Managing Kerberos in Oracle® Solaris 11.4

Part No: E61026
November 2020

ORACLE

Managing Kerberos in Oracle Solaris 11.4

Part No: E61026

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customers that have purchased support 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.

Référence: E61026

Restrictions de licence/Avis d'exclusion de responsabilité en cas de dommage indirect et/ou consécutif

Ce logiciel et la documentation qui l'accompagne 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 stipulation expresse de votre contrat de licence ou de la loi, vous ne pouvez pas copier, reproduire, traduire, diffuser, modifier, accorder de licence, 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.

Exonération de garantie

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.

Avis sur la limitation des droits

Si ce logiciel, ou la documentation qui l'accompagne, est livré sous licence au Gouvernement des Etats-Unis, ou à quiconque qui aurait souscrit la licence de ce logiciel pour le compte du Gouvernement des Etats-Unis, la notice suivante s'applique :

U.S. GOVERNMENT END USERS: Oracle programs (including any operating system, integrated software, any programs embedded, installed or activated on delivered hardware, and modifications of such programs) and Oracle computer documentation or other Oracle data delivered to or accessed by U.S. Government end users are "commercial computer software" or "commercial computer software documentation" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, reproduction, duplication, release, display, disclosure, modification, preparation of derivative works, and/or adaptation of i) Oracle programs (including any operating system, integrated software, any programs embedded, installed or activated on delivered hardware, and modifications of such programs), ii) Oracle computer documentation and/or iii) other Oracle data, is subject to the rights and limitations specified in the license contained in the applicable contract. The terms governing the U.S. Government's use of Oracle cloud services are defined by the applicable contract for such services. No other rights are granted to the U.S. Government.

Avis sur les applications dangereuses

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 un risque de 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 des applications dangereuses.

Marques

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 Inside 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 de SPARC International, Inc. AMD, Epyc, et le logo AMD sont des marques ou des marques déposées d'Advanced Micro Devices. UNIX est une marque déposée de The Open Group.

Avis d'exclusion de responsabilité concernant les services, produits et contenu tiers

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 déclinent toute responsabilité ou garantie expresse quant aux contenus, produits ou services émanant de tiers, sauf mention contraire stipulée dans un contrat entre vous et Oracle. En aucun cas, 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, sauf mention contraire stipulée dans un contrat entre vous et Oracle.

Date de publication et mention de la version préliminaire de Disponibilité Générale ("Pre-GA")

Version préliminaire de Disponibilité Générale ("Pre-GA") : 15.01.2020

Avis sur la version préliminaire de Disponibilité Générale ("Pre-GA") de la documentation

Si ce document est fourni dans la Version préliminaire de Disponibilité Générale ("Pre-GA") à caractère public ou privé :

Cette documentation est fournie dans la Version préliminaire de Disponibilité Générale ("Pre-GA") et uniquement à des fins de démonstration et d'usage à titre préliminaire de la version finale. Celle-ci n'est pas toujours spécifique du matériel informatique sur lequel vous utilisez ce logiciel. Oracle Corporation et ses affiliés déclinent expressément toute responsabilité ou garantie expresse quant au contenu de cette documentation. Oracle Corporation et ses affiliés ne sauraient en aucun cas être tenus pour responsables des pertes subies, des coûts occasionnés ou des dommages causés par l'utilisation de cette documentation.

Mention sur les informations confidentielles Oracle

INFORMATIONS CONFIDENTIELLES ORACLE. Destinées uniquement à un usage autorisé. Ne pas distribuer à des tiers.

Avis sur la reconnaissance du revenu

Si ce document est fourni dans la Version préliminaire de Disponibilité Générale ("Pre-GA") à caractère privé :

Les informations contenues dans ce document sont fournies à titre informatif uniquement et doivent être prises en compte en votre qualité de membre du customer advisory board ou conformément à votre contrat d'essai de Version préliminaire de Disponibilité Générale ("Pre-GA") uniquement. Ce document ne constitue en aucun cas un engagement à fournir des composants, du code ou des fonctionnalités et ne doit pas être retenu comme base d'une quelconque décision d'achat. Le développement, la commercialisation et la mise à disposition des fonctions ou fonctionnalités décrites restent à la seule discrétion d'Oracle.

Accessibilité de la documentation

Pour plus d'informations sur l'engagement d'Oracle pour l'accessibilité de la documentation, visitez le site Web Oracle Accessibility Program, à l'adresse : http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Accès aux services de support Oracle

Les clients Oracle qui ont souscrit un contrat de support ont accès au support électronique via My Oracle Support. Pour plus d'informations, visitez le site http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info ou le site http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs si vous êtes malentendant.

Contents

Tables

Examples

Using This Documentation

- **Overview** – Describes how to administer MIT Kerberos in Oracle Solaris.
- **Audience** – System, security, and network security administrators.
- **Required knowledge** – Access requirements and network security requirements.

Product Documentation Library

Documentation and resources for this product and related products are available at http://www.oracle.com/pls/topic/lookup?ctx=E37838-01.

Feedback

Provide feedback about this documentation at http://www.oracle.com/goto/docfeedback.

♦ ♦ ♦ CHAPTER 1

Kerberos on Oracle Solaris

This chapter introduces how Kerberos runs on Oracle Solaris. This chapter contains the following information:

What's New in Kerberos in Oracle Solaris 11.4

Kerberos in the Oracle Solaris 11.4 release is based on a recent version of MIT Kerberos. To see which version is installed, run the `klist -V` command.

Introduction to MIT Kerberos on Oracle Solaris

MIT Kerberos on Oracle Solaris takes advantage of Oracle Solaris features, such as the Image Packaging Service (IPS), SMF services, Security Extensions, and Automated Installation (AI). See also "Native Oracle Solaris Features Integrated With Kerberos" on page 27.

Comparison of MIT Kerberos and Oracle Solaris Kerberos

The following table describes the differences between MIT Kerberos and the Oracle Solaris version.

TABLE 1 Differences Between MIT Kerberos and Oracle Solaris Kerberos

MIT Kerberos Behavior	Oracle Solaris Kerberos Behavior	Difference in Oracle Solaris
Users download MIT Kerberos from the web.	Administrators install Kerberos as IPS packages.	IPS repositories provide security for data at rest and data in transit.
`k*` commands run Kerberos.	`svc*` commands run Kerberos, which is an SMF service.	Some Kerberos commands are replaced by SMF commands. See "Differences in Defaults Between MIT Kerberos and Oracle Solaris Kerberos" on page 18.
Users create scripts to configure Kerberos clients identically.	Kerberos is integrated with the Automated Install (AI) feature.	Kerberos clients can be installed automatically and identically through AI.
Users create scripts to configure Kerberos clients identically.	Oracle Solaris provides a `kclient` configuration script.	The `kclient` configuration script can configure clients similarly.
Users configure KDCs manually.	Oracle Solaris provides a `kdcmgr` configuration script.	The `kdcmgr` configuration script can configure the KDC with minimal input.
Tickets cannot be automatically renewed.	Oracle Solaris provides the `ktkt_warnd` daemon.	The `ktkt_warnd` daemon can enable automatic ticket renewal.
Relation default values can be different.	Oracle Solaris changes the default for some relations and adds relations.	Oracle Solaris changes the defaults of some Kerberos relations. See "Differences in Defaults Between MIT Kerberos and Oracle Solaris Kerberos" on page 18.

For additional information, see "Documentation About Kerberos" on page 19 and Chapter 3, "Configuring the Kerberos Service".

Differences in Defaults Between MIT Kerberos and Oracle Solaris Kerberos

SMF services for Kerberos and some relations are unique to Oracle Solaris Kerberos. Also, some relations in Oracle Solaris have different default values than the relations in MIT Kerberos.

`kadmin` service

The `svc:/network/security/kadmin:default` SMF service manages the Kerberos database administration daemon in Oracle Solaris. SMF administrative commands include `svcs` for determining the status of the service and `svcadm` for administering the service.

`krb5kdc` service

The `svc:/network/security/krb5kdc:default` SMF service manages the KDC in Oracle Solaris.

`krb5_prop` service

The `svc:/network/security/krb5_prop:default` SMF service manages the Kerberos database propagation daemon in Oracle Solaris.

`-u` permission

In the `kadm5.acl` file, allows or disallows the creation of one-component user principals whose password can be validated with PAM.

`kdc_max_tcp_connections` relation

In the `kdc.conf` file, controls the maximum number of TCP connections that the KDC allows. The minimum value is 10. If this relation is not specified, the Kerberos server allows a maximum of 30 TCP connections.

`admin_server_rotate` and `kdc_rotate` relations

In the `kdc.conf` file, enables log files to be rotated to multiple files on a schedule. The `admin_server_rotate` relation controls the `kadmin` log file and the `kdc_rotate` relation controls the `kdc` log file.

Rotation can be used to avoid logging to a file which might grow too large and halt the KDC. See the `kdc.conf(5)` man page for how to set file versions and the time interval.

`auth_to_local_realm` relation

In the `krb5.conf` file, enables non-default realms to equate with the default realm for authenticated name-to-local name mapping. Unique to Oracle Solaris.

`verify_ap_req_nofail` relation

In the `krb5.conf` file, causes credential verification to fail if the client system does not have a keytab. The default value in Oracle Solaris is `true`.

Documentation About Kerberos

Kerberos documentation for features that Oracle Solaris does not change is on the MIT Kerberos Documentation web site (`http://web.mit.edu/kerberos/krb5-1.14/doc/index.html`). This guide documents Oracle Solaris changes to default Kerberos behavior or Kerberos behaviors that are integrated with Oracle Solaris features.

Kerberos Documentation

Kerberos documentation from MIT covers the following topics:

- What is Kerberos? – Describes the Kerberos environment.

- Administrator Documentation – Includes planning; administering the Key Distribution Center (KDC), also called the database; configuring Kerberos in an LDAP environment; and so on. Includes man pages and troubleshooting. See the Table of Contents.
- User Documentation – Includes ticket and password management, configuration files, and user commands.

Other topics on the MIT Kerberos Documentation web site include developer and build information, plugins, and advanced configuration.

Oracle Solaris Documentation for Kerberos

Supplementary information or information specific to Oracle Solaris is covered in this guide in the following sections:

- "How the Kerberos Service Works" on page 20 – Discusses details about ticket handling by Kerberos.
- "Kerberos and FIPS 140-2 Mode;" on page 25 – Describes configuring Kerberos in FIPS 140-2 mode in Oracle Solaris.
- Chapter 2, "Planning for the Kerberos Service" – Describes planning issues that are specify to Oracle Solaris.
- Chapter 3, "Configuring the Kerberos Service" – Describes procedures that use Oracle Solaris features to install and configure Kerberos.
- Chapter 4, "Users Using Kerberos" – Describes Kerberos password, ticketing, and remote login considerations in an Oracle Solaris environment.
- Modified MIT Kerberos man pages – Delivered in the Kerberos IPS packages to describe Oracle Solaris-specific features of Kerberos.

How the Kerberos Service Works

This section provides an overview of the Kerberos authentication system.

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 request an NFS-mounted file), 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 NFS server. "Transparently" means that you do not need to explicitly request a ticket. The request happens when you attempt to access the server. Because only authenticated clients can get a ticket for a specific service, another client cannot access the NFS server 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 system 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 is set by *policies*, for example, to specify which users are allowed to obtain which types of ticket. Policies are determined when the Kerberos service is installed or administered.

Note - You will frequently see the terms *credential* and *ticket*. While they are often used interchangeably, technically a credential is a ticket plus the *session key* for that session.

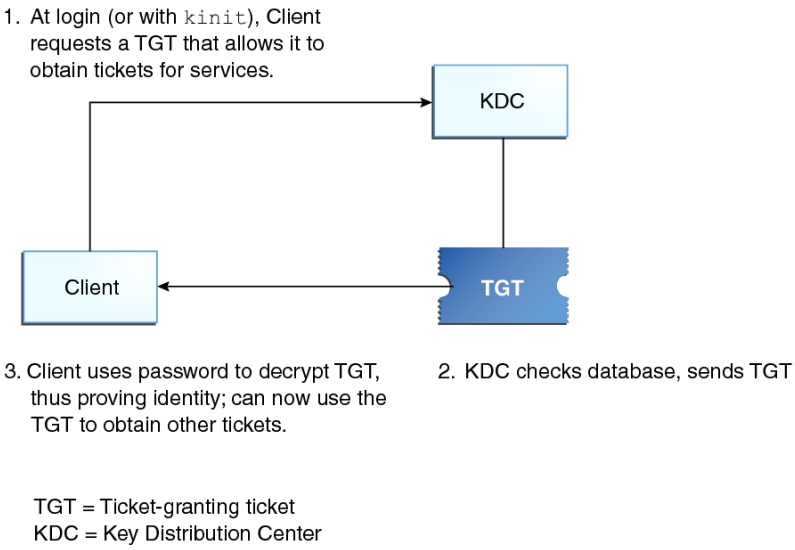
The following sections further explain the Kerberos authentication process.

Initial Authentication: the Ticket-Granting Ticket

Kerberos authentication has two phases: an initial authentication that enables all subsequent authentications, and the subsequent authentications themselves.

The following figure shows how the initial authentication takes place.

FIGURE 1 Initial Authentication for a Kerberos Session



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" (tickets), which instead of granting access to foreign countries enable you to access remote systems 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.

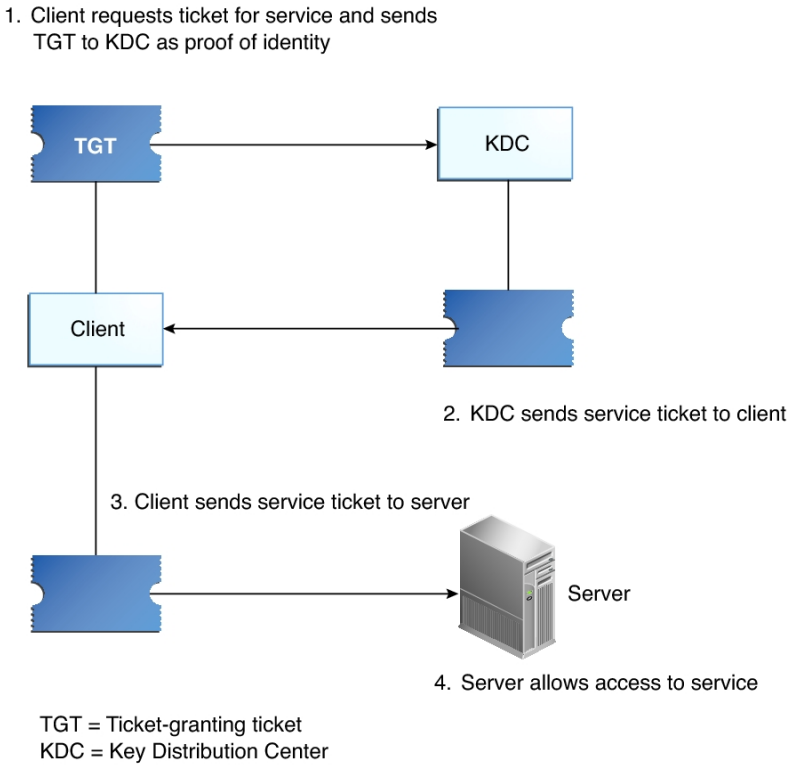
   Another analogy for the ticket-granting ticket is that of a three-day ski pass that is good at four different ski resorts. You show the pass at whichever resort you decide to go to and you receive a lift ticket for that resort as long as the pass has not expired. Once you have the lift ticket, you can 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 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 `nfs` or `ssh`, 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.

FIGURE 2 Obtaining Access to a Service Using Kerberos Authentication



1. The client requests a ticket for a particular service, for example, to log in remotely to another system, 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.

   Suppose user `jdoe` wants to access an NFS file system that has been shared with `krb5` authentication required. Because `jdoe` is already authenticated (that is, `jdoe` already has a ticket-granting ticket), as `jdoe` attempts to access the files, the NFS client system automatically and transparently obtains a ticket from the KDC for the NFS service. To use a different Kerberized service, `jdoe` 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.

Although these steps imply that the server never communicates with the KDC, the server does register itself with the KDC, just as the first client does. For simplicity's sake, that section has been omitted.

Kerberos Authentication of Batch Jobs

Batch jobs, such as cron, at, and batch, are delayed execution processes. In a Kerberos environment, all processes including delayed execution processes require credentials. However, users' credentials are relatively short-lived. By default, user credentials are valid for 8 hours and renewable for as long as a week. These times are designed to limit the exposure of sensitive keys to malicious users, but can prevent the execution of jobs at arbitrary times.

In Oracle Solaris, batch jobs that access Kerberos services can run without exposing the user's longterm key. The solution involves storing credentials that include the Kerberos service, the user name, and the client host name in a per-session user credential cache. A PAM module is used to authenticate the batch job. Which services a host can obtain tickets for can be centrally stored in the LDAP directory server.

For more information, see the `pam_krb5_keytab`(7) and `pam_gss_s4u`(7) man pages and “Configuring Delayed Execution for Access to Kerberos Services” on page 73.

Kerberos, DNS, and the Naming Service

The Kerberos service is compiled to use DNS to resolve host names. The `nsswitch` service is not checked at all to resolve host names.

Kerberos and Strong Encryption

Kerberos tickets, described in "How the Kerberos Service Works" on page 20, are encrypted. Kerberos provides several algorithms, or *encryption types*, for encrypting tickets. By default, weak types, such as `des` and `arcfour-hmac` are disallowed. These types should only be allowed for backward compatibility or interoperability. For instructions about limiting encryption to the strongest encryption types, see "How to Require Strong Encryption in Kerberos" on page 38 and the `krb5.conf(5)` man page.

Kerberos and FIPS 140-2 Mode;

You can configure Kerberos to run in FIPS 140-2 mode in Oracle Solaris. If your realm contains legacy applications or systems that are not FIPS 140-2-compliant, then the realm cannot run in FIPS 140-2 mode.

When running in FIPS 140-2 mode, Kerberos is said to be a *consumer* of the FIPS 140-2 *provider*. The provider in Oracle Solaris is the OpenSSL FIPS 140-2 provider. For instructions, see "How to Configure Kerberos to Run in FIPS 140-2 Mode" on page 38 and *Using a FIPS 140-2 Enabled System in Oracle Solaris 11.4*.

♦ ♦ ♦ CHAPTER 2

Planning for the Kerberos Service

This chapter covers several installation and configuration options that are specific to Oracle Solaris that you must resolve before you configure or deploy the Kerberos service:

Native Oracle Solaris Features Integrated With Kerberos

Kerberos uses many features that are native to Oracle Solaris, including the Image Packaging Service (IPS), Automated Installation (AI), the Service Management Facility (SMF), and privileges. Oracle Solaris Kerberos may require the use of features that are available but not required on other operating systems, such as PAM. Also, Oracle Solaris can have different defaults than MIT Kerberos. You should plan accordingly.

- Image Packaging Service (IPS) – In Oracle Solaris, MIT Kerberos software is stored in packages in your IPS repository. You install the packages from the repository rather than download the software from the MIT web site.
- Automated Installation (AI) – In Oracle Solaris, AI enables you to install your Kerberos clients identically. For pointers, see "Using Automatic Installation to Install Kerberos Clients" on page 29.
- Security Extensions – On SPARC systems, Oracle Solaris, security extensions protect the heap and the stack. For more information, see "Preventing Intentional Misuse of System Resources" in *Securing Systems and Attached Devices in Oracle Solaris 11.4*
- Pluggable Authentication Modules (PAM) – All authentication on Oracle Solaris systems calls a PAM stack.

  Oracle Solaris provides several PAM stacks that are specific to Kerberos. These stacks are likely different from PAM stacks on other UNIX systems. For more information, read

the `/etc/pam.conf` file, then list the modules in the `/etc/pam.d` and `/etc/security/pam_policy` directories and review their corresponding man pages.

- Relation defaults – See "Differences in Defaults Between MIT Kerberos and Oracle Solaris Kerberos" on page 18 for the differences.

  Oracle Solaris provides modified Kerberos man pages on your installed system. Use these pages rather than the man pages on the MIT Kerberos Documentation web site.

Planning KDCs

KDCs use specific ports, require additional servers to handle larger ticket loads, and then require propagation techniques to keep the servers synchronized. Additionally, encryption types are centrally managed. You have several options for initially configuring your KDCs.

You can configure a KDC manually as described on the MIT Kerberos Documentation web site, while using Oracle Solaris features such as PAM. Or, you can use the Oracle Solaris `kdcmgr` utility.

The `kdcmgr` utility provides a simple way to configure the KDC automatically or interactively. In the automatic version, you define the configuration parameters as options on the command line. This version is especially useful for scripts. The interactive version prompts you for all information that is needed. For pointers to the instructions for using this command, see "Configuring KDC Servers" on page 36.

You can also use LDAP to manage the database files for Kerberos. For instructions, see "Configuring KDC Servers on LDAP Directory Servers" on page 43. LDAP simplifies administration at sites that require coordination between the Kerberos databases and their existing directory server setup.

Planning for Kerberos Clients

You can use choose one of three ways to install Kerberos clients:

- Automatically – See "Using Automatic Installation to Install Kerberos Clients" on page 29.
- By script – See "Using the `kclient` Profile to Install Kerberos Clients" on page 29.
- Manually – See the MIT Kerberos Documentation web site while taking into account Oracle Solaris features such as PAM.

Client configuration planning includes which PAM module to use and whether to allow delegation of services.

- In Oracle Solaris, the PAM framework provides protected network logins with the `pam_krb5` module and other PAM modules. See "Kerberos Client Login Security" on page 30 and the `pam_krb5(7)` man page.
- When a client requests a service, that service can be granted by a server other than the master KDC. For more information, see "Trusted Delegated Services in Kerberos" on page 30.

For the procedures, see "Configuring Kerberos Clients" on page 54.

Using Automatic Installation to Install Kerberos Clients

Kerberos clients can be configured quickly and easily by using the Oracle Solaris Automated Installer (AI) feature. AI server administrators create and assign Kerberos configuration profiles to AI clients. Additionally, the AI server delivers the client keys. Therefore, at installation, the Kerberos client is a fully provisioned Kerberos system, capable of hosting secure services. Using the Automated Installer can lower system administration and maintenance costs.

You run the `kclient` command to create Kerberos configuration profiles for AI. For more information, see the `kclient(8)` man page. For instructions to configure Kerberos clients by using AI, see "How to Configure Kerberos Clients Using AI" in *Automatically Installing Oracle Solaris 11.4 Systems*.

You can use AI for clients that are not clients of an Oracle Solaris KDC. For the list of KDC vendors, see the `kclient(8)` man page.

For all KDC types, pre-generated keytab transfer is supported. Oracle Solaris KDC and MS AD also support auto-registering.

Using the `kclient` Profile to Install Kerberos Clients

In addition to AI configuration, Oracle Solaris provides the `kclient` configuration utility. This utility runs in interactive mode and noninteractive mode. Interactive mode prompts you for Kerberos-specific parameter values, so you can make changes when configuring each client. In

noninteractive mode, you supply a file with parameter values and you can supply command-line options. The `kclient` utility requires fewer steps than manual configuration and are quicker and less prone to error.

If the following setup is in effect, then no explicit configuration of your Kerberos client is necessary:

- 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 keys that are different from the KDC server's keys.

You still might want to explicitly configure the Kerberos client for the following reasons:

- The zero-configuration process performs more DNS lookups than a directly configured client, and therefore is less efficient than direct configuration.
- If referrals are not used, the zero-configuration logic depends on the DNS domain name of the host to determine the realm. This configuration 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 file. Although this requirement can be disabled in the `krb5.conf` file, doing so is not recommended for security reasons. For more information, see "Kerberos Client Login Security" on page 30 and the `krb5.conf`(5) man page.

Kerberos Client Login Security

At login, a client uses a Kerberos PAM module to verify that the KDC that issued the latest TGT is the same KDC that issued the client host principal that is stored in the `/etc/krb5/krb5.keytab` file. The PAM module must be configured in the authentication stack to verify the KDC.

For some configurations, such as DHCP clients that do not store a client host principal, this check needs to be disabled. To disable the check, see "Verifying Kerberos Clients Without a Host Principal" on page 61.

Trusted Delegated Services in Kerberos

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 can be trusted to accept delegated credentials. The `ok_to_auth_as_delegate` option to the `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 encrypted part of the KDC reply to the client can include a copy of the credential ticket flags with the `ok_to_auth_as_delegate` option set. A client can use this setting to determine whether to delegate credentials (by granting either a proxy or a forwarded TGT) to this server. When setting this option, 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.

Planning Kerberos Use of UNIX Names and 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. Also, UNIX users who do not have valid user accounts in the default Kerberos realm can be automatically migrated by using the PAM framework.

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 by using the PAM framework. Specifically, you add the `pam_krb5_migrate.so` module to the authentication stack of the PAM service. Services are then configured so that whenever a user who does not have a Kerberos principal performs a successful password login to a system, a Kerberos principal would be automatically created for that user. The new principal password is then the same as the UNIX password. For instructions about using the `pam_krb5_migrate.so` module, see "How to Configure Automatic Migration of Users in a Kerberos Realm" on page 63.

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 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 lower this default in the `libdefaults` section of the `krb5.conf` file.

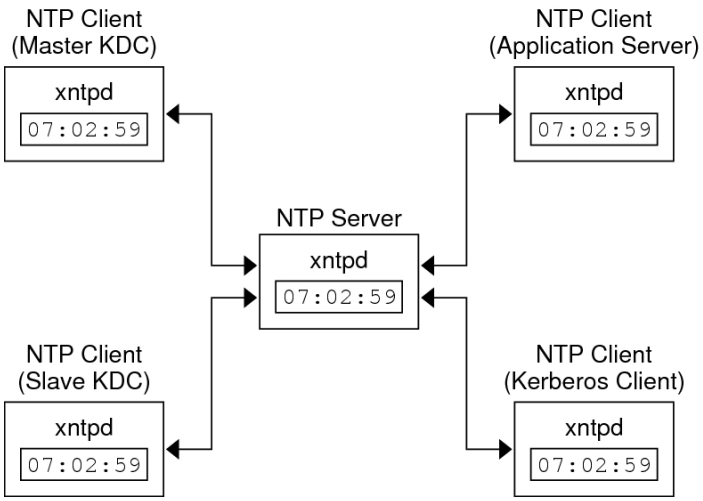
Note - For security reasons, do not increase the clock skew beyond 300 seconds.

Because maintaining synchronized clocks between the KDCs and Kerberos clients is important, use the Precision Time Protocol (PTP) or Network Time Protocol (NTP) software to synchronize the clocks. For how to configure clock synchronization in Oracle Solaris, see *Managing Clock Synchronization in Oracle Solaris 11.4*.

The NTP software is installed by default on most Oracle Solaris systems. You can install the PTP software by using the `pkg install ptp` command.

The following figure shows an example of NTP clock synchronization.

FIGURE 3 Synchronizing Clocks by Using NTP



Ensuring that the KDCs and Kerberos clients maintain synchronized clocks involves implementing the following steps:

1. Setting up a PTP or an NTP server on your Kerberos network. This server can be any system except the master KDC.
2. As you configure the KDCs and Kerberos clients on the network, make them clients of the clock synchronization server. Return to the master KDC to configure the KDC as a client of the clock synchronization server.

3. Enabling the clock synchronization service on all systems.

   For the procedures, see *Managing Clock Synchronization in Oracle Solaris 11.4*.

♦ ♦ ♦ CHAPTER 3

Configuring the Kerberos Service

This chapter provides configuration procedures for KDC servers, network application servers, NFS servers, and Kerberos clients. Many of these procedures require `root` access, so they should be performed by system administrators or advanced users. Cross-realm configuration procedures and other topics related to KDC servers are also covered.

This chapter covers the following topics:

Configuring the Kerberos Service

Because some procedures in the configuration process depend on other procedures, they must be done in a specific order. These procedures often establish services that are required to use the Kerberos service. Other procedures are not ordered, and so can be performed when appropriate. The following task map shows a suggested order for a Kerberos installation.

Note - The examples in these sections use default encryption types, which are not FIPS 140-2-validated for Oracle Solaris. To run in FIPS 140-2 mode, you must limit the encryption types to only FIPS 140-2-validated encryption types for the database, servers, and client communications. For more information, see "How to Configure Kerberos to Run in FIPS 140-2 Mode" on page 38.

TABLE 2 Task Map: Configuring the Kerberos Service

Task	Description	For Instructions
1. Plan your Kerberos installation.	Resolves configuration issues before you start the software configuration process. Planning ahead saves you time and other resources later.	Chapter 2, "Planning for the Kerberos Service"
2. Configure the KDC servers.	Configures and builds the master KDC and the slave KDC servers and KDC database for a realm.	"Configuring KDC Servers" on page 36
2a. (Optional) Configure Kerberos to run in FIPS 140-2 mode.	Enables the use of FIPS 140-2-validated algorithms only.	"How to Configure Kerberos to Run in FIPS 140-2 Mode" on page 38
2b. (Optional) Configure Kerberos to run on LDAP.	Configures the KDC to use an LDAP Directory Server.	"Configuring KDC Servers on LDAP Directory Servers" on page 43
3. Install clock synchronization software.	Creates a central clock that provides the time for all hosts on the network.	"Synchronizing Clocks Between KDCs and Kerberos Clients" on page 31
4. (Optional) Increase security on the KDC servers.	Prevents security breaches on the KDC servers.	"Restricting Access to KDC Servers" on page 77

After you have completed the required steps, perform the following procedures when appropriate:

- Configure Kerberos application servers, such as an `ftp` server – "Configuring Kerberos Network Application Servers" on page 66
- Enable a `cron` host to execute tasks at arbitrary times – "Configuring Delayed Execution for Access to Kerberos Services" on page 73
- Enable a server to share a file system that requires Kerberos authentication – "Configuring Kerberos NFS Servers" on page 69
- Enable a client to use Kerberos services – "Configuring Kerberos Clients" on page 54
- Maintain the Kerberos database – "Administering the Kerberos Database" on page 75

Configuring KDC Servers

After you install the Kerberos software, you must connect clients to existing Key Distribution Center (KDC) servers, or configure a master KDC and at least one slave KDC. KDCs issue credentials. These credentials are the basis for the Kerberos service, so the KDCs must be configured before you attempt other tasks.

Note - If you have an existing KDC, go to "Configuring Kerberos Clients" on page 54.

You can choose to configure and build the master KDC server, the database, and additional servers on Oracle Solaris with the `kdcmgr` utility or manually.

Note - For all configuration methods, you must install the KDC package from your repository as described in "How to Install the KDC Package" on page 37.

Perform the following procedures to configure KDC servers:

- Install the KDC package from your IPS repository – "How to Install the KDC Package" on page 37
- Enable the use of FIPS 140-2-validated algorithms only – "How to Configure Kerberos to Run in FIPS 140-2 Mode" on page 38
- Use a script to configure the KDCs –
  - "How to Use `kdcmgr` to Configure the Master KDC" on page 39
  - "How to Use `kdcmgr` to Configure a Slave KDC" on page 42
  - Example 1, "Running the `kdcmgr` Command Without Arguments," on page 41
- If you are configuring your KDC server manually, follow the instructions on the MIT Kerberos Documentation web site (`http://web.mit.edu/kerberos/krb5-1.14/doc/index.html`), while keeping in mind "Native Oracle Solaris Features Integrated With Kerberos" on page 27.

How to Install the KDC Package

By default, Kerberos client software is installed on your system, but the Key Distribution Center (KDC) software is not. To install a KDC, you must add the KDC package.

Before You Begin You must be assigned the Software Installation rights profile to add packages to the system. The initial user has this right as does the `root` role. For more information, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.4*.

1. **Install the KDC package.**

```
$ pkg install security/kerberos-5/kdc
```

For more information about package installation, see the `pkg`(1) man page.

2. **(Optional) List the Kerberos services.**

With the addition of the server package, your system has three Kerberos services, two for the KDC and one for the Kerberos client. These services are disabled until you configure Kerberos and then explicitly enable the services.

```
$ svcs -a krb5
STATE          STIME    FMRI
```

```
disabled       Sep_10   svc:/security/kerberos-5/krb5kdc:default
disabled       Sep_10   svc:/security/kerberos-5/krb5_prop:default
$ svcs -a | grep kerb
STATE          STIME    FMRI
disabled       Sep_07   svc:/security/kerberos-5/install:default
```

How to Require Strong Encryption in Kerberos

This procedure completely disables the use of the `arcfour-hmac` and `des3-cbc-sha1` encryption types.

Caution - This procedure breaks interoperability for deployments that join Oracle Solaris systems to domains and forests that are using weaker encryption.

1. **On the KDC, require strong encryption types for all tickets.**

   Modify the permitted encryption types in the `[libdefaults]` section of the `krb5.conf` file.

   ```
   kdc # cd /etc/krb5
   kdc # pfedit krb5.conf
        [libdefaults]
        ...
        permitted_enctypes = aes256-cts-hmac-sha1-96 aes128-cts-hmac-sha1-96
   ```

2. **On Kerberos clients, require strong encryption types for all tickets.**

   Modify the default encryption types in the `[libdefaults]` section of the `krb5.conf` file.

   ```
   Kerberos-client # cd /etc/krb5
   Kerberos-client # pfedit krb5.conf
       [libdefaults]
        ...
       default_tgs_enctypes = aes256-cts-hmac-sha1-96 aes128-cts-hmac-sha1-96
       default_tkt_enctypes = aes256-cts-hmac-sha1-96 aes128-cts-hmac-sha1-96
   ```

How to Configure Kerberos to Run in FIPS 140-2 Mode

Before You Begin For Kerberos to run in FIPS 140-2 mode, you must enable FIPS 140-2 mode on your system. See "How to Create a Boot Environment With FIPS 140-2 Enabled" in *Managing Encryption and Certificates in Oracle Solaris 11.4*.

1. **On the master KDC, edit the encryption types for the KDC.**

   In the `[realms]` section of the `kdc.conf` file, set the master key type for the KDC database:

   ```
   # pfedit  /etc/krb5/kdc.conf
   ...
   master_key_type = des3-cbc-sha1-kd
   ```

2. **In the same file, explicitly forbid other encryption types.**

   Because you can also set encryption by running a command, the configuration files should prevent the use of a non-FIPS 140-2 algorithm argument to a command.

   ```
           supported_enctypes = des3-cbc-sha1-kd:normal
   ```

3. **Edit the encryption types for transactions in the `[libdefaults]` section of the `krb5.conf` file.**

   These parameters limit the encryption types for the Kerberos servers, services, and clients.

   ```
   # pfedit /etc/krb5/krb5.conf
           default_tgs_enctypes = des3-cbc-sha1-kd
           default_tkt_enctypes = des3-cbc-sha1-kd
           permitted_enctypes = des3-cbc-sha1-kd
   ```

4. **In the same file, explicitly forbid weak encryption types.**

   ```
           allow_weak_enctypes = false
   ```

Troubleshooting For the encryption types that Kerberos recognizes, see Kerberos Encryption Types on the MIT Kerberos Documentation web site. For the encryption types that OpenSSL provides, see the documentation links at OpenSSL Cryptography and SSL/TLS Toolkit. An encryption type that is both in Kerberos and in the Oracle OpenSSL FOM 1.0 can be used to run Kerberos in FIPS 140-2 mode.

See Also For information about Oracle OpenSSL FOM 1.0, see *Using a FIPS 140-2 Enabled System in Oracle Solaris 11.4*.

▼ How to Use `kdcmgr` to Configure the Master KDC

The `kdcmgr` script provides a command-line interface to install the master and slave KDCs. For the master, you must create a password for the Kerberos database and a password for the administrator. On the slave KDCs, you must supply these passwords to complete the installation. For information about these passwords, see the `kdcmgr`(8) man page.

Before You Begin You must assume the `root` role. For more information, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.4*.

1. **Create the master KDC.**

   On the command line, run the `kdcmgr` command and name the administrator and the realm.

   You are prompted for the Kerberos database password, called the *master key* and the password for the administrative principal. The script prompts for the passwords.

```
kdc1# kdcmgr -a admin-name/admin -r DOMAIN.SUFFIX 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@DOMAIN.SUFFIX'
You will be prompted for the database Master Password.
It is important that you NOT FORGET this password.
Enter KDC database master key:     /** Type strong password **/
Re-enter KDC database master key to verify: xxxxxxxx

Authenticating as principal root/admin@DOMAIN.SUFFIX with password.
WARNING: no policy specified for admin-name/admin@DOMAIN.SUFFIX; defaulting to no
 policy
Enter password for principal "admin-name/admin@DOMAIN.SUFFIX":     /** Type strong
 password **/
Re-enter password for principal "admin-name/admin@DOMAIN.SUFFIX": xxxxxxxx
Principal "admin-name/admin@DOMAIN.SUFFIX" created.

Setting up /etc/krb5/kadm5.acl.

-------------------------------------------------
Setup COMPLETE.

kdc1#
```

   Note - Save and store these passwords in a safe location.

2. **(Optional) Display the status of the master KDC.**

```
# kdcmgr status
```

3. **Synchronize this system's clock with other clocks in the realm.**

Note - A master KDC cannot be the clock synchronization server.

For more information and pointers to procedures, see "Synchronizing Clocks Between KDCs and Kerberos Clients" on page 31. See also the krb5.conf(5) man page.

Example 1 Running the kdcmgr Command Without Arguments

In this example, the administrator supplies the realm name and admin principal when prompted by the script.

```
kdc1# kdcmgr 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 strong password **/
Re-enter KDC database master key to verify: xxxxxxxx

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 strong password **/
Re-enter password for principal "kws/admin@EXAMPLE.COM": xxxxxxxx
Principal "kws/admin@EXAMPLE.COM" created.

Setting up /etc/krb5/kadm5.acl.

-------------------------------------------------
Setup COMPLETE.

kdc1#
```

How to Use kdcmgr to Configure a Slave KDC

Before You Begin The master KDC server is configured.

You must assume the root role. For more information, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.4*.

1. Create a slave KDC.

On the command line, run the kdcmgr command and name the administrator, the realm, and the master KDC.

The script prompts for the two passwords that you created when you created the master KDC, one for the administrative principal and one for the KDC database. For the EXAMPLE.COM example, you created the passwords in Example 1, "Running the kdcmgr Command Without Arguments," on page 41.

```
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: xxxxxxxx

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: xxxxxxxx

-------------------------------------------------
Setup COMPLETE.

kdc2#
```

2. (Optional) Display the status of the KDC.

```
# kdcmgr status
```

3. **Synchronize this system's clock with other clocks in the realm.**
   For more information and pointers to procedures, see "Synchronizing Clocks Between KDCs and Kerberos Clients" on page 31. See also the `krb5.conf`(5) man page.

4. **Return to the master KDC to make it a client of the clock synchronization server.**

Configuring KDC Servers on LDAP Directory Servers

In order to configure and build a master KDC server and secondary servers on LDAP, you have to create the LDAP back end and the Kerberos KDC, and then configure Kerberos and LDAP to recognize each other. This section shows how to configure an OpenLDAP back end and an Oracle Unified Directory (OUD) back end and connect them to the KDC. It also describes how to add Kerberos attributes to the LDAP people object class and how to destroy a Kerberos realm on an LDAP directory server.

The following tasks are in this section:

Configuring a Master KDC on an OpenLDAP Directory Server

By installing the KDC and OpenLDAP on the same server you get better performance.

The main steps involved in configuring the KDC and OpenLDAP on the same server are:

1. Installing the OpenLDAP package
2. Enabling the LDAP service
3. Configuring access to the OpenLDAP server
4. Ensuring that the OpenLDAP daemon is listening on `ldapi://`
5. Adding organizational entries to the OpenLDAP server
6. Adding the OpenLDAP server to the KDC configuration file
7. Creating LDAP entries in the Kerberos database
8. Adding the KDC and `kadmin` roles to the OpenLDAP server

9. Creating the Kerberos database keys
10. Synchronizing the master KDC's clock with the clock synchronization server
11. Enabling the KDC and `kadmin` services

▼ How to Configure a Master KDC on an OpenLDAP Directory Server

This procedure configures a KDC master and an OpenLDAP server on the same system. The KDC uses the OpenLDAP client library, as will the Kerberos clients that you configure later.

Before You Begin Make sure the system is configured to use DNS. For more information about OpenLDAP, see the OpenLDAP Home Page.

You are in the `root` role. For more information, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.4*.

1. Install the `openldap` server package.

```
# pkg install service/network/ldap/openldap
```

2. Enable the OpenLDAP service.

This step enables the directory server to read the configuration file and be populated.

```
# svcadm enable ldap/server
```

3. Configure access to the OpenLDAP server.

Modify access information for the OpenLDAP configuration by creating and loading the `access.ldif` file.

```
# cat <<- EOF >access.ldif
dn: olcDatabase={1}mdb,cn=config
changetype: modify
add: olcAccess
olcAccess: {0}to dn.subtree="cn=example.com,cn=krbcontainer,dc=example,dc=com"
   by dn.base="cn=kdc service,ou=profile,dc=example,dc=com" write
   by dn.base="cn=kadmin service,ou=profile,dc=example,dc=com" write
   by * none
-
add: olcAccess
olcAccess: {1}to dn.subtree="ou=users,dc=example,dc=com"
   by dn.base="cn=kdc service,ou=profile,dc=example,dc=com" write
   by dn.base="cn=kadmin service,ou=profile,dc=example,dc=com" write
   by * none
```

```
EOF
# ldapmodify -D "cn=config" -W -f access.ldif
```

Note - In the "Providing access to" sections, `kdc service` needs `write` access to any accounts that account lockout should apply to. Write access enables the service to lock out an account after its account password has expired.

4. Ensure that the `slapd` daemon is listening on the `ldapi://` UNIX domain socket.

```
# ldapsearch -H ldapi:/// -x -b "" -s base '(objectclass=*)' namingContexts
```

5. Add organizational entries to the OpenLDAP server.

```
# cat <<- EOF >entries.ldif
dn: ou=groups,dc=example,dc=com
objectClass: top
objectClass: organizationalunit
ou: groups

dn: ou=users,dc=example,dc=com
objectClass: top
objectClass: organizationalunit
ou: users
EOF
# ldapadd -D "cn=Manager,dc=example,dc=com" -W -f entries.ldif
```

6. Add the OpenLDAP server to the Kerberos configuration file.

```
# pfedit /etc/krb5/krb5.conf
[realms]
        EXAMPLE.COM = {
                kdc = krb1.example.com
                admin_server = krb1.example.com
                database_module = LDAP
        }

[dbmodules]
        LDAP = {
                db_library = kldap
                ldap_kerberos_container_dn = "cn=krbcontainer,dc=example,dc=com"
                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_servers = ldapi:///
        }
...
```

7. Create the LDAP entries in the Kerberos database.

```
# kdb5_ldap_util -D "cn=Manager,dc=example,dc=com" create \
   -subtrees ou=users,dc=example,dc=com -r EXAMPLE.COM -s
```

For more information, see the kdb5_ldap_util(8) man page.

8. **Create and add KDC and kadmin roles.**

```
# cat <<- EOF >kdc_roles.ldif
dn: cn=kdc service,ou=profile,dc=example,dc=com
cn: kdc service
sn: kdc service
objectclass: top
objectclass: person
userpassword: nnnnnnnn

dn: cn=kadmin service,ou=profile,dc=example,dc=com
cn: kadmin service
sn: kadmin service
objectclass: top
objectclass: person
userpassword: nnnnnnnn
EOF
# ldapadd -D "cn=Manager,dc=example,dc=com" -W -f kdc_roles.ldif
```

The passwords for the kdc service and the kadmin service should be different and difficult to guess. Remember these passwords. You use them in the following step.

9. **Create stash files for LDAP binding to the KDC and kadmin services.**

```
# kdb5_ldap_util -D "cn=Manager,dc=example,dc=com" stashsrvpw \
   cn="kdc service,ou=profile,dc=example,dc=com"
Password for "cn=Manager,dc=example,dc=com": nnnnnnnn
Password for "cn=kdc service,ou=profile,dc=example,dc=com": nnnnnnnn
Re-enter password for "cn=kdc service,ou=profile,dc=example,dc=com": nnnnnnnn
# kdb5_ldap_util -D "cn=Manager,dc=example,dc=com" stashsrvpw \
   cn="kadmin service,ou=profile,dc=example,dc=com"
Password for "cn=Manager,dc=example,dc=com": nnnnnnnn
Password for "cn=kadmin service,ou=profile,dc=example,dc=com": nnnnnnnn
Re-enter password for "cn=kadmin service,ou=profile,dc=example,dc=com": nnnnnnnn
```

10. **Synchronize this system's clock with other clocks in the realm.**

Note - A master KDC cannot be the clock synchronization server.

For more information and pointers to procedures, see “Synchronizing Clocks Between KDCs and Kerberos Clients” on page 31. See also the krb5.conf(5) man page.

11. **Enable the KDC and `kadmin` services.**

```
# svcadm enable krb5kdc
# svcadm enable kadmin
```

Configuring a Master KDC on an Oracle Unified Directory Server

By installing the KDC and LDAP on the same server you get better performance.

The main steps are:

1. Installing the OUD package
2. Configuring the OUD server
3. Adding the OUD server to the Kerberos configuration file
4. Creating keys for the KDC and specifying a privileged port for the OUD servers
5. Configuring KDC roles and services on the OUD server
6. Creating and installing a certificate and keys for the OUD server
7. Testing
8. Synchronizing the master KDC's clock with the clock synchronization server

▼ How to Configure a Master KDC on an Oracle Unified Directory LDAP Directory Server

This procedure configures a KDC master and an OUD server on the same system. The KDC uses the OpenLDAP client library, as will the Kerberos clients that you configure later.

Before You Begin Ensure that the system is configured to use DNS. This procedure uses OUD for LDAP. For more information, see the Oracle Fusion Middleware Installation Guide for Oracle Unified Directory 11g Release 2 (11.1.2) .

You are in the `root` role. For more information, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.4*.

1. **Download the OUD package.**

   Follow the directions on the Downloads for Oracle Identity Management web site.

2. **Configure the OUD LDAP server.**

   See the links to OUD information on the Oracle Identity Management page.

This sample configuration uses the following parameters:

- Listener port: `1389`
- TLS port: `1636` (privileged port)
- Administrator connector port: `4444`
- Password: *nnnnnnn*
- Certificates: StartTLS and TLS
- Process: `java -server -Dorg.opends.server.scriptName=sta...`

```
# cd Oracle/Middleware/Oracle_OUD1
# export JAVA_HOME=/usr/jdk/instances/jdkversion
# ./oud-setup
```

3. **Verify that the LDAP server is listening.**

```
# ldapsearch -x -p 1389 -D "cn=directory manager" -h $HOSTNAME -b "" -s base
 objectclass='*'
```

4. **Add the initial profile entries to the OUD configuration.**

```
# pfedit profile.ldif
  dn: ou=profile,dc=example,dc=com
  ou: profile
  objectclass: top
  objectclass: organizationalUnit
# ldapmodify -a -h $HOSTNAME -D "cn=directory manager" -f profile.ldif
```

5. **Remove the newlines from all the attribute types in the `kerberos.ldif` file, then add the file to the OUD configuration.**

```
# pfedit /usr/share/lib/ldif/kerberos.ldif
# ldapmodify -p 1389 -a -h $HOSTNAME -D "cn=directory manager" \
   -f /usr/share/lib/ldif/kerberos.ldif
```

6. **Add the OUD server to the Kerberos configuration file.**

```
# pfedit /etc/krb5/krb5.conf
 [realms]
        EXAMPLE.COM = {
                kdc = krb1.example.com
                admin_server = krb1.example.com
                database_module = LDAP
        }

 [dbmodules]
        LDAP = {
```

```
                db_library = kldap
                ldap_kerberos_container_dn = "cn=krbcontainer,dc=example,dc=com"
                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 = ldap://krb1:1389
        }
...
```

7. Create the keys and stash files for LDAP binding to the KDC and `kadmin` services.

```
# kdb5_ldap_util -D "cn=directory manager" create -P nnnnnnnn -r EXAMPLE.COM -s
# 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"
```

8. Modify the `ldap_servers` entry in the Kerberos configuration file to use a privileged port.

```
# pfedit /etc/krb5/krb5.conf
       ldap_servers = ldaps://krb1:1636
```

9. Add Kerberos entries to the OUD server.

a. Create and add KDC roles.

```
# pfedit kdc_roles.ldif
 dn: cn=kdc service,ou=profile,dc=example,dc=com
 cn: kdc service
 sn: kdc service
 objectclass: top
 objectclass: person
 userpassword: nnnnnnnn

 dn: cn=kadmin service,ou=profile,dc=example,dc=com
 cn: kadmin service
 sn: kadmin service
 objectclass: top
 objectclass: person
 userpassword: nnnnnnnn

# ldapmodify -p 1389 -a -h $HOSTNAME -D "cn=directory manager" -f kdc_roles.ldif
```

b. Create and add administrative users.

```
# pfedit example.ldif
dn: dc=example,dc=com
changetype: modify
```

```
replace: aci
aci: (target ="ldap:///dc=example,dc=com")(targetattr !=
 "userPassword")(version 3.0;acl "Anonymous read-search access";
 allow (read, search, compare)(userdn = "ldap:///anyone");)
aci: (target="ldap:///dc=example,dc=com") (targetattr =
  "*")(version 3.0; acl "allow all Admin group"; allow(all) groupdn =
  "ldap:///cn=Directory Administrators,ou=Groups,dc=example,dc=com";)

dn: ou=Groups, dc=example,dc=com
objectclass: top
objectclass: organizationalunit
ou: Groups

dn: cn=Directory Administrators, ou=Groups, dc=example,dc=com
cn: Directory Administrators
objectclass: top
objectclass: groupofuniquenames
ou: Groups
uniquemember: uid=kvaughan, ou=People, dc=example,dc=com
uniquemember: uid=rdaugherty, ou=People, dc=example,dc=com
uniquemember: uid=hmiller, ou=People, dc=example,dc=com

dn: ou=People, dc=example,dc=com
objectclass: top
objectclass: organizationalunit
ou: People
aci: (target ="ldap:///ou=People,dc=example,dc=com")(targetattr =
 "userpassword || telephonenumber || facsimiletelephonenumber")(version 3.0;
 acl "Allow self entry modification";allow (write)(userdn = "ldap:///self");)
aci: (target ="ldap:///ou=People,dc=example,dc=com")(targetattr !=
 "cn || sn || uid")(targetfilter ="(ou=Accounting)")(version 3.0;
 acl "Accounting Managers Group Permissions";allow (write) (groupdn =
  "ldap:///cn=Accounting Managers,ou=groups,dc=example,dc=com");)
aci: (target ="ldap:///ou=People,dc=example,dc=com")(targetattr !=
 "cn || sn || uid")(targetfilter ="(ou=Human Resources)")(version 3.0;
 acl "HR Group Permissions";allow (write)(groupdn = "ldap:///cn=HR Managers,
 ou=groups,dc=example,dc=com
 ");)
aci: (target ="ldap:///ou=People,dc=example,dc=com")(targetattr !=
 "cn ||sn || uid")(targetfilter ="(ou=Product Testing)")(version 3.0;
 acl "QA Group Permissions";allow (write)(groupdn = "ldap:///cn=QA Managers,
 ou=groups,dc=example,dc=com");)
aci: (target ="ldap:///ou=People,dc=example,dc=com")(targetattr !=
 "cn || sn || uid")(targetfilter ="(ou=Product Development)")(version 3.0;
 acl "Engineering Group Permissions";allow (write)(groupdn = "ldap:///
 cn=PD Managers,ou=groups,dc=example,dc=com");)

dn: ou=Special Users,dc=example,dc=com
```

```
objectclass: top
objectclass: organizationalUnit
ou: Special Users
description: Special Administrative Accounts

# ldapmodify -p 1389 -a -h $HOSTNAME -D "cn=directory manager" -f example.ldif
```

c. Create and add ACLs for LDAP entries.

```
# pfedit kadmin.aci
## Set kadmin ACL for everything under krbcontainer.
dn: cn=krbcontainer,dc=example,dc=com
changetype: modify
replace: aci
aci: (target="ldap:///cn=krbcontainer,dc=example,dc=com") (targetattr="*")
 (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
replace: aci
aci: (target="ldap:///ou=people,dc=example,dc=com") (targetattr="*")
 (version 3.0; acl "kadmin_ACL"; allow (all)
 userdn="ldap:///cn=kadmin service,ou=profile,dc=example,dc=com";)

# ldapmodify -h $HOSTNAME -D "cn=directory manager" -f kadmin.aci
```

10. Generate and store the TLS certificate for the OUD server.

This set of commands also creates the key manager provider, trust manager provider, and connection handler.

```
# export LDAPHOME=~OUD-base-location/ORACLE_HOME
# export LDAPHOME=$PWD
# export LDAP_SERVER_DN=krb1.example.com
# export STORE_PASSWD=xxxxxxxx
# export LDAP_BINDPWF=$LDAPHOME/config/keystore.pin
# export LDAP_ADMIN_PORT=4444
# export LDAP_BINDDN="cn=directory manager"
# export LDAP_SERVER=krb1.example.com
# rm $LDAPHOME/config/keystore
# rm $LDAPHOME/config/truststore
# echo $STORE_PASSWD > LDAP_BINDPWF
# keytool -genkeypair -alias server-cert -keyalg rsa \
   -dname "CN=$LDAP_SERVER_DN" -keystore $LDAPHOME/config/keystore \
   -storepass $STORE_PASSWD -keypass $STORE_PASSWD
```

```
# keytool -selfcert -alias server-cert -validity 1825 \
   -keystore $LDAPHOME/config/keystore -storetype JKS -storepass $STORE_PASSWD
# keytool -list -alias server-cert -keystore $LDAPHOME/config/keystore \
   -storepass $STORE_PASSWD
# keytool -exportcert -alias server-cert -file $LDAPHOME/config/server-cert.txt \
   -rfc -keystore $LDAPHOME/config/keystore -storepass $STORE_PASSWD
# cp $LDAPHOME/config/server-cert.txt /var/ldap/certdb.pem
```

11. **Enable the key manager provider, trust manager provider, and connection handler.**

```
# ldapservercfg -X -n -h $LDAP_SERVER -p $LDAP_ADMIN_PORT -D "$LDAP_BINDDN" \
  -j $LDAP_BINDPWF set-connection-handler-prop \
  --handler-name "LDAPS Connection Handler" \
  --set key-manager-provider:JKS --set trust-manager-provider:JKS \
  --set listen-port:1636 --set enabled:true
# bin/stop-ds
```

12. **(Optional) Verify the configuration with an SSL LDAP query.**

```
# /usr/lib/openldap/bin/ldapsearch -x -v -x -D "$LDAP_BINDDN" -w $LDAP_BINDPW \
   -H ldapi://$LDAP_SERVER_DN:1636 -b "" -s base objectclass='*'
```

13. **Synchronize this system's clock with other clocks in the realm.**

Note - A master KDC cannot be the clock synchronization server.

For more information and pointers to procedures, see “Synchronizing Clocks Between KDCs and Kerberos Clients” on page 31. See also the `krb5.conf`(5) man page.

How to Mix Kerberos Principal Attributes in a Non-Kerberos Object Class Type on an OpenLDAP Server

In this procedure, the `krbprincipalaux`, and `krbTicketPolicyAux` and `krbPrincipalName` attributes are associated with the people object class.

This procedure uses the following configuration parameters:

- OpenLDAP Server = `krb1.example.com`
- User principal = `mre@EXAMPLE.COM`

Before You Begin You must assume the `root` role. For more information, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.4*.

1. **Prepare each entry in the people object class.**

   On the OpenLDAP server, repeat this step for each entry.

   ```
   cat << EOF | ldapmodify -h openldap.example.com \
   -D "cn=directory manager,dc=example,dc=com"
   dn: uid=mre,ou=people,dc=example,dc=com
   changetype: modify
   objectClass: krbprincipalaux
   objectClass: krbTicketPolicyAux
   krbPrincipalName: mre@EXAMPLE.COM
   EOF
   ```

2. **Add a subtree attribute to the realm container.**

   This example enables searching 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 the DB2 entries.**

   a. **Dump the DB2 entries.**

   ```
   # kdb5_util dump > dumpfile
   ```

   b. **Load the database into the LDAP server.**

   ```
   # kdb5_ldap_util load -update dumpfile
   ```

4. **(Optional) Add the principal attributes to the KDC.**

   ```
   # kadmin.local -q 'addprinc mre'
   ```

How to Destroy a Kerberos Realm on an LDAP Directory Server

Use this procedure if a different LDAP Directory Server is handling a realm.

Before You Begin You must assume the `root` role. For more information, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.4*.

- **Destroy the realm.**

```
# kdb5_ldap_util -D "cn=directory manager" destroy
```

Configuring Kerberos Clients

Kerberos clients include any host on the network that is not a KDC server and that provides or uses Kerberos services. This section provides procedures for installing a Kerberos client, as well as information about using `root` authentication to mount NFS file systems. For an overview of client configuration options, see "Planning for Kerberos Clients" on page 28.

The following task map describes the tasks that are covered in this section.

TABLE 3 Task Map: Configuring Kerberos Clients

Task	Description	For Instructions
Install clients by using the Automated Installer (AI).	Appropriate when you want the Kerberos client to be configured during system installation.	"How to Configure Kerberos Clients Using AI" in *Automatically Installing Oracle Solaris 11.4 Systems*
Create an installation profile for similar Kerberos clients.	Creates a kclient installation profile.	"How to Create a Kerberos Client Installation Profile" on page 55
Install clients with a script.	Appropriate when the installation parameters for each client are the same.	"How to Use a Kerberos Client Profile" on page 55
Install clients by answering prompts.	Appropriate when only a few of the installation parameters need to change.	"How to Use the kclient Utility Without an Installation Profile" on page 57
Install clients manually.	Appropriate when each client installation requires unique installation parameters.	How to Manually Configure a KDC
Join a Kerberos client to an Active Directory Server.	Automatically installs a Kerberos client of an Active Directory server.	"How to Join a Kerberos Client to an Active Directory Server" on page 60
Enable a client to access an NFS file system as the `root` user	Enables the client to mount an NFS file system with `root` access. Also, enables the client to access the NFS file system so that cron jobs can run.	"How to Access a Kerberos Protected NFS File System as the `root` User" on page 62
Enable a client without a fixed IP address to use Kerberos services.	Changes the default security requirement to verify the KDC of the client's ticket.	"Verifying Kerberos Clients Without a Host Principal" on page 61

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 profile, you reduce the likelihood of typing errors. Also, using the profile reduces user intervention as compared to the interactive process.

Note - To create systems that initially boot as fully configured Kerberos clients, see "Configuring Security" in *Installing Oracle Solaris 11.2 Systems* .

Before You Begin A KDC server is installed and configured.

The `security/kerberos-5` package must be on your system. If you installed the `security/kerberos-5/kdc` package, the `kerberos-5` package with the client features is installed.

You must assume the `root` role. For more information, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.4*.

1. **Create a `kclient` installation profile.**

   The following is a sample `kclient` profile:

```
client# pfedit kcprofile
REALM EXAMPLE.COM
KDC kdc1.example.com
ADMIN clntconfig
FILEPATH /net/denver.example.com/export/install/krb5.conf
NFS 1
DNSLOOKUP none
```

2. **Protect the file and store it for use by other clients.**

```
client# cp kcprofile /net/denver.example.com/export/install
denver# chown root kcprofile; chmod 644 kcprofile
```

How to Use a Kerberos Client Profile

Before You Begin You must assume the `root` role. For more information, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.4*.

You have access to a `kclient` profile, such as the profile in "How to Create a Kerberos Client Installation Profile" on page 55.

- **Run the `kclient` command with a profile argument.**

  You must provide the password for the `clntconfig` principal to complete the process. You created this password when you configured your master KDC. For more information, see the `kclient`(8) man page.

```
client# /usr/sbin/kclient -p /net/denver.example.com/export/install/kcprofile

Starting client setup
-------------------------------------------------

kdc1.example.com

Setting up /etc/krb5/krb5.conf.

Obtaining TGT for clntconfig/admin ...
Password for clntconfig/admin@EXAMPLE.COM: xxxxxxxx

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#
```

Example 2 Sample Use of `kclient` Utility

The following example uses the `kcprofile` client profile and two command-line overrides to configure the client.

```
# /usr/sbin/kclient -p /net/denver.example.com/export/install/kcprofile \
-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:  xxxxxxxx
```

```
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 Use the kclient Utility Without an Installation Profile

This procedure uses the `kclient` installation utility without an installation profile. If the client is to join an Active Directory server, go to "How to Join a Kerberos Client to an Active Directory Server" on page 60.

Before You Begin You must assume the `root` role. For more information, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.4*.

1. **Run the `kclient` command with no arguments.**

   ```
   client# /usr/sbin/kclient
   ```

   The script prompts you for 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

   For more information, see the `kclient`(8) man page.

2. **If the KDC server is not running an Oracle Solaris release, answer `y` and define the type of server that is running the KDC.**

   For the list of available servers, see the `-T` option in the `kclient`(8) man page.

3. **If DNS should be used for Kerberos lookups, answer `y` and indicate the DNS lookup option to use.**

   Valid options are `dns_lookup_kdc`, `dns_lookup_realm`, and `dns_fallback`. For more information about these values, see the `krb5.conf`(5) man page.

4. **Define the name of the Kerberos realm and the master KDC host name.**

   This information is added to the `/etc/krb5/krb5.conf` configuration file.

5. **If slave KDCs are in the realm, answer `y` and provide the slave KDC host names.**

   This information is used to create additional KDC entries in the client's configuration file.

6. **If service or host keys are required, answer `y`.**

   Tip - For security, all clients should have a host key. See "Verifying Kerberos Clients Without a Host Principal" on page 61.

   Service or host keys are *required* only when the client system is hosting Kerberized services.

7. **If the client is a member of a cluster, answer `y` and 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.

8. **Identify any domains or hosts to map to the current realm.**

   This mapping enables the client to recognize other domains as belonging to the client's default domain.

9. **Specify whether the client will use Kerberized NFS.**

   NFS service keys need to be created if the client will host NFS services using Kerberos.

10. **Indicate whether a new PAM policy needs to be created.**

    To set which PAM services use Kerberos for authentication, you provide the service name and a flag that indicates 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

    For information about provided PAM services for Kerberos, review the files in the `/etc/security/pam_policy` directory.

11. Specify whether the master `/etc/krb5/krb5.conf` file should be copied.

This option enables specific configuration information to be used when the arguments to `kclient` are not sufficient.

Example 3 Sample Run of the `kclient` Script

```
...
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]: y
...
Enter the Kerberos realm: EXAMPLE.COM
Specify the KDC host name for the above realm: kdc1.example.com

Note, this host and the KDC's time must be within 5 minutes of each other for
Kerberos to function. Both hosts 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/hosts to map to realm ? [y/n]: y
Enter a comma-separated list of domain/hosts to map to the default realm: corphdqtrs.
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}: gdm:first
Configuring /etc/pam.conf.

Do you want to copy over the master krb5.conf file ? [y/n]: n
No action performed.

--------------------------------------------------
Setup COMPLETE.
```

How to Join a Kerberos Client to an Active Directory Server

This procedure uses the `kclient` command without an installation profile.

Before You Begin You must assume the `root` role. For more information, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.4*.

1. **(Optional) Enable DNS resource record creation for the client.**

```
client# sharectl set -p ddns_enable=true smb
```

2. **Run the `kclient` command.**

The following output shows sample output from running the `kclient` command to join the client to the AD domain, `EXAMPLE.COM`.

The `-T` option selects a KDC server type, in this case, a Microsoft Active Directory (AD) server type. By default, you must provide the password for the Administrator principal of the AD server.

```
client# /usr/sbin/kclient -T ms_ad
Starting client setup
-------------------------------------------------

Attempting to join 'CLIENT' to the 'EXAMPLE.COM' domain.
Password for Administrator@EXAMPLE.COM: xxxxxxxx
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.
-------------------------------------------------
Setup COMPLETE.
#
```

For more information, see the `kclient`(8) man page.

Example 4 Sample Kerberos Client of a Non-Oracle Solaris KDC

A Kerberos client can be set up to work with a non-Oracle Solaris KDC by adding a line to 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 following excerpt shows the format of this line.

```
[realms]
EXAMPLE.COM = {
kdc = kdc1.example.com
kdc = kdc2.example.com
admin_server = kdc1.example.com
kpasswd_protocol = SET_CHANGE
}
```

Verifying Kerberos Clients Without a Host Principal

By default, Kerberos checks that the KDC of the host principal that is stored in the local `/etc/krb5/krb5.keytab` file is the same KDC that issued the ticket-granting ticket (TGT). This check, `verify_ap_req_nofail`, prevents DNS spoofing attacks.

However, this check must be disabled for client configurations where the host principal is unavailable. The following configurations require this check to be disabled:

- The client IP address is dynamically assigned, for example, 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.

To disable TGT verification, set the `verify_ap_req_nofail` option to `false` in the `krb5.conf` file. The `verify_ap_req_nofail` option can be entered in either the `[libdefaults]` or the `[realms]` section of the `krb5.conf` file. In the `[libdefaults]` section, the setting is used for all realms:

```
client # pfedit /etc/krb5/krb5.conf
[libdefaults]
default_realm = EXAMPLE.COM
verify_ap_req_nofail = false
...
```

If the option is in the `[realms]` section, the setting applies only to the defined realm. For more information about this option, see the `krb5.conf`(5) man page.

How to Access a Kerberos Protected NFS File System as the root User

This procedure enables a client to access an NFS file system that requires Kerberos authentication with the root principal and in particular, when the NFS file system is shared with options like: -o sec=krb5p,root=client1.example.com.

1. **Run the kadmin command.**

```
denver # /usr/sbin/kadmin -p kws/admin
Enter password: xxxxxxxx
kadmin:
```

2. **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. The second component should be 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 naming service.

```
kadmin: addprinc -randkey root/client.example.com
Principal "root/client.example.com" created.
kadmin:
```

3. **Add the root principal to the server's keytab file and quit kadmin.**

   This step is required for the client to have root access to NFS-mounted file systems. This step is also required for non-interactive root access, 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.
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 by using PAM. You customize per-system PAM configuration files on the migration server and the master server to handle the recognition of UNIX credentials and the re-authentication in the Kerberos realm.

For information about PAM, see Chapter 1, "Using Pluggable Authentication Modules" in *Managing Authentication in Oracle Solaris 11.4* and the `pam.conf`(5) man page.

In this procedure, the login service names are configured to use automatic migration. This example uses the following configuration parameters:

- 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 You must assume the `root` role. For more information, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.4*.

1. **Ensure that 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
   ...
   ```

   For information about the options to the `klist` command, see the `klist`(1) man page.

2. **If `server1` is not listed, configure it as a Kerberos client of the realm `EXAMPLE.COM`.**

   For the steps, see the examples in "Configuring Kerberos Clients" on page 54.

3. **Modify the PAM policy for `server1`.**

   For more information, see "Assigning a Per-User PAM Policy" in *Managing Authentication in Oracle Solaris 11.4*.

Note - If you are using the `account-policy` SMF stencil and the `config/etc_default_login` property is enabled, the following substeps will not work. Rather, PAM policy is determined by the value of the `login_policy/pam_policy` SMF property for `server1`. For examples of changing `account-policy` properties, see the procedures in "Modifying Rights System-Wide As SMF Properties" in *Securing Users and Processes in Oracle Solaris 11.4*. See also the `account-policy`(8S) man page.

a. Determine which Kerberos policy is in use on `server1`.

```
$ grep PAM_POLICY /etc/security/policy.conf
# PAM_POLICY specifies the system-wide PAM policy (see pam_user_policy(5))
...
PAM_POLICY=krb5_first
```

b. Copy that PAM policy file, then modify the new policy file to append the `pam_krb5_migrate.so.1` module to each authentication stack.

```
server1 # cd /etc/security/pam_policy/; cp krb5_first krb5_firstmigrate
server1 # pfedit /etc/security/pam_policy/krb5_firstmigrate
# login service (explicit because of pam_dial_auth)
...
login auth required  pam_unix_auth.so.1
login    auth optional  pam_krb5_migrate.so.1
#
# PPP service (explicit because of pam_dial_auth)
...
ppp auth required  pam_unix_auth.so.1
ppp auth optional  pam_krb5_migrate.so.1
#
# GDM Autologin (explicit because of pam_allow). ...
#
gdm-autologin auth  required    pam_unix_cred.so.1
gdm-autologin auth  sufficient  pam_allow.so.1
gdm-autologin auth optional    pam_krb5_migrate.so.1
#
# Default definitions for Authentication management
...
OTHER   auth required           pam_unix_auth.so.1
OTHER   auth optional    pam_krb5_migrate.so.1
#
# passwd command (explicit because of a different authentication module)
#
passwd  auth required           pam_passwd_auth.so.1
passwd  auth optional    pam_krb5_migrate.so.1
#
```

```
...
```

Note - PPP was removed in the Oracle Solaris 11.4 SRU 24 release.

c. (Optional) Edit the `krb5_firstmigrate` file to force an immediate password change.

For the newly created Kerberos accounts, set the password expiration time to the current time by adding the `expire_pw` option to the `pam_krb5_migrate` entries. For more information, see the `pam_krb5_migrate`(7) man page.

```
service-name auth optional     pam_krb5_migrate.so.1 expire_pw
```

d. In this policy file, modify the `OTHER` account stack to block access if the Kerberos password has expired.

```
# Definition for Account management
# Used when service name is not explicitly mentioned for account management
# Re-ordered pam_krb5 causes a Kerberos password expiration to block access
#
OTHER   account requisite   pam_roles.so.1
OTHER   account required pam_krb5.so.1
OTHER   account required pam_unix_account.so.1
OTHER   account required pam_tsol_account.so.1
## OTHER account required pam_krb5.so.1
#
...
```

e. Change the `PAM_POLICY` entry in the `policy.conf` file to use the modified configuration file.

Note - If you are using the `account-policy` SMF stencil and the `config/etc_default_login` property is enabled, editing the `policy.conf` file will not work. Rather, system-wide PAM policy is set by the value of the `login_policy/pam_policy` SMF property. For examples of changing `account-policy` properties, see the procedures in "Modifying Rights System-Wide As SMF Properties" in *Securing Users and Processes in Oracle Solaris 11.4*. See also the `account-policy`(8S) man page.

```
server1 # pfedit /etc/security/policy.conf
...
# PAM_POLICY=krb5_first
PAM_POLICY=krb5_firstmigrate
```

For more information, review the comments in the `policy.conf` file.

4. **On the master KDC, update the `kadm5.acl` access control file.**

   The following entries grant migrate and inquire privileges to the `host/server1.example.com` service principal for all users except the `root` user. Use the `U` privilege to list users who must not be migrated. These exceptions must precede the permit all or `ui` entry. For more information. see the `kadm5.acl`(5) man page.

   ```
   kdc1# pfedit /etc/krb5/kadm5.acl
   host/server1.example.com@EXAMPLE.COM U root
   host/server1.example.com@EXAMPLE.COM ui *
   */admin@EXAMPLE.COM *
   ```

5. **On the master KDC, enable the `kadmind` daemon to use the `k5migrate` PAM service.**

   If a `k5migrate` service file is not in the `/etc/pam.d` directory, add the service file to the directory. The contents are as follows:

   ```
   kdc1# cat /etc/pam.d/k5migrate
   ...
   ## Permits validation of migrated UNIX accounts
   auth    required        pam_unix_auth.so.1
   account required        pam_unix_account.so.1
   ```

   This modification enables the validation of UNIX user passwords for accounts that require migration. For more information, see the `pam.d`(5) man page.

   Note - `k5migrate` is the name of a PAM service. The file must be named `k5migrate`.

6. **Test your configuration before putting it in production.**

   - **As a regular user, test each modified PAM service.**
   - **As `root`, test each modified PAM service.**
   - **Force a password change, then test the modified PAM services.**

Configuring Kerberos Network Application Servers

Network application servers are hosts that provide access using secure network applications, such as `ftp`. Only a few steps are required to enable the Kerberos version of these applications 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 Make sure the master KDC is configured and the clocks are synchronized as described in "Synchronizing Clocks Between KDCs and Kerberos Clients" on page 31. To fully test the process, you need several clients.

You must assume the `root` role on the application server. For more information, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.4*.

1. **Determine if a host principal exists for the new server.**
   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 returns a principal, you are done. If it does not return a principal, then create new principals by using the following steps.

2. **Log in to the server with one of the `admin` principal names that you created when configuring the master KDC.**

```
boston # /usr/sbin/kadmin -p kws/admin
Enter password: xxxxxxxx
kadmin:
```

3. **Create the server's `host` principal.**

```
kadmin: addprinc -randkey host/boston.example.com
Principal "host/boston.example.com" created.
kadmin:
```

   The `host` principal is used in the following ways:

   - To authenticate traffic when using remote commands such as `ftp`.

- 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 enable the `root` user to automatically acquire a Kerberos credential without requiring that a `root` principal exist. This capability 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 host names associated with it, then create a principal for each host name using the FQDN form of the host name.

4. **Add the server's `host` principal to the server's keytab file and quit `kadmin`.**

   If the `kadmin` command is not running, restart it with a command similar to the following:
   `/usr/sbin/kadmin -p` *kws*`/admin`

   If the server has multiple host names associated with it, then add a principal to the keytab for each host name.

```
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.
kadmin: quit
```

How to Use the Generic Security Service With Kerberos When Running FTP

The generic security service (GSS) can be used by Kerberos network applications 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 "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.4*.

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: xxxxxxxx
kadmin:
```

b. Add the `ftp` service principal for the FTP server.

```
kadmin: addprinc -randkey ftp/ftpserver1.example.com
```

c. Add the `ftp` service principal to a new keytab file.

A new keytab file makes this information available to the `ftp` service without exposing all of the information in the server's keytab file.

```
kadmin: ktadd -k /etc/krb5/ftp.keytab ftp/ftpserver1.example.com
```

For more information, see the `ktadd` command in the `kadmin(1)` man page.

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.

```
# pfedit /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, you might need to use the `auth_to_local` relation or a custom `auth_to_local` plugin. Kerberos NFS servers can be protected with multiple security modes.

This section contains two procedures:

- "How to Configure Kerberos NFS Servers" on page 70

- "How to Set Up a Secure NFS Environment With Multiple Kerberos Security Modes" on page 71

▼ How to Configure Kerberos NFS Servers

This procedure uses the following configuration parameters:

- 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 "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.4*.

Make sure the master KDC is configured and the clocks are synchronized as described in "Synchronizing Clocks Between KDCs and Kerberos Clients" on page 31. To fully test the process, you need several clients.

1. **Configure the NFS server as a Kerberos client.**

   Follow the instructions in "Configuring Kerberos Clients" on page 54.

2. **Add the NFS service principal.**

   Use the `kadmin` command.

   ```
   denver # /usr/sbin/kadmin -p kws/admin
   Enter password: xxxxxxxx
   kadmin:
   ```

   a. **Create the 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 naming service.

      Repeat this step for each unique interface on the host 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 and quit `kadmin`.**

Repeat this step for each unique service principal that you created in Step 2a.

```
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.
kadmin: quit
```

3. **Share the NFS file system with Kerberos security modes.**

For more information, see "How to Set Up a Secure NFS Environment With Multiple Kerberos Security Modes" on page 71.

▼ How to Set Up a Secure NFS Environment With Multiple Kerberos Security Modes

This procedure enables an NFS server to provide secure NFS access by using several security modes. When a client negotiates a security mode with the NFS server, the client uses the first mode that is offered by the server. This mode is used for all subsequent client requests of the file system shared by that server.

Before You Begin You must assume the `root` role on the NFS server. For more information, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.4*.

1. **Verify that an NFS service principal entry is in the keytab file.**

The `klist` command reports if a keytab file exists 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 70.

```
# 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
```

For more information, see the klist(1) man page.

2. **Enable Kerberos security modes in the `/etc/nfssec.conf` file.**

   In the `/etc/nfssec.conf` file, remove the "#" that comments out the Kerberos security modes.

```
# pfedit /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.**

   - Choose `krb5p` to provide `krb5` authentication, integrity and privacy protection for confidential data transmitted over NFS. Use this mode unless it strains the server's processing resources.
   - Choose `krb5i` to provide `krb5` authentication and integrity protection in addition to the minimum protection that TCP/IP provides for NFS data.
   - Choose `krb5` for `krb5` authentication only. This security mode provides the least protection of the security modes but also has the smallest impact on the processor.

   `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) Mount a file system by using a security mode other than the default.**

   Do not perform this procedure if the default security mode is acceptable.

   - **If the automounter is being used, edit the `auto_master` database to enter a security mode other than the default.**

file-system `auto_home -nosuid,sec=`*mode*

- **Manually issue the `mount` command to access the file system by using a non-default mode.**

  `# mount -F nfs -o sec=`*mode* *file-system*

Example 5 Sharing a File System With One Kerberos Security Mode

In this example, authentication with the `krb5` security mode must succeed before any files can be accessed through the NFS service.

```
# share -F nfs -o sec=krb5p /export/home
```

Example 6 Sharing a File System With Multiple Kerberos Security Modes

In this example, all three Kerberos security modes have been selected. The mode that is used is negotiated between the client and the NFS server. If the first mode in the command fails, then the next mode is tried. For more information, see the `nfssec`(7) man page.

```
# share -F nfs -o sec=krb5p:krb5i:krb5 /export/home
```

Configuring Delayed Execution for Access to Kerberos Services

In the default Kerberos environment, credentials expire after a limited amount of time. For processes that can execute at arbitrary times, such as `cron` and `at`, the limited time presents a problem.

This procedure describes how to configure the Kerberos environment to support delayed execution processes that require authenticated services through Kerberos. Oracle Solaris provides PAM modules, uses service keys, and uses `kclient` configuration options to make delayed execution with Kerberos authentication possible and more secure than alternative solutions.

Note - If the `cron` server becomes compromised, an attacker could impersonate users to gain access to target services that are configured for the `cron` server. Therefore, consider that the `cron` host that is configured in this procedure as a more sensitive system, because it provides intermediate services for users.

How to Configure a cron Host for Access to Kerberos Services

This procedure uses the following configuration parameters:

- `cron` host = `host1.example.com`
- NFS server = `host2.example.com`
- LDAP server = `host3.example.com`

Note - Delayed execution works only with an LDAP back end.

1. **Configure the `cron` service to support Kerberos.**

   - **If the `cron` host is not configured for Kerberos, then run the `kclient` command on the system.**

     For more information, see the `kclient`(8) man page.

     For example, the following command configures the client in the `EXAMPLE.COM` realm. The command includes the `pam_gss_s4u` file in the `/etc/pam.d/cron` service file by using the `include` mechanism.

     ```
     # kclient -s cron:optional -R EXAMPLE.COM
     ```

   - **If the `cron` host is already configured for Kerberos, then you must modify the PAM configuration for the `cron` service on that host manually.**

     Ensure that the PAM configuration for the `cron` service includes the `pam_gss_s4u` file.

     ```
     # cd /etc/pam.d ; cp cron cron.orig
     # pfedit cron
           # PAM include file for optional set credentials
           # through Kerberos keytab and GSS-API S4U support
           auth include          pam_gss_s4u
     ```

2. **Enable the `cron` host to act as a delegate.**

   For example:

   ```
   # kadmin -p kws/admin
   Enter password: xxxxxxxx
   kadmin: modprinc +ok_as_delegate host/host1.example.com@EXAMPLE.COM
   Principal host/host1.example.com@EXAMPLE.COM modified.
   ```

3. **Enable the `cron` host to request tickets for itself on behalf of the user who created the `cron` job.**

```
kadmin: modprinc +ok_to_auth_as_delegate host/host1.example.com@EXAMPLE.COM
Principal host/host1.example.com@EXAMPLE.COM modified.
kadmin: quit
```

4. **In LDAP, configure the `cron` host to specify the services that it uses as a delegate.**

   For example, to enable the `cron` host to access the user's home directory on `host2`, a Kerberized NFS server, add the NFS host to the `krbAllowedToDelegateTo` parameter in the `cron` server's LDAP definition.

   a. **Create the delegate assignment.**

```
# pfedit /tmp/delghost.ldif
dn: krbprincipalname=host/host1.example.com@EXAMPLE.COM,cn=EXAMPLE.COM,
cn=krbcontainer,dc=example,dc=com
changetype: modify
krbAllowedToDelegateTo: nfs/host2.example.com@EXAMPLE.COM
```

   b. **Add the assignment to LDAP.**

```
# ldapmodify -h host3 -D "cn=directory manager" -f delghost.ldif
```

Administering the Kerberos Database

The Kerberos database is the backbone of Kerberos and must be maintained properly. This section provides some procedures for administering 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 MIT Kerberos Installation Guide.

▼ How to Convert a Kerberos Database After a Server Upgrade

If your KDC database was created on a server that was running an old release, converting the database enables you to take advantage of the improved database format.

Before You Begin Use this procedure only if the database is using an older format.

On the KDC master, you must assume the `root` role. For more information, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.4*.

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
```

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, make the `gssd` service read the `/etc/gss/gsscred.conf` file.

```
# pkill -HUP gssd
```

Now you can monitor the credential mappings as `gssd` requests them. The mappings are recorded by the `rsyslog` daemon, if the `rsyslog.conf` file is configured for the `auth` system facility with the `debug` severity level.

Increasing Security on Kerberos Servers

This section provides advice about increasing security on Kerberos application servers and on KDC servers.

Restricting 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.

- Restrict physical access to the hardware that supports the KDC.

  Make sure that the KDC server and its monitor are located in a secure facility. Regular users should not be able to access this server in any way.
- 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.

  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.

Using 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 for new credentials. Preventing the use of dictionary words 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.

Add a line to the KDC configuration file, `kdc.conf` to instruct the service to use a dictionary file. In this example, the administrator uses the dictionary that is included with the `spell` utility,

then restarts the Kerberos services. For a full description of the configuration file, see the kdc.conf(5) man page.

```
kdc1# pfedit /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
        iprop_enable = true
        iprop_master_ulogsize = 1000
        dict_file = /usr/share/lib/dict/words
     }
kdc1#
kdc1# svcadm restart -r network/security/krb5kdc
kdc1# svcadm restart -r network/security/kadmin
```

♦ ♦ ♦ CHAPTER 4

Users Using Kerberos

This chapter is intended for Kerberos users. It briefly explains Kerberos password and ticket management, and describes remote access to Kerberos applications.

Kerberos Password and Ticket Management

Kerberos is a *single sign-on* environment, which means that you type your password only once when using network applications. Kerberos authentication and encryption is built into each of a suite of existing, familiar network applications. The Kerberos V5 applications are versions of existing UNIX network applications with Kerberos features added.

Administrative Responsibilities for Kerberos Password and Ticket Management

The administrator configures Kerberos to handle user passwords and tickets.

- In Oracle Solaris, Kerberos is built into the `login` command.

  If the administrator configures the PAM service for the applicable login services, users can obtain tickets automatically. For more information, see the `pam_krb5`(7) man page.
- If the administrator configures the `ssh` command to forward copies of user tickets to the other hosts, then users do not have to explicitly ask for tickets to get access to those hosts.

  For security reasons, the administrator might prevent ticket forwarding. For more information, see the discussion about agent forwarding in the `ssh`(1) man page.

User Responsibilities for Kerberos Ticket Management

Typically, Kerberos creates a ticket for you when you log in, so you need not do anything special to obtain a ticket.

User responsibilities for Kerberos tickets include the following:

- Create a ticket if your ticket expires.

  The `kinit` command prompts you for a password, then creates the ticket.
- Create a ticket for a different principal.

  When you use a different principal besides your default principal, you might need to create a ticket. For example, you might use the `ssh -l` command to log in to a host as another user.
- Create a ticket for a new host when your tickets are not forwarded.

  If the administrator configures the `ssh` command to forward copies of your tickets to the other hosts, then you do not have to explicitly ask for tickets to get access to those hosts. For security reasons, the administrator might prevent ticket forwarding. For more information, see the discussion about agent forwarding in the `ssh`(1) man page.
- List the properties of your ticket, such as whether it can be forwarded or is invalid.

  Not all tickets are alike. For example, one ticket might be `forwardable`, another ticket might be `postdated`, and a third ticket might be both forwardable and postdated. You can list the properties of your tickets with the `klist -f` command.
- Destroy your tickets at the end of a session.

  The `kdestroy` command destroys your credential cache, which destroys all your credentials and tickets. While this destruction is not usually necessary, running `kdestroy` reduces the chance of the credential cache being compromised during times that you are not logged in.

  If you are going to be away from your system, you should either use the `kdestroy` command or lock the screen with a screen saver.

For more information, see the MIT Kerberos User Commands Documentation (`http://web.mit.edu/kerberos/krb5-1.14/doc/user/user_commands/index.html`).

User Responsibilities for Kerberos Password Management

In a Kerberos environment, you have two passwords: the regular Oracle Solaris UNIX password and a Kerberos password. You can make both passwords the same, or they can be different.

Note - The behavior of the `passwd` command depends on how the PAM module is configured. The administrator might require users to change both passwords. For some sites, the UNIX password must be changed, while other sites require the Kerberos password to change.

If PAM is properly configured, you can change your Kerberos password in two ways.

- Use the `passwd` command. With the Kerberos service configured, the `passwd` command also automatically prompts for a new Kerberos password.

  By using the `passwd` command, you can set both your UNIX and Kerberos passwords at the same time. You can also change only one password and leave the other password untouched.
- Use the `kpasswd` command. `kpasswd` changes only Kerberos passwords. You must use `passwd` if you want to change your UNIX password.

  A primary use for `kpasswd` is to change a password for a Kerberos principal that is not a valid UNIX user. For example, `jdoe/admin` is a Kerberos principal but not an actual UNIX user, so you must use `kpasswd` to change the password.

For more information, see the MIT Kerberos User Commands Documentation.

After you change your password, the password must propagate through the network. The size of the Kerberos network affects the time that is required for the propagation.

Tip - If you need 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 policy defines the criteria for passwords. The administrator configures the policy. Password character classes are lowercase, uppercase, numbers, punctuation, and all other characters.

User Remote Logins in Kerberos

When you use a Kerberized application to connect to a remote host, the application, the KDC, and the remote host perform a set of rapid negotiations. When these negotiations are completed and the application has proven your identity on your behalf to the remote host, then the remote host grants you access.

You can log in remotely by using `ssh` or `ftp`.

- After installation, the `ssh` command is the only network service in Oracle Solaris, including Kerberos, that accepts network requests. See the `ssh`(1) man page.
- The `OPTIONS` section in the `ftp`(1) man page describes the Kerberos features in the FTP application. Your administrator must configure access to FTP.

Kerberos Glossary

These glossary entries cover words that have different meanings in different parts of the operating system, or have different meanings in Oracle Solaris and Kerberos. For definitions of Kerberos components, see the documentation on the MIT Kerberos web site (`http://web.mit.edu/kerberos/`).

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 Oracle Solaris rights-based access control (RBAC), a right that can be assigned to a role or user (or as part of a rights profile) for performing a class of operations that are otherwise prohibited by security policy. Authorizations are enforced at the user application level, not in the kernel.

instance 1. In Kerberos, 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.

2. In Oracle Solaris, a specific service of a class of SMF services. For example, the `compliance:default` instance and the `compliance:generate-guide` instance are separate instances of the `compliance` SMF service.

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.

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.

See also Kerberos policy.

privilege In general, a power or capability to perform an operation on a computer system that is beyond the powers of a regular user. A privileged user or privileged application is a user or application that has been granted additional rights.

1. In Kerberos, a right granted to a principal by an entry in the `kadm5.acl` file.

2. In Oracle Solaris, one of around one hundred discrete kernel rights that can be granted to a user or a process to allow the performance of an action.

relation In Kerberos, a configuration variable or relationship that is defined in the `kdc.conf` or `krb5.conf` files. In the Oracle Solaris OS, relations are typically called *variables* or *keyword-value pairs*.

service 1. In Kerberos, a resource that is provided to network clients, often by more than one server. For example, if you `ssh` to the `central.example.com` host, then that host is the server that provides the `ssh` service.

2. In Oracle Solaris, a program that is managed by the System Management Facility (SMF) as a service. Services can be enabled, disabled, refreshed, and restarted through SMF commands. The status of services is constantly monitored and logged for ease in tracking and troubleshooting. In Oracle Solaris, the Kerberos client is the `kadmin` service and the KDC is two services, `krb5kdc` and `krb5_prop`.

Index

F

I

K

L

U

V