit queue is full behind the `audit_remote` plugin. This audited system is configured to audit many classes and is transmitting across a high-traffic, slow network. The administrator enlarges the plugin’s buffer size to enable the audit queue to grow and not exceed the buffer’s limit before records are removed from the queue.

```
audsys1 # auditconfig -setplugin audit_remote "" 1000
audsys1 # audit -s
```

▼ **How to Configure a Remote Repository for Audit Files**

In this procedure, you configure a remote system, the Audit Remote Server (ARS), to receive and store audit records from one or more audited systems. Then, you activate the audit daemon on the remote server.

The configuration is twofold. First, you configure the underlying security mechanisms to securely transport the audit data, that is, you configure the KDC. Second, you configure the audit service on both the audited system and the ARS. This procedure illustrates a scenario with one audited client and one ARS, where the ARS and the KDC are on the same server. More complex scenarios can be configured similarly.

**Before You Begin**

You must assume the root role. You have installed the Kerberos packages, as described in “How to Prepare to Stream Audit Records to Remote Storage” on page 545. You are working with an administrator who has configured the audited system, as described in “How to Send Audit Files to a Remote Repository” on page 572.

1 **Configure the underlying security transport mechanism.**

You need a KDC on a system that both the audited system and the ARS can use, a host principal for each system, and an audit service principal.

a. **Configure the KDC.**

If your site has configured a KDC, use it and continue with Step c. The following series of commands illustrate a KDC configuration strategy:

```
arstore # kdcmgr -a audr/admin -r EXAMPLE.COM create master
```

This command uses the administrative principal `audr/admin` to create a master KDC in the `EXAMPLE.COM` realm, enables the master KDC, and starts the Kerberos service.
b. Verify that the KDC is available.

For more information, see the `kdcmgr(1M)` man page.

```bash
# kdcmgr status
KDC Status Information
--------------------------
svc:/network/security/krb5kdc:default (Kerberos key distribution center)
  State: online since Wed Feb 29 01:59:27 2012
  See: man -M /usr/share/man -s 1M krb5kdc
  See: /var/svc/log/network-security-krb5kdc:default.log
Impact: None.

KDC Master Status Information
-----------------------------
svc:/network/security/kadmin:default (Kerberos administration daemon)
  State: online since Wed Feb 29 01:59:28 2012
  See: man -M /usr/share/man -s 1M kadmind
  See: /var/svc/log/network-security-kadmin:default.log
Impact: None.

Transaction Log Information
----------------------------
Kerberos update log (/var/krb5/principal.ulo)
Update log dump:
  Log version #: 1
  Log state: Stable
  Entry block size: 2048
  Number of entries: 13
  First serial #: 1
  Last serial #: 13
  First time stamp: Wed Feb 29 01:59:27 2012
  Last time stamp: Mon Mar 5 19:29:28 2012
```

c. Add the audit service principal to the KDC keytab file.

You can add the principal by typing the `kadmin.local` command on the KDC system. Or, you can remotely add the principal by using the `kadmin` command and providing a password. In this example, the arstore system is running the KDC.

```bash
# kadmin -p audr/admin
kadmin: addprinc -randkey audit/arstore.example.com@EXAMPLE.COM
kadmin: ktadd audit/arstore.example.com@EXAMPLE.COM
```

d. On each audited system, add keys.

The receiver and the sender must have keys.

```bash
enigma # kclient
  .. Enter the Kerberos realm: EXAMPLE.COM
  .. KDC hostname for the above realm: arstore.example.com
  .. Will this client need service keys? [y/n]: y
```
Configure the audit service on the ARS.

- To create a group that accepts audit records from any audited system in the Kerberos realm, name a connection group.
  
  ```bash
  # auditconfig -setremote group create Bank_A
  ```

  Bank_A is a connection group. Because the hosts attribute is not defined, this group accepts all connections, which means that it is a wildcard group. Any audited system in this Kerberos realm whose audit_remote plugin is correctly configured can reach this ARS.

- To limit connections to this group, specify the audited systems that can use this repository.
  
  ```bash
  # auditconfig -setremote group Bank_A "hosts=enigma.example.com"
  ```

  Connection group Bank_A now accepts only connections from the enigma system. A connection from any other host is refused.

- To prevent an audit file in this group from growing too large, set a maximum size.
  
  ```bash
  # auditconfig -setremote group Bank_A "binfile_size=4GB"
  ```

3 Configure the audit service on the audited system.

To specify the ARS, use the p_hosts attribute.

- To specify the ARS, use the p_hosts attribute.
  
  ```bash
  enigma # auditconfig -setplugin audit_remote active p_hosts=arstore.example.com
  ```

4 Refresh the audit service.

The audit service reads the audit plugin change upon refresh.

  ```bash
  # audit -s
  ```

The KDC now manages the connection between the audited system enigma and the ARS.

**Example 28–19 Streaming Audit Records to Different File Locations on the Same ARS**

This example extends the example in the procedure. The administrator separates audit records by host on the ARS by creating two connection groups.

Audit files from audsys1 stream to the Bank_A connection group on this ARS.

- To create a group that accepts audit records from any audited system in the Kerberos realm, name a connection group.
  
  ```bash
  # auditconfig -setremote group create Bank_A
  ```

- To specify the ARS, use the p_hosts attribute.
  
  ```bash
  arstore # auditconfig -setremote group active Bank_A "hosts=audsys1"
  ```
Audit files from audsys2 stream to the Bank_B connection group.

```
arstore # auditconfig -setremote group create Bank_B
arstore # auditconfig -setremote group active Bank_B \n    "hosts=audsys2;binfile_dir=/var/audit/audsys2;binfile_fsize=4M;"
```

For easier maintenance, the administrator sets other attribute values identically.

```
arstore # auditconfig -getremote
Audit Remote Server
   Attributes: listen_address=;login_grace_time=30;max_startups=10;listen_port=0;
Connection group: Bank_A
   Attributes: binfile_dir=/var/audit/audsys1;binfile_fsize=4M;binfile_minfree=1;
               hosts=audsys1
Connection group: Bank_B
   Attributes: binfile_dir=/var/audit/audsys2;binfile_fsize=4M;binfile_minfree=1;
               hosts=audsys2
```

**Example 28–20** Placing the ARS on a Different System From the KDC

In this example, the administrator places the ARS on a different system from the KDC. First, the administrator creates and configures the master KDC.

```
kserv # kdcadm -a audr/admin -r EXAMPLE.COM create master
kserv # kadmin.local -p audr/admin
kadmin: addprinc -randkey audit/arstore.example.com@EXAMPLE.COM
kadmin: ktadd -t /tmp/krb5.keytab.audit audit/arstore.example.com@EXAMPLE.COM
```

After securely transmitting the `/tmp/krb5.keytab.audit` file to the ARS, `arstore`, the administrator moves the file to the correct location.

```
arstore # chown root:root krb5.keytab.audit
arstore # chmod 600 krb5.keytab.audit
arstore # mv krb5.keytab.audit /etc/krb5/krb5.keytab
```

Rather than rewrite the file, the administrator also has the option to use the `ktutil` command on the ARS to merge the KDC `krb5.keytab.audit` file with existing keys in the `arstore`'s `/etc/krb5/krb5.keytab` file.

Finally, the administrator generates keys on the audited system.

```
enigma # kclient
   .. Enter the Kerberos realm: EXAMPLE.COM
   .. KDC hostname for the above realm: kserv.example.com
   .. Will this client need service keys? [y/n]: y
```
How to Configure syslog Audit Logs

You can instruct the audit service to copy some or all of the audit records in the audit queue to the syslog utility. If you record both binary audit data and text summaries, the binary data provide a complete audit record, while the summaries filter the data for real-time review.

Before You Begin

To configure the audit_syslog plugin, you must become an administrator who is assigned the Audit Configuration rights profile. To configure the syslog utility and create the auditlog file, you must assume the root role.

1. Select audit classes to be sent to the audit_syslog plugin, and make the plugin active.

   Note – p_flags audit classes must be preselected as either system defaults or in a user’s or a rights profile’s auditflags. Records are not collected for a class that is not preselected.

   # auditconfig -setplugin audit_syslog active p_flags=lo,+as,-ss

2. Determine which system-log service instance is online.

   # svc system-log

<table>
<thead>
<tr>
<th>STATE</th>
<th>STIME</th>
<th>FMRI</th>
</tr>
</thead>
<tbody>
<tr>
<td>disabled</td>
<td>13:11:55</td>
<td>svc:/system/system-log:rsyslog</td>
</tr>
<tr>
<td>online</td>
<td>13:13:27</td>
<td>svc:/system/system-log:default</td>
</tr>
</tbody>
</table>

   Note – If the rsyslog service instance is online, modify the rsyslog.conf file.

3. Configure the syslog utility.

   a. Add an audit.notice entry to the syslog.conf file.

      The entry includes the location of the log file.

      # cat /etc/syslog.conf

      ...audit.notice /var/adm/auditlog

   b. Create the log file.

      # touch /var/adm/auditlog

   c. Refresh the configuration information for the system-log service.

      # svcadm refresh system-log:default

      Note – Refresh the system-log:rsyslog service instance if the rsyslog service is online.
4 Refresh the audit service.
The audit service reads the changes to the audit plugin upon refresh.

    # audit -s

5 Regularly archive the syslog log files.
The audit service can generate extensive output. To manage the logs, see the `logadm(1M)` man page.

Example 28–21 Specifying Audit Classes for syslog Output

In the following example, the syslog utility collects a subset of the preselected audit classes. The pf class is created in Example 28–10.

    # auditconfig -setnaflags lo,na
    # auditconfig -setflags lo,ss
    # usermod -K audit_flags=pf:no jdoe
    # auditconfig -setplugin audit_syslog active p_flags=lo,+na,-ss,+pf

The arguments to the auditconfig command instruct the system to collect all login/logout, non-attributable, and change of system state audit records. The audit_syslog plugin entry instructs the syslog utility to collect all logins, successful non-attributable events, and failed changes of system state.

For the jdoe user, the binary utility collects successful and failed calls to the pfexec command. The syslog utility collects successful calls to the pfexec command.

Example 28–22 Putting syslog Audit Records on a Remote System

You can change the audit.notice entry in the syslog.conf file to point to a remote system. In this example, the name of the local system is sys1.1. The remote system is remote1.

    sys1.1 # cat /etc/syslog.conf
    ...
    audit.notice @remote1

The audit.notice entry in the syslog.conf file on the remote1 system points to the log file.

    remote1 # cat /etc/syslog.conf
    ...
    audit.notice /var/adm/auditlog
Configuring the Audit Service in Zones (Tasks)

The audit service audits the entire system, including audit events in zones. A system that has installed non-global zones can audit all zones identically, or can configure auditing per zone. For more information, see “How to Plan Auditing in Zones” on page 540.

When you audit the non-global zones exactly as the global zone is audited, the non-global zone administrators might not have access to the audit records. Also, the global zone administrator can modify the audit preselection masks of users in non-global zones.

When you audit the non-global zones individually, the audit records are visible to the non-global zone and to the global zone from the non-global zone root.

◆ How to Configure All Zones Identically for Auditing

This procedure enables audits every zone identically. This method requires the least computer overhead and administrative resources.

**Before You Begin**

You must assume the root role. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1. **Configure the global zone for auditing.**
   
   Complete the tasks in “Configuring the Audit Service (Task Map)” on page 551, with the following exceptions:
   - Do not enable per zone audit policy.
   - Set the zonename policy. This policy adds the name of the zone to every audit record.
     
     ```
     # auditconfig -setpolicy +zonename
     ```
   
2. **If you modified audit configuration files, copy them from the global zone to every non-global zone.**
   
   If you modified the audit_class or audit_event file, copy it in one of two ways:
   - You can loopback mount the files.
   - You can copy the files.
   
   The non-global zone must be running.
   
   - **Mount the changed audit_class and audit_event files as a loopback file system (lofs).**
     
     a. From the global zone, halt the non-global zone.
     
     ```
     # zoneadm -z non-global-zone halt
     ```
b. Create a read-only loopback mount for every audit configuration file that you modified in the global zone.

```
# zonecfg -z non-global-zone
add fs
  set special=/etc/security/audit-file
  set dir=/etc/security/audit-file
  set type=lofs
  add options [ro,nodevices,nosetuid]
  commit
end
exit
```

To make the changes effective, boot the non-global zone.

```
# zoneadm -z non-global-zone boot
```

Later, if you modify an audit configuration file in the global zone, you reboot each zone to refresh the loopback-mounted files in the non-global zones.

- Copy the files.

a. From the global zone, list the `/etc/security` directory in each non-global zone.

```
# ls /zone/zonename/root/etc/security/
```

b. Copy the changed `audit_class` and `audit_event` files to each zone's `/etc/security` directory.

```
# cp /etc/security/audit-file /zone/zonename/root/etc/security/audit-file
```

Later, if you change one of these files in the global zone, you must copy the changed file to the non-global zones.

The non-global zones are audited when the audit service is restarted in the global zone or when the zones are rebooted.

### Example 28–23 Mounting Audit Configuration Files as Loopback Mounts in a Zone

In this example, the system administrator has modified the `audit_class`, `audit_event`, and `audit_warn` files.

The `audit_warn` file is read in the global zone only, so does not have to be mounted into the non-global zones.

On this system, `machine1`, the administrator has created two non-global zones, `machine1–webserver` and `machine1–appserver`. The administrator has finished modifying the audit configuration files. If the administrator later modifies the files, the zone must be rebooted to re-read the loopback mounts.

```
# zoneadm -z machine1-webserver halt
# zoneadm -z machine1-appserver halt
# zonecfg -z machine1-webserver
```
When the non-global zones are rebooted, the audit_class and audit_event files are read-only in the zones.

▼ How to Configure Per-Zone Auditing

This procedure enables separate zone administrators to control the audit service in their zone. For the complete list of policy options, see the auditconfig(1M) man page.

Before You Begin

To configure auditing, you must become an administrator who is assigned the Audit Configuration rights profile. To enable the audit service, you must become an administrator who is assigned the Audit Control rights profile. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1 In the global zone, configure auditing.
   a. Complete the tasks in “Configuring the Audit Service (Task Map)” on page 551.
   b. Add the per-zone audit policy. For the command, see Example 28–8.

   Note – You are not required to enable the audit service in the global zone.

2 In each non-global zone that you plan to audit, configure the audit files.
   a. Complete the tasks in “Configuring the Audit Service (Task Map)” on page 551.
b. Do not configure system-wide audit settings. Specifically, do not add the perzone or ahlt policy to the non-global zone.

3   Enable auditing in your zone.
    myzone# audit -s

**Example 28–24** Disabling Auditing in a Non-Global Zone

This example works if the perzone audit policy is set. The zone administrator of the noaudit zone disables auditing for that zone.

noauditzone # auditconfig -getcond
audit condition = auditing
noauditzone # audit -t
noauditzone # auditconfig -getcond
audit condition = noaudit

---

**Enabling and Disabling the Audit Service (Tasks)**

The audit service is enabled by default. If the perzone audit policy is set in the global zone, zone administrators can enable, refresh, and disable the audit service in their non-global zones.

**▼ How to Refresh the Audit Service**

This procedure updates the audit service when you have changed the configuration of an audit plugin after the audit service is enabled.

**Before You Begin**
You must become an administrator who is assigned the Audit Control rights profile. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1   Refresh the audit service.
    # audit -s

**Note** – When you refresh the audit service, all temporary configuration settings are lost. Audit policy and queue controls allow temporary settings. For more information, see the auditconfig(1M) man page.

2   Update the preselection masks of users who are currently being audited.
Refresh the audit service does not change the masks of existing processes. To explicitly reset the preselection mask for an existing process, see “How to Update the Preselection Mask of Logged In Users” on page 604.
Example 28-25 Refreshing an Enabled Audit Service

In this example, the administrator reconfigures auditing, verifies the changes, then refreshes the audit service.

- First, the administrator adds a temporary policy.

  ```
  # auditconfig -t -setpolicy +zonename
  # auditconfig -getpolicy
  configured audit policies = ahlt,arge,argv,perzone
  active audit policies = ahlt,arge,argv,perzone,zonename
  ```

- Then, the administrator specifies queue controls.

  ```
  # auditconfig -setqctrl 200 20 0 0
  # auditconfig -getqctrl
  configured audit queue hiwater mark (records) = 200
  configured audit queue lowater mark (records) = 20
  configured audit queue buffer size (bytes) = 8192
  configured audit queue delay (ticks) = 20
  active audit queue hiwater mark (records) = 200
  active audit queue lowater mark (records) = 20
  active audit queue buffer size (bytes) = 8192
  active audit queue delay (ticks) = 20
  ```

- Then, the administrator specifies plugin attributes.

  - For the `audit_binfile` plugin, the administrator removes the `qsize` value.

    ```
    # auditconfig -getplugin audit_binfile
    Plugin: audit_binfile
    Attributes: p_dir=/audit/sys1.1,/var/audit;
    p_minfree=2;p_fsize=4G;
    Queue size: 200
    # auditconfig -setplugin audit_binfile "" 0
    ```

    - For the `audit_syslog` plugin, the administrator specifies that successful login and logout events and failed executables be sent to `syslog`. The `qsize` for this plugin is set to 150.

      ```
      # auditconfig -setplugin audit_syslog active p_flags=+lo,-ex 150
      ```

  - The administrator does not configure or use the `audit_remote` plugin.

- Then, the administrator refreshes the audit service and verifies the configuration.

  ```
  # audit -s
  # auditconfig -getpolicy
  configured audit policies = ahlt,arge,argv,perzone
  active audit policies = ahlt,arge,argv,perzone
  ```
The queue controls remain the same.

```bash
# auditconfig -getqctrl
configured audit queue hiwater mark (records) = 200
configured audit queue lowater mark (records) = 20
configured audit queue buffer size (bytes) = 8192
configured audit queue delay (ticks) = 20
active audit queue hiwater mark (records) = 200
active audit queue lowater mark (records) = 20
active audit queue buffer size (bytes) = 8192
active audit queue delay (ticks) = 20
```

The `audit_binfile` plugin does not have a specified queue size. The `audit_syslog` plugin has a specified queue size.

```bash
# auditconfig -getplugin
Plugin: audit_binfile
Attributes: p_dir=/var/audit;p_fsize=4G;p_minfree=2;

Plugin: audit_syslog
Attributes: p_flags=+lo,-ex;
Queue size: 50
...```

### How to Disable the Audit Service

This procedure shows how to disable auditing in the global zone and in a non-global zone when the `perzone` audit policy is set. After the `perzone` policy is set in the global zone, a non-global zone that has enabled auditing continues to collect audit records across global zone reboots and non-global zone reboots.

**Before You Begin**

To disable or enable the audit service, you must become an administrator who is assigned the Audit Control rights profile. For more information, see "How to Use Your Assigned Administrative Rights" on page 157.

- **Run the `audit -t` command to disable the audit service.**
  
  For more information, see the `audit(1M)` and `audited(1M)` man pages.

  - **In the global zone, disable the audit service.**
    
    ```bash
    # audit -t
    ```

    If the `perzone` audit policy is not set, this command disables auditing in all zones. If the `perzone` audit policy is set, the non-global zones are not affected.

  - **In a non-global zone, disable the audit service.**
    
    If the `perzone` audit policy is set, the non-global zone administrator must disable the service in the non-global zone.

    ```bash
    zone1 # audit -t
    ```
How to Enable the Audit Service

This procedure enables the audit service for all zones after the service is disabled by an administrator. To start the audit service in a non-global zone, see Example 28–26.

Before You Begin

To enable or disable the audit service, you must become an administrator who is assigned the Audit Control rights profile. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1 Use the audit -s command to enable the audit service.
   
   # audit -s
   
   For more information, see the audit(1M) man page.

2 Verify that auditing is enabled.
   
   # auditconfig -getcond
   
   audit condition = auditing

Example 28–26 Enabling Auditing in a Non-Global Zone

In this example, the zone administrator enables the audit service for zone1 after taking the following actions are taken:

- The global zone administrator sets the per zone policy in the global zone.
- The zone administrator of the non-global zone configures the audit service and per-user customizations.

Then, the zone administrator enables the audit service for the zone.

zone1# audit -s

Managing Audit Records on Local Systems (Tasks)

The default plugin, audit_binfile, creates an audit trail. The trail can contain large amounts of data. The following tasks show you how to work with all this data.

Managing Audit Records on Local Systems (Task Map)

The following task map points to procedures for selecting, analyzing, and managing audit records.
How to Display Audit Record Definitions

The `auditrecord` command displays audit record definitions. The definitions provide the audit event number, audit class, selection mask, and record format of an audit event.

- **Put the definitions of all audit event records in an HTML file.**

  The `-a` option lists all audit event definitions. The `-h` option puts the list in HTML format.

  ```bash
  % auditrecord -ah > audit.events.html
  ```

  **Tip** – When you display the HTML file in a browser, use the browser's Find tool to find specific audit record definitions.

  For more information, see the `auditrecord(1M)` man page.

Example 28–27  Displaying the Audit Record Definitions of a Program

In this example, the definition of all audit records that are generated by the `login` program are displayed. The login programs include `rlogin`, `telnet`, `newgrp`, and the Secure Shell feature of Oracle Solaris.

```bash
% auditrecord -p login
...
login: logout
  program  logout
  event ID  6153
  class  lo

newgrp
```
Displaying the Audit Record Definitions of an Audit Class

In this example, the definitions of all audit records in the pf class that was created in Example 28–10 is displayed.

```
% auditrecord -c pf
```

The `use_of_privilege` token is recorded whenever privilege is used. The `privileges` tokens are recorded if the limit or inheritable set is changed. The `process` token is recorded if an ID is changed. No policy option is required for these tokens to be included in the record.

How to Merge Audit Files From the Audit Trail

By merging the audit files from all the audit directories, you can analyze the contents of the entire audit trail.
Note – Because the timestamps in the audit trail are in Coordinated Universal Time (UTC), the date and hour must be translated to the current time zone to be meaningful. Be aware of this point whenever you manipulate these files with standard file commands rather than with the auditreduce command.

Before You Begin

You must become an administrator who is assigned the Audit Review rights profile. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1 Create a file system for storing merged audit files.

To lessen the chance of reaching the limit of disk space, this file system should be in a different zpool from the file systems that you created in “How to Create ZFS File Systems for Audit Files” on page 566 to store the original files.

2 Merge the audit records in the audit trail.

Go to the directory for storing merged audit files. From this directory, merge the audit records into a file with a named suffix. All directories in the audit trail on the local system are merged and placed in this directory.

```
# cd audit-storage-directory
# auditreduce -Uppercase-option -O suffix
```

The uppercase options to the auditreduce command manipulate files in the audit trail. The uppercase options include the following:

- **-A** Selects all of the files in the audit trail.
- **-C** Selects complete files only.
- **-M** Selects files with a particular suffix. The suffix can be a machine name, or it can be a suffix that you have specified for a summary file.
- **-O** Creates an audit file with 14-character timestamps for both the start time and the end time, with the suffix suffix in the current directory.
- **-R pathname** Specifies to read audit files in pathname, an alternate audit root directory.
- **-S server** Specifies to read audit files from the specified server.

For the full list of options, see the audit reduce(1M) man page.

Example 28–29 Copying Audit Files to a Summary File

In the following example, an administrator who is assigned the System Administrator rights profile copies all files from the audit trail into a merged file on a different file system. The /var/audit/storage file system is on a separate disk from the /var/audit file system, the audit root file system.
$ cd /var/audit/storage
$ auditreduce -A -O All
$ ls /var/audit/storage/*All
20100827183214.20100827215318.All

In the following example, only complete files are copied from the audit trail into a merged file. The complete path is specified as the value of the -O option. The last component of the path, Complete, is used as the suffix.

$ auditreduce -C -O /var/audit/storage/Complete
$ ls /var/audit/storage/*Complete
20100827183214.20100827214217.Complete

In the following example, by adding the -D option, the original audit files are deleted.

$ auditreduce -C -O daily_sys1.1 -D sys1.1
$ ls *sys1.1
20100827183214.20100827214217.daily_sys1.1

▼ How to Select Audit Events From the Audit Trail

You can filter audit records for examination. For the complete list of filtering options, see the auditreduce(1M) man page.

Before You Begin

You must become an administrator who is assigned the Audit Review rights profile. For more information, see "How to Use Your Assigned Administrative Rights" on page 157.

Select the kinds of records that you want from the audit trail, or from a specified audit file.

auditreduce -lowercase-option argument [optional-file]

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-d</td>
<td>Selects all of the events on a particular date. The date format for argument is yyyymmdd. Other date options, -b and -a, select events before and after a particular date.</td>
</tr>
<tr>
<td>-u</td>
<td>Selects all of the events attributable to a particular user. The argument is a user name. Another user option, -e, selects all of the events attributable to an effective user ID.</td>
</tr>
<tr>
<td>-c</td>
<td>Selects all of the events in a preselected audit class. The argument is an audit class name.</td>
</tr>
<tr>
<td>-m</td>
<td>Selects all of the instances of a particular audit event. The argument is an audit event.</td>
</tr>
</tbody>
</table>
Selects by object type. Use this option to select by file, group, file owner, FMRI, pid, and other object types.

optional-file Is the name of an audit file.

For the full list of options, see the auditreduce(1M) man page.

Example 28–30 Combining and Reducing Audit Files

The auditreduce command can eliminate the less interesting records as it combines the input files. For example, you might use the auditreduce command to retain only the login and logout records in audit files that are over a month old. If you need to retrieve the complete audit trail, you could recover the trail from backup media.

```
# cd /var/audit/audit_summary
# auditreduce -O lo.summary -b 20100827 -c lo; compress *lo.summary
```

Example 28–31 Copying One User’s Audit Records to a Summary File

In this example, the records in the audit trail that contain the name of a particular user are merged. The -e option finds the effective user. The -u option finds the login user.

```
$ cd /var/audit/audit_summary
$ auditreduce -e tamiko -O tamiko
```

You can look for specific events in this file. In the following example, what time the user logged in and out on Sept 7, 2010, your time, is checked. Only those files with the user’s name as the file suffix are checked. The short form of the date is yyyymmdd.

```
# auditreduce -M tamiko -O tamikolo -d 20100907 -u tamiko -c lo
```

Example 28–32 Copying Selected Records to a Single File

In this example, login and logout records for a particular day are selected from the audit trail. The records are merged into a target file. The target file is written in a file system other than the file system that contains the audit root directory.

```
# auditreduce -c lo -d 20100827 -O /var/audit/audit_summary/logins
# ls /var/audit/audit_summary/*logins
/var/audit/audit_summary/20100827183936.20100827232326.logins
```
How to View the Contents of Binary Audit Files

The `praudit` command enables you to view the contents of binary audit files. You can pipe the output from the `auditreduce` command, or you can read a particular audit file. The `-x` option is useful for further processing.

### Before You Begin
You must become an administrator who is assigned the Audit Review rights profile. For more information, see "How to Use Your Assigned Administrative Rights" on page 157.

- Use one of the following `praudit` commands to produce the output that is best for your purposes.

  The following examples show `praudit` output from the same audit event. Audit policy has been set to include the sequence token.

  - The `praudit -s` command displays audit records in a short format, one token per line. Use the `-l` option to place each record on one line.

    ```bash
    $ auditreduce -c lo | praudit -s
    header,69,2,AUE_screenlock,,mach1,2010-10-14 08:02:56.348 -07:00
    subject,jdoe,root,staff,jdoe,staff,856,50036632,82 0 mach1
    return,success,0
    sequence,1298
    ```

  - The `praudit -r` command displays audit records in their raw format, one token per line. Use the `-l` option to place each record on one line.

    ```bash
    $ auditreduce -c lo | praudit -r
    21,69,2,6222,0x0000,10.132.136.45,1287070091,698391050
    36,26700,0,10,26700,10,856,50036632,82 0 10.132.136.45
    39,0,0
    47,1298
    ```

  - The `praudit -x` command displays audit records in XML format, one token per line. Use the `-l` option to place each record on one line. The following listing divides two lines of output to fit on this printed page:

    ```xml
    <record version="2" event="screenlock - unlock" host="mach1"
    iso8601="2010-10-14 08:28:11.698 -07:00">
    <subject audit-uid="jdoe" uid="root" gid="staff" ruid="jdoe"
    rgid="staff" pid="856" sid="50036632" tid="82 0 mach1"/>
    <return errval="success" retval="0"/>
    <sequence seq-num="1298"/>
    </record>
    ```

### Example 28–33 Printing the Entire Audit Trail
With a pipe to the print command, the output for the entire audit trail goes to the printer. For security reasons, the printer has limited access.
# auditreduce | praudit | lp -d example.protected.printer

**Example 28–34** Viewing a Specific Audit File

In this example, a summary login file is examined in a terminal window.

```bash
# cd /var/audit/audit_summary/logins
# praudit 20100827183936.20100827232326.logins | more
```

**Example 28–35** Putting Audit Records in XML Format

In this example, the audit records are converted to XML format.

```bash
# praudit -x 20100827183214.20100827215318.logins > 20100827.logins.xml
```

The XML file can be displayed in a browser. The contents of the file can be operated on by a script to extract the relevant information.

**Example 28–36** Processing praudit Output With a Script

You might want to process output from the praudit command as lines of text. For example, you might want to select records that the auditreduce command cannot select. You can use a simple shell script to process the output of the praudit command. The following sample script puts one audit record on one line, searches for a user-specified string, then returns the audit file to its original form.

```bash
#!/bin/sh
#
## This script takes an argument of a user-specified string.
# The sed command prefixes the header tokens with Control-A
# The first tr command puts the audit tokens for one record
# onto one line while preserving the line breaks as Control-A
# praudit | sed -e '1,2d' -e '$s/^file.*$/\012\001\002\012\012/' -e 's/^header/^aheader/' \ |
| tr \012\001 \002\012 \ \ | tr \002 \012 \ Fonts the user-specified string
| tr \002 \012 \ Restores the original newline breaks
```

Note that the ^a in the script is Control-A, not the two characters ^ and a. The prefix distinguishes the header token from the string header that might appear as text.

**Troubleshooting**

A message similar to the following indicates that you do not have enough privilege to use the praudit command:

```
praudit: Can’t assign 20090408164827.20090408171614.sys1.1 to stdin.
```
Run the `praudit` command in a profile shell. You must become an administrator who is assigned the Audit Review rights profile. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

### How to Clean Up a `not_terminated` Audit File

When anomalous system interruptions occur, the audit service exits while its audit file is still open. Or, a file system becomes inaccessible and forces the system to switch to a new file system. In such instances, an audit file remains with the string `not_terminated` as the end timestamp, even though the file is no longer used for audit records. Use the `auditreduce -O` command to give the file the correct timestamp.

**Before You Begin**

You must become an administrator who is assigned the Audit Review rights profile. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1. **List the files with the `not_terminated` string on your audit file system in order of creation.**
   
   ```bash
   # ls -Rlt audit-directory/* | grep not_terminated
   #R     Lists files in subdirectories.
   #  -t     Lists files from most recent to oldest.
   #  -1     Lists the files in one column.
   ```

2. **Clean up the old `not_terminated` file.**
   
   Specify the name of the old file to the `auditreduce -O` command.
   
   ```bash
   # auditreduce -O system-name old-not-terminated-file
   ```

3. **Remove the old `not_terminated` file.**
   
   ```bash
   # rm system-name old-not-terminated-file
   ```

**Example 28–37** Cleaning Up Closed `not_terminated` Audit Files

In the following example, `not_terminated` files are found, renamed, then the originals are removed.

```
ls -Rlt */* | grep not_terminated
.../egret.1/2010090162220.not_terminated.egret
.../egret.1/2010090162220.2010090162220.not_terminated.egret
# cd */egret.1
# auditreduce -O egret 2010090162220.not_terminated.egret
# ls -lt
2010090162220.not_terminated.egret     Current audit file
2010090162220.2010090162220.not_terminated.egret  Cleaned up audit file
20100827230920.201008300909.egret     Input (old) audit file
```

Chapter 28 • Managing Auditing (Tasks) 593
The start timestamp on the new file reflects the time of the first audit event in the `not_terminated` file. The end timestamp reflects the time of the last audit event in the file.

### How to Prevent Audit Trail Overflow

If your security policy requires that all audit data be saved, prevent audit record loss.

**Before You Begin**

You must assume the root role. For more information, see "How to Use Your Assigned Administrative Rights" on page 157.

1. **Set a minimum free size on the audit_binfile plugin.**
   
   Use the `p_minfree` attribute.
   
   The `audit_warn` email alias sends a warning when the disk space fills to the minimum free size. See Example 28–17.

2. **Set up a schedule to regularly archive audit files.**
   
   Archive audit files by backing up the files to offline media. You can also move the files to an archive file system.
   
   If you are collecting text audit logs with the `syslog` utility, archive the text logs. For more information, see the `logadm(1M)` man page.

3. **Set up a schedule to delete the archived audit files from the audit file system.**

4. **Save and store auxiliary information.**
   
   Archive information that is necessary to interpret audit records along with the audit trail. Minimally, you save the `passwd`, `group`, and `hosts` files. You also might archive the `audit_event` and `audit_class` files.

5. **Keep records of which audit files have been archived.**

6. **Store the archived media appropriately.**

7. **Reduce the amount of file system capacity that is required by enabling ZFS compression.**
   
   On a ZFS file system that is dedicated to audit files, compression shrinks the files considerably. For an example, see “How to Compress Audit Files on a Dedicated File System” on page 606.
See also “Interactions Between ZFS Compression, Deduplication, and Encryption Properties” in Oracle Solaris 11.1 Administration: ZFS File Systems.

8 Reduce the volume of audit data that you store by creating summary files.

You can extract summary files from the audit trail by using options to the audit reduce command. The summary files contain only records for specified types of audit events. To extract summary files, see Example 28–30 and Example 28–32.

Troubleshooting the Audit Service (Tasks)

This section covers various auditing error messages, preferences, and the auditing that is provided by other tools. These procedures can help you record required audit events and debug audit problems.

Troubleshooting the Audit Service (Task Map)

The following task map points to procedures for troubleshooting auditing.

<table>
<thead>
<tr>
<th>Problem</th>
<th>Solution</th>
<th>For Instructions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Why are audit records not being logged when I have configured auditing?</td>
<td>Troubleshoot the audit service.</td>
<td>&quot;How to Determine That Auditing Is Running&quot; on page 596</td>
</tr>
<tr>
<td>How can I reduce the amount of audit information that is being collected?</td>
<td>Audit just the events that you want to audit.</td>
<td>&quot;How to Lessen the Volume of Audit Records That Are Produced&quot; on page 598</td>
</tr>
<tr>
<td>How can I audit everything that a user does on the system?</td>
<td>Audit one or more users for every command.</td>
<td>&quot;How to Audit All Commands by Users&quot; on page 600</td>
</tr>
<tr>
<td>How can I change the audit events that are being recorded and have the change affect existing sessions?</td>
<td>Update a user’s preselection mask.</td>
<td>&quot;How to Update the Preselection Mask of Logged In Users&quot; on page 604</td>
</tr>
<tr>
<td>How can I locate modifications to particular files?</td>
<td>Audit file modifications, then use the audit reduce command to find particular files.</td>
<td>&quot;How to Find Audit Records of Changes to Specific Files&quot; on page 602</td>
</tr>
<tr>
<td>How can I reduce the size of my audit files?</td>
<td>Limit the size of the binary audit file.</td>
<td>&quot;How to Limit the Size of Binary Audit Files&quot; on page 606</td>
</tr>
<tr>
<td>How can I use less file system space for audit files?</td>
<td>Use ZFS quotas and compression.</td>
<td>&quot;How to Compress Audit Files on a Dedicated File System&quot; on page 606</td>
</tr>
<tr>
<td>How can I remove audit events from the audit_event file?</td>
<td>Correctly update the audit_event file.</td>
<td>&quot;How to Prevent the Auditing of Specific Events&quot; on page 605</td>
</tr>
</tbody>
</table>
Troubleshooting the Audit Service (Tasks)

<table>
<thead>
<tr>
<th>Problem</th>
<th>Solution</th>
<th>For Instructions</th>
</tr>
</thead>
<tbody>
<tr>
<td>How can I audit all logins to an Oracle Solaris system?</td>
<td>Audit logins from any system.</td>
<td>“How to Audit Logins From Other Operating Systems” on page 607</td>
</tr>
<tr>
<td>Why are auditing records not being kept for my FTP transfers?</td>
<td>Use the appropriate auditing tool for utilities that generate their own logs.</td>
<td>“How to Audit FTP and SFTP File Transfers” on page 608</td>
</tr>
</tbody>
</table>

**How to Determine That Auditing Is Running**

Auditing is enabled by default. If you believe that auditing has not been disabled, but no audit records are being sent to the active plugin, use the following procedure to isolate the issue.

**Before You Begin**

To modify a system file, you must be assigned the `solaris.admin.edit/path-to-system-file` authorization. By default, the root role has this authorization. To configure auditing, you must become an administrator who is assigned the Audit Configuration rights profile.

1. **Determine that auditing is running.**
   Use any of the following methods:

   - **Verify the current audit condition.**
     The following listing indicates that auditing is not running:
     
     ```shell
     # auditconfig -getcond
     audit condition = noaudit
     ```
     
     The following listing indicates that auditing is running:
     
     ```shell
     # auditconfig -getcond
     audit condition = auditing
     ```

   - **Verify that the audit service is running.**
     The following listing indicates that auditing is not running:
     
     ```shell
     # svcs -x auditd
     svc:/system/auditd:default (Solaris audit daemon)
     State: disabled since Sun Oct 10 10:10:10 2010
     Reason: Disabled by an administrator.
     See: http://support.oracle.com/msg/SMF-8000-05
     See: auditd(1M)
     See: audit(1M)
     See: auditconfig(1M)
     See: audit_flags(5)
     See: audit_binfile(5)
     See: audit_syslog(5)
     See: audit_remote(5)
     See: /var/svc/log/system-auditd:default.log
     Impact: This service is not running.
     ```
The following listing indicates that the audit service is running:

```
# svcs auditd
STATE STIME FMRI
online 10:10:10 svc:/system/auditd:default
```

If the audit service is not running, enable it. For the procedure, see “How to Enable the Audit Service” on page 585.

2  Verify that at least one plugin is active.

```
# audit -v
audit: no active plugin found
```

If no plugin is active, make one active.

```
# auditconfig -setplugin audit_binfile active
# audit -v
configuration ok
```

3  If you created a customized audit class, verify that you assigned events to the class.

For example, the following list of flags contains the pf class, which Oracle Solaris software did not deliver:

```
# auditconfig -getflags
active user default audit flags = pf,lo(0x0100000000000000,00x0100000000000000)
configured user default audit flags = pf,lo(0x0100000000000000,00x0100000000000000)
```

For a description of creating the pf class, see “How to Add an Audit Class” on page 563.

a.  Verify that the class is defined in the audit_class file.

The audit class must be defined, and its mask must be unique.

```
# grep pf /etc/security/audit_class Verify class exists
0x0100000000000000:pf:profile
# grep 0x0100000000000000 /etc/security/audit_class Ensure mask is unique
0x0100000000000000:pf:profile
```

Replace a mask that is not unique. If the class is not defined, define it. Otherwise, run the auditconfig -setflags command with valid values to reset the current flags.

b.  Verify that events have been assigned to the class.

Use one of the following methods:

```
# auditconfig -lsevent | egrep " pf|pf|pf,"
AUE_PFEXEC 116 pf execve(2) with pfexec enabled
```

```
# auditrecord -c pf
List of audit events assigned to pf class
```

If events are not assigned to the class, assign the appropriate events to this class.
4 If the previous steps did not indicate a problem, review your email and the log files.

a. **Read the email sent to the audit_warn alias.**
   The audit_warn script sends alert messages to the audit_warn email alias. In the absence of a correctly configured alias, the messages are sent to the root account.

b. **Review the log files for the audit service.**
   The output from the svc -s auditd command lists the full path to the audit logs that the audit service produces. For an example, see the listing in Step 1.

c. **Review the system log files.**
   The audit_warn script writes daemon.alert messages to the /var/log/syslog file.
   The /var/adm/messages file might contain information.

5 After you locate and fix the problems, enable or restart the audit service.
   
   # audit -s

### How to Lessen the Volume of Audit Records That Are Produced

After you have determined which events must be audited at your site, use the following suggestions to create audit files with just the information that you require.

**Before You Begin**

To preselect audit classes and set audit policy, you must be assigned the Audit Configuration rights profile. To modify a system file, you must be assigned the solaris.admin.edit/path-to-system-file authorization. By default, the root role has this authorization. To assign audit flags to users, roles, and rights profiles, you must assume the root role.

1 **Use the default audit policy.**

Specifically, avoid adding events and audit tokens to the audit trail. The following policies grow the size of the audit trail.

- **arge policy** – Adds environment variables to execv audit events. While auditing execv events can be costly, adding variables to the audit record is not costly.

- **argv policy** – Adds command parameters to execv audit events. While auditing execv events can be costly, adding command parameters to the audit record is not costly.

- **public policy** – If file events are being audited, adds an event to the audit trail every time an auditable event happens to a public object. File classes include fa, fc, fd, fm, fr, fw, and cl. For the definition of a public file, see “Audit Terminology and Concepts” on page 526.

- **path policy** – Adds a path token to audit events that include an optional path token.
- **group policy** – Adds a group token to audit events that include an optional `newgroups` token.
- **seq policy** – Adds a sequence token to every audit event.
- **trail policy** – Adds a trailer token to every audit event.
- **windata_down policy** – On a system that is configured with Trusted Extensions, adds events when information in a labeled window is downgraded.
- **windata_up policy** – On a system that is configured with Trusted Extensions, adds events when information in a labeled window is upgraded.
- **zonename policy** – Adds the zone name to every audit event. If the global zone is the only configured zone, adds the string `zone, global` to every audit event.

The following audit record shows the use of the `ls` command. The `ex` class is being audited and the default policy is in use:

```
header,129,2,AUE_EXECVE,,mach1,2010-10-14 11:39:22.480 -07:00
path,/usr/bin/ls
attribute,100555,root,bin,21,320271,18446744073709551615
subject,jdoe,root,root,root,root,2404,50036632,82 0 mach1
return,success,0
```

The following is the same record when all policies are turned on:

```
header,1578,2,AUE_EXECVE,,mach1,2010-10-14 11:45:46.658 -07:00
path,/usr/bin/ls
attribute,100555,root,bin,21,320271,18446744073709551615
exec_args,2,ls,/etc/security
exec_env,49,MANPATH=/usr/share/man,USER=jdoe,GDM_KEYBOARD_LAYOUT=us,EDITOR=gedit,
LANG=en_US.UTF-8,GDM_LANG=en_US.UTF-8,PS1=#,GDMSESSION=gnome,SESSIONTYPE=1,SHLVL=2,
HOME=/home/jdoe,LOGNAME=jdoe,FILENAME_ENCODING=@locale,UTF-8,PRINTER=example-dbl,
path,/lib/ld.so.1
attribute,100755,root,bin,21,393073,18446744073709551615
subject,jdoe,root,root,root,root,2424,50036632,82 0 mach1
group,root,other,bin,syn,adm,uucp,mail,ttv,lp,uucp,daemon
return,success,0
zone,global
sequence,197
trailer,1578
```

2 **Use the audit_syslog plugin to send some audit events to syslog.**

And do not send those audit events to the `audit_binfile` or `audit_remote` plugin. This strategy works only if you are not required to keep binary records of the audit events that you send to the syslog logs.

3 **Set fewer system-wide audit flags and audit individual users.**

Reduce the amount of auditing for all users by reducing the number of audit classes that are audited system-wide.
Use the audit_flags keyword to the roleadd, rolemod, useradd, and usermod commands to audit events for specific users and roles. For examples, see Example 28–21 and the usermod(1M) man page.

Use the always_audit and never_audit properties of the profiles command to audit events for specific rights profiles. For information, see the profiles(1) man page.

**Note** – Like other security attributes, audit flags are affected by search order. For more information, see “Order of Search for Assigned Security Attributes” on page 197.

4 **Create your own customized audit class.**

You can create audit classes at your site. Into these classes, put only those audit events that you need to monitor. For the procedure, see “How to Add an Audit Class” on page 563.

**Note** – For information about the effects of modifying an audit configuration file, see “Audit Configuration Files and Packaging” on page 615.

### How to Audit All Commands by Users

As part of site security policy, some sites require audit records of all commands that are run by the root account and administrative roles. Some sites can require audit records of all commands by all users. Additionally, sites can require that the command arguments and environment be recorded.

**Before You Begin**

To preselect audit classes and set audit policy, you must become an administrator who is assigned the Audit Configuration rights profile. To assign audit flags to users, roles, and rights profiles, you must assume the root role.

1 **Audit the lo and ex classes.**

The ex class audits all calls to the exec() and execl() functions.

The lo class audits logins, logouts, and screen locks. The following output lists all the events in the ex and lo classes.

```bash
% auditconfig -lsevent | grep "^lo $"
AUE_login 6152 lo login - local
AUE_logout 6153 lo logout
AUE_telnet 6154 lo login - telnet
AUE_rlogin 6155 lo login - rlogin
AUE_rshd 6158 lo rsh access
AUE_su 6159 lo su
AUE_rexecd 6162 lo rexecd
AUE_passwd 6163 lo passwd
AUE_rexd 6164 lo rexd
AUE_ftpd 6165 lo ftp access
```
To audit these classes for administrative roles, modify the roles' security attributes.

In the following example, root is a role. The site has created three roles, sysadm, auditadm, and netadm. All roles are audited for the success and failure of events in the ex and lo classes.

```bash
# rolemod -K audit_flags=lo,ex:no root
# rolemod -K audit_flags=lo,ex:no sysadm
# rolemod -K audit_flags=lo,ex:no auditadm
# rolemod -K audit_flags=lo,ex:no netadm
```

To audit these classes for all users, set the system-wide flags.

```bash
# auditconfig -setflags lo,ex
```

The output appears similar to the following:

```
header,129,2,AUE_EXECVE,,mach1,2010-10-14 12:17:12.616 -07:00
path,/usr/bin/ls
attribute,100555,root,bin,21,320271,18446744073709551615
subject,jdoe,root,root,root,root,2486,50036632,820 mach1
return,success,0
```

2. To record the arguments to commands, add the argv policy.

```bash
# auditconfig -setpolicy +argv
```

The exec_args token records the command arguments:

```
header,151,2,AUE_EXECVE,,mach1,2010-10-14 12:26:17.373 -07:00
path,/usr/bin/ls
attribute,100555,root,bin,21,320271,18446744073709551615
exec_args,2,ls, /etc/security
subject,jdoe,root,root,root,root,2494,50036632,820 mach1
return,success,0
```

3. To record the environment in which the command is run, add the arge policy.

```bash
# auditconfig -setpolicy +arge
```
The exec_env token records the command environment:

header,1460,2,AUE_EXECVE,,mach1,2010-10-14 12:29:39.679 -07:00
path,/usr/bin/ls
attribute,100555,root,bin,21,320271,1844674073709551615
exec_args,2,ls,/etc/security
exect_env,49,MANPATH=/usr/share/man,USER=jdoe,GDM_KEYBOARD_LAYOUT=us,EDITOR=gedit,
LANG=en_US.UTF-8,GDM_LANG=en_US.UTF-8,PS1=#,GDMSESSION=gnome,SESSIONTYPE=1,SHLVL=2,
HOME=/home/jdoe,LOGNAME=jdoe,G_FILENAME_ENCODING=@locale,UTF-8,
PRINTER=example-dbl,...,_=/usr/bin/ls
subject,jdoe,root,root,root,root,2502,50036632,820,mach1
return,success,0

▼ How to Find Audit Records of Changes to Specific Files

If your goal is to log file writes against a limited number of files, such as /etc/passwd and the
files in the /etc/default directory, you can use the audit reduce command to locate the files.

Before You Begin

The root role can perform every task in this procedure.

If administrative rights are distributed in your organization, consider the following:

■ An administrator with the Audit Configuration rights profile can run the audit config
command.
■ An administrator with the Audit Review rights profile can run the audit reduce command.
■ Only the root role can assign audit flags.

For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1 Audit the fw class.

Adding the class to the audit flags of a user or role generates fewer records than adding the class
to the system-wide audit preselection mask. Perform one of the following steps:

■ Add the fw class to specific roles.

# rolemod -K audit_flags=fw:no root
# rolemod -K audit_flags=fw:no sysadm
# rolemod -K audit_flags=fw:no auditadm
# rolemod -K audit_flags=fw:no netadm

■ Add the fw class to the system-wide flags.

# auditconfig -getflags
active user default audit flags = lo(0x1000,0x1000)
configured user default audit flags = lo(0x1000,0x1000)
# auditconfig -setflags lo,fw
user default audit flags = lo,fw(0x1002,0x1002)
2 Or, audit successful file-writes.

Auditing successes generates fewer records than auditing failures and successes. Perform one of the following steps:

- Add the +fw flag to specific roles.
  
  ```
  # rolemod -K audit_flags=+fw:no root
  # rolemod -K audit_flags=+fw:no sysadm
  # rolemod -K audit_flags=+fw:no auditadm
  # rolemod -K audit_flags=+fw:no netadm
  ```

- Add the +fw flag to the system-wide flags.
  
  ```
  # auditconfig -getflags
  active user default audit flags = lo(0x1000,0x1000)
  configured user default audit flags = lo(0x1000,0x1000)
  # auditconfig -setflags lo,+fw
  user default audit flags = lo,+fw(0x1002,0x1000)
  ```

- If the system-wide flags are auditing for success and for failure, audit successes only for specific users and roles.
  
  ```
  # auditconfig -getflags
  active user default audit flags = lo,fw(0x1002,0x1002)
  configured user default audit flags = lo,fw(0x1002,0x1002)
  # rolemod -K audit_flags=-:fw:no root
  # rolemod -K audit_flags=-:fw:no sysadm
  # rolemod -K audit_flags=-:fw:no auditadm
  # rolemod -K audit_flags=-:fw:no netadm
  ```

  The system-wide flags are still unchanged, but the preselection mask for these four roles is changed.
  
  ```
  # auditconfig -getflags
  active user default audit flags = lo,fw(0x1002,0x1000)
  configured user default audit flags = lo,fw(0x1002,0x1000)
  ```

3 To find the audit records for specific files, use the auditreduce command.

  ```
  # auditreduce -o file=/etc/passwd,/etc/default -O filechg
  ```

The auditreduce command searches the audit trail for all instances of the file argument. The command creates a binary file with the suffix filechg which contains all records that include the pathnames of the files of interest. See the auditreduce(1M) man page for the syntax of the -o file=pathname option.

4 To read the filechg file, use the praudit command.

  ```
  # praudit *filechg
  ```
How to Update the Preselection Mask of Logged In Users

You want the users who are already logged in to be audited for changes to the system-wide audit preselection mask.

Before You Begin

You must become an administrator who is assigned the Audit Configuration rights profile. To terminate user sessions, you must become an administrator who is assigned the Process Management rights profile. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

Update the preselection mask of users who are already logged in.

You have two options. You can terminate the existing sessions or use the auditconfig command to update the preselection masks.

- Terminate the users’ existing sessions.
  Users can log out and log back in. Or, you in a role that is assigned the Process Management rights profile can manually terminate (kill) active sessions. The new sessions will inherit the new preselection mask. However, terminating users could be impractical.

- Dynamically change each logged-in user’s preselection mask.
  In this example, assume that you changed the system-wide audit preselection mask from lo to lo,ex.

```
# auditconfig -setflags lo,ex
```

a. List the regular users who are logged in and their process IDs.

```
# who -a
jdoe - vt/2 Jan 25 07:56 4:10 1597 (:0)
jdoe + pts/1 Jan 25 10:10 . 1706 (:0.0)
...
jdoe + pts/2 Jan 25 11:36 3:41 1706 (:0.0)
```

b. For later comparison, display each user’s preselection mask.

```
# auditconfig -getpinfo 1706
audit id = jdoe(1234)
process preselection mask = lo(0x1000,0x1000)
terminal id (maj,min,host) = 9426,65559,mach1(192.168.123.234)
audit session id = 103203403
```

c. Modify the appropriate preselection mask by running one or more of the following commands:

```
# auditconfig -setpmask 1706 lo,ex /* for this process */
# auditconfig -setumask jdoe lo,ex /* for this user */
# auditconfig -setsmask 103203403 lo,ex /* for this session */
```
d. **Verify that the preselection mask for the user has changed.**

   For example, check a process that existed before you changed the mask.

   ```
   # auditconfig -getpinfo 1706
   audit id = jdoe(1234)
   process preselection mask = ex,lo(0x40001000,0x40001000)
   terminal id (maj,min,host) = 9426,65559,mach1(192.168.123.234)
   audit session id = 10323403
   ```

**How to Prevent the Auditing of Specific Events**

For maintenance purposes, sometimes a site wants to prevent events from being audited.

**Before You Begin**

You must assume the root role. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1. **Change the class of the event to the no class.**

   **Note** – For information about the effects of modifying an audit configuration file, see “Audit Configuration Files and Packaging” on page 615.

   For example, events 26 and 27 belong to the pm class.

   ```
   ## audit_event file
   ...
   25:AUE_VFORK:vfork(2):ps
   26:AUE_SETGROUPS:setgroups(2):pm
   27:AUE_SETGRP:setgrp(2):pm
   28:AUE_SWAPON:swapon(2):no
   ...
   ```

   Change these events to the no class.

   ```
   ## audit_event file
   ...
   25:AUE_VFORK:vfork(2):ps
   26:AUE_SETGROUPS:setgroups(2):no
   27:AUE_SETGRP:setgrp(2):no
   28:AUE_SWAPON:swapon(2):no
   ```

   If the pm class is currently being audited, existing sessions will still audit events 26 and 27. To stop these events from being audited, you must update the users’ preselection masks by following the instructions in “How to Update the Preselection Mask of Logged In Users” on page 604.
Caution – Never comment out events in the audit_event file. This file is used by the praudit command to read binary audit files. Archived audit files might contain events that are listed in the file.

2 Refresh the kernel events.
   
   # auditconfig -conf
   Configured 283 kernel events.

▼ How to Limit the Size of Binary Audit Files

Binary audit files grow without limit. For ease of archiving and searching, you might want to limit the size. You can also create smaller binary files from the original file.

Before You Begin
You must become an administrator who is assigned the Audit Configuration rights profile to set the p_fsize attribute. You must become an administrator who is assigned the Audit Review rights profile to use the auditreduce command. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

1 Use the p_fsize attribute to limit the size of individual binary audit files.
   For a description of the p_fsize attribute, see the OBJECT ATTRIBUTES section of the audit_binfile(5) man page.

   For an example, see Example 28–14.

2 Use the auditreduce command to select records and write those records to a smaller file for further analysis.
   The auditreduce - lowercase options find specific records.

   The auditreduce - Uppercase options write your selections to a file. For more information, see the auditreduce(1M) man page. See also “Managing Audit Records on Local Systems (Tasks)” on page 585.

▼ How to Compress Audit Files on a Dedicated File System

Audit files can grow large. You can set an upper limit to the size of a file, as shown in Example 28–14. In this procedure, you use compression to reduce the size.

Before You Begin
You must become an administrator who is assigned the ZFS File System Management and ZFS Storage Management rights profiles. The latter profile enables you to create storage pools. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.
Dedicate a ZFS file system for audit files.
For the procedure, see “How to Create ZFS File Systems for Audit Files” on page 566.

Compress the ZFS storage pool by using one of the following options.
With both options, the audit file system is compressed. After the audit service is refreshed, the compression ratio is displayed.

To set compression, use the `zfs set compression=on dataset` command. In the following examples, the ZFS pool `auditp/auditf` is the dataset.

- Use the default compression algorithm.

```
# zfs set compression=on auditp/auditf
# audit -s
# zfs get compressratio auditp/auditf
NAME    PROPERTY VALUE SOURCE
auditp/auditf  compressratio 4.54x -
```

- Use a higher compression algorithm.

```
# zfs set compression=gzip-9 auditp/auditf
# zfs get compression auditp/auditf
NAME    PROPERTY VALUE SOURCE
auditp/auditf  compression  gzip-9  local
# audit -s
# zfs get compressratio auditp/auditf
NAME    PROPERTY VALUE SOURCE
auditp/auditf  compressratio 16.89x -
```

The `gzip-9` compression algorithm results in files that occupy one-third less space than the default compression algorithm, `lzjb`. For more information, see Chapter 5, “Managing Oracle Solaris ZFS File Systems,” in Oracle Solaris 11.1 Administration: ZFS File Systems.

How to Audit Logins From Other Operating Systems

The Oracle Solaris OS can audit all logins, independent of source.

**Before You Begin**
You must become an administrator who is assigned the Audit Configuration rights profile. For more information, see “How to Use Your Assigned Administrative Rights” on page 157.

Audit the `lo` class for attributable events and non-attributable events.
This class audits logins, logouts, and screen locks. These classes are audited by default.

```
# auditconfig -getflags
active user default audit flags = lo(0x1000,0x1000)
configured user default audit flags = lo(0x1000,0x1000)
# auditconfig -getnaflags
active non-attributable audit flags = lo(0x1000,0x1000)
configured non-attributable audit flags = lo(0x1000,0x1000)
```
If the values have been changed, add the lo flag.

```
# auditconfig -getflags
active user default audit flags = as,st(0x20800,0x20800)
configured user default audit flags = as,st(0x20800,0x20800)
# auditconfig -setflags lo,as
user default audit flags = as,lo,st(0x21800,0x21800)
# auditconfig -getnaflags
active non-attributable audit flags = na(0x400,0x400)
configured non-attributable audit flags = na(0x400,0x400)
# auditconfig -setnaflags lo,na
non-attributable audit flags = lo,na(0x1400,0x1400)
```

**Note** – To audit ssh logins, your system must be running the ssh daemon from Oracle Solaris. This daemon is modified for the audit service on an Oracle Solaris system. For more information, see “Secure Shell and the OpenSSH Project” on page 283.

### How to Audit FTP and SFTP File Transfers

The FTP service creates logs of its file transfers. The SFTP service, which runs under the ssh protocol, can be audited by preselecting the ft audit class. Logins to both services can be audited.

**Before You Begin**

You must become an administrator who is assigned the Audit Configuration rights profile. For more information, see "How to Use Your Assigned Administrative Rights" on page 157.

1. **To log commands and file transfers of the FTP service, see the proftpd(8) man page.**
   For the available logging options, read ProFTPD Logging (http://www.proftpd.org/docs/howto/Logging.html).

2. **To log sftp access and file transfers, audit the ft class.**
   The ft class includes the following SFTP transactions:

   ```
   % auditrecord -c ft
   file transfer: chmod ...
   file transfer: chown ...
   file transfer: get ...
   file transfer: mkdir ...
   file transfer: put ...
   file transfer: remove ...
   file transfer: rename ...
   file transfer: rmdir ...
   file transfer: session start ...
   file transfer: session end ...
   file transfer: symlink ...
   file transfer: utimes ...
   ```
3 To record access to the FTP server, audit the \lo class.

As the following output indicates, logging in to and out of the proftpd daemon generates audit records.

```
% auditrecord -c lo | more
...
FTP server login
  program  proftpd  See in.ftpd(1M)
  event ID  6165  AUE_ftpd
  class    lo
    header
    subject
      [text]  error message
    return

FTP server logout
  program  proftpd  See in.ftpd(1M)
  event ID  6171  AUE_ftpd_logout
  class    lo
    header
    subject
    return
...
```
This chapter describes the important components of auditing, and covers the following topics:

- “Audit Service” on page 611
- “Audit Service Man Pages” on page 612
- “Rights Profiles for Administering Auditing” on page 614
- “Auditing and Oracle Solaris Zones” on page 614
- “Audit Configuration Files and Packaging” on page 615
- “Audit Classes” on page 615
- “Audit Plugins” on page 616
- “Audit Remote Server” on page 617
- “Audit Policy” on page 617
- “Process Audit Characteristics” on page 619
- “Audit Trail” on page 620
- “Conventions for Binary Audit File Names” on page 620
- “Audit Record Structure” on page 620
- “Audit Token Formats” on page 622

For an overview of auditing, see Chapter 26, “Auditing (Overview).” For planning suggestions, see Chapter 27, “Planning for Auditing.” For procedures to configure auditing at your site, see Chapter 28, “Managing Auditing (Tasks).”

**Audit Service**

The audit service, auditd, is enabled by default. To enable, refresh, or disable the service, see “Enabling and Disabling the Audit Service (Tasks)” on page 582.

Without customer configuration, the following defaults are in place:

- All login events are audited.
  - Both successful and unsuccessful login attempts are audited.
- All users are audited for login and logout events, including role assumption and screenlock.
The audit_binfile plugin is active. The /var/audit directory stores audit records, the size of an audit file is not limited, and the queue size is 100 records.

The cnt policy is set.
When audit records fill the available disk space, the system tracks the number of dropped audit records. A warning is issued when one percent of available disk space remains.

The following audit queue controls are set:
- Maximum number of records in the audit queue before generating the records locks - 100
- Minimum number of records in the audit queue before blocked auditing processes unblock - 10
- Buffer size for the audit queue - 8192 bytes
- Interval between writing audit records to the audit trail - 20 seconds

To display the defaults, see “How to Display Audit Service Defaults” on page 552.

The audit service enables you to set temporary, or active, values. These values can differ from configured, or property, values.
- Temporary values are not restored when you refresh or restart the audit service.
  Audit policy and audit queue controls accept temporary values. Audit flags do not have a temporary value.
- Configured values are stored as property values of the service, so they are restored when you refresh or restart the audit service.

Rights profiles control who can administer the audit service. For more information, see “Rights Profiles for Administering Auditing” on page 614.

By default, all zones are audited identically. See “Auditing and Oracle Solaris Zones” on page 614.

### Audit Service Man Pages

The following table summarizes the major administrative man pages for the audit service.
<table>
<thead>
<tr>
<th>Man Page</th>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>audit(1M)</td>
<td>Command that controls the actions of the audit service</td>
</tr>
<tr>
<td></td>
<td><code>audit -n</code> starts a new audit file for the <code>audit_binfile</code> plugin.</td>
</tr>
<tr>
<td></td>
<td><code>audit -s</code> enables and refreshes auditing.</td>
</tr>
<tr>
<td></td>
<td><code>audit -t</code> disables auditing.</td>
</tr>
<tr>
<td></td>
<td><code>audit -v</code> verifies that at least one plugin is active.</td>
</tr>
<tr>
<td>audit_binfile(5)</td>
<td>Default audit plugin, which sends audit records to a binary file. See also “Audit Plugins” on page 616.</td>
</tr>
<tr>
<td>audit_remote(5)</td>
<td>Audit plugin that sends audit records to a remote receiver.</td>
</tr>
<tr>
<td>audit_syslog(5)</td>
<td>Audit plugin that sends text summaries of audit records to the <code>syslog</code> utility.</td>
</tr>
<tr>
<td>audit_class(4)</td>
<td>File that contains the definitions of audit classes. The eight high-order bits are available for customers to create new audit classes. For the effect of modifying this file on system upgrade, see “How to Add an Audit Class” on page 563.</td>
</tr>
<tr>
<td>audit_event(4)</td>
<td>File that contains the definitions of audit events and maps the events to audit classes. The mapping can be modified. For the effect of modifying this file on system upgrade, see “How to Change an Audit Event’s Class Membership” on page 564.</td>
</tr>
<tr>
<td>audit_flags(5)</td>
<td>Describes the syntax of audit class preselection, the prefixes for selecting only failed events or only successful events, and the prefixes that modify an existing preselection.</td>
</tr>
<tr>
<td>audit.log(4)</td>
<td>Describes the naming of binary audit files, the internal structure of a file, and the structure of every audit token.</td>
</tr>
<tr>
<td>audit_warn(1M)</td>
<td>Script that notifies an email alias when the audit service encounters an unusual condition while writing audit records. You can customize this script for your site to warn of conditions that might require manual intervention. Or, you could specify how to handle those conditions automatically.</td>
</tr>
<tr>
<td>auditconfig(1M)</td>
<td>Command that retrieves and sets audit configuration parameters. Type <code>auditconfig</code> with no options for a list of parameters that can be retrieved and set.</td>
</tr>
<tr>
<td>auditrecord(1M)</td>
<td>Command that displays the definition of audit events in the <code>/etc/security/audit_event</code> file. For sample output, see “How to Display Audit Record Definitions” on page 586.</td>
</tr>
<tr>
<td>auditreduce(1M)</td>
<td>Command that post-selects and merges audit records that are stored in binary format. The command can merge audit records from one or more input audit files. The records remain in binary format.</td>
</tr>
</tbody>
</table>
Rights Profiles for Administering Auditing

Oracle Solaris provides rights profiles for configuring the audit service, for enabling and disabling the service, and for analyzing the audit trail. To edit an audit configuration file requires the privileges of root.

- **Audit Configuration** – Enables an administrator to configure the parameters of the audit service and to run the `auditconfig` command.
- **Audit Control** – Enables an administrator to start, refresh, and disable the audit service and to run the `audit` command to start, refresh, or stop the service.
- **Audit Review** – Enables an administrator to analyze audit records. This rights profile grants authorization to read audit records with the `praudit` and `auditreduce` commands. This administrator can also run the `auditstat` command.
- **System Administrator** – Includes the Audit Review rights profile. An administrator with the System Administrator rights profile can analyze audit records.

To configure roles to handle the audit service, see “Initially Configuring RBAC (Task Map)” on page 160.

Auditing and Oracle Solaris Zones

Non-global zones can be audited exactly as the global zone is audited, or non-global zones can set their own flags, storage, and audit policy.

When all zones are being audited identically, the `audit_class` and `audit_event` files in the global zone provide the class-event mappings for auditing in every zone. The `+zonename` policy option is useful for post-selecting records by zone name.
Zones can also be audited individually. When the policy option, per zone, is set in the global zone, each non-global zone runs its own audit service, handles its own audit queue, and specifies the content and location of its audit records. A non-global zone can also set most audit policy options. It cannot set policy that affects the entire system, so a non-global zone cannot set the allt or perzone policy. For further discussion, see “Auditing on a System With Oracle Solaris Zones” on page 537 and “How to Plan Auditing in Zones” on page 540.


Audit Configuration Files and Packaging

The audit configuration files in Oracle Solaris are marked in the package with the preserve=rename new package attribute. This attribute preserves any modifications you make to the files across updates. For information about the effects of the preserve values, see the pkg(5) man page.

These configuration files are also marked with the overlay=allow package attribute. This attribute enables you to create your own package that contains these files and replace the Oracle Solaris files with files from your package. When you set the overlay attribute to true in your package, the pkg subcommands, such as verify, fix, revert, and so on, will return results on your packages. For more information, see the pkg(1) and pkg(5) man pages.

Audit Classes

Oracle Solaris defines audit classes as convenient containers for large numbers of audit events.

You can reconfigure audit classes and make new audit classes. Audit class names can be up to 8 characters in length. The class description is limited to 72 characters. Numeric and non-alphanumeric characters are allowed. For more information, see the audit_class(4) man page and “How to Add an Audit Class” on page 563.

Caution – The all class can generate large amounts of data and quickly fill disks. Use the all class only if you have extraordinary reasons to audit all activities.

Audit Class Syntax

Events in an audit class can be audited for success, for failure, and for both.

- Without a prefix, a class of events is audited for success and for failure.
- With a plus (+) prefix, a class of events is audited for success only.
With a minus (-) prefix, a class of events is audited for failure only.

With a caret (^) preceding a prefix or an audit flag, a current preselection is modified. For example,

- If \texttt{ot} is preselected for the system, and a user's preselection is \texttt{^ot}, that user is not audited for events in the other class.
- If \texttt{+ot} is preselected for the system, and a user's preselection is \texttt{^+ot}, that user is not audited for successful events in the other class.
- If \texttt{-ot} is preselected for the system, and a user's preselection is \texttt{^-ot}, that user is not audited for failed events in the other class.

To review the syntax of audit class preselection, see the \texttt{audit\_flags(5)} man page.

The audit classes and their prefixes can be specified in the following commands:

- As arguments to the \texttt{auditconfig} command options \texttt{-setflags} and \texttt{-setnaflags}.
- As values for the \texttt{p\_flags} attribute to the \texttt{audit\_syslog} plugin. You specify the attribute as an option to the \texttt{auditconfig -setplugin audit\_syslog active} command.
- As values for the \texttt{-K audit\_flags=always-audit-flags:never-audit-flags} option to the \texttt{useradd}, \texttt{usermod}, \texttt{roleadd}, and \texttt{rolemod} commands.
- As values for the \texttt{-always\_audit} and \texttt{-never\_audit} properties of the \texttt{profiles} command.

### Audit Plugins

Audit plugins specify how to handle the audit records in the audit queue. The audit plugins are specified by name: \texttt{audit\_binfile}, \texttt{audit\_remote}, and \texttt{audit\_syslog}, as arguments to the \texttt{auditconfig -setplugin} command. The plugins can be further specified by the following attributes:

- \texttt{audit\_binfile} plugin
  - Where to send binary data - \texttt{p\_dir} attribute
  - The minimum space remaining on a disk before the administrator is warned - \texttt{p\_minfree} attribute
  - The maximum size of an audit file - \texttt{p\_fsize} attribute

- \texttt{audit\_remote} plugin
  - A remote authenticated audit server to send the binary audit data to - \texttt{p\_hosts} attribute
  - The number of attempts to make to reach a remote authenticated audit server - \texttt{p\_retries} attribute
  - The number of seconds between attempts to reach a remote authenticated audit server - \texttt{p\_timeout} attribute

- \texttt{audit\_syslog} plugin
A selection of text summaries of audit records to be sent to syslog - p_flags attribute

- For all plugins, the maximum number of audit records that are queued for the plugin - qsize attribute

Refer to the audit_binfile(5), audit_remote(5), audit_syslog(5), and auditconfig(1M) man pages.

**Audit Remote Server**

The Audit Remote Server (ARS) receives audit records over a secure link from audited systems and stores the records.

The reception relies on the following being configured:

- A Kerberos realm with specific audit principals and a GSS-API mechanism
- The ARS with at least one configured and active connection group
- At least one audited system in the connection group and a configured and active audit_remote plugin

A connection group is specified in the group property of the ARS. For file management, group can limit the size of an audit file and specify the minimum free space. The primary reason to specify different connection groups is to specify different storage locations on the ARS, as shown in Example 28–19.

For more information about ARS, see the ars(5) man page. For ARS configuration information, see the -setremote options in the auditconfig(1M) man page.

To configure the audited systems, see the audit_remote(5) man page and the -setplugin option in the auditconfig(1M) man page.

**Audit Policy**

Audit policy determines if additional information is added to the audit trail.

The following policies add tokens to audit records: arge, argv, group, path, seq, trail, windata_down, windata_up, and zonename. The windata_down and windata_up policies are used by the Trusted Extensions feature of Oracle Solaris. For more information, see Chapter 22, "Trusted Extensions Auditing (Overview)," in Trusted Extensions Configuration and Administration.

The remaining policies do not add tokens. The public policy limits auditing of public files. The perzone policy establishes separate audit queues for non-global zones. The ahlt and cnt policies determine what happens when audit records cannot be delivered. For details, see "Audit Policies for Asynchronous and Synchronous Events" on page 618.
The effects of the different audit policy options are described in “Understanding Audit Policy” on page 546. For a description of audit policy options, see the -setpolicy option in the auditconfig(1M) man page. For a list of available policy options, run the command auditconfig -lspolicy. For the current policy, run the command auditconfig -getpolicy.

Audit Policies for Asynchronous and Synchronous Events

Together, the ahlt policy and the cnt policy govern what happens when the audit queue is full and cannot accept more events.

Note – The cnt or ahlt policy is not triggered if the queue for at least one plugin can accept audit records.

The cnt and ahlt policies are independent and related. The combinations of the policies have the following effects:

- ahlt +cnt is the default policy that is shipped. This default lets an audited event be processed even if the event cannot be logged.
  The ahlt policy states that if an audit record of an asynchronous event cannot be placed in the kernel audit queue, the system will count the events and continue processing. In the global zone, the as_dropped counter records the count.
  The +cnt policy states that if a synchronous event arrives and the event cannot be placed in the kernel audit queue, the system will count the event and continue processing. The zone’s as_dropped counter records the count.
  The ahlt +cnt configuration is generally used at sites where processing must continue, even if continued processing could result in a loss of audit records. The auditstat drop field shows the number of audit records that are dropped in a zone.
- The +ahlt -cnt policy states that processing halts when an asynchronous event cannot be added to the kernel audit queue.
  The +ahlt policy states that if an audit record of an asynchronous event cannot be placed in the kernel audit queue, all processing is stopped. The system will panic. The asynchronous event will not be in the audit queue and must be recovered from pointers on the call stack.
  The -cnt policy states that if a synchronous event cannot be placed in the kernel audit queue, the thread that is attempting to deliver the event will be blocked. The thread is placed in a sleep queue until audit space becomes available. No count is kept. Programs might appear to hang until audit space becomes available.
The `+ahlt -cnt` configuration is generally used in sites where a record of every audit event takes precedence over system availability. Programs will appear to hang until audit space becomes available. The `auditstat wblk` field shows the number of times that threads were blocked.

However, if an asynchronous event occurs, the system will panic, leading to an outage. The kernel queue of audit events can be manually recovered from a saved crash dump. The asynchronous event will not be in the audit queue and must be recovered from pointers on the call stack.

- The `+ahlt -cnt` policy states that if an asynchronous event cannot be placed in the kernel audit queue, the event will be counted and processing will continue. When a synchronous event cannot be placed in the kernel audit queue, the thread that is attempting to deliver the event will be blocked. The thread is placed in a sleep queue until audit space becomes available. No count is kept. Programs might appear to hang until audit space becomes available.

  The `+ahlt -cnt` configuration is generally used in sites where the recording of all synchronous audit events takes precedence over some potential loss of asynchronous audit records. The `auditstat wblk` field shows the number of times that threads were blocked.

- The `+ahlt +cnt` policy states that if an asynchronous event cannot be placed in the kernel audit queue, the system will panic. If a synchronous event cannot be placed in the kernel audit queue, the system will count the event and continue processing.

### Process Audit Characteristics

The following audit characteristics are set at initial login:

- **Process preselection mask** – A combination of the system-wide audit mask and the user-specific audit mask, if a user audit mask has been specified. When a user logs in, the login process combines the preselected classes to establish the process preselection mask for the user’s processes. The process preselection mask specifies the events that generate audit records.

  The following algorithm describes how the system obtains the user’s process preselection mask:

  \[
  \text{(system-wide default flags} + \text{always-audit-classes}) - \text{never-audit-classes}
  \]

  Add the system-wide audit classes from the results of the `auditconfig -getflags` command to the classes from the `always-audit-classes` value for the user’s `always_audit` keyword. Then, from the total subtract the classes from the user’s `never-audit-classes`. See also the `audit_flags(5)` man page.
Audit Trail

The audit trail contains binary audit files. The trail is created by the audit_binfile plugin. The audit service collects the records in the audit queue and sends them to the plugin, which writes them to disk.

Conventions for Binary Audit File Names

The audit_binfile plugin creates binary audit files. Each binary audit file is a self-contained collection of records. The file’s name identifies the time span during which the records were generated and the system that generated them. The time stamps that indicate the time span are specified in Coordinated Universal Time (UTC) to ensure that they sort in proper order, even across time zones.

For more information, see the audit.log(4) man page. For examples of open and closed audit file names, see “How to Clean Up a not_terminated Audit File” on page 593.

Audit Record Structure

An audit record is a sequence of audit tokens. Each audit token contains event information such as user ID, time, and date. A header token begins an audit record, and an optional trailer token concludes the record. Other audit tokens contain information relevant to the audit event. The following figure shows a typical kernel audit record and a typical user-level audit record.
Audit Record Analysis

Audit record analysis involves post-selecting records from the audit trail. You can use one of two approaches to parsing the binary data that was collected.

- You can use the `praudit` command. Options to the command provide different text output. For example, the `praudit -x` command provides XML for input into scripts and browsers. `praudit` output does not include fields whose sole purpose is to help to parse the binary data. Note that the order and format of `praudit` output is not guaranteed between Oracle Solaris releases.
  
  For examples of `praudit` output, see “How to View the Contents of Binary Audit Files” on page 591.
  
  For examples of `praudit` output for each audit token, see the individual tokens in "Audit Token Formats" on page 622.

- You can write a program to parse the binary data stream. The program must take into account the variants of an audit record. For example, the `ioctl()` system call creates an audit record for “Bad file name”. This record contains different tokens from the `ioctl()` audit record for “Invalid file descriptor”.

  - For a description of the order of binary data in each audit token, see the `audit.log(4)` man page.

  - For manifest values, see the `/usr/include/bsm/audit.h` file.

  - To view the order of tokens in an audit record, use the `auditrecord` command. Output from the `auditrecord` command includes the different tokens for different manifest values. Square brackets ([ ]) indicate that an audit token is optional. For more information, see the `auditrecord(1M)` man page.
Audit Token Formats

Each audit token has a token type identifier, which is followed by data that is specific to the token. The following table shows the token names with a brief description of each token. Obsolete tokens are maintained for compatibility with previous Solaris releases.

<table>
<thead>
<tr>
<th>Token Name</th>
<th>Description</th>
<th>For More Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>acl</td>
<td>Access Control Entry (ACE) and Access Control List (ACL) information</td>
<td>“acl Token” on page 623</td>
</tr>
<tr>
<td>arbitrary</td>
<td>Data with format and type information</td>
<td>See the audit.log(4) man page.</td>
</tr>
<tr>
<td>argument</td>
<td>System call argument value</td>
<td>“argument Token” on page 624</td>
</tr>
<tr>
<td>attribute</td>
<td>File vnode information</td>
<td>“attribute Token” on page 624</td>
</tr>
<tr>
<td>cmd</td>
<td>Command arguments and environment variables</td>
<td>“cmd Token” on page 624</td>
</tr>
<tr>
<td>exec_args</td>
<td>Exec system call arguments</td>
<td>“exec_args Token” on page 625</td>
</tr>
<tr>
<td>exec_env</td>
<td>Exec system call environment variables</td>
<td>“exec_env Token” on page 625</td>
</tr>
<tr>
<td>exit</td>
<td>Program exit information</td>
<td>See the audit.log(4) man page.</td>
</tr>
<tr>
<td>file</td>
<td>Audit file information</td>
<td>“file Token” on page 625</td>
</tr>
<tr>
<td>fmri</td>
<td>Framework Management Resource Indicator</td>
<td>“fmri Token” on page 625</td>
</tr>
<tr>
<td>group</td>
<td>Process groups information</td>
<td>“group Token” on page 626</td>
</tr>
<tr>
<td>header</td>
<td>Indicates start of audit record</td>
<td>“header Token” on page 626</td>
</tr>
<tr>
<td>ip</td>
<td>IP header information</td>
<td>See the audit.log(4) man page.</td>
</tr>
<tr>
<td>ip address</td>
<td>Internet address</td>
<td>“ip address Token” on page 626</td>
</tr>
<tr>
<td>ip port</td>
<td>Internet port address</td>
<td>“ip port Token” on page 627</td>
</tr>
<tr>
<td>ipc</td>
<td>System V IPC information</td>
<td>“ipc Token” on page 627</td>
</tr>
<tr>
<td>IPC_perm</td>
<td>System V IPC object access information</td>
<td>“IPC_perm Token” on page 628</td>
</tr>
<tr>
<td>opaque</td>
<td>Unstructured data (unspecified format)</td>
<td>See the audit.log(4) man page.</td>
</tr>
<tr>
<td>path</td>
<td>Path information</td>
<td>“path Token” on page 628</td>
</tr>
<tr>
<td>path_attr</td>
<td>Access path information</td>
<td>“path_attr Token” on page 628</td>
</tr>
<tr>
<td>privilege</td>
<td>Privilege set information</td>
<td>“privilege Token” on page 628</td>
</tr>
<tr>
<td>process</td>
<td>Process information</td>
<td>“process Token” on page 629</td>
</tr>
<tr>
<td>Token Name</td>
<td>Description</td>
<td>For More Information</td>
</tr>
<tr>
<td>---------------</td>
<td>--------------------------------------------------</td>
<td>----------------------------------------</td>
</tr>
<tr>
<td>return</td>
<td>Status of system call</td>
<td>“return Token” on page 629</td>
</tr>
<tr>
<td>sequence</td>
<td>Sequence number</td>
<td>“sequence Token” on page 629</td>
</tr>
<tr>
<td>socket</td>
<td>Socket type and addresses</td>
<td>“socket Token” on page 629</td>
</tr>
<tr>
<td>subject</td>
<td>Subject information (same format as process)</td>
<td>“subject Token” on page 630</td>
</tr>
<tr>
<td>text</td>
<td>ASCII string</td>
<td>“text Token” on page 630</td>
</tr>
<tr>
<td>trailer</td>
<td>Indicates end of audit record</td>
<td>“trailer Token” on page 630</td>
</tr>
<tr>
<td>use of authorization</td>
<td>Use of authorization</td>
<td>“use of authorization Token” on page 631</td>
</tr>
<tr>
<td>use of privilege</td>
<td>Use of privilege</td>
<td>“use of privilege Token” on page 631</td>
</tr>
<tr>
<td>user</td>
<td>User ID and user name</td>
<td>“user Token” on page 631</td>
</tr>
<tr>
<td>xclient</td>
<td>X client identification</td>
<td>“xclient Token” on page 631</td>
</tr>
<tr>
<td>zonename</td>
<td>Name of zone</td>
<td>“zonename Token” on page 631</td>
</tr>
<tr>
<td>Trusted Extensions tokens</td>
<td>label and X Window System information</td>
<td>See “Trusted Extensions Audit Reference” in Trusted Extensions Configuration and Administration.</td>
</tr>
</tbody>
</table>

The following tokens are obsolete:

- liaison
- host
- tid

For information about obsolete tokens, see the reference material for the release that included the token.

An audit record always begins with a header token. The header token indicates where the audit record begins in the audit trail. In the case of attributable events, the subject and the process tokens refer to the values of the process that caused the event. In the case of non-attributable events, the process token refers to the system.

**acl Token**

The acl token has two forms to record information about Access Control Entries (ACEs) for a ZFS file system and Access Control Lists (ACLs) for a UFS file system.

When the acl token is recorded for a UFS file system, the `praudit -x` command shows the fields as follows:
Whenthe acl token is recorded for a ZFS dataset, the praudit -x command shows the fields as follows:

<acl type="l" value="root" mode="6"/>

The argument token contains information about the arguments to a system call: the argument number of the system call, the argument value, and an optional description. This token allows a 32-bit integer system-call argument in an audit record.

The praudit -x command shows the fields of the argument token as follows:

<argument arg-num="2" value="0x5401" desc="cmd"/>

The attribute token contains information from the file vnode.

The attribute token usually accompanies a path token. The attribute token is produced during path searches. If a path-search error occurs, there is no vnode available to obtain the necessary file information. Therefore, the attribute token is not included as part of the audit record. The praudit -x command shows the fields of the attribute token as follows:

<attribute mode="20620" uid="root" gid="tty" fsid="0" nodeid="9267" device="108233"/>

The cmd token records the list of arguments and the list of environment variables that are associated with a command.

The praudit -x command shows the fields of the cmd token. The following is a truncated cmd token. The line is wrapped for display purposes.

<cmd><arge>WINDOWID=6823679</arge>
<arge>COLORTERM=gnome-terminal</arge>
<arge>...LANG=C</arge>...<arge>HOST=machine1</arge>
<arge>LPDEST=printer1</arge>...<cmd>
exec_args Token

The exec_args token records the arguments to an exec() system call.

The praudit -x command shows the fields of the exec_args token as follows:

```
<exec_args><arg>/usr/bin/sh</arg><arg>/usr/bin/hostname</arg></exec_args>
```

**Note** – The exec_args token is output only when the argv audit policy option is active.

exec_env Token

The exec_env token records the current environment variables to an exec() system call.

The praudit -x command shows the fields of the exec_env token. The line is wrapped for display purposes.

```
<exec_env><env>_/=/usr/bin/hostname</env>
<env>LANG=C</env><env>PATH=/usr/bin:/usr/ucb</env>
<env>LOGNAME=jdoe</env><env>USER=jdoe</env>
<env>DISPLAY=:0</env><env>SHELL=/bin/csh</env>
<env>HOME=/home/jdoe</env><env>PWD=/home/jdoe</env><env>TZ=US/Pacific</env>
</exec_env>
```

**Note** – The exec_env token is output only when the arge audit policy option is active.

file Token

The file token is a special token that marks the beginning of a new audit file and the end of an old audit file as the old file is deactivated. The initial file token identifies the previous file in the audit trail. The final file token identifies the next file in the audit trail. These tokens “link” together successive audit files into one audit trail.

The praudit -x command shows the fields of the file token. The line is wrapped for display purposes.

```
<file iso8601="2009-04-08 14:18:26.200 -07:00">
/var/audit/machine1/files/20090408211826.not_terminated.machine1</file>
```

fmri Token

The fmri token records the use of a fault management resource indicator (FMRI). For more information, see the smf(5) man page.

The praudit -x command shows the content of the fmri token:
group Token

The group token records the group entries from the process's credential.

The praudit -x command shows the fields of the groups token as follows:

<group><gid>staff</gid><gid>other</gid></group>

Note – The group token is output only when the group audit policy option is active.

header Token

The header token is special in that it marks the beginning of an audit record. The header token combines with the trailer token to bracket all the other tokens in the record.

Infrequently, a header token can include one or more event modifiers:

- fe indicates a failed audit event
- fp indicates the failed use of privilege
- na indicates a non-attributable event
- rd indicates that data is read from the object
- sp indicates the successful use of privilege

The praudit command displays the header token as follows:

header,52,2,system booted,na,machine1,2011-10-10 10:10:20.564 -07:00

ip address Token

The ip address token contains an Internet Protocol address (IP address). The IP address can be displayed in IPv4 or IPv6 format. The IPv4 address uses 4 bytes. The IPv6 address uses 1 byte to describe the address type, and 16 bytes to describe the address.
The `praudit -x` command shows the content of the `ip address` token as follows:

```<ip_address>machine1</ip_address>```

**ip port Token**

The `ip port` token contains the TCP or UDP port address.

The `praudit` command displays the `ip port` token as follows:

```ip port,0xf6d6```

**ipc Token**

The `ipc` token contains the System V IPC message handle, semaphore handle, or shared-memory handle that is used by the caller to identify a particular IPC object.

---

**Note** – The IPC object identifiers violate the context-free nature of the audit tokens. No global “name” uniquely identifies IPC objects. Instead, IPC objects are identified by their handles. The handles are valid only during the time that the IPC objects are active. However, the identification of IPC objects should not be a problem. The System V IPC mechanisms are seldom used, and the mechanisms all share the same audit class.

The following table shows the possible values for the IPC object type field. The values are defined in the `/usr/include/bsm/audit.h` file.

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AU_IPC_MSG</td>
<td>1</td>
<td>IPC message object</td>
</tr>
<tr>
<td>AU_IPC_SEM</td>
<td>2</td>
<td>IPC semaphore object</td>
</tr>
<tr>
<td>AU_IPC_SHM</td>
<td>3</td>
<td>IPC shared-memory object</td>
</tr>
</tbody>
</table>

The `praudit -x` command shows the fields of the `ipc` token as follows:

```<IPC ipc-type="shm" ipc-id="15"/>```
**IPC_perm Token**

The **IPC_perm** token contains a copy of the System V IPC access permissions. This token is added to audit records that are generated by IPC shared-memory events, IPC semaphore events, and IPC message events.

The `praudit -x` command shows the fields of the **IPC_perm** token. The line is wrapped for display purposes.

```
<IPC_perm uid="jdoe" gid="staff" creator-uid="jdoe"
 creator-gid="staff" mode="100600" seq="0" key="0x0"/>
```

The values are taken from the **IPC_perm** structure that is associated with the IPC object.

**path Token**

The **path** token contains access path information for an object.

The `praudit -x` command shows the content of the **path** token:

```
<path>/export/home/srv/.xsession-errors</path>
```

**path_attr Token**

The **path_attr** token contains access path information for an object. The access path specifies the sequence of attribute file objects below the **path** token object. Systems calls such as `openat()` access attribute files. For more information about attribute file objects, see the `fsattr(5)` man page.

The `praudit` command displays the **path_attr** token as follows:

```
path_attr,1,attr_file_name
```

**privilege Token**

The **privilege** token records the use of privileges on a process. The **privilege** token is not recorded for privileges in the basic set. If a privilege has been removed from the basic set by administrative action, then the use of that privilege is recorded. For more information about privileges, see “Privileges (Overview)” on page 138.

The `praudit -x` command shows the fields of the **privilege** token.

```
<privilege set-type="Inheritable">ALL</privilege>
```
The `process` token contains information about a user who is associated with a process, such as the recipient of a signal.

The `praudit -x` command shows the fields of the `process` token. The line is wrapped for display purposes.

```xml
<process audit-uid="-2" uid="root" gid="root" ruid="root" rgid="root" pid="567" sid="0" tid="0 0.0.0.0"/>
```

The `return` token contains the return status of the system call (`u_error`) and the process return value (`u_rval1`).

The `return` token is always returned as part of kernel-generated audit records for system calls. In application auditing, this token indicates exit status and other return values.

The `praudit` command displays the return token for a system call as follows:

```
return,failure: Operation now in progress,-1
```

The `praudit -x` command shows the fields of the `return` token as follows:

```xml
<return errval="failure: Operation now in progress" retval="-1"/>
```

The sequence token contains a sequence number. The sequence number is incremented every time an audit record is added to the audit trail. This token is useful for debugging.

The `praudit -x` command shows the content of the sequence token:

```xml
<sequence seq-num="1292"/>
```

**Note** – The sequence token is output only when the `seq` audit policy option is active.

The `socket` token contains information that describes an Internet socket. In some instances, the token includes only the remote port and remote IP address.

The `praudit` command displays this instance of the `socket` token as follows:
The expanded token adds information, including socket type and local port information.

The `praudit -x` command displays this instance of the socket token as follows. The line is wrapped for display purposes.

```
<socket sock_domain="0x0002" sock_type="0x0002" lport="0x83cf"
laddr="example1" fport="0x2383" faddr="server1.Subdomain.Domain.COM"/>
```

**subject Token**

The subject token describes a user who performs or attempts to perform an operation. The format is the same as the process token.

The subject token is always returned as part of kernel-generated audit records for system calls. The `praudit` command displays the subject token as follows:

```
subject,jdoe,root,root,root,root,1631,1421584480,8243 65558 machine1
```

The `praudit -x` command shows the fields of the subject token. The line is wrapped for display purposes.

```
<subject audit-uid="jdoe" uid="root" gid="root" ruid="root"
rgid="root" pid="1631" sid="1421584480" tid="8243 65558 machine1"/>
```

**text Token**

The text token contains a text string.

The `praudit -x` command shows the content of the text token:

```
<text>booting kernel</text>
```

**trailer Token**

The two tokens, header and trailer, are special in that they distinguish the end points of an audit record and bracket all the other tokens. A header token begins an audit record. A trailer token ends an audit record. The trailer token is an optional token. The trailer token is added as the last token of each record only when the trail audit policy option has been set.

When an audit record is generated with trailers turned on, the `auditreduce` command can verify that the trailer correctly points back to the record header. The trailer token supports backward seeks of the audit trail.
The `praudit` command displays the trailer token as follows:

```
trailer,136
```

**use of authorization Token**

The use of authorization token records the use of authorization.

The `praudit` command displays the use of authorization token as follows:

```
use of authorization,solaris.role.delegate
```

**use of privilege Token**

The use of privilege token records the use of privilege.

The `praudit -x` command shows the fields of the use of privilege token as follows:

```
<use_of_privilege result="successful use of priv">proc_setid</use_of_privilege>
```

**user Token**

The user token records the user name and user ID. This token is present if the user name is different from the caller.

The `praudit -x` command shows the fields of the user token as follows:

```
<user uid="123456" username="tester1"/>
```

**xclient Token**

The xclient token contains the number of the client connection to the X server.

The `praudit -x` command shows the content of the xclient token as follows:

```
<X_client>15</X_client>
```

**zonename Token**

The zonename token records the zone in which the audit event occurred. The string “global” indicates audit events that occur in the global zone.

The `praudit -x` command shows the content of the zonename token:
Audit Token Formats

<zone name="graphzone"/>

Glossary

Access Control List (ACL)  An access control list (ACL) provides finer-grained file security than traditional UNIX file protection provides. For example, an ACL enables you to allow group read access to a file, while allowing only one member of that group to write to the file.

admin principal  A user principal with a name of the form *username@admin* (as in *jdoe@admin*). An admin principal can have more privileges (for example, to change policies) than a regular user principal. See also *principal name*, *user principal*.


algorithm  A cryptographic algorithm. This is an established, recursive computational procedure that encrypts or hashes input.

application server  See *network application server*.

asynchronous audit event  Asynchronous events are the minority of system events. These events are not associated with any process, so no process is available to be blocked and later woken up. Initial system boot and PROM enter and exit events are examples of asynchronous events.

audit files  Binary audit logs. Audit files are stored separately in an audit file system.

audit policy  The global and per-user settings that determine which audit events are recorded. The global settings that apply to the audit service typically affect which pieces of optional information are included in the audit trail. Two settings, *cnt* and *ahl t*, affect the operation of the system when the audit queue fills. For example, audit policy might require that a sequence number be part of every audit record.

audit trail  The collection of all audit files from all hosts.

authentication  The process of verifying the claimed identity of a principal.

authenticator  Authenticators are passed by clients when requesting tickets (from a KDC) and services (from a server). They contain information that is generated by using a session key known only by the client and server, that can be verified as of recent origin, thus indicating that the transaction is secure. When used with a ticket, an authenticator can be used to authenticate a user principal. An authenticator includes the principal name of the user, the IP address of the user’s host, and a time stamp. Unlike a ticket, an authenticator can be used only once, usually when access to a service is requested. An authenticator is encrypted by using the session key for that client and that server.
authorization

1. In Kerberos, the process of determining if a principal can use a service, which objects the principal is
   allowed to access, and the type of access that is allowed for each object.

2. In role-based access control (RBAC), a permission that can be assigned to a role or user (or embedded in
   a rights profile) for performing a class of actions that are otherwise prohibited by security policy.

basic set

The set of privileges that are assigned to a user’s process at login. On an unmodified system, each user’s
initial inheritable set equals the basic set at login.

Blowfish

A symmetric block cipher algorithm that takes a variable-length key from 32 bits to 448 bits. Its author,
Bruce Schneier, claims that Blowfish is optimized for applications where the key does not change often.

client

Narrowly, a process that makes use of a network service on behalf of a user; for example, an application
that uses rlogin. In some cases, a server can itself be a client of some other server or service.

More broadly, a host that a) receives a Kerberos credential, and b) makes use of a service that is provided
by a server.

Informally, a principal that makes use of a service.

client principal

(RPCSEC_GSS_API) A client (a user or an application) that uses RPCSEC_GSS-secured network services.
Client principal names are stored in the form of rpc_gss_principal_t structures.

clock skew

The maximum amount of time that the internal system clocks on all hosts that are participating in the
Kerberos authentication system can differ. If the clock skew is exceeded between any of the participating
hosts, requests are rejected. Clock skew can be specified in the krb5.conf file.

confidence

See privacy.

consumer

In the Cryptographic Framework feature of Oracle Solaris, a consumer is a user of the cryptographic
services that come from providers. Consumers can be applications, end users, or kernel operations.
Kerberos, IKE, and IPsec are examples of consumers. For examples of providers, see provider.

credential

An information package that includes a ticket and a matching session key. Used to authenticate the
identity of a principal. See also ticket, session key.

credential cache

A storage space (usually a file) that contains credentials that are received from the KDC.

cryptographic algorithm

See algorithm.

DES

Data Encryption Standard. A symmetric-key encryption method developed in 1975 and standardized by
ANSI in 1981 as ANSI X.3.92. DES uses a 56-bit key.

device allocation

Device protection at the user level. Device allocation enforces the exclusive use of a device by one user at a
time. Device data is purged before device reuse. Authorizations can be used to limit who is permitted to
allocate a device.

device policy

Device protection at the kernel level. Device policy is implemented as two sets of privileges on a device.
One set of privileges controls read access to the device. The second set of privileges controls write access to
the device. See also policy.
Diffie-Hellman protocol

Also known as public key cryptography. An asymmetric cryptographic key agreement protocol that was developed by Diffie and Hellman in 1976. The protocol enables two users to exchange a secret key over an insecure medium without any prior secrets. Diffie-Hellman is used by Kerberos.

digest

See message digest.

DSA

Digital Signature Algorithm. A public key algorithm with a variable key size from 512 to 4096 bits. The U.S. Government standard, DSS, goes up to 1024 bits. DSA relies on SHA1 for input.

effective set

The set of privileges that are currently in effect on a process.

flavor

Historically, security flavor and authentication flavor had the same meaning, as a flavor that indicated a type of authentication (AUTH_UNIX, AUTH_DESC, AUTH_KERB). RPCSEC_GSS is also a security flavor, even though it provides integrity and privacy services in addition to authentication.

forwardable ticket

A ticket that a client can use to request a ticket on a remote host without requiring the client to go through the full authentication process on that host. For example, if the user david obtains a forwardable ticket while on user jennifer's machine, he can log in to his own machine without being required to get a new ticket (and thus authenticate himself again). See also proxiable ticket.

FQDN

Fully qualified domain name. For example, central.example.com (as opposed to simply denver).

GSS-API

The Generic Security Service Application Programming Interface. A network layer that provides support for various modular security services, including the Kerberos service. GSS-API provides for security authentication, integrity, and privacy services. See also authentication, integrity, privacy.

hardening

The modification of the default configuration of the operating system to remove security vulnerabilities that are inherent in the host.

hardware provider

In the Cryptographic Framework feature of Oracle Solaris, a device driver and its hardware accelerator. Hardware providers offload expensive cryptographic operations from the computer system, thus freeing CPU resources for other uses. See also provider.

host

A system that is accessible over a network.

host principal

A particular instance of a service principal in which the principal (signified by the primary name host) is set up to provide a range of network services, such as ftp, rcp, or rlogin. An example of a host principal is host/central.example.com@EXAMPLE.COM. See also server principal.

inheritable set

The set of privileges that a process can inherit across a call to exec.

initial ticket

A ticket that is issued directly (that is, not based on an existing ticket-granting ticket). Some services, such as applications that change passwords, might require tickets to be marked initial so as to assure themselves that the client can demonstrate a knowledge of its secret key. This assurance is important because an initial ticket indicates that the client has recently authenticated itself (instead of relying on a ticket-granting ticket, which might existed for a long time).

instance

The second part of a principal name, an instance qualifies the principal's primary. In the case of a service principal, the instance is required. The instance is the host's fully qualified domain name, as in host/central.example.com. For user principals, an instance is optional. Note, however, that jdoe and jdoe/admin are unique principals. See also primary, principal name, service principal, user principal.
integrity  A security service that, in addition to user authentication, provides for the validity of transmitted data through cryptographic checksumming. See also authentication, privacy.

invalid ticket  A postdated ticket that has not yet become usable. An invalid ticket is rejected by an application server until it becomes validated. To be validated, an invalid ticket must be presented to the KDC by the client in a TGS request, with the VALIDATE flag set, after its start time has passed. See also postdated ticket.

KDC  Key Distribution Center. A machine that has three Kerberos V5 components:
   ■ Principal and key database
   ■ Authentication service
   ■ Ticket-granting service

Each realm has a master KDC and should have one or more slave KDCs.

Kerberos  An authentication service, the protocol that is used by that service, or the code that is used to implement that service.

The Oracle Solaris Kerberos implementation that is closely based on Kerberos V5 implementation.

While technically different, ”Kerberos” and ”Kerberos V5” are often used interchangeably in the Kerberos documentation.

Kerberos (also spelled Cerberus) was a fierce, three-headed mastiff who guarded the gates of Hades in Greek mythology.

Kerberos policy  A set of rules that governs password usage in the Kerberos service. Policies can regulate principals' accesses, or ticket parameters, such as lifetime.

key  1. Generally, one of two main types of keys:
   ■ A symmetric key – An encryption key that is identical to the decryption key. Symmetric keys are used to encrypt files.
   ■ An asymmetric key or public key – A key that is used in public key algorithms, such as Diffie-Hellman or RSA. Public keys include a private key that is known only by one user, a public key that is used by the server or general resource, and a private-public key pair that combines the two. A private key is also called a secret key. The public key is also called a shared key or common key.
   ■ 2. An entry (principal name) in a keytab file. See also keytab file.
   3. In Kerberos, an encryption key, of which there are three types:
      ■ A private key – An encryption key that is shared by a principal and the KDC, and distributed outside the bounds of the system. See also private key.
      ■ A service key – This key serves the same purpose as the private key, but is used by servers and services. See also service key.
      ■ A session key – A temporary encryption key that is used between two principals, with a lifetime limited to the duration of a single login session. See also session key.

keystore  A keystore holds passwords, passphrases, certificates, and other authentication objects for retrieval by applications. A keystore can be specific to a technology, or a location that several applications use.
keytab file
A key table file that contains one or more keys (principals). A host or service uses a keytab file in the much the same way that a user uses a password.

kvno
Key version number. A sequence number that tracks a particular key in order of generation. The highest kvno is the latest and most current key.

least privilege
A security model which gives a specified process only a subset of superuser powers. The least privilege model assigns enough privilege to regular users that they can perform personal administrative tasks, such as mount file systems and change the ownership of files. On the other hand, processes run with just those privileges that they need to complete the task, rather than with the full power of superuser, that is, all privileges. Damage due to programming errors like buffer overflows can be contained to a non-root user, which has no access to critical abilities like reading or writing protected system files or halting the machine.

limit set
The outside limit of what privileges are available to a process and its children.

MAC
1. See message authentication code (MAC).
2. Also called labeling. In government security terminology, MAC is Mandatory Access Control. Labels such as Top Secret and Confidential are examples of MAC. MAC contrasts with DAC, which is Discretionary Access Control. UNIX permissions are an example of DAC.
3. In hardware, the unique system address on a LAN. If the system is on an Ethernet, the MAC is the Ethernet address.

master KDC
The main KDC in each realm, which includes a Kerberos administration server, kadmind, and an authentication and ticket-granting daemon, krb5kdc. Each realm must have at least one master KDC, and can have many duplicate, or slave, KDCs that provide authentication services to clients.

MD5
An iterative cryptographic hash function that is used for message authentication, including digital signatures. The function was developed in 1991 by Rivest. Its use is deprecated.

mechanism
1. A software package that specifies cryptographic techniques to achieve data authentication or confidentiality. Examples: Kerberos V5, Diffie-Hellman public key.
2. In the Cryptographic Framework feature of Oracle Solaris, an implementation of an algorithm for a particular purpose. For example, a DES mechanism that is applied to authentication, such as CKM_DES_MAC, is a separate mechanism from a DES mechanism that is applied to encryption, CKM_DES_CBC_PAD.

message authentication code (MAC)
MAC provides assurance of data integrity and authenticates data origin. MAC does not protect against eavesdropping.

message digest
A message digest is a hash value that is computed from a message. The hash value almost uniquely identifies the message. A digest is useful for verifying the integrity of a file.

minimization
The installation of the minimal operating system that is necessary to run the server. Any software that does not directly relate to the operation of the server is either not installed, or deleted after the installation.

name service scope
The scope in which a role is permitted to operate, that is, an individual host or all hosts that are served by a specified naming service such as NIS LDAP.
### network application server
A server that provides a network application, such as FTP. A realm can contain several network application servers.

### nonattributable audit event
An audit event whose initiator cannot be determined, such as the AUE_BOOT event.

### NTP
Network Time Protocol. Software from the University of Delaware that enables you to manage precise time or network clock synchronization, or both, in a network environment. You can use NTP to maintain clock skew in a Kerberos environment. See also clock skew.

### PAM
Pluggable Authentication Module. A framework that allows for multiple authentication mechanisms to be used without having to recompile the services that use them. PAM enables Kerberos session initialization at login.

### passphrase
A phrase that is used to verify that a private key was created by the passphrase user. A good passphrase is 10-30 characters long, mixes alphabetic and numeric characters, and avoids simple prose and simple names. You are prompted for the passphrase to authenticate use of the private key to encrypt and decrypt communications.

### password policy
The encryption algorithms that can be used to generate passwords. Can also refer to more general issues around passwords, such as how often the passwords must be changed, how many password attempts are permitted, and other security considerations. Security policy requires passwords. Password policy might require passwords to be encrypted with the AES algorithm, and might make further requirements related to password strength.

### permitted set
The set of privileges that are available for use by a process.

### policy
Generally, a plan or course of action that influences or determines decisions and actions. For computer systems, policy typically means security policy. Your site’s security policy is the set of rules that define the sensitivity of the information that is being processed and the measures that are used to protect the information from unauthorized access. For example, security policy might require that systems be audited, that devices be protected with privileges, and that passwords be changed every six weeks.

For the implementation of policy in specific areas of the Oracle Solaris OS, see audit policy, policy in the Cryptographic Framework, device policy, Kerberos policy, password policy, and RBAC policy.

### policy for public key technologies
In the Key Management Framework (KMF), policy is the management of certificate usage. The KMF policy database can put constraints on the use of the keys and certificates that are managed by the KMF library.

### policy in the Cryptographic Framework
In the Cryptographic Framework feature of Oracle Solaris, policy is the disabling of existing cryptographic mechanisms. The mechanisms then cannot be used. Policy in the Cryptographic Framework might prevent the use of a particular mechanism, such as CCM_DES_CBC, from a provider, such as DES.

### postdated ticket
A postdated ticket does not become valid until some specified time after its creation. Such a ticket is useful, for example, for batch jobs that are intended to run late at night, since the ticket, if stolen, cannot be used until the batch job is run. When a postdated ticket is issued, it is issued as invalid and remains that way until a) its start time has passed, and b) the client requests validation by the KDC. A postdated ticket is normally valid until the expiration time of the ticket-granting ticket. However, if the postdated ticket is marked renewable, its lifetime is normally set to be equal to the duration of the full life time of the ticket-granting ticket. See also invalid ticket, renewable ticket.
primary The first part of a principal name. See also instance, principal name, realm.

principal 1. A uniquely named client/user or server/service instance that participates in a network communication. Kerberos transactions involve interactions between principals (service principals and user principals) or between principals and KDCs. In other words, a principal is a unique entity to which Kerberos can assign tickets. See also principal name, service principal, user principal.

2. (RPCSEC_GSS API) See client principal, server principal.

principal name 1. The name of a principal, in the format primary.instance@REALM. See also instance, primary, realm.

2. (RPCSEC_GSS API) See client principal, server principal.

privacy A security service, in which transmitted data is encrypted before being sent. Privacy also includes data integrity and user authentication. See also authentication, integrity, service.

private key A key that is given to each user principal, and known only to the user of the principal and to the KDC. For user principals, the key is based on the user’s password. See also key.

private-key encryption In private-key encryption, the sender and receiver use the same key for encryption. See also public-key encryption.

privilege A discrete right on a process in an Oracle Solaris system. Privileges offer a finer-grained control of processes than does root. Privileges are defined and enforced in the kernel. For a full description of privileges, see the privileges(5) man page.

privilege-aware Programs, scripts, and commands that turn on and off the use of privilege in their code. In a production environment, the privileges that are turned on must be supplied to the process, for example, by requiring users of the program to use a rights profile that adds the privileges to the program. For a full description of privileges, see the privileges(5) man page.

privilege escalation Gaining access to resources that are outside the range of resources that your assigned security attributes, including overrides, permit. The result is that a process can perform unauthorized actions.

privilege model A stricter model of security on a computer system than the superuser model. In the privilege model, processes require privilege to run. Administration of the system can be divided into discrete parts that are based on the privileges that administrators have in their processes. Privileges can be assigned to an administrator’s login process. Or, privileges can be assigned to be in effect for certain commands only.

privilege set A collection of privileges. Every process has four sets of privileges that determine whether a process can use a particular privilege. See limit set, effective set set, permitted set set, and inheritable set set.

Also, the basic set of privileges is the collection of privileges that are assigned to a user’s process at login.

privileged application An application that can override system controls. The application checks for security attributes, such as specific UIDs, GIDs, authorizations, or privileges.

profile shell In RBAC, a shell that enables a role (or user) to run from the command line any privileged applications that are assigned to the role’s rights profiles. The profile shells are pfsh, pfcs, and pfks. They correspond to the Bourne shell (sh), C shell (csh), and Korn shell (ksh), respectively.
In the Cryptographic Framework feature of Oracle Solaris, a cryptographic service that is provided to consumers. PKCS #11 libraries, kernel cryptographic modules, and hardware accelerators are examples of providers. Providers plug in to the Cryptographic Framework, so are also called plugins. For examples of consumers, see consumer.

A ticket that can be used by a service on behalf of a client to perform an operation for the client. Thus, the service is said to act as the client's proxy. With the ticket, the service can take on the identity of the client. The service can use a proxiable ticket to obtain a service ticket to another service, but it cannot obtain a ticket-granting ticket. The difference between a proxiable ticket and a forwardable ticket is that a proxiable ticket is only valid for a single operation. See also forwardable ticket.

An encryption scheme in which each user has two keys, one public key and one private key. In public-key encryption, the sender uses the receiver’s public key to encrypt the message, and the receiver uses a private key to decrypt it. The Kerberos service is a private-key system. See also private-key encryption.

A file that is owned by the root user and readable by the world, such as any file in the /etc directory.

Quality of Protection. A parameter that is used to select the cryptographic algorithms that are used in conjunction with the integrity service or privacy service.

Role-based access control, a feature of the Oracle Solaris. An alternative to the all-or-nothing superuser model. RBAC lets an organization separate superuser’s capabilities and assign them to special user accounts called roles. Roles can be assigned to specific individuals according to their responsibilities.

The security policy that is associated with a command. Currently, solaris is the valid policy. The solaris policy recognizes privileges, authorizations, and setuid security attributes.

1. The logical network that is served by a single Kerberos database and a set of Key Distribution Centers (KDCs).

2. The third part of a principal name. For the principal name j doe/admin@ENG.EXAMPLE.COM, the realm is ENG.EXAMPLE.COM. See also principal name.

A configuration variable or relationship that is defined in the kdc.conf or krb5.conf files.

Because having tickets with very long lives is a security risk, tickets can be designated as renewable. A renewable ticket has two expiration times: a) the time at which the current instance of the ticket expires, and b) maximum lifetime for any ticket. If a client wants to continue to use a ticket, the client renews the ticket before the first expiration occurs. For example, a ticket can be valid for one hour, with all tickets having a maximum lifetime of ten hours. If the client that holds the ticket wants to keep it for more than an hour, the client must renew the ticket. When a ticket reaches the maximum ticket lifetime, it automatically expires and cannot be renewed.

Also referred to as a right or a profile. A collection of overrides used in RBAC that can be assigned to a role or user. A rights profile can consist of authorizations, privileges, commands with security attributes, and other rights profiles.

A special identity for running privileged applications that only assigned users can assume.

A method for obtaining digital signatures and public key cryptosystems. The method was first described in 1978 by its developers, Rivest, Shamir, and Adleman.
<table>
<thead>
<tr>
<th>Term</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>scan engine</td>
<td>A third-party application, residing on an external host, that examines a file for known viruses.</td>
</tr>
<tr>
<td>SEAM</td>
<td>Sun Enterprise Authentication Mechanism. The product name for the initial versions of a system for authenticating users over a network, based on the Kerberos V5 technology that was developed at the Massachusetts Institute of Technology. The product is now called the Kerberos service. SEAM refers to parts the Kerberos service that were not included in various Solaris releases.</td>
</tr>
<tr>
<td>secret key</td>
<td>See private key.</td>
</tr>
<tr>
<td>Secure Shell</td>
<td>A special protocol for secure remote login and other secure network services over an insecure network.</td>
</tr>
<tr>
<td>security attributes</td>
<td>In RBAC, overrides to security policy that enable an administrative command to succeed when the command is run by a user other than superuser. In the superuser model, the setuid and setgid programs are security attributes. When these attributes are applied to a command, the command succeeds no matter who runs the command. In the privilege model, security attributes are privileges. When a privilege is given to a command, the command succeeds. The privilege model is compatible with the superuser model, in that the privilege model also recognizes the setuid and setgid programs as security attributes.</td>
</tr>
<tr>
<td>security flavor</td>
<td>See flavor.</td>
</tr>
<tr>
<td>security mechanism</td>
<td>See mechanism.</td>
</tr>
<tr>
<td>security policy</td>
<td>See policy.</td>
</tr>
<tr>
<td>security service</td>
<td>See service.</td>
</tr>
<tr>
<td>seed</td>
<td>A numeric starter for generating random numbers. When the starter originates from a random source, the seed is called a random seed.</td>
</tr>
<tr>
<td>separation of duty</td>
<td>Part of the notion of least privilege. Separation of duty prevents one user from performing or approving all actions that complete a transaction. For example, in RBAC, you can separate the creation of a login user from the assignment of security overrides. One role creates the user. A separate role can assign security attributes, such as rights profiles, roles, and privileges to existing users.</td>
</tr>
<tr>
<td>server</td>
<td>A principal that provides a resource to network clients. For example, if you ssh to the system central.example.com, then that system is the server that provides the ssh service. See also service principal.</td>
</tr>
<tr>
<td>server principal</td>
<td>(RPCSEC_GSS API) A principal that provides a service. The server principal is stored as an ASCII string in the form service@host. See also client principal.</td>
</tr>
<tr>
<td>service</td>
<td>1. A resource that is provided to network clients, often by more than one server. For example, if you rlogin to the machine central.example.com, then that machine is the server that provides the rlogin service.</td>
</tr>
<tr>
<td></td>
<td>2. A security service (either integrity or privacy) that provides a level of protection beyond authentication. See also integrity and privacy.</td>
</tr>
<tr>
<td>service key</td>
<td>An encryption key that is shared by a service principal and the KDC, and is distributed outside the bounds of the system. See also key.</td>
</tr>
</tbody>
</table>
service principal  
A principal that provides Kerberos authentication for a service or services. For service principals, the primary name is a name of a service, such as ftp, and its instance is the fully qualified host name of the system that provides the service. See also host principal, user principal.

session key  
A key that is generated by the authentication service or the ticket-granting service. A session key is generated to provide secure transactions between a client and a service. The lifetime of a session key is limited to a single login session. See also key.

SHA1  
Secure Hashing Algorithm. The algorithm operates on any input length less than $2^{64}$ to produce a message digest. The SHA1 algorithm is input to DSA.

single-system image  
A single-system image is used in Oracle Solaris auditing to describe a group of audited systems that use the same naming service. These systems send their audit records to a central audit server, where the records can be compared as if the records came from one system.

slave KDC  
A copy of a master KDC, which is capable of performing most functions of the master. Each realm usually has several slave KDCs (and only one master KDC). See also KDC, master KDC.

software provider  
In the Cryptographic Framework feature of Oracle Solaris, a kernel software module or a PKCS #11 library that provides cryptographic services. See also provider.

stash file  
A stash file contains an encrypted copy of the master key for the KDC. This master key is used when a server is rebooted to automatically authenticate the KDC before it starts the kadmin and krb5kdc processes. Because the stash file includes the master key, the stash file and any backups of it should be kept secure. If the encryption is compromised, then the key could be used to access or modify the KDC database.

superuser model  
The typical UNIX model of security on a computer system. In the superuser model, an administrator has all-or-nothing control of the system. Typically, to administer the machine, a user becomes superuser (root) and can do all administrative activities.

synchronous audit event  
The majority of audit events. These events are associated with a process in the system. A non-attributable event that is associated with a process is a synchronous event, such as a failed login.

TGS  
Ticket-Granting Service. That portion of the KDC that is responsible for issuing tickets.

TGT  
Ticket-Granting Ticket. A ticket that is issued by the KDC that enables a client to request tickets for other services.

ticket  
An information packet that is used to securely pass the identity of a user to a server or service. A ticket is valid for only a single client and a particular service on a specific server. A ticket contains the principal name of the service, the principal name of the user, the IP address of the user's host, a timestamp, and a value that defines the lifetime of the ticket. A ticket is created with a random session key to be used by the client and the service. Once a ticket has been created, it can be reused until the ticket expires. A ticket only serves to authenticate a client when it is presented along with a fresh authenticator. See also authenticator, credential, service, session key.

ticket file  
See credential cache.
| **user principal** | A principal that is attributed to a particular user. A user principal’s primary name is a user name, and its optional instance is a name that is used to described the intended use of the corresponding credentials (for example, jdoe or jdoe/admin). Also known as a user instance. See also service principal. |
| **virtual private network (VPN)** | A network that provides secure communication by using encryption and tunneling to connect users over a public network. |
Index

Numbers and Symbols

$(double dollar sign)$, parent shell process number, 189
[ (square brackets), audit record output, 621
* (asterisk)
  checking for in RBAC authorizations, 174
  device_allocate file, 86, 87
  wildcard character
    in RBAC authorizations, 198
@ (at sign), device_allocate file, 87
\ (backslash)
  device_allocate file, 86
  device_maps file, 85
^ (caret)
  audit class prefix modifier, 615–616
  in audit class prefixes, 555–558, 603
.(dot)
  authorization name separator, 198
  displaying hidden files, 115
= (equal sign), file permissions symbol, 112
- (minus sign)
  audit class prefix, 615–616
  file permissions symbol, 112
  file type symbol, 108
  su/log file, 60
+ (plus sign)
  audit class prefix, 615–616
  file permissions symbol, 112
  in audit class prefixes, 577
  su/log file, 60
# (pound sign)
  device_allocate file, 86

$(Continued)$
  device_maps file, 85
; (semicolon), device_allocate file, 86
> (redirect output), preventing, 44
>> (append output), preventing, 44
~/.gkadmin file, description, 507
~/.k5login file, description, 507
~/.rhosts file, description, 313
~/.hosts file, description, 313
~/.ssh/authorized_keys file
  description, 312
  override, 314
~/.ssh/config file
  description, 313
  override, 314
~/.ssh/environment file, description, 313
~/.ssh/id_dsa file, override, 314
~/.ssh/id_rsa file, override, 314
~/.ssh/identity file, override, 314
~/.ssh/known_hosts file
  description, 312
  override, 314
~/.ssh/rc file, description, 313
32-bit executables, protecting from compromising security, 114–115
3des -cbc encryption algorithm, ssh_config file, 307
3des encryption algorithm, ssh_config file, 306

A
- A option, auditreduce command, 588–589
Index

a option
  audit record command, 586
  digest command, 224
  encrypt command, 228
  Kerberized commands, 501
  mac command, 226
absolute mode
  changing file permissions, 112, 119–120
  changing special file permissions, 120–121
  description, 112
  setting special permissions, 113
access
  address space, 42–43
  control lists
    See ACL
  getting to server
    with Kerberos, 516–519
  granting to your account, 498–500
  login authentication with Secure Shell, 296–297
  obtaining for a specific service, 518–519
  restricting for
    devices, 40–42, 71
    system hardware, 62–63
  restricting for KDC servers, 429–430
  root access
    displaying attempts on console, 61–62
    monitoring su command attempts, 43, 60
    restricting, 48, 61–62
  Secure RPC authentication, 321
  security
    ACLs, 47
    controlling system usage, 42–46
    devices, 71
    file access restriction, 45
    firewall setup, 51
    login access restrictions, 36
    login authentication, 296–297
    login control, 36
    monitoring system usage, 46
    network control, 48–52
    NFS client-server, 323–325
    PATH variable setting, 44
    peripheral devices, 41
    physical security, 35–36
  access, security (Continued)
    remote systems, 281
    reporting problems, 52
    root login tracking, 43
    setuid programs, 45
    system hardware, 62–63
    UFS ACLs, 113–114
    sharing files, 48
  access control list
    See ACL
  Access Control Lists (ACLs), See ACL
  ACL
    description, 47, 113–114
    format of entries, 113–114
    kadm5.acl file, 464, 466, 470
    acl audit token, format, 623–624
    active audit policy, temporary audit policy, 558–560
    add_drv command, description, 82
  adding
    administration principals (Kerberos), 367
    allocatable device, 74
    audit classes, 563–564
    audit file systems, 566–569
    audit policy, 558–560
    auditing
      of individual users, 555–558, 599
      of roles, 166–167
      of zones, 539–546
    authorizations
      to command, 172
  cryptomgt role, 165–166
  DH authentication to mounted file systems, 326
  hardware provider mechanisms and features, 245
  library plugin, 236–237
  new authorization, 171–172
  new rights profile, 167–169
  new rights profile from existing one, 169–171
  PAM modules, 269
  plugins
    auditing, 572–573, 573–576, 577–578
    Cryptographic Framework, 235–237
    KMF, 260–261
  privileged users, 157
adding (Continued)

privileges
  directly to role, 180
  directly to user, 157
  to command in rights profile, 169

RBAC properties
to legacy applications, 172–174
roles, 163–165
security attributes
to legacy applications, 172–174
to roles, 179–180
to users, 155–157
security-related role, 165–166
security to devices, 73–78
security to system hardware, 62–63
service principal to keytab file (Kerberos), 484–485
software provider, 235–237
temporary audit policy, 560
user-level software provider, 236–237
address space, random layout, 42–43

admin_server section

krb5.conf file, 360, 366

administering

auditing
  audit -s command, 582–584, 585
  audit -t command, 584
  audit classes, 529–530
  audit events, 528
  audit files, 591–593
  audit records, 530
  audit_remote plugin, 572–573, 573–576
  audit_syslog plugin, 577–578
  audit trail overflow prevention, 594–595
  auditconfig command, 551–552, 554–555
  auditreduce command, 587–589
  configuring, 551–552
  cost control, 548
  description, 536
  disabling, 584
  efficiency, 550
  enabling, 585
  plugins, 572–573, 573–576
  policy, 558–560
  praudit command, 591–593

administrating, auditing (Continued)
  queue controls, 560–561
  reducing space requirements, 549
  refreshing, 582–584
  rights profiles required, 614
  in zones, 537, 614–615
  zones, 579–582
  auditing in zones, 540–541
  authorizations, 171–172
  Cryptographic Framework and zones, 217
  Cryptographic Framework commands, 215
  Cryptographic Framework plus FIPS–140, 217
  Cryptographic Framework task map, 230
  device allocation, 73–74
  device policy, 71–72
  file permissions, 115–123
  Kerberos
    keytabs, 483–489
    policies, 471–478
    principals, 458–471
  metaslot, 215
  NFS client-server file security, 323–325
  password algorithms, 57–59
  privileges, 185
  RBAC properties, 167–169, 171–172
  remote logins with Secure Shell, 292–294
  rights profiles, 167–169
    of a user, 180–181, 182–183
    role password, 178–179
  roles to replace superuser, 161–163
  Secure RPC task map, 326
  Secure Shell
    clients, 305–306
    overview, 303–305
    servers, 306
    task map, 285–286
  security properties
    authorizations, 171–172
    of a legacy application, 172–174
    of a rights profile, 167–169, 169–171
    of a user, 155–157
  user password to assume role, 180–181, 182–183
  without privileges, 140
administering (Continued)
  ZFS remotely with Secure Shell, 297–299
administrators, restricting rights, 181–182
AES kernel provider, 231
aes128-cbc encryption algorithm, ssh_config file, 307
aes128-ctr encryption algorithm, ssh_config file, 307
agent daemon, Secure Shell, 296–297
ahlt audit policy
description, 546
setting, 559
  with cnt policy, 618–619
algorithms
definition in Cryptographic Framework, 213
file encryption, 227–230
listing in the Cryptographic Framework, 231–235
passphrase protection in ssh-keygen, 284
password configuration, 57–58
password encryption, 37, 57–59
All (RBAC), rights profile, 196
all audit class, caution for using, 615
allocate command
  allocate error state, 85
  authorizations required, 85, 203
  removable media, 79
  user authorization, 75
  using, 78–79
allocate error state, 85
allocating devices
  by users, 78–79
  forcibly, 76
  troubleshooting, 79
AllowGroups keyword, sshd_config file, 306
AllowTcpForwarding keyword
  changing, 289
  sshd_config file, 306
AllowUsers keyword, sshd_config file, 306
ALTSHARE in Secure Shell, 311
always-audit classes, process preselection mask, 619
antivirus software, See virus scanning
appending arrow (>>), preventing appending, 44
application server, configuring, 378–381
arcfour encryption algorithm, ssh_config file, 307
ARCFOUR kernel provider, 231
archiving, audit files, 594–595
arge audit policy
  and exec_env token, 625
description, 546
  setting, 601
argument audit token, format, 624
argv audit policy
  and exec_args token, 625
description, 547
  setting, 601
assigning
  authorizations in a rights profile, 172
  privileges to commands in a rights profile, 169
  privileges to commands in a script, 193
  privileges to role, 180
  privileges to user, 157
  rights profile
to a role, 179–180
role to a user locally, 165–166
assuming role
  how to, 160–177
  in a terminal window, 154–155
  root, 155
asterisk (*)
  wildcard character
  in RBAC authorizations, 198
asterisk (*)
  checking for in RBAC authorizations, 174
device_allocate file, 86, 87
asynchronous audit events, 618–619
at command, authorizations required, 203
at sign (@), device_allocate file, 87
atq command, authorizations required, 203
attribute audit token, 624
attributes, keyword in BART, 95–96
audio devices, security, 88
audit -s command, 582–584, 585
audit -t command, 584
audit_binfile plugin, 531
  getting attributes, 570, 571
  limiting audit file size, 570
  removing queue size, 571
audit_binfile plugin (Continued)
  setting attributes, 569–571
  setting free space warning, 571
audit characteristics
  audit user ID, 620
  processes, 619–620
  session ID, 620
  terminal ID, 620
  user process preselection mask, 619

audit_class file
  adding a class, 563–564
  troubleshooting, 564

audit classes
  adding, 563–564
  configuration, 615–616
  description, 526, 528
  displaying defaults, 552–554
  exceptions to system-wide settings, 529
  mapping events, 530
  modifying default, 563–564
  overview, 529–530
  post-selection, 527
  prefixes, 615–616
  preselecting
    effect on public objects, 528
    for failure, 556–557, 577, 578
    for success, 556–557, 577, 578
    for success and failure, 554–555
  preselection, 527
  process preselection mask, 619
  replacing, 554–555
  syntax, 615
  user exceptions, 555–558

audit command
  disabling audit service, 584
  options, 613
  refreshing audit service, 582–584

Audit Configuration rights profile, 614
  auditing a role, 166–167
  configuring audit policy, 558–560
  displaying auditing defaults, 552–554
  preselecting audit classes, 554–555

Audit Control rights profile, 614
  disabling audit service, 584
  Audit Control rights profile (Continued)
    enabling audit service, 585
    refreshing audit service, 582–584
  audit directory, creating file systems for, 566–569
  audit_event file
    changing class membership, 564–565
    description, 528
    removing events safely, 605–606
  audit event-to-class mappings, changing, 564–565
  audit events
    asynchronous, 618–619
    audit_event file, 528
    changing class membership, 564–565
    description, 528
    mapping to classes, 530
    removing from audit_event file, 605–606
    selecting from audit trail, 589–590
    selecting from audit trail in zones, 615
    summary, 526
    synchronous, 618–619
    viewing from binary files, 591–593
  audit file system, description, 526
  audit files
    combining, 587–589
    compressing on disk, 606–607
    copying messages to single file, 590
    creating summary files, 590
    effects of Coordinated Universal Time (UTC), 588
    limiting size of, 606
    managing, 594–595
    printing, 591–592
    reading with praudit, 591–593
    reducing, 587–589
    reducing space requirements, 549
    reducing storage-space requirements, 550
    setting aside disk space for, 566–569
    time stamps, 620
    ZFS file systems, 566–569, 606–607
  audit flags, summary, 526
  audit_event_flag keyword, 553
  specifying user exceptions to audit
    preselection, 555–558
  use, 616
  using caret (^) prefix, 556–557
audit logs
  See also audit files
  comparing binary and text summaries, 531
  configuring, 565–578
  configuring text summary audit logs, 577–578
  modes, 531

audit notice entry, syslog.conf file, 577

audit plugins
  audit_binfile plugin, 560–561, 569–571
  audit_remote plugin, 572–573, 573–576
  audit_syslog plugin, 577–578
  description, 526
  qsize attribute, 560–561
  summary, 612–614, 616–617, 617

audit policy
  audit tokens from, 617
  defaults, 546–548
  description, 527
  displaying defaults, 552–554
  effects of, 546–548
  public, 547
  setting, 558–560
  setting ahlt, 559
  setting arge, 601
  setting argv, 601
  setting in global zone, 537, 614–615
  setting perzone, 560
  that does not affect tokens, 617
  tokens added by, 617

audit preselection mask
  modifying for existing users, 604–605
  modifying for individual users, 555–558

audit queue, events included, 530

audit queue controls
  displaying defaults, 552–554
  getting, 560–561

audit records
  converting to readable format, 592
  copying to single file, 590
  description, 527
  displaying, 591–593
  displaying definitions of procedure, 586–587
  displaying formats of a program, 586–587
  (Continued)
  displaying formats of an audit class, 587
  displaying in XML format, 592
  event modifiers, 626
  events that generate, 535
  format, 620
  formatting example, 586
  merging, 587–589
  overview, 530
  reducing audit files, 587–589
  sequence of tokens, 620
  /var/adm/auditlog file, 577
  audit_remote plugin, 531
    configuring, 573–576
    getting attributes, 572–573, 573–576
    setting attributes, 572–573, 573–576
    troubleshooting audit queue overfull, 573
  Audit Remote Server (ARS), managing, 534
  Audit Review rights profile, 614

audit session ID, 620

overview, 525–526

audit_syslog plugin, 531
  setting attributes, 577–578

audit tokens
  See also individual audit token names
  added by audit policy, 617
  audit record format, 620
  description, 527, 530
  format, 622
  list of, 622
  xclient token, 631

audit trail
  adding disk space, 569–571
audit trail (Continued)
analysis costs, 549
cleaning up not terminated files, 593–594
creating
summary files, 590
description, 527
effect of audit policy, 546
monitoring in real time, 550
no public objects, 528
overview, 536
preventing overflow, 594–595
reducing size of, 598–600, 606–607
selecting events from, 589–590
sending files to remote repository, 572–573, 573–576
viewing events from, 591–593
viewing events from different zones, 615
audit user ID
mechanism, 620
overview, 525–526
audit_warn script
configuring, 562
description, 613
auditconfig command (Continued)
setting system-wide audit parameters, 529
viewing default audit preselection, 554–555
auditd daemon
refreshing audit service, 582, 583
auditing
adding audit flags to a group of users, 558
all commands by users, 600–602
Audit Remote Server (ARS), 534
changes in current release, 538
changes in device policy, 72
configuring
all zones, 551–565
global zone, 559
identically for all zones, 579–581
per zone, 581–582
configuring in global zone, 540
defaults, 611–612
determining if running, 596–598
device allocation, 78
disabling, 584
enabling, 585
finding changes to specific files, 602–603
getting queue controls, 560–561
local definition, 527
logins, 607–608
man page summaries, 612–614
planning, 539–546
planning in zones, 540–541
plugin modules, 531
post-selection definition, 527
preselection definition, 527
privileges and, 205–206
remote definition, 528
removing user-specific audit flags, 557–558
rights profiles for, 614
roles, 166–167
setting queue controls, 560–561
sftp file transfers, 608–609
troubleshooting, 595–596
troubleshooting praudit command, 592
updating information, 582–584
users only, 557
zones and, 537, 614–615
auditlog file, text audit records, 577

auditrecord command
[] (square brackets) in output, 621
description, 613
displaying audit record definitions, 586–587
to example, 586
listing all formats, 586
listing formats of class, 587
listing formats of program, 586–587
optional tokens ([ ]), 621

auditreduce command
-A option, 588–589
-b option, 590
-C option, 589
-c option, 590
cleaning up audit files, 593–594
description, 613
e-option, 590
examples, 587–589
filtering options, 589
-M option, 589
merging audit records, 587–589
-o option, 587–589, 589, 590
selecting audit records, 589–590
time-stamp use, 588
trailer tokens and, 630
using lowercase options, 589
using uppercase options, 588

auditstat command, description, 614

auth_attr database
description, 200
summary, 199

AUTH_DES authentication, See AUTH_DH authentication
AUTH_DH authentication, and NFS, 321

authentication
AUTH_DH client-server session, 323–325
configuring cross-realm, 376–378
description, 49–50
DH authentication, 322–325
disabling with -X option, 502
Kerberos and, 333
naming services, 321
network security, 49–50

authentication (Continued)
NFS-mounted files, 329
overview of Kerberos, 516
Secure RPC, 321
Secure Shell
methods, 282–283
process, 304–305
terminology, 511–512
types, 49–50
use with NFS, 321

authentication methods
GSS-API credentials in Secure Shell, 282
host-based in Secure Shell, 283, 286–288
password in Secure Shell, 283
public keys in Secure Shell, 283
Secure Shell, 282–283

authenticator
in Kerberos, 512, 518

authorizations
adding to rights profile, 172
device allocation, 83–84
Kerberos and, 333
removing from rights profile, 170–171
troubleshooting, 174–177
types, 49–50

authorizations (RBAC)
checking for wildcards, 174
checking in privileged application, 135
commands that require authorizations, 203–204
database, 199–202
definition, 133–134
delegating, 198–199
description, 130, 198–199
for allocating device, 74–75
for device allocation, 84–85
granularity, 198
naming convention, 198
not requiring for device allocation, 77
solaris.device.allocate, 75, 84
solaris.device.revoke, 85

authorized_keys file, description, 312
AuthorizedKeysFile keyword, sshd_config file, 306
auths command, description, 202
AUTHS_GRANTED keyword, policy.conf file, 201
auto_transition option, SASL and, 319
automatic login
  disabling, 502
  enabling, 501
automatically configuring
  Kerberos
    master KDC server, 357–358
    slave KDC server, 369–370
automating principal creation, 459
auxprop_login option, SASL and, 319

B
  -b option, auditreduce command, 590
backup
  Kerberos database, 411–412
  slave KDCs, 348–349
Banner keyword, sshd_config file, 306
BART
  components, 92–93
  overview, 91–93
  programmatic output, 106
  security considerations, 93
  task map, 94
  verbose output, 105
bart create command, 92, 94–96
Basic Audit Reporting Tool, See BART
basic privilege set, 143
Basic Solaris User (RBAC), rights profile, 196
Batchmode keyword, ssh_config file, 306
BindAddress keyword, ssh_config file, 306
binding control flag, PAM, 276
blowfish-cbc encryption algorithm, ssh_config file, 307
Blowfish encryption algorithm
  allowing in heterogeneous environment, 58
Blowfish encryption algorithm, kernel provider, 231
Blowfish encryption algorithm
  policy.conf file, 58
  ssh_config file, 306
Bourne shell, privileged version, 137

C
  -C option, auditreduce command, 589
C shell, privileged version, 137
  -c option
    auditrecord command, 587
    auditreduce command, 590
  cache, credential, 516
canonical_user_plugin option, SASL and, 319
caret (^)
  in audit class prefixes, 555–558, 603
  using prefix in audit_flags value, 556–557
CD-ROM drives
  allocating, 80
  security, 88
cdrw command, authorizations required, 203
Certificate signing requests (CSR), See certificates
certificates
  exporting for use by another system, 253–254
  generating with pktool gencert command, 251–252
  importing into keystore, 252–253
  signing PKCS #10 CSR using the pktool command, 259–260
ChallengeResponseAuthentication keyword, See
  KbdInteractiveAuthentication keyword
changing
  allocatable devices, 77–78
  audit_class file, 563–564
  audit_event file, 564–565
  auditing defaults, 554–555
default password algorithm, 57–59
file ownership, 116–117
file permissions
  absolute mode, 119–120
  special, 120–121
  symbolic mode, 118
group ownership of file, 117–118
NFS secret keys, 323
passphrase for Secure Shell, 294
password algorithm for a domain, 58–59
password algorithm task map, 57–59
password of role, 178–179
properties of role, 179–180
rights profile contents, 167–169
changing (Continued)
  root password, 54
  root role into user, 183–185
  special file permissions, 120–121
  your password with kpasswd, 496
  your password with passwd, 496
CheckHostIP keyword, ssh_config file, 306
chgrp command
  description, 108
  syntax, 117
chkey command, 323, 328
chmod command
  changing special permissions, 120–121, 121
  description, 108
  syntax, 120
choosing, your password, 495–496
chown command, description, 108
chroot directory, sftp and, 290–291
ChrootDirectory keyword, ssh_config file, 306
Ciphers keyword, ssh_config file, 306
Ciphers keyword, Secure Shell, 307
classes, See audit classes
cleaning up, binary audit files, 593–594
clear protection level, 502
ClearAllForwardings keyword, Secure Shell port forwarding, 307
client names, planning for in Kerberos, 347–348
ClientAliveCountMax keyword, ssh_config file, 307
ClientAliveInterval keyword, ssh_config file, 307
clients
  AUTH DH client-server session, 323–325
  configuring for Secure Shell, 304, 305–306
  configuring Kerberos, 387–405
  definition in Kerberos, 511
cntconf principal
  creating, 362, 369
clock skew
  Kerberos and, 405–407
  Kerberos planning and, 350
clock synchronizing
  Kerberos master KDC and, 363, 369
  Kerberos planning and, 350
  Kerberos slave KDC and, 375
  Kerberos slave server and, 421
cloning, rights profile contents, 169–171
cmd audit token, 624
cnt audit policy
  description, 547
  with ah1t policy, 618–619
combining audit files
  auditreduce command, 587–589
  from different zones, 615
command execution, Secure Shell, 305
command-line equivalents of SEAM Tool, 454–455
commands
  See also individual commands
  Cryptographic Framework commands, 215
determining user’s privileged commands, 188–189
device allocation commands, 84
device policy commands, 82–83
file protection commands, 107
for administering privileges, 204
Kerberos, 509–510
RBAC administration commands, 202–203
Secure RPC commands, 323
Secure Shell commands, 314–316
that assign privileges, 144
that check for privileges, 135
user-level cryptographic commands, 216
common keys
  calculating, 324
  DH authentication and, 322–325
components
  BART, 92–93
device allocation mechanism, 83
RBAC, 130–132
Secure Shell user session, 305
compressing, audit files on disk, 606–607
Compression keyword, Secure Shell, 307
CompressionLevel keyword, ssh_config file, 307
Computer Emergency Response Team/Coordination Center (CERT/CC), 52
computer security, See system security
computing
  DH key, 327
digest of a file, 224–225
MAC of a file, 225–227
secret key, 220–224
configuration decisions
auditing
  file storage, 544
  policy, 546–548
  remote file storage, 545–546
  who and what to audit, 541–544
  zones, 540–541
Kerberos
  client and service principal names, 347–348
  clients, 350–351
  clock synchronization, 350
  database propagation, 350
  encryption types, 352–353
  KDC server, 351
  mapping host names onto realms, 347
  number of realms, 346
  ports, 348
  realm hierarchy, 347
  realm names, 346
  realms, 346–347
  slave KDCs, 348–349
password algorithm, 37
configuration files
  auditing, 612–614
  device_maps file, 85
PAM
  syntax, 272
  for password algorithms, 37
  policy.conf file, 37, 57–58, 202
Secure Shell, 303
  syslog.conf file, 205
  with privilege information, 205
configured audit policy, permanent audit policy, 558–560
configuring
  active audit policy, 560
  ahlt audit policy, 559
  audit_class file, 563–564
  audit classes, 554–555
  audit_event file, 564–565
  audit logs task map, 565–566
  audit policy, 558–560
  audit policy temporarily, 560
  audit queue controls, 560–561
configuring (Continued)
  audit service policy, 558–560
  audit trail overflow prevention, 594–595
  audit_warn script, 562
  auditing, 551–565
  auditing in zones, 537, 614–615
  auditing task map, 551–552
  authorizations, 171–172
  chroot directory for sftp, 290–291
  device allocation, 73–74
  device policy, 71–72
  DH key for NIS user, 327–328
  DH key in NIS, 326–327
  exceptions to Secure Shell system defaults, 289–290
  hardware security, 62–63
  host-based authentication for Secure Shell, 286–288
  identical auditing for non-global zones, 579–581
Kerberos
  adding administration principals, 367
  clients, 387–405
  cross-realm authentication, 376–378
  master KDC server, 357–358, 358–359, 359–363
  master KDC server using LDAP, 363–369
  NFS servers, 382–383
  overview, 355–431
  task map, 355–356
password for hardware access, 62–63
per-zone auditing, 581–582
permanent audit policy, 558–560
per zone audit policy, 560
port forwarding in Secure Shell, 288–289
privileged users, 157
RBAC, 160–177
RBAC task map, 160–161
rights profiles, 167–169
roles, 163–165, 179–180
root role as user, 183–185
Secure Shell
  clients, 305–306
  servers, 306
Secure Shell task map, 285–286
space for audit trail, 569–571
temporary audit policy, 558–560
configuring (Continued)
  text summaries of audit records, 577–578
configuring application servers, 378–381
Connection Attempts keyword, ssh_config file, 307
ConnectTimeout keyword, ssh_config file, 307
console, displaying su command attempts, 61–62
CONSOLE in Secure Shell, 311
Console User (RBAC), rights profile, 196
CONSOLE_USER keyword, policy.conf file, 201
consumers, definition in Cryptographic Framework, 214
context-sensitive help, SEAM Tool, 455
control flags, PAM, 276
control manifests (BART), 91
controlling, system usage, 42–46
conversation keys
  decrypting in secure RPC, 324
generating in secure RPC, 323
converting, audit records to readable format, 592
Coordinated Universal Time (UTC)
  time stamp use in auditing, 588, 620
copying, files using Secure Shell, 300–301
copying audit records to single file, 590
cost control, and auditing, 548
cryptmd5.so.1 plug-in, SASL and, 318
creating
  audit trail, 620
  authorization, 171–172
  credential table, 383–384
  file digests, 224–225
  key pair, 255–259
  new device-cleant scripts, 88–89
  new policy (Kerberos), 463, 475–476
  new principal (Kerberos), 463–465
  privileged users, 157
  rights profile for a group of users, 558
  rights profiles, 167–169
  roles, 163–165
  root user, 183–185
  secret keys
    for encryption, 220–224
  Secure Shell keys, 292–294
  stash file, 374, 421
  storage for binary audit files, 566–569
creating (Continued)
  tickets with kinit, 492
cred database, DH authentication, 322–325
cred table
  DH authentication and, 323
  information stored by server, 325
credential
  cache, 516
description, 324, 512
mapping, 349
obtaining for a server, 517–518
obtaining for a TGS, 516–517
or tickets, 335
credential table, adding single entry to, 384
crontab files, authorizations required, 203
cross-realm authentication, configuring, 376–378
CRYPT_ALGORITHMS_ALLOW keyword, policy.conf file, 38
CRYPT_ALGORITHMS_DEPRECATE keyword, policy.conf file, 38
crypt bsdbf password algorithm, 38
crypt bsmd5 password algorithm, 38
crypt command, file security, 47
CRYPT_DEFAULT keyword, policy.conf file, 38
CRYPT_DEFAULT system variable, 57
crypt_sha256 password algorithm, 38, 57–59
crypt_sunmd5 password algorithm, 38
crypt_unix password algorithm, 38
Crypto Management (RBAC), creating role, 165–166
cryptoadm command
  description, 215
disabling cryptographic mechanisms, 239, 240
disabling hardware mechanisms, 244–245
installing PKCS #11 library, 237
listing providers, 231
-m option, 239, 240
-p option, 239, 240
restoring kernel software provider, 241
cryptoadm install command, installing PKCS #11 library, 237
Cryptographic Framework
  administering with role, 165–166
  connecting providers, 216–217
  consumers, 213
Cryptographic Framework (Continued)
cryptoadm command, 215
definition of terms, 213
description, 211–213
elfsign command, 216
error messages, 230
FIPS–140 and, 217
hardware plugins, 213
interacting with, 215–217
listing providers, 231–235
PKCS #11 library, 213
providers, 213
refreshing, 245–246
registering providers, 217
restarting, 245–246
signing providers, 216
SPARC T4 series optimizations, 217–218
user-level commands, 216
zones and, 217, 245–246
cryptographic mechanisms, optimized for SPARC T4 series, 217–218
cryptographic services, See Cryptographic Framework
Cryptoki, See PKCS #11 library
csh command, privileged version, 137
customizing, manifests, 96–97
customizing a report (BART), 101–102
databases (Continued)
backing up and propagating KDC, 411–412
creating KDC, 361
cred for Secure RPC, 322
eexec_attr, 201
KDC propagation, 350
NFS secret keys, 323
prof_attr, 201
publickey for Secure RPC, 322
RBAC, 199–202
user_attr, 200
deallocate command
allocate error state, 85
authorizations required, 85, 203
device-clean scripts and, 88–89
using, 81–82
deallocating
devices, 81–82
forcibly, 76–77
microphone, 81
debugging sequence number, 629
decrypt command
description, 216
syntax, 229
decrypting
collection keys for Secure RPC, 324
files, 229
NFS secret keys, 323
secret keys, 323
default/login file, description, 312
default_realm section
krb5.conf file, 360, 366
defaults
audit service, 611–612
privilege settings in policy.conf file, 205
system-wide in policy.conf file, 37
umask value, 111
definitive control flag, PAM, 276
delegating, RBAC authorizations, 198–199
delete_entry command, ktutil command, 488
deleting
archived audit files, 594
audit files, 587
host’s service, 488
deleting (Continued)
   not_terminated audit files, 593–594
   policies (Kerberos), 478
   principal (Kerberos), 467–468
DenyGroups keyword, sshd_config file, 307
DenyUsers keyword, sshd_config file, 307
DES encryption, kernel provider, 231
DES encryption, Secure NFS, 322
destroying, tickets with kdestroy, 494
determining
   audit ID of a user, 604
   auditing is running, 596–598
   files with setuid permissions, 121
   privileges on a process, 189–190
   privileges task map, 185
/dev/arp device, getting IP MIB-II information, 73
devfsadm command, description, 82
device_allocate file
description, 86–87
format, 86
sample, 77, 86
device allocation
   adding devices, 73–74
   allocatable devices, 87
   allocate error state, 85
   allocating devices, 78–79
   auditing, 78
   authorizations, 83–84
   authorizations for commands, 84–85
   authorizing users to allocate, 74–75
   changing allocatable devices, 77–78
   commands, 84
   components of mechanism, 83
   configuration file, 85
deallocate command
   device-clean scripts and, 88–89
   using, 81–82
deallocating devices, 81–82
device_allocate file, 86–87
device-clean scripts
description, 87–89
options, 88
writing new scripts, 88–89
device_maps file, 85–86
device allocation (Continued)
   disabling, 74
   enabling, 74
   examples, 79
   forcibly allocating devices, 76
   forcibly deallocating devices, 76–77
   making device allocatable, 74
   managing devices, 73–74
   mounting devices, 79–81
   not requiring authorization, 77
   preventing, 77–78
   requiring authorization, 77–78
   rights profiles, 83–84
   SMF service, 83
   task map, 73–74
   troubleshooting, 79, 81
   troubleshooting permissions, 76
   unmounting allocated device, 81–82
   user procedures, 73–78
   using, 73–78
   using allocate command, 78–79
   viewing information, 75–76
device-clean scripts
description, 87–89
media, 87
object reuse, 87–89
options, 88
writing new scripts, 88–89
device management, See device policy
Device Management rights profile, 83–84
device_maps file
description, 85
format, 85
sample entries, 85
device policy
   add_drv command, 82
   auditing changes, 72
   commands, 82
   configuring, 71–73
   kernel protection, 82–89
   managing devices, 71–72
   overview, 40–42
   task map, 71–72
   update_drv command, 82
device policy (Continued)
viewing, 72
Device Security rights profile, 74, 83–84
devices
allocating for use, 73–78
auditing allocation of, 78
auditing policy changes, 72
authorizing users to allocate, 74–75
changing which are allocatable, 77–78
deallocating a device, 81–82
device allocation
See device allocation
forcibly allocating, 76
forcibly deallocating, 76–77
getting IP MIB-II information, 73
listing, 72
listing device names, 75
login access control, 40
making allocatable, 74
managing, 71–72
managing allocation of, 73–74
mounting allocated devices, 79–81
not requiring authorization for use, 77
policy commands, 82–83
preventing use of all, 77–78
preventing use of some, 77
privilege model and, 145–146
protecting by device allocation, 41
protecting in the kernel, 41
security, 40–42
superuser model and, 145–146
unmounting allocated device, 81–82
viewing allocation information, 75–76
viewing device policy, 72
zones and, 41
DH authentication
configuring in NIS, 326–327
description, 322–325
for NIS client, 326–327
mounting files with, 329
sharing files with, 328–329
Diffie-Hellman authentication, See DH authentication
digest command (Continued)
example, 225
digestmd5.so.1 plug-in, SASL and, 318
digests
computing for file, 224–225
digests
computing for file, 224–225
direct realms, 377–378
directories
See also files
displaying files and related information, 107, 115–116
permissions
defaults, 111
description, 108–109
public directories, 111
DisableBanner keyword, ssh_config file, 307
disabling
32-bit executables that compromise security, 114–115
abort sequence, 63
audit policy, 558–560
audit service, 584
cryptographic mechanisms, 239
device allocation, 74
executable stacks, 122–123
hardware mechanisms, 244–245
keyboard abort, 63
keyboard shutdown, 63
logging of executable stack messages, 123
logins temporarily, 56–57
programs from using executable stacks, 122–123
remote root access, 61–62
service on a host (Kerberos), 487–489
system abort sequence, 63
user logins, 56–57
disk space, for binary audit files, 566–569
disk space requirements, audit files, 549
displaying
allocatable devices, 75–76
audit policies, 558
audit policy defaults, 552–554
audit queue controls, 552–554, 561
audit record definitions, 586–587
enabling
audit service, 585
auditing, 585
cryptographic mechanisms, 240
device allocation, 74
kernel software provider use, 241
keyboard abort, 63
mechanisms and features on hardware provider, 245
encrypt command
description, 216
error messages, 230
troubleshooting, 230
encrypting
communications between hosts, 295
crypt command, 227–230
dotted dollar sign ($$), parent shell process number, 189
encryption
algorithms
Kerberos and, 352–353
DES algorithm, 322
generating symmetric key
using the pktool command, 220–224
list of password algorithms, 37
modes
Kerberos and, 352–353
password algorithm, 37
private service, 333
specifying algorithms in ssh_config file, 306
specifying password algorithm
locally, 57–59
specifying password algorithms in policy.conf file, 37
types
Kerberos and, 352–353, 519–521
with -x option, 502
environment variables
See also variables
audit token for, 625
environment variables (Continued)
  overriding proxy servers and ports, 301
  PATH, 44
  presence in audit records, 546, 622
  Secure Shell and, 310–311
  use with ssh-agent command, 314
  equal sign (=), file permissions symbol, 112
error messages
  encrypt command, 230
  Kerberos, 433–447
  with kpasswd, 496
errors, allocate error state, 85
EscapeChar keyword, ssh_config file, 307
/etc/default/kbd file, 63
/etc/default/login file
description, 312
  restricting remote root access, 61–62
  Secure Shell and, 310–311
/etc/default/su file
displaying su command attempts, 61–62
monitoring access attempts, 61–62
monitoring su command, 60
/etc/hosts.equiv file, description, 313
/etc/krb5/kadm5.acl file, description, 507
/etc/krb5/kadm5.keytab file, description, 508
/etc/krb5/kdc.conf file, description, 508
/etc/krb5/kpropd.acl file, description, 508
/etc/krb5/krb5.conf file, description, 508
/etc/krb5/krb5.keytab file, description, 508
/etc/krb5/warn.conf file, description, 508
/etc/logindevperm file, 40
/etc/nologin file
description, 313
  disabling user logins temporarily, 56–57
/etc/pam.conf file, Kerberos and, 508
/etc/publckey file, DH authentication and, 322
/etc/syslog.conf file, PAM and, 270
/etc/security/audit_event file, audit events and, 528
/etc/security/device_allocate file, 86
/etc/security/device_maps file, 85
/etc/security/policy.conf file, algorithms configuration, 57–58
/etc/ssh_host_dsa_key.pub file, description, 312
/etc/ssh_host_rsa_key.pub file, description, 312
/etc/ssh/hosts.equiv file, description, 313
/etc/ssh/ssh_config file
configuring Secure Shell, 305–306
description, 313
host-specific parameters, 310
keywords, 306–311
override, 313
/etc/ssh/ssh_host_dsa_key file, description, 312
/etc/ssh/ssh_host_rsa_key file, describe, 312
/etc/ssh/ssh_known_hosts file
  controlling distribution, 311
description, 312
override, 314
secure distribution, 312
/etc/ssh/sshd_config file
description, 312
keywords, 306–311
/etc/ssh/sshrb file, description, 313
/etc/syslog.conf file
  auditing and, 577, 614
event, description, 528
  event modifiers, audit records, 626
  exec_args audit token
    argv policy and, 625
    format, 625
exec_attr database
  description, 401
  summary, 199
exec_env audit token, format, 625
  executable stacks
    disabling logging messages, 123
    logging messages, 114
    protecting against, 122–123
    protecting against 32-bit processes, 114
execute permissions, symbolic mode, 112
export subcommand, pktool command, 253–254
extended policy, privileges, 144
EXTERNAL security mechanism plug-in, SASL
  and, 318
files (Continued)
displaying hidden files, 115
displaying information about, 107
encrypting, 219–220, 227–230
file types, 108
finding files with setuid permissions, 121
for administering Secure Shell, 312
hashing, 219–220
kdc.conf, 513
Kerberos, 507–509
manifests (BART), 103–104
mounting with DH authentication, 329
ownership
and setgid permission, 110
and setuid permission, 110
permissions
absolute mode, 112, 119–120
changing, 108, 111–113, 118
defaults, 111
description, 108–109
setgid, 110
setuid, 110
sticky bit, 110–111
symbolic mode, 112, 118
umask value, 111
PKCS #12, 254
privileges relating to, 139
protecting with UNIX permissions, 115
public objects, 528
rsyslog.conf, 270–271, 577–578
scanning for integrity, 91–106
security
access restriction, 45
ACL, 47
changing ownership, 116–117
changing permissions, 111–113, 118
directory permissions, 108–109
displaying file information, 107, 115–116
encryption, 47, 219–220
file permissions, 108–109
file types, 108
special file permissions, 113
umask default, 111
UNIX permissions, 107–113
files, security (Continued)
  user classes, 108
  sharing with DH authentication, 328–329
  special files, 109–111
  symbols of file type, 108
  syslog.conf, 270–271, 577–578
  syslog.conf, 614
  tracking integrity, 91–106
  verifying integrity with digest, 224–225
  with privilege information, 205
find command, finding files with setuid permissions, 121
FIPS-140, Cryptographic Framework and, 217
FIPS-140 support
  Secure Shell remote access, 285
  Secure Shell using a Sun Crypto Accelerator 6000 card, 285
firewall systems
  connecting from outside, 302
  outside connections with Secure Shell
    from command line, 302
    from configuration file, 301–302
  packet smashing, 51–52
  packet transfers, 51–52
  secure host connections, 301
  security, 51
  trusted hosts, 51
flags line, process preselection mask, 619
fmr1 audit token format, 625–626
forced cleanup, st_clean script, 89
format of audit records, auditrecord command, 586
forwardable tickets
  definition, 512
  description, 334
  example, 492
  with -F option, 501, 503
  with -f option, 501, 503
ForwardAgent keyword, Secure Shell forwarded authentication, 307
ForwardX11 keyword, Secure Shell port forwarding, 307
ForwardX11Trusted keyword, Secure Shell port forwarding, 307
fp audit event modifier, 626
FQDN (Fully Qualified Domain Name), in
  Kerberos, 347–348
ftp command
  Kerberos and, 500–503, 509
  logging file transfers, 608–609
  setting protection level in, 502
G
  GatewayPorts keyword, Secure Shell, 307
gateways, See firewall systems
gencert subcommand, pktool command, 251–252
  generating
    certificates with pktool command, 251–252
    key pair
      using the pktool command, 255–259
    keys for Secure Shell, 292–294
    NFS secret keys, 323
    passphrases with pktool command, 254–255
    random number
      using the pktool command, 220–224
    Secure Shell keys, 292–294
    symmetric key
      using the pktool command, 220–224
    X.509 v3 certificate, 259–260
getdevpolicy command, description, 82
gentet command, description, 202
  -getflags option
    auditconfig command, 552–554, 554–555
  -getflags option
    auditconfig command, 552–554, 554–555
  -getplugin option
    auditconfig command, 552–554, 554–555
  -getpolicy option
    auditconfig command, 552–554, 554–555
  -getqctrl option, auditconfig command, 552–554
  getting
    access to a specific service, 518–519
    credential for a server, 517–518
    credential for a TGS, 516–517
gkadmin command
  See also SEAM Tool
description, 509
Index

.gkadmin file
description, 507
SEAM Tool and, 455
GlobalKnownHostsFile keyword
See GlobalKnownHostsFile keyword
ssh_config file, 307
granting access to your account, 498–500
group audit policy
and groups token, 547
description, 547
group token, and, 626
group audit token
format, 626
group policy, and, 626
groups
changing file ownership, 117–118
exceptions to Secure Shell defaults, 289–290
GSS-API
authentication in Secure Shell, 282
credentials in Secure Shell, 304
Kerberos and, 334
gssapi.so.1 plug-in, SASL and, 318
GSSAPIAuthentication keyword, Secure Shell, 307
GSSAPIDelegateCredentials keyword, ssh_config file, 307
GSSAPIKeyExchange keyword, Secure Shell, 307
GSSAPIStoreDelegatedCredentials keyword, ssh_config file, 307
gsscred command, description, 509
gsscred table, using, 521
gssd daemon, Kerberos and, 510

H
-h option, auditrecord command, 586
hard disk, space requirements for auditing, 549
hardware
Cryptographic Framework and, 217–218
listing attached hardware accelerators, 243–244
protecting, 35–36, 62–63
requiring password for access, 62–63
SPARC T4 series, 217–218
hardware providers
disabling cryptographic mechanisms, 244–245
hardware providers (Continued)
   enabling mechanisms and features on, 245
listing, 243–244
loading, 243
hash
   algorithms
      Kerberos and, 352–353
hashing, files, 219–220
HashKnownHosts keyword, ssh_config file, 307
header audit token
event modifiers, 626
format, 626
order in audit record, 626
help
SEAM Tool, 455–456
URL for online, 353
Help Contents, SEAM Tool, 456
hierarchical realms
configuring, 376–377
in Kerberos, 339, 347
hmac-md5 algorithm, ssh_config file, 308
hmac-sha1 encryption algorithm, ssh_config file, 308
host-based authentication
configuring in Secure Shell, 286–288
description, 282
Host keyword
   ssh_config file, 308, 310
host names, mapping onto realms, 347
host principal
creating, 362, 368
HostBasedAuthentication keyword, Secure Shell, 308
HostUsesNameFromPacketOnly keyword,
   sshd_config file, 308
HostKey keyword, sshd_config file, 308
HostKeyAlgorithms keyword, ssh_config file, 308
HostKeyAlias keyword, ssh_config file, 308
HostName keyword, ssh_config file, 308
hosts
   disabling Kerberos service on, 487–489
   exceptions to Secure Shell defaults, 289–290
   Secure Shell hosts, 282
   trusted hosts, 51
hosts.equiv file, description, 313
I
- I option
  bart create command, 94
  st_clean script, 88
- i option
  bart create command, 94, 97
  encrypt command, 228
  st_clean script, 88
identity files (Secure Shell), naming conventions, 312
IDs
  audit
    mechanism, 620
    overview, 525–526
  audit session, 620
  mapping UNIX to Kerberos principals, 521
IgnoreIfUnknown keyword, ssh_config file, 308
IgnoreRhosts keyword, sshd_config file, 308
IgnoreUserKnownHosts keyword, sshd_config file, 308
import subcommand, pktool command, 252–253
in.rlogind daemon, Kerberos and, 510
in.rshd daemon, Kerberos and, 510
in.telnetd daemon, Kerberos and, 510
include control flag, PAM, 276
inheritable privilege set, 142
initial ticket, definition, 512
install subcommand, cryptoadm command, 237
installing, Secure by Default, 45
instance, in principal names, 338–339
integrity
  Kerberos and, 333
  security service, 340
interactively configuring
  Kerberos
    master KDC server, 358–359
    slave KDC server, 370–371
INTERNAL plug-in, SASL and, 318
Internet firewall setup, 51
Internet-related audit tokens
  ip address token, 626–627
  ip port token, 627
  socket token, 629–630
invalid ticket, definition, 512
ioctl() system calls, AUDIO_SETINFO(), 88
ip address audit token, format, 626–627
IP addresses
  exceptions to Secure Shell defaults, 289–290
  Secure Shell checking, 306
IP MIB-II, getting information from /dev/arp not /dev/Ip, 73
ip port audit token, format, 627
ipc audit token, 627
format, 627
IPC_perm audit token, format, 628
IPC privileges, 140
ipc type field values (ipc token), 627

K
- k option
  encrypt command, 228
  Kerberized commands, 502
  mac command, 226
- K option
  encrypt command, 228
  mac command, 226
  Kerberized commands, 502
  rolemod command, 180
  usermod command, 157
  .k5.REALM file, description, 508
  .k5login file
description, 498–500, 507
rather than revealing password, 499
kadm5.acl file
description, 507
format of entries, 470
master KDC entry, 361, 367, 410
new principals and, 464, 466
kadm5.keytab file, description, 508
kadmin command
  creating host principal, 362, 368
description, 509
  ktadd command, 484–485
  ktremove command, 486
removing principals from keytab with, 485–486
SEAM Tool and, 453
kadmin.local command
  adding administration principals, 367
kadmin.local command (Continued)
    automating creation of principals, 459
    description, 509
kadmin.log file, description, 508
kadmind daemon
    Kerberos and, 510
    master KDC and, 511
kbd file, 63
KbdInteractiveAuthentication keyword, Secure Shell, 308
kcfd daemon, 215, 245–246
kclient command, description, 509
kdb5_ldap_util command, description, 509
kdb5_util command
    creating KDC database, 361
    creating stash file, 374, 421
    description, 510
KDC
    backing up and propagating, 411–412
    configuring master
        automatic, 357–358
        interactive, 358–359
        manual, 359–363
        with LDAP, 363–369
    configuring slave
        automatic, 369–370
        interactive, 370–371
        manual, 371–375
    copying administration files from slave to master, 373, 419
    creating database, 361
    creating host principal, 362, 368
    database propagation, 350
    master
        definition, 510
        planning, 348–349
        ports, 348
        restricting access to servers, 429–430
        slave, 348–349
        definition, 510
    slave or master, 339–340, 356
    starting daemon, 375, 421
    swapping master and slave, 407–411
KDC (Continued)
    synchronizing clocks
        master KDC, 363, 369
        slave KDC, 375, 421
kdc.conf file
    description, 508
    ticket lifetime and, 513
kdc.log file, description, 508
kdcmgr command
    configuring master
        automatic, 357
        interactive, 358
    configuring slave
        automatic, 370
        interactive, 370
    server status, 359
kdestroy command
    example, 494
    Kerberos and, 509
KeepAlive keyword, Secure Shell, 308
Kerberos
    administering, 453–489
    Administration Tool
        See SEAM Tool
    commands, 500–505, 509–510
    components of, 341–342
    configuration decisions, 345–353
    configuring KDC servers, 356–375
    daemons, 510
    encryption types
        overview, 352–353
        using, 519–521
    error messages, 433–447
    examples of using Kerberized commands, 504–505
    files, 507–509
    gaining access to server, 516–519
    granting access to your account, 498–500
    Kerberos V5 protocol, 333
    online help, 353
    options to Kerberized commands, 501
    overview
        authentication system, 334–340, 516
        Kerberized commands, 500–503
        password management, 495–500
Kerberos (Continued)
    planning for, 345–353
realms
    See realms (Kerberos)
    reference, 507–522
remote applications, 338
table of network command options, 502
terminology, 510–516
troubleshooting, 447
using, 491–505
Kerberos authentication, and Secure RPC, 322
Kerberos commands, 500–505
    examples, 504–505
kern.notice entry, syslog.conf file, 114
kernel providers, listing, 231
Key Distribution Center, See KDC
key management framework (KMF), See KMF
key pairs
    creating, 255–259
    generating
        using the pktool command, 255–259
KEYBOARD_ABORT system variable, 63
keylogin command, use for Secure RPC, 323
KeyRegenerationInterval keyword, sshd_config file, 308
keys
    creating DH key for NIS user, 327–328
    creating for Secure Shell, 292–294
    definition in Kerberos, 511
    generating for Secure Shell, 292–294
    generating key pair
        using the pktool command, 255–259
        generating symmetric key
            using the pktool command, 220–224
    service key, 483–489
    session keys
        Kerberos authentication and, 516
        using for MAC, 226
keyserv daemon, 326
keyserver
    description, 323
    starting, 326
keystores (Continued)
    exporting certificates, 253–254
    importing certificates, 252–253
    listing contents, 251
    managed by KMF, 248
    protecting with password in KMF, 254–255
    supported by KMF, 248, 249
keytab file
    adding master KDC's host principal to, 362, 369
    adding service principal to, 483, 484–485
    administering, 483–489
    administering with ktutil command, 484
    disabling a host's service with delete_entry command, 488
    read into keytab buffer with read_kt command, 487, 488
    removing principals with ktremove command, 486
    removing service principal from, 485–486
    viewing contents with ktutil command, 486
    viewing keylist buffer with list command, 487, 488
keytab option, SASL and, 319
keywords
    See also specific keyword
    attribute in BART, 95–96
    command-line overrides in Secure Shell, 315
    Secure Shell, 306–311
kgcmgr command, description, 510
kinit command
    example, 492
    -f option, 492
    Kerberos and, 509
    ticket lifetime, 513
klist command
    example, 493–494
    -f option, 493–494
    Kerberos and, 509
KMF
    adding plugin, 260–261
    creating
        passphrases for keystores, 249
        password for keystore, 254–255
        self-signed certificate, 251–252
    exporting certificates, 253–254
    importing certificates into keystore, 252–253
Index

KMF (Continued)
  keystores, 248, 249
  library, 248
  listing plugins, 260–261
  managing
    keystores, 249
    PKI policy, 248
    plugins, 249
    public key technologies (PKI), 247
  removing plugin, 260–261
  utilities, 248
kmfcfg command
  list plugin subcommand, 260–261
  plugin subcommands, 247, 249
known_hosts file
  controlling distribution, 311
  description, 312
Korn shell, privileged version, 137
kpasswd command
  error message, 496
  example, 497
  Kerberos and, 509
  passwd command and, 496
kprop command, description, 509
kpropd.acl file, description, 508
kpropd daemon, Kerberos and, 510
kproplog command, description, 510
krb-warn.conf file, description, 508
krb5.conf file
  description, 508
  domain_realm section, 347
  editing, 360, 366
  ports definition, 348
krb5.keytab file, description, 508
krb5cc_uid file, description, 508
krb5kdcd daemon
  Kerberos and, 510
  master KDC and, 511
  starting, 375, 421
ksh command, privileged version, 137
ktadd command
  adding service principal, 483, 484–485
  syntax, 485
ktkt_warn daemon, Kerberos and, 510
ktremove command, 486
ktutil command
  administering keytab file, 484
  delete_entry command, 488
  Kerberos and, 509
  list command, 487, 488
  read_kt command, 487, 488
  viewing list of principals, 486
kvno command, Kerberos and, 509

L
-L option, ssh command, 299–300
-l option
  digest command, 224
  mac command, 225
  ssh command, 295–296
layout of address space, load-time randomization, 42–43
LDAP, configuring master KDC using, 363–369
LDAP naming service
  passwords, 37
  specifying password algorithm, 59
  least privilege, principle of, 139
libraries, user-level providers, 231
lifetime of ticket, in Kerberos, 513–515
limit privilege set, 142
limiting
  audit file size, 606
  use of privileges in a rights profile, 168
list command, 487, 488
list_devices command
  authorizations required, 85, 203
list plugin subcommand, kmfcfg command, 260–261
list privilege, SEAM Tool and, 482
list subcommand, pktool command, 251
ListenAddress keyword, sshd_config file, 308
listing
  all RBAC security attributes, 150–151
  available providers in Cryptographic Framework, 231–235
  contents of keystore, 251
  Cryptographic Framework providers, 243–244
  device policy, 72
listing (Continued)
  hardware providers, 243–244
  providers in the Cryptographic
    Framework, 231–235
  rights of initial user, 151–154
  roles you can assume, 154, 202
  users with no passwords, 56
  your RBAC rights, 151–154
load-time randomization, address space layout, 42–43
local auditing, 527
LocalForward keyword, ssh_config file, 308
log files
  audit records, 531, 592
  BART
    programmatic output, 105–106
    verbose output, 105–106
  configuring for audit service, 577–578
  failed login attempts, 57
  monitoring su command, 60
  syslog audit records, 614
  /var/adm/messages, 598
  /var/log/syslog, 598
log_level option, SASL and, 319
logadm command, archiving text summary audit
  files, 594
logging, ftp file transfers, 608–609
logging in
  and AUTH_DH, 323
  auditing logins, 607–608
  disabling temporarily, 56–57
  displaying user’s login status, 55
  log of failed logins, 57
  root login
    restricting to console, 61–62
    tracking, 43
security
  access control on devices, 40
  access restrictions, 36
  system access control, 36
  tracking root login, 43
  task map, 53–54
  users’ basic privilege set, 143
with Secure Shell, 295–296
with Secure Shell to display a GUI, 296
login access restrictions,
  svc:/system/name-service/switch:default, 36
login environment variables, Secure Shell
  and, 310–311
login file, restricting remote root access, 61–62
LoginGraceTime keyword, sshd_config file, 308
logins command
  displaying user’s login status, 55
  displaying users with no passwords, 56
  syntax, 55
LogLevel keyword, Secure Shell, 308
LookupClientHostnames keyword, sshd_config
  file, 308
-1spolicy option, auditconfig command, 558–560

M
  -M option, auditreduce command, 589
  -m option
    cryptoadm command, 239, 240
    Kerberized commands, 502
mac command
  description, 216
  syntax, 225
machine security, See system security
MACS keyword, Secure Shell, 308
mail, using with Secure Shell, 299
man pages
  audit service, 612–614
  commands that require authorizations, 203–204
  device allocation, 84
  policy, 31
  RBAC, 202–203
  Secure Shell, 314–316
managing
  See also administering
  audit files, 587–589, 594–595
  audit records task map, 585–586
  audit trail overflow, 594–595
  auditing in zones, 537, 614–615
  device allocation task map, 73–74
  devices, 73–74
  file permissions, 115–123
  keystores with KMF, 249
Index

managing (Continued)
  passwords with Kerberos, 495–500
  privileges task map, 185
  RBAC task map, 177–178
manifests
  See also bart create
  control, 91
  customizing, 96–97
  file format, 103–104
  test in BART, 92–93
manually configuring
  Kerberos
    master KDC server, 359–363
    master KDC server using LDAP, 363–369
    slave KDC server, 371–375
mapping
  host names onto realms (Kerberos), 347
  UIDs to Kerberos principals, 521
mapping GSS credentials, 349
mappings, events to classes (auditing), 530
mask (auditing), description of process
  preselection, 619
master KDC
  automatically configuring, 357–358
  configuring with LDAP, 363–369
  definition, 510
  interactively configuring, 358–359
  manually configuring, 359–363
  slave KDCs and, 339–340, 356
  swapping with slave KDC, 407–411
Match blocks
  chroot directory and, 290–291
  exceptions to Secure Shell defaults, 289–290
  Match keyword, sshd_config file, 308
  max_life value, description, 513
  max_renewable_life value, description, 514
  MaxStartups keyword, sshd_config file, 308
  MD4 encryption algorithm, kernel provider, 231
  MD5 encryption algorithm
    allowing in heterogeneous environment, 58
    encryption algorithm, kernel provider, 231
  MD5 encryption algorithm
    policy.conf file, 57–58, 58
  mech_dh mechanism, GSS-API credentials, 304
  mech_krb mechanism, GSS-API credentials, 304
  mech_list option, SASL and, 319
  mechanism, definition in Cryptographic Framework, 214
    mechanisms
      disabling all on hardware provider, 244–245
      enabling some on hardware provider, 245
      media, device-clean scripts, 87–89
      Media Backup rights profile
        assigning to trusted users, 129, 163
      Media Restore rights profile, assigning to trusted users, 163
      merging, binary audit records, 587–589
      message authentication code (MAC), computing for
        file, 225–227
      messages file, executable stack messages, 114
      metaslot
        administering, 215
        definition in Cryptographic Framework, 214
      microphone
        allocating, 78–79
        deallocating, 81
      minus sign (-)
        audit class prefix, 615–616
        entry in sulog file, 60
        file permissions symbol, 112
        symbol of file type, 108
      mode, definition in Cryptographic Framework, 214
    modifying
      policies (Kerberos), 477–478
      principal’s password (Kerberos), 467
      principals (Kerberos), 466–467
      roles (RBAC), 179–180
      user security attributes, 555–558
      users (RBAC), 155–157
  modules, password encryption, 37
monitoring
  audit trail in real time, 550
  root access, 60–62
  root access attempts, 61–62
  su command attempts, 43, 60
  system usage, 46
  use of privileged commands, 166–167
  mount command, with security attributes, 75
mounting
allocated CD-ROM, 80
allocated devices, 79–81
files with DH authentication, 329
nt command, 88

N
-n option, bart create command, 94
n2cp driver
   hardware plugin to Cryptographic Framework, 213
   listing mechanisms, 243–244
na audit event modifier, 626
names
   device names
      device_maps file, 86
naming conventions
   audit files, 620
devices, 75
   RBAC authorizations, 198
Secure Shell identity files, 312
naming service configuration, login access restrictions, 36
naming services
   See individual naming services
   scope and RBAC, 137
ncp driver
   hardware plugin to Cryptographic Framework, 213
   listing mechanisms, 243–244
NET privileges, 140
netservices limited installation option, 45
network, privileges relating to, 140
Network IPsec Management rights profile, adding solaris.admin.edit authorization, 170
network security
   authentication, 49–50
   authorizations, 49–50
   controlling access, 48–52
firewall systems
   need for, 51
   packet smashing, 51–52
   trusted hosts, 51
   overview, 49
   reporting problems, 52
Network Time Protocol, See NTP
never-audit classes, process preselection mask, 619
new features
   auditing enhancements, 538
   SASL, 317
   Secure Shell and FIPS-140, 285
   Secure Shell enhancements, 283–284
newkey command
   creating key for NIS user, 327–328
   generating keys, 323
NFS file systems
   authentication, 321
   providing client-server security, 323–325
   secure access with AUTH_DH, 329
NFS servers, configuring for Kerberos, 382–383
NIS naming service
   authentication, 321
   passwords, 37
   specifying password algorithm, 58–59
nisaddcred command, generating keys, 323
nobody user, 48
noexec_user_stack_log variable, 114, 123
noexec_user_stack variable, 114, 122–123
NoHostAuthenticationForLocalHost keyword, ssh_config file, 309
nologin file, description, 313
nonhierarchical realms, in Kerberos, 339
nscd (name service cache daemon), use, 202
NSS, managing keystore, 249
NTP
   Kerberos planning and, 350
   master KDC and, 363, 369
   slave KDC and, 375, 421
NumberOfPasswordPrompts keyword, ssh_config file, 309

O
-o option
   auditreduce command, 587–589, 589, 590
-o option, encrypt command, 228
object reuse requirements
   device-clean scripts
      writing new scripts, 88–89
object reuse requirements (Continued)
   for devices, 87–89

obtaining
   access to a specific service, 518–519
   credential for a server, 517–518
   credential for a TGS, 516–517
   forwardable tickets, 492
   privileged commands, 179–180
   privileges, 143, 144, 157, 180
   privileges on a process, 189–190
   tickets with kinit, 492

online help
   SEAM Tool, 455–456
   URL for, 353

OpenSSH, See Secure Shell

OpenSSL
   managing keystore, 249
   version, 248

Operator (RBAC)
   recommended role, 129
   rights profile, 196

optional control flag, PAM, 276

options to Kerberized commands, 501

order of search
   security attributes, 197
   user security attributes, 197

overflow prevention, audit trail, 594–595

ovsecadm xxxx file, description, 508

ownership of files
   ACLs and, 47
   changing, 108, 116–117
   changing group ownership, 117–118
   UFS ACLs and, 113–114

PAM
   adding a module, 269
   configuration file
      Kerberos and, 508
   configuration files
      control flags, 276
      introduction, 273
      stacking, 275–277
      syntax, 273
   /etc/rsyslog.conf file, 270
   /etc/syslog.conf file, 270
   framework, 266
   Kerberos and, 342
   overview, 265
   planning, 268
   stack to cache authentication, 159
   stacking diagrams, 276
   example, 278–279
   explained, 275
   task map, 267

pam.conf file, See PAM configuration files

PAM configuration file, adding su stack, 159

PAM modules, 159

pam_roles command, description, 202

pam_tty_tickets.so.1 module, PAM, 159

PAMServicename keyword, sshd_config file, 309

PAMServicexPrefix keyword, sshd_config file, 309

panels, table of SEAM Tool, 479–482

passphrases
   changing for Secure Shell, 294
   encrypt command, 227
   example, 295
   generating in KMF, 254–255
   mac command, 226
   storing safely, 229
   using for MAC, 226
   using in Secure Shell, 296–297

PASSREQ in Secure Shell, 311

passwd command
   and kpasswd command, 496
   and naming services, 37
   changing password of role, 178–179
   syntax, 54
password authentication, Secure Shell, 282
PasswordAuthentication keyword, Secure Shell, 309
passwords
authentication in Secure Shell, 282
changing role password, 178–179
changing with kpasswd command, 496
changing with passwd -r command, 37
changing with passwd command, 496
constraining encryption algorithms in
heterogeneous environment, 58
displaying users with no passwords, 56
eliminating in Secure Shell, 296–297
cipher algorithms, 37
finding users with no passwords, 56
granting access without revealing, 498–500
hardware access and, 62–63
LDAP, 37
 specifying new password algorithm, 59
local, 37
login security, 36
managing, 495–500
modifying a principal’s password, 467
NIS, 37
 specifying new password algorithm, 58–59
policies and, 496
PROM security mode, 36, 62–63
protecting
 keystore, 254
PKCS #12 file, 254
requiring for hardware access, 62–63
secret-key decryption for Secure RPC, 323
specifying algorithm, 57–58
 in naming services, 58–59
 locally, 57–59
suggestions on choosing, 495–496
system logins, 36
task map, 53–54
UNIX and Kerberos, 495–500
using Blowfish in heterogeneous environment, 58
using MD5 encryption algorithm for, 57–58
using new algorithm, 58
using user’s to assume role, 180–181, 182–183
path_attr audit token, 628
path audit policy, description, 547
path audit token, format, 628
PATH environment variable
 and security, 44
setting, 44
PATH in Secure Shell, 311
permanent audit policy, configured audit
policy, 558–560
permissions
ACLs and, 47
changing file permissions
 absolute mode, 112, 119–120
 chmod command, 108
 symbolic mode, 112, 118
defaults, 111
directory permissions, 108–109
file permissions
 absolute mode, 112, 119–120
 changing, 111–113, 118
description, 108–109
special permissions, 111, 113
symbolic mode, 112, 118
finding files with setuid permissions, 121
setgid permissions
 absolute mode, 113, 121
description, 110
symbolic mode, 112
setuid permissions
 absolute mode, 113, 121
description, 110
security risks, 110
symbolic mode, 112
special file permissions, 109–111, 111, 113
sticky bit, 110–111
UFS ACLs and, 113–114
umask value, 111
user classes and, 108
PermitEmptyPasswords keyword, sshd_config
file, 309
PermitRootLogin keyword, sshd_config file, 309
permitted privilege set, 142
PermitUserEnvironment keyword, sshd_config
file, 309
perzone audit policy
description, 547
planning, Kerberos (Continued)
   realm hierarchy, 347
   realm names, 346
   realms, 346–347
   slave KDCs, 348–349
   PAM, 268
   RBAC, 161–163
pluggable authentication module, See PAM
plug_in_list option, SASL and, 319
plugins
   adding to KMF, 260–261
   auditing, 531
   Cryptographic Framework, 213
   managed in KMF, 249
   removing from KMF, 260–261
   SASL and, 318
plus sign (+)
   audit class prefix, 615–616
   entry in su log file, 60
   file permissions symbol, 112
   in audit class prefixes, 577
policies
   administering, 453–489
   creating (Kerberos), 463
   creating new (Kerberos), 475–476
   deleting, 478
   for auditing, 546–548
   modifying, 477–478
   on devices, 72
   overview, 30–31
   passwords and, 496
   SEAM Tool panels for, 479–482
   specifying password algorithm, 57–59
   task map for administering, 471
   viewing attributes, 473–475
   viewing list of, 471–473
policy
   definition in Cryptographic Framework, 214
   definition in Oracle Solaris, 30–31
policy.conf file
   description, 201–202, 202
   keywords
      for password algorithms, 38
      for privileges, 201, 205
policy.conf file, keywords (Continued)
  for RBAC authorizations, 201
  for rights profiles, 201
  for workstation owner, 201
  specifying encryption algorithms in, 57–58
  specifying password algorithm
    in naming services, 58–59
  specifying password algorithms, 57–58
Port forwarding
  configuring in Secure Shell, 288–289
  Secure Shell, 299, 300
Port keyword, Secure Shell, 309
ports, for Kerberos KDC, 348
post-selection in auditing, 527
postdated ticket
  definition, 513
  description, 334
pound sign (#)
  device_allocate file, 86
  device_maps file, 85
ppriv command, listing privileges, 189
praudit command
  converting audit records to readable format, 592
  description, 614
  piping audit reduce output to, 591–592
  use in a script, 592
  viewing audit records, 591–593
  XML format, 592
PreferredAuthentications keyword, ssh_config
  file, 309
prefixes for audit classes, 615–616
preselecting, audit classes, 554–555
preselection in auditing, 527
preselection mask (auditing), description, 619
PreUserAuthHook keyword, ssh_config file, 309
preventing
  audit trail overflow, 594–595
  kernel software provider use, 240–243
  use of hardware mechanism, 244–245
primary, in principal names, 338–339
principal
  adding administration, 367
  adding service principal to keytab, 483, 484–485
  administering, 453–489
principal (Continued)
  automating creation of, 459
  creating, 463–465
  creating clntconfig, 362, 369
  creating host, 362, 368
  deleting, 467–468
  duplicating, 466
  Kerberos, 338–339
  modifying, 466–467
  principal name, 338–339
  removing from keytab file, 486
  removing service principal from keytab, 485–486
  SEAM Tool panels for, 479–482
  service principal, 338
  setting up defaults, 468–469
  task map for administering, 458–459
  user ID comparison, 383–384
  user principal, 338
  viewing attributes, 461–463
  viewing list of, 459–461
  viewing sublist of principals, 460
principal file, description, 508
principal.kadm5 file, description, 508
principal.kadm5.lock file, description, 508
principal.ok file, description, 508
principal. ulog file, description, 508
principle of least privilege, 139
Printer Management (RBAC), rights profile, 196
printing, audit log, 591–592
PrintLastLog keyword, ssh_config file, 309
PrintMotd keyword, sshd_config file, 309
priv.debug entry, syslog.conf file, 205
PRIV_DEFAULT keyword
  policy.conf file, 201, 205
PRIV_LIMIT keyword
  policy.conf file, 201, 205
PRIV_PROC_LOCK_MEMORY privilege, 141–142
privacy
  availability, 502
  Kerberos and, 333
  security service, 340
private keys
  See also secret keys
  definition in Kerberos, 511
private keys (Continued)
- Secure Shell identity files, 312
- private protection level, 502
- privilege audit token, 628
- privilege checking, in applications, 135
- privilege sets
  - adding privileges to, 144
    - basic, 143
    - effective, 142
    - inheritable, 142
    - limit, 142
    - listing, 143
    - permitted, 142
    - removing privileges from, 145
- privileged application
  - authorization checking, 135
    - description, 131
    - ID checking, 134
    - privilege checking, 135
- privileged ports, alternative to Secure RPC, 50
- privileges (Continued)
  - implemented in sets, 142
    - inherited by processes, 143
    - limiting use in a rights profile, 168
    - listing, 186–187
    - listing on a process, 189–190
  - PRIV_PROC_LOCK_MEMORY, 141–142
  - processes with assigned privileges, 143
  - programs aware of privileges, 143
  - protecting kernel processes, 138
  - removing from a user, 145
  - removing from basic set, 168
  - removing from limit set, 157
  - removing from limit set in rights profile, 168
  - task map, 185
  - troubleshooting
    - to users, 174–177
    - troubleshooting requirements for, 191–192
    - using in shell script, 193
- PROC privileges, 140
- process audit characteristics
  - audit session ID, 620
  - audit user ID, 620
  - process preselection mask, 619
  - terminal ID, 620
- process audit token, format, 629
- process preselection mask, description, 619
- process privileges, 140
- process rights management, See privileges
- processing time costs, of audit service, 548
- prof_attr database
  - description, 201
  - summary, 199
- profile shells
  - description, 137
  - opening, 157–160
  - restricting rights, 181–182
- profiles, See rights profiles
- profiles command
  - creating rights profiles, 167–169
    - description, 202
    - listing user's rights profiles, 153
    - modifying rights profile, 168
- PROFS_GRANTED keyword, policy.conf file, 201
proftpd daemon, Kerberos and, 510
programs
checking for RBAC authorizations, 173
privilege-aware, 142, 143
project.max-locked-memory resource control, 141–142
PROM security mode, 62–63
propagation
KDC database, 350
Kerberos database, 411–412
protecting
32-bit executables from compromising security, 114–115
BIOS, pointer to, 62–63
by using passwords with Cryptographic Framework, 250–251
contents of keystore, 254
files with Cryptographic Framework, 219–220
PROM, 62–63
sftp transfer directory, 290–291
system from risky programs, 121
protecting files
user procedures, 115
with UFS ACLs, 113–114
with UNIX permissions, 107–113, 115
with UNIX permissions task map, 115
protection level
clear, 502
private, 502
safe, 502
setting in ftp, 502
Protocol keyword, Secure Shell, 309
providers (Continued)
restoring use of kernel software provider, 241
signing, 216
proxiable ticket, definition, 513
proxy ticket, definition, 513
ProxyCommand keyword, ssh_config file, 309
pseudo-tty, use in Secure Shell, 305
PublicAuthentication keyword, Secure Shell, 309
public audit policy
description, 547
read-only events, 547
public directories
auditing, 528
sticky bit and, 111
public key authentication, Secure Shell, 282
public key cryptography
AUTH_DH client-server session, 323–325
changing NFS public keys and secret keys, 323
common keys
calculation, 324
database of public keys for Secure RPC, 323
generating keys
conversation keys for Secure NFS, 323
using Diffie-Hellman, 323
NFS secret keys, 323
public key cryptography, See PKI
public keys
changing passphrase, 294
DH authentication and, 322–325
generating public-private key pair, 292–294
Secure Shell identity files, 312
public objects, auditing, 528
publickey map, DH authentication, 322–325
pwcheck_method option, SASL and, 319
Q
qsize attribute, audit plugins, 560–561
quoting syntax in BART, 104–105
rcp command
   Kerberos and, 500–503, 509
rd audit event modifier, 626
read .kt command, 487, 488
read permissions, symbolic mode, 112
readable audit record format, converting audit records to, 592
realms (Kerberos)
   configuration decisions, 346–347
   configuring cross-realm authentication, 376–378
   contents of, 340
   direct, 377–378
   hierarchical, 376–377
   hierarchical or nonhierarchical, 339
   hierarchy, 347
   in principal names, 338–339
   mapping host names onto, 347
   names, 346
   number of, 346
   requesting tickets for specific, 502
   servers and, 339–340
reauth_timeout option, SASL and, 319
redirecting arrow (>), preventing redirection, 44
reducing
   audit files, 587–589
   disk space required for audit files, 606–607
   storage-space requirements for audit files, 550
refreshing
   audit service, 582–584, 583–584
   cryptographic services, 245–246
registering providers, Cryptographic Framework, 217
RekeyLimit keyword, ssh_config file, 309
rem_drv command, description, 82
remote auditing, 528
remote logins
   authentication, 49–50
   authorization, 49–50
   preventing root access, 61–62
   security and, 325
RemoteForward keyword, ssh_config file, 309
removable media, allocating, 79
removing
   audit events from audit_event file, 605–606
   cryptographic providers, 240

RC4, See ARCFOUR kernel provider
removing (Continued)
plugins from KMF, 260–261
principals with ktremove command, 486
privileges from basic set, 168
privileges from limit set, 157
privileges from limit set in rights profile, 168
service principal from keytab file, 485–486
software providers
permanently, 242
temporarily, 241
user-specific auditing, 557–558
renewable ticket, definition, 513
replacing
preselected audit classes, 554–555
superuser with roles, 161–163
replayed transactions, 325
reporting tool, See bart compare
repository, installing third-party providers, 236
required control flag, PAM, 276
requisite control flag, PAM, 276
resource controls
privileges, and, 141–142
project_max-locked-memory, 141–142
zone_max-locked-memory, 141–142
restarting
cryptographic services, 245–246
ssh service, 289
sshd daemon, 289
restoring, cryptographic providers, 241
restricted shell (rsh), 44
restricting
remote root access, 61–62
root access, 60–62
user privileges in an assigned rights profile, 168
restricting access for KDC servers, 429–430
RETRIES in Secure Shell, 311
return audit token, format, 629
reoffl option, mt command, 88
.rhosts file, description, 313
RhostsAuthentication keyword, Secure Shell, 309
RhostsRSAAuthentication keyword, Secure Shell, 309
right, See rights profiles
rights, restricting administrator to explicitly
assigned, 181–182
rights profiles
adding privileges to command, 169
adding solaris.admin.edit authorization, 170
All, 196
assigning to trusted users, 129, 163
for audit service, 614
authenticating with user’s password, 180–181, 182–183
Basic Solaris User, 196
changing contents of, 167–169
cloning contents of, 169–171
Console User, 196, 197
contents of typical, 195
creating for Sun Ray users, 168
databases
See prof_attr database and exec_attr database
description, 131, 135–136
Device Management, 83–84
Device Security, 74, 83–84
major rights profiles descriptions, 195
modifying, 167–169, 169–171
Network IPSec Management, 170
Operator, 196
order of search, 197
preventing privilege escalation, 129, 163
Printer Management, 196
removing authorizations, 170–171
removing privileges from limit set, 168
restricting basic privileges, 168
Stop, 196, 197
System Administrator, 196
troubleshooting, 174–177
using the System Administrator profile, 62
viewing contents, 197
VSCAN Management, 170–171
rlogin command
Kerberos and, 500–503, 509
rlogind daemon, Kerberos and, 510
role-based access control, See RBAC
roleadd command
description, 202
using, 163
roleauth keyword
  passwords for roles, 180–181, 182–183

rolemod command
  changing properties of role, 179, 182
  description, 202
  passwords for roles, 180–181, 182–183

roles
  assigning privileges to, 180
  assigning with usermod command, 165–166
  assuming, 154–155
  assuming after login, 136
  assuming in a terminal window, 137, 154–155
  assuming root role, 155
  auditing, 166–167
  authenticating with user’s password, 180–181, 182–183
  changing password of, 178–179
  changing properties of, 179–180
  creating, 163–165
    - Crypto Management role, 165–166
      description, 136
  determining directly assigned privileges, 187–188
  determining role’s privileged commands, 189
  listing local roles, 154, 202
  making root role into user, 183–185
  modifying, 179–180
  recommended roles, 128
  summary, 131
  use in RBAC, 128
  using an assigned role, 154–155
  using to access the hardware, 62–63
  using user password, 132, 182–183

roles command
  description, 202
  using, 154

root access
  monitoring and restricting, 60–62
  monitoring attempts, 61–62
  troubleshooting remote, 62

root account, description, 39

root principal, adding to host’s keytab, 483

root role
  assuming role, 155
  changing from root user, 184

root role (Continued)
  changing password, 54
  changing to root user, 183–185
  provided role, 129
  troubleshooting, 185

root user
  changing into root role, 184
  displaying access attempts on console, 61–62
  monitoring su command attempts, 43, 60
  replacing in RBAC, 136
  restricting access, 48
  restricting remote access, 61–62
  tracking logins, 43

RSA kernel provider, 231

RSAAuthentication keyword, Secure Shell, 309

rsh command
  Kerberos and, 500–503, 509
  rsh command (restricted shell), 44
  rshd daemon, Kerberos and, 510
  rstchown system variable, 117
  rsyslog.conf entry
    - creating for PAM, 270–271
    - creating for real-time audit logs, 577–578
  rules file (BART), 93
  rules file attributes, See keywords
  rules file format (BART), 104–105
  rules file specification language, See quoting syntax

S
  -S option, st_clean script, 89
  -s option
    - audit command, 582–584, 585
  safe protection level, 502

SASL
  environment variable, 318
  options, 319
  overview, 317
  plug-ins, 318
  saslauthd_path option, SASL and, 319
  scope (RBAC), description, 137

scp command
  copying files with, 300–301
  description, 315
scp command (Continued)
   Kerberos and, 509

scripts
   audit_warn script, 562, 613
   checking for RBAC authorizations, 173
device-clean scripts
   See also device-clean scripts
   for cleaning devices, 87–89
   monitoring audit file example, 550
   processing praudit output, 592
   running with privileges, 145
   securing, 173
   use of privileges in, 193

SCSI devices, st_clean script, 87

SEAM Tool (Continued)
   starting, 457
   table of panels, 479–482
   viewing a principal's attributes, 461–463
   viewing list of policies, 471–473
   viewing list of principals, 459–461
   viewing policy attributes, 473–475

secret keys
   creating, 220–224
   generating
      using the pktool command, 220–224
   generating for Secure RPC, 323
Secure by Default installation option, 45
secure connection
   across a firewall, 301
   logging in, 295–296
Secure NFS, 322
Secure RPC
   alternative, 50
   and Kerberos, 322
   description, 321
   implementation of, 323–325
   keys server, 323
   overview, 49–50
Secure Shell
   administering, 303–305
   administering ZFS, 297–299
   administrator task map, 285–286
   authentication
      requirements for, 282–283
   authentication methods, 282–283
   authentication steps, 304–305
   basis from OpenSSH, 283–284
   changes in current release, 283–284
   changing passphrase, 294
   command execution, 305
   configuring chroot directory, 290–291
   configuring clients, 305–306
   configuring port forwarding, 288–289
   configuring server, 306
   connecting across a firewall, 301
   connecting outside firewall
      from command line, 302
      from configuration file, 301–302

secret keys
creating, 220–224

generating
   using the pktool command, 220–224

generating for Secure RPC, 323

Secure by Default installation option, 45
secure connection
   across a firewall, 301
   logging in, 295–296

Secure NFS, 322

Secure RPC
   alternative, 50
   and Kerberos, 322
   description, 321
   implementation of, 323–325
   keys server, 323
   overview, 49–50

Secure Shell
   administering, 303–305
   administering ZFS, 297–299
   administrator task map, 285–286
   authentication
      requirements for, 282–283
   authentication methods, 282–283
   authentication steps, 304–305
   basis from OpenSSH, 283–284
   changes in current release, 283–284
   changing passphrase, 294
   command execution, 305
   configuring chroot directory, 290–291
   configuring clients, 305–306
   configuring port forwarding, 288–289
   configuring server, 306
   connecting across a firewall, 301
   connecting outside firewall
      from command line, 302
      from configuration file, 301–302

Index
Secure Shell (Continued)
copying files, 300–301
creating keys, 292–294
data forwarding, 305
description, 281
files, 312
FIPS-140 support, 285
forwarding mail, 299
generating keys, 292–294
keywords, 306–311
local port forwarding, 299, 300
logging in fewer prompts, 296–297
logging in to display remote GUI, 296
logging in to remote host, 295–296
login environment variables and, 310–311
naming identity files, 312
protocol versions, 282
public key authentication, 282
remote port forwarding, 300
scp command, 300–301
specifying exceptions to system defaults, 289–290
TCP and, 288
typical session, 303–305
user procedures, 292
using port forwarding, 299–300
using without password, 296–297
xauth package, 296

security
across insecure network, 301
auditing, 525–538
auditing and, 534–535
BART, 91–106
computing digest of files, 224–225
computing MAC of files, 225–227
Cryptographic Framework, 211–218
device allocation, 71–89
devices, 40–42
DH authentication, 323–325
encrypting files, 227–230

security (Continued)
installation options, 45
key management framework, 247–261
netservices limited installation option, 45
NFS client-server, 323–325
password encryption, 37
policy overview, 30–31
preventing remote login, 61–62
protecting against denial of service, 45
protecting against Trojan horse, 44
protecting devices, 87–89
protecting hardware, 62–63
protecting PROM, 62–63
Secure by Default, 45
Secure Shell, 281–302
system hardware, 62–63
systems, 35–52
security attributes
checking for, 134
considerations when directly assigning, 137–138
description, 131
listing all RBAC, 150–151
Network Security rights profile, 132
order of search, 197
privileges on commands, 135
special ID on commands, 135
usability considerations when directly assigning, 138
using to mount allocated device, 75
security mechanism, specifying with -m option, 502
security modes, setting up environment with multiple, 385–387
security policy, default (RBAC), 199
security service, Kerberos and, 340
selecting
audit classes, 554–555
audit records, 589–590
events from audit trail, 589–590
semicolon (;), device allocate file, 86
sendmail command, authorizations required, 204
seq audit policy
and sequence token, 548, 629
description, 548
sequence audit token
  and seq audit policy, 629
  format, 629
ServerAliveCountMax keyword, ssh_config file, 309
ServerAliveInterval keyword, ssh_config file, 309
ServerKeyBits keyword, sshd_config file, 310
servers
  AUTH_DH client-server session, 323–325
  configuring for Secure Shell, 306
  definition in Kerberos, 511
  gaining access with Kerberos, 516–519
  obtaining credential for, 517–518
  realms and, 339–340
service
  definition in Kerberos, 511
  disabling on a host, 487–489
  obtaining access for specific service, 518–519
service keys
  definition in Kerberos, 511
  keytab files and, 483–489
service management facility, refreshing Cryptographic Framework, 236
Service Management Facility (SMF), See SMF
service principal
  adding to keytab file, 483, 484–485
  description, 338
  planning for names, 347–348
  removing from keytab file, 485–486
session ID, audit, 620
session keys
  definition in Kerberos, 511
  Kerberos authentication and, 516
-setflags option, auditconfig command, 554–555
setgid permissions
  absolute mode, 113, 121
  description, 110
  security risks, 110
  symbolic mode, 112
-setnaflags option, auditconfig command, 554–555
-setpin subcommand, pktool command, 254–255
-setplugin option
  auditconfig command, 572–573, 573–576, 577–578
  -setpolicy option, auditconfig command, 558–560
setting
 arge policy, 601
argv policy, 601
audit policy, 558–560
audit queue controls, 560–561
principal defaults (Kerberos), 468–469
setuid permissions
  absolute mode, 113, 121
  description, 110
  finding files with permissions set, 121
  security risks, 45, 110
  symbolic mode, 112
sftp command
  auditing file transfers, 608–609
  chroot directory and, 290–291
  copying files with, 301
  Kerberos and, 509
sftpcmdmand, description, 315
sh command, privileged version, 137
SHA1 kernel provider, 231
SHA2 kernel provider, 231
sharing files
  and network security, 48
  with DH authentication, 328–329
shell, privileged versions, 137
shell commands, passing parent shell process number, 189
shell process, listing its privileges, 189–190
shell scripts, writing privileged, 193
hosts . equiv file, description, 313
hosts file, description, 313
signing
  PKCS #10 CSR, 259–260
  using the pktool command, 259–260
signing providers, Cryptographic Framework, 216
double-sign-on system, 500–505
Kerberos and, 333
size of audit files
  reducing, 587–589
  reducing storage-space requirements, 550
slave datatrans file
  description, 508
KDC propagation and, 411–412
slave_data_trans_slave file, description, 508
slave KDCs
  automatically configuring, 369–370
  configuring, 371–375
definition, 510
  interactively configuring, 370–371
master KDC and, 339–340
or master, 356
planning for, 348–349
swapping with master KDC, 407–411
slot, definition in Cryptographic Framework, 215
SMF
  auditd service, 611–612
  Cryptographic Framework service, 215
device allocation service, 83
  enabling keyserver, 326
cf k service, 215
  managing Secure by Default configuration, 45
restarting Cryptographic Framework, 245–246
  restarting Secure Shell, 289
  ssh service, 289
socket audit token, 629–630
  solaris.admin.edit authorization, adding to rights profile, 170
solaris.device.revoke authorization, 85
  solaris.smf.value authorization, removing from rights profile, 170–171
sp audit event modifier, 626
SPARC T4 series, cryptographic optimizations, 217–218
special permissions
  setgid permissions, 110
  setuid permissions, 110
  sticky bit, 110–111
square brackets ([]), audit record output, 621
sr_clean script, description, 88
ssh-add command
  description, 314
  example, 296–297, 297
  storing private keys, 296–297
ssh-agent command
  description, 314
  from command line, 296–297
ssh command
  description, 314
  Kerberos and, 509
  overriding keyword settings, 315
  port forwarding options, 299–300
  remotely administering ZFS, 297–299
  using, 295–296
  using a proxy command, 302
  .ssh/config file
    description, 313
    override, 314
ssh_config file
  configuring Secure Shell, 305–306
  host-specific parameters, 310
  keywords, 306–311
    See specific keyword override, 313
  .ssh/environment file, description, 313
  .ssh_host_dsa_key file, description, 312
  .ssh_host_dsa_key.pub file, description, 312
  .ssh_host_key file, override, 314
  .ssh_host_key.pub file, description, 312
  .ssh_host_rsa_key file, description, 312
  .ssh_host_rsa_key.pub file, description, 312
  .ssh/id_dsa file, 314
  .ssh/id_rsa file, 314
  .ssh/identity file, 314
ssh-keygen command
  description, 314
  passphrase protection, 284
  using, 292–294
ssh-keyscan command, description, 314
ssh-keysign command, description, 314
  .ssh/known_hosts file
    description, 312
    override, 314
ssh_known_hosts file, 312
  .ssh/rc file, description, 313
sshd command, description, 314
  .sshd_config file
    description, 312
    keywords, 306–311
      See specific keyword override, 313
overrides of /etc/default/login entries, 310–311
sshd daemon, Kerberos and
sshd.pid file, description
sshrc file, description
st_clean script
standard cleanup, st_clean script
starting
auditing
device allocation
KDC daemon
Secure RPC keyservers
stash file
sticky bit permissions
absolute mode
description
symbolic mode
Stop (RBAC), rights profile
storage costs, and auditing
storage overflow prevention, audit trail
storing
audit files
audit files remotely
passphrase
StrictHostKeyChecking keyword, ssh_config file
StrictModes keyword, sshd_config file
su command
displaying access attempts on console
in role assumption
monitoring use
su file, monitoring su command
subject audit token, format
Subsystem keyword, sshd_config file
success, audit class prefix
sufficient control flag, PAM
su log file
monitoring contents of
Sun Crypto Accelerator 1000 board, listing mechanisms
Sun Crypto Accelerator 6000 board
hardware plugin to Cryptographic Framework
Secure Shell and FIPS-140
SUPATH in Secure Shell
superuser
See also root role
compared to privilege model
compared to RBAC model
differences from privilege model
eliminating in RBAC
troubleshooting becoming root as a role
svc:/system/device/allocate, device allocation service
svcadm command
administering Cryptographic Framework
enabling Cryptographic Framework
enabling keyservers
refocusing Cryptographic Framework
restarting
Secure Shell
syslog daemon
svcs command
listing cryptographic services
listing keyservers
swapping master and slave KDCs
symbolic links, file permissions
symbolic mode
changing file permissions
description
synchronizing clocks
master KDC
overview
slave KDC
SYS privileges
syslog.conf entry
creating for PAM
creating for real-time audit logs
syslog.conf file
and auditing
audit notice level
executable stack messages
kern notice level
priv debug entry
privilege debugging
SYSLOG_FAILED_LOGINS, in Secure Shell
SyslogFacility keyword, sshd_config file
System Administrator (RBAC)
  protecting hardware, 62
  recommended role, 129
  rights profile, 196
system calls
  argument audit token, 624
  exec_args audit token, 625
  exec_env audit token, 625
  ioctl to clean audio device, 88
  return audit token, 629
system hardware, controlling access to, 62–63
system properties, privileges relating to, 140
system security
  access, 35–52
  changing
    user’s password, 54
  displaying
    user’s login status, 55
    users with no passwords, 56
  firewall systems, 51
  hardware protection, 35–36, 62–63
  login access restrictions, 36
  machine access, 35–36
  overview, 35–52
  password encryption, 37
  passwords, 36
  privileges, 138–146
  protecting from risky programs, 121
  restricted shell, 44, 45
  restricting remote root access, 61–62
  role-based access control (RBAC), 43, 127–130
  root access restrictions, 48, 61–62
  special accounts, 39
  su command monitoring, 43, 60
  task map, 121
UFS ACLs, 113–114
System V IPC
  ipc audit token, 627
  IPC_perm audit token, 628
  privileges, 140
system variables (Continued)
  noexec_user_stack, 122–123
  noexec_user_stack_log, 123
  rstchown, 117
/system/volatile/sshd.pid file, description, 312
  systems
    protecting from risky programs, 121
    tracking file integrity, 91–106
T
  -T option
    encrypt command, 228
    mac command, 226
  -t option, audit command, 584
T4, See SPARC T4 series
tables, gsscred, 521
tail command, example of use, 550
task maps
  administering Cryptographic Framework, 230–231
  administering policies (Kerberos), 471
  administering principals (Kerberos), 458–459
  administering Secure RPC, 326
  configuring audit logs, 565–566
  configuring auditing, 551–552
  configuring device policy, 71–72
  configuring Kerberos NFS servers, 381
  configuring RBAC, 160–161
  configuring Secure Shell, 285–286
  device allocation, 73–74
  device policy, 71–72
  Kerberos configuration, 355–356
  Kerberos maintenance, 356
  managing audit records, 585–586
  managing device allocation, 73–74
  managing device policy, 71–72
  managing RBAC, 177–178
  PAM, 267
  planning auditing, 539–546
  privileges, listing, managing, and using, 185
  protecting against programs with security risk, 121
  protecting files with cryptographic mechanisms, 219–220
  protecting files with UNIX permissions, 115
task maps (Continued)

securing logins and passwords, 53–54
troubleshooting auditing, 595–596
Using BART task map, 94
using RBAC, 149–150
using Secure Shell, 292
using the default RBAC configuration, 150
Using the Key Management Framework (Task Map), 250–251
TCP
addresses, 627
Secure Shell and, 288, 305
telnet command
Kerberos and, 500–503, 509
telnetd daemon, Kerberos and, 510
temporary audit policy
active audit policy, 558–560
setting, 560
terminal ID, audit, 620
terminology
authentication-specific, 511–512
Kerberos, 510–516
Kerberos-specific, 510–511
test manifests, BART, 92–93
text audit token, format, 630
TGS, getting credential for, 516–517
TGT, in Kerberos, 335–336
ticket file, See credential cache
ticket-granting service, See TGS
ticket-granting ticket, See TGT
tickets
creating, 491–492
creating with kinit, 492
definition, 334
definition in Kerberos, 511
destroying, 494
-F option or -f option, 501
file
See credential cache
forwardable, 334, 492, 503, 512
initial, 512
invalid, 512
-k option, 502
klist command, 493–494
tickets (Continued)
life time, 513–515
maximum renewable lifetime, 514
obtaining, 491–492
or credentials, 335
postdateable, 513
postdated, 334
proxiable, 513
proxy, 513
renewable, 513
requesting for specific realm, 502
types of, 512–516
viewing, 493–494
warning about expiration, 398
time stamps, audit files, 620
TIMEOUT in Secure Shell, 311
/tmp/krb5cc_uid file, description, 508
/tmp/ovsec_adm_xxxxx file, description, 508
TMPFS file system, security, 111
token, definition in Cryptographic Framework, 215
trail audit policy
and trailer token, 548
description, 548
trailer audit token
format, 630–631
order in audit record, 630–631
praudit display, 631
transparency, definition in Kerberos, 334
Trojan horse, 44
troubleshooting
active plugin, 597
allocating a device, 79
audit classes
customized, 564, 597
auditing, 595–596
encrypt command, 230
finding files with setuid permissions, 121
Kerberos, 447
lack of privilege, 191–192
list_devices command, 76
mounting a device, 81
praudit command, 592
preventing programs from using executable stacks, 122–123
troubleshooting (Continued)
  privilege requirements, 191–192
  remote root access, 62
  root as a role, 185
  security properties, 174–177
  terminal where su command originated, 60
  too many audit records in queue, 573
  user running privileged commands, 188–189
  truss command, for privilege debugging, 191
  trusted hosts, 51
  types of tickets, 512–516
  TZ in Secure Shell, 311

U
- U option, allocate command, 85
  UDP
    addresses, 627
    port forwarding and, 288
    Secure Shell and, 288
    using for remote audit logs, 531
  umask value
    and file creation, 111
    typical values, 111
  umount command, with security attributes, 75
  uninstalling, cryptographic providers, 240
  UNIX file permissions, See files, permissions
  unmounting, allocated devices, 81–82
  update_drv command, description, 82
  URL for online help, Graphical Kerberos Tool, 353
  use_authid option, SASL and, 319
  use of authorization audit token, 631
  use of privilege audit token, 631
  UseOpenSSLEngine keyword
    Secure Shell, 310
  UsePrivilegedPort keyword, Secure Shell, 310
  user accounts
    See also users
    changing root password, 54
    displaying login status, 55
  user_attr database
    description, 199, 200
    listing user exceptions to audit
      preselection, 555–558
  user_attr file, exceptions to system-wide audit
    classes, 529
  user audit token, 631
  user classes of files, 108
  user database (RBAC), See user_attr database
  user ID
    audit ID and, 525–526, 620
    in NFS services, 383–384
  user ID numbers (UIDs), special accounts and, 39
  User keyword, ssh_config file, 310
  user principal, description, 338
  user procedures
    adding plugins to KMF, 260–261
    allocating devices, 73–78
    assuming a role, 154–155
    chkey command, 328
    computing digest of a file, 224–225
    computing MAC of a file, 225–227
    creating self-signed certificate, 251–252
    decrypting files, 227–230
    encrypting files, 219–220
    encrypting NIS user's private key, 328
    exporting certificates, 253–254
  generating a symmetric key
    using the pktool command, 220–224
    generating passphrase for keystore, 254–255
    importing certificates, 252–253
    protecting files, 115
    using an assigned role, 154–155
    using pktool command, 250–251
    using Secure Shell, 292
  user rights management, See privileges
  User Security rights profile, modifying audit
    preselection for users, 555–558
  useradd command, description, 202
  userattr command
    description, 202
    displaying exceptions to system-wide
      auditing, 552–554
  userdel command, description, 202
  UserKnownHostsFile keyword, ssh_config file, 310
  UserKnownHostsFile2 keyword, See
    UserKnownHostsFile keyword
usermod command
    audit_flags keyword, 555–558
usermod command
    changing user’s RBAC properties, 155
description, 203
exceptions to system-wide auditing, 529
usermod command
    specifying user exceptions to audit
    preselection, 555–558
    using caret (^) prefix for audit_flags
    exception, 556–557
usermod command
    using to assign role, 165–166
users
    allocating devices, 78–79
    assigning allocate authorization to, 74–75
    assigning privileges to, 157
    assigning RBAC defaults, 201–202
    assigning rights profiles, 157
    auditing all of their commands, 600–602
    auditing individual users, 557
    authenticating to rights profile, 180–181, 182–183
    authenticating to role, 180–181, 182–183
    basic privilege set, 143
    computing digest of files, 224–225
    computing MAC of files, 225–227
    creating rights profile for a group, 558
    creating root user, 183–185
deallocating devices, 81–82
determining directly assigned privileges, 187–188
determining own privileged commands, 188–189
disabling login, 56–57
displaying login status, 55
derouting files, 227–230
exceptions to Secure Shell defaults, 289–290
generating a symmetric key, 220–224
having no passwords, 56
initial inheritable privileges, 143
modifying audit preselection mask of, 555–558
modifying properties (RBAC), 155–157
mounting allocated devices, 79–81
removing audit flags, 557–558
troubleshooting running privileged
    commands, 188–189
users (Continued)
    unmounting allocated devices, 81–82
    using rights profile, 180–181, 182–183
UseRsh keyword, ssh_config file, 310
using
    allocate command, 78–79
    BART, 93
cryptoadm command, 230
deallocate command, 81
default RBAC configuration task map, 150
device allocation, 78–79
digest command, 224–225
encrypt command, 227–230
file permissions, 115–123
mac command, 225–227
new password algorithm, 58
pktool command, 220–224, 255–259
ppriv command, 189
privileges task map, 185
RBAC defaults, 150–160
RBAC task map, 149–150
rolemod command, 180
Secure Shell task map, 292
ssh-add command, 296–297
ssh-agent daemon, 296–297
truss command, 191
umount command, 81
usermod command, 157
Using the Key Management Framework (Task
    Map), 250–251
/usr/bin/ftp command, Kerberos and, 509
/usr/bin/kdestroy command, Kerberos and, 509
/usr/bin/kinit command, Kerberos and, 509
/usr/bin/klist command, Kerberos and, 509
/usr/bin/kpasswd command, Kerberos and, 509
/usr/bin/ktutil command, Kerberos and, 509
/usr/bin/kvno command, Kerberos and, 509
/usr/bin/rcp command, Kerberos and, 509
/usr/bin/rlogin command, Kerberos and, 509
/usr/bin/rsh command, Kerberos and, 509
/usr/bin/scp command, Kerberos and, 509
/usr/bin/sftp command, Kerberos and, 509
/usr/bin/ssh command, Kerberos and, 509
/usr/bin/telnet command, Kerberos and, 509
/usr/lib/inet/proftpd daemon, Kerberos and, 510
/usr/lib/kprop command, description, 509
/usr/lib/krb5/kadm5 daemon, Kerberos and, 510
/usr/lib/krb5/kpropd daemon, Kerberos and, 510
/usr/lib/krb5/krb5kdc daemon, Kerberos and, 510
/usr/lib/krb5/kttk_warn daemon, Kerberos and, 510
/usr/lib/libsasl.so library, overview, 317
/usr/lib/ssh/sshd daemon, Kerberos and, 510
/usr/sbin/gkadmin command, description, 509
/usr/sbin/gsscred command, description, 509
/usr/sbin/in.rlogind daemon, Kerberos and, 510
/usr/sbin/in.rshd daemon, Kerberos and, 510
/usr/sbin/in.telnetd daemon, Kerberos and, 510
/usr/sbin/kadmin command, description, 509
/usr/sbin/kadmin.local command, description, 509
/usr/sbin/kclient command, description, 509
/usr/sbin/kdb5_idap_util command, description, 509
/usr/sbin/kdb5_util command, description, 510
/usr/sbin/kgcmgr command, description, 510
/usr/sbin/kproplog command, description, 510
/var/adm/auditlog file, text audit records, 577
/var/adm/messages file
    executable stack messages, 114
troubleshooting auditing, 598
/var/adm/sulog file, monitoring contents of, 60
/var/krb5/.k5.REALM file, description, 508
/var/krb5/kadmin.log file, description, 508
/var/krb5/kdc.log file, description, 508
/var/krb5/principal file, description, 508
/var/krb5/principal.kadm5 file, description, 508
/var/krb5/principal.kadm5.lock file, description, 508
/var/krb5/principal.ok file, description, 508
/var/krb5/principal.ulog file, description, 508
/var/krb5/slave_datatrans file, description, 508
/var/krb5/slave_datatrans_slave file, description, 508
/var/log/syslog file, troubleshooting auditing, 598
/var/user/$USER/krb-warn.conf file, description, 508

variables
    adding to audit record, 546, 625
    auditing those associated with a command, 624
    for proxy servers and ports, 301
    KEYBOARD_ABORT, 63
    login and Secure Shell, 310–311
    noexec_user_stack, 114
    noexec_user_stack_log, 114
    rstatchown, 117
    setting in Secure Shell, 311

verifiers
    description, 324
    returned to NFS client, 325
    window, 324

VerifyReverseMapping keyword, ssh_config file, 310

viewing
    audit record definitions, 586–587
    available cryptographic mechanisms, 233, 240
    binary audit files, 591–593
    contents of rights profiles, 197
    cryptographic mechanisms
        available, 233, 240
        existing, 232, 233, 240
        purpose, 234
    device allocation information, 75–76
    device policy, 72
    digest of a file, 225
    directly assigned privileges, 187
    existing cryptographic mechanisms, 233, 240
    file permissions, 115–116
    keylist buffer with list command, 487, 488
    list of policies, 471–473
    list of principals, 459–461
    MAC of a file, 226
    policy attributes, 473–475
    principal’s attributes, 461–463
    privilege definitions, 186–187
viewing (Continued)
  privileges, 185–193
  privileges in a shell, 187–188, 190
  privileges on a process, 189
  privileges task map, 185
  rights of initial user, 151–154
  tickets, 493–494
  user’s login status, 55
  users with no passwords, 56
  verbose listing of cryptographic mechanisms, 234
  XML audit records, 591
  your RBAC rights, 151–154
virus scanning
  configuring, 66–69
  described, 66
  engines, 65–66
  files, 65–66
  viruses
    denial of service attack, 45
    Trojan horse, 44
vnode audit token, format, 624
VSCAN Management rights profile, removing authorizations, 170–171

W
warn.conf file, description, 508
warning about ticket expiration, 398
wildcard characters
  for hosts in Secure Shell, 301
  in RBAC authorizations, 198
window verifier, 324
wr audit event modifier, 626
write permissions, symbolic mode, 112

X
X.509 v3 certificate, generating, 259–260
  -X option
    Kerberized commands, 502
X Window system, and SEAM Tool, 454–455
  -X option
    ssh command, 296
X11 forwarding
  configuring in ssh_config file, 307
  in Secure Shell, 305
X11DisplayOffset keyword, ssd_config file, 310
X11Forwarding keyword, sshd_config file, 310
X11UseLocalHost keyword, sshd_config file, 310
  -x option, Kerberized commands, 502
xauth command, X11 forwarding, 310
XAuthLocation keyword, Secure Shell port forwarding, 310
xclient audit token, 631
XML format, audit records, 592

Z
ZFS File System Management rights profile, creating audit file systems, 566–569
ZFS file systems, creating for binary audit files, 566–569
ZFS Storage Management rights profile, creating pools for audit files, 566–569
zone.max.locked-memory resource control, 141–142
zonename audit policy
  description, 548
  using, 541, 614–615
zonename audit token, 631–632
zones
  auditing and, 537, 614–615
  configuring auditing in global zone, 559
  Cryptographic Framework and, 217
  cryptographic services and, 245–246
  devices and, 41
  perzone audit policy, 537, 541, 614–615
  planning auditing in, 540–541
  zonename audit policy, 541, 614–615