Securing Systems and Attached Devices in Oracle Solaris 11.4

E61021-03
September 2023

Securing Systems and Attached Devices in Oracle Solaris 11.4,

E61021-03

Primary Author: Cathleen Reiher

Contents

2 Protecting Oracle Solaris System Integrity

Index

Using This Documentation

- **Overview** – Describes how to control and monitor access to an Oracle Solaris system. Also shows how to use antivirus software.
- **Audience** – System administrators responsible for implementing security on the corporate network.
- **Required knowledge** – Familiarity with security concepts and features that are supported in Oracle Solaris.

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.

1

Managing Computer System Security

Keeping a computer system safe from tampering and malware is an important system administration responsibility. This chapter provides overview information about managing security for your computer systems.

What's New in Securing Systems and Devices in Oracle Solaris 11.4

This section highlights information for existing customers about important new features in this release that secure systems and devices from unauthorized access.

For new password and login features, see the following:

- Password Parameters
- Two-Factor Authentication With OTP and Smart Cards
- For using the `account-policy` SMF service to track system security values rather than files in the `/etc` directory, see the `account-policy`(8S) man page and Modifying Rights System-Wide As SMF Properties in *Securing Users and Processes in Oracle Solaris 11.4*.
- For changes in verified boot, see SPARC: Firmware Upgrade for Verified Boot
- For changes in security extensions, see:
  - Preventing Process Heap Corruption Using adiheap
  - ADI-Based Stack Protection Using adistack
  - Enabling Inheritance of Security Extension Status
  - Specifying Per-Object Security Extensions

Controlling Access to a Computer System

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

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

Maintaining Physical Security

To control access to your system, you must maintain the physical security of your computing environment. For instance, a system that is logged in and left unattended is vulnerable to unauthorized access. An intruder can gain access to the operating system and to the network. The computer's surroundings and the computer hardware must be physically protected from unauthorized access.

You can protect a SPARC system from unauthorized access to the hardware settings. Use the `eeprom` command to require a password to access the PROM. For more information, see How to Require a Password for SPARC Hardware Access. To protect x86 hardware, consult the vendor documentation.

Controlling Access to Boot Processes

Oracle Solaris provides two technologies to control access to boot processes.

- Verified Boot – Allows only signed boot and kernel software to run on the system.

  The value of the `boot_policy` property controls verified boot. The policy includes checking the `bootblk` and the loading of all kernel modules, including `unix` and `genunix`.

  The policy settings are stored in a Service Processor (SP), such as Oracle ILOM, Fujitsu SPARC M12, or Fujitsu M10 XSCF. The SP manages the hardware platform. For security reasons, the policy settings are purposely stored outside of the booted Oracle Solaris environment.

  For further information, see Policy for Verified Boot.
- Trusted Platform Module (TPM) – Dedicated microcontroller that provides cryptographic functions to secure a system. TPM provides a cryptographic keystore and records hashes of firmware and software that are used to boot the system.

Controlling Access to USB Ports

Some Oracle platforms, such as the SPARC T7 platform, include external USB ports. Devices such as system controllers, power distribution units (PDUs), and network switches can use these USB connections. System administrators must protect their systems from attacks that exploit USB connections. Oracle Integrated Lights Out Manager (ILOM) can deny or limit system access through external USB ports.

For more information, see Using ILOM to Prevent Access to USB Ports.

Controlling Logins

You can prevent unauthorized logins to a system or the network through password assignment and login control. A password is a simple authentication mechanism. All accounts on a system must have a password. An account without a password makes

your entire network accessible to an intruder who guesses a user name. A strong password algorithm protects against brute force attacks.

When a user logs in to a system, the `login` command checks the appropriate naming service or directory service database according to the information in the name switch service, `svc:/system/name-service/switch`. To change values in a naming service database, you use the SMF commands. The naming services indicate the location of the databases that affect login:

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

For a description of the naming service, see the `nscd`(8) man page. For information about naming services and directory services, see *Working With Oracle Solaris 11.4 Directory and Naming Services: DNS and NIS* and *Working With Oracle Solaris 11.4 Directory and Naming Services: LDAP*.

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

PAM modules can streamline logging in to applications after a successful system login. For more information, see Chapter 1, Using Pluggable Authentication Modules in *Managing Authentication in Oracle Solaris 11.4*.

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

Managing Password Information

When users log in to a system, they must supply both a user name and a password. Although logins are publicly known, passwords must be kept secret. Passwords should be known only to each user.

Your organization should have a password policy that follows industry standards. Users must choose their passwords carefully and follow your site's password policy.

You create a user's initial password when you set up the user's account. You can disable a user account by locking the password. For more information, review the following:

- Chapter 1, About User Accounts and User Environments in *Managing User Accounts and User Environments in Oracle Solaris 11.4*
- Passwords and Password Policy in *Oracle Solaris 11.4 Security and Hardening Guidelines*
- `passwd`(1)

Local Passwords

If your network uses local files to authenticate users, the password information is kept in the system's `/etc/passwd` and `/etc/shadow` files. The user names and other information are

kept in the `/etc/passwd` file. The encrypted passwords themselves are kept in a separate *shadow* file, `/etc/shadow`. This security measure prevents a user from gaining access to the encrypted passwords. While the `/etc/passwd` file is available to anyone who can log in to a system, only the `root` account can read the `/etc/shadow` file. You can use the `passwd` command to change a user's password on a local system.

NIS Passwords

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

LDAP Passwords

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

Password policy is enforced on the Oracle Directory Server Enterprise Edition. Specifically, the client's `pam_ldap` module follows the password policy controls that are enforced on Oracle Directory Server Enterprise Edition. For more information, see LDAP Naming Service Security Model in *Working With Oracle Solaris 11.4 Directory and Naming Services: LDAP*.

Password Parameters

The Oracle Solaris 11.4 release includes changes to the default password length and supported password hashes, and adds duration parameters. The legacy method of viewing and changing password parameter information in files is replaced by the SMF service, `account-policy`.

> **Note:**
>
> If you are using the `account-policy` SMF stencil and the `config/etc_default_passwd` property is enabled, review 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.

The following list of password attributes in the `account-policy` SMF stencil indicate the password parameters that are configurable:

```
password/history count
password/value_authorization astring solaris.account.setpolicy
password/aging_defaults/max_days count
password/aging_defaults/min_days count
password/aging_defaults/warn_days count
password/complexity/max_repeats count
password/complexity/min_alpha count
password/complexity/min_diff count
password/complexity/min_digit count
```

```
password/complexity/min_lower count
password/complexity/min_nonalpha count
password/complexity/min_special count
password/complexity/min_upper count
password/complexity/namecheck boolean
password/complexity/passlength count
password/complexity/whitespace boolean
password/crypt/algorithms_allow astring 2a 5 6
password/crypt/algorithms_deprecate astring
password/crypt/default astring 5
password/dictionary/db_dir astring
password/dictionary/min_word_length count
password/dictionary/word_list astring
```

Passwords can no longer be less than 8 characters in length. You can modify the password length for users.

For an example, see Modifying Password Policy in *Securing Users and Processes in Oracle Solaris 11.4*.

Password Hashes

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

> **Note:**
>
> To be FIPS 140-2 approved, use the SHA algorithms. For information, see passwd Command as a FIPS 140-2 Consumer in *Using a FIPS 140-2 Enabled System in Oracle Solaris 11.4*.

Password Algorithm Identifiers

You can specify the algorithms configuration for your site by enabling the `config/etc_default_passwd` property in the `account-policy` SMF stencil. For more information, review 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.

You indicate the algorithms by their identifier, as shown in the following table. For the identifier-algorithm mapping, see the `/etc/security/crypt.conf` file.

> **Note:**
>
> Use FIPS 140-2 approved algorithms when possible. For a list of FIPS 140-2 approved algorithms, see FIPS 140-2 Algorithm Lists and Certificate References for Oracle Solaris Systems in *Using a FIPS 140-2 Enabled System in Oracle Solaris 11.4*.

Table 1-1 Password Hashing Algorithms

Identifier	Description	Algorithm Man Page
1	The MD5 algorithm that is compatible with MD5 algorithms on BSD and Linux systems.	crypt_bsdmd5(7)
2a	The Blowfish algorithm that is compatible with the Blowfish algorithm on BSD systems. **To promote FIPS 140-2 security, remove the Blowfish algorithm (2a) from `password/crypt/algorithms_allow`.**	crypt_bsdbf(7)
md5	The Sun MD5 algorithm, which is considered stronger than the BSD and Linux version of MD5.	crypt_sunmd5(7)
5	The SHA256 algorithm. SHA stands for Secure Hash Algorithm. This algorithm is a member of the SHA-2 family. SHA256 supports 255-character passwords. This algorithm is the default, (CRYPT_DEFAULT).	crypt_sha256(7)
6	The SHA512 algorithm.	crypt_sha512(7)
__unix_ _	Deprecated. The traditional UNIX encryption algorithm. This algorithm can be of use when connecting to old systems.	crypt_unix(7)

Note:

The algorithm that is used for a user's initial password continues to be used for new password generation for that user even though a different default algorithm might have been selected prior to generating a new password for that user. This mechanism applies under the following conditions:

- The algorithm is included in the list of allowed algorithms to be used for password encryption.
- The identifier is not _unix_.

For procedures describing how to switch algorithms for password encryption, see Changing the Default Algorithm for Password Encryption.

Password Hashes Configuration

Note:

If you are using the account-policy SMF stencil and the config/etc_default_passwd property is enabled, you change password configuration in SMF. For more information, see 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.

The account-policy service has three parameters that affect password hashes:

```
password/crypt/algorithms_allow astring 2a 5 6
password/crypt/algorithms_deprecate astring
password/crypt/default astring 5
```

When you change the value for `password/crypt/default`, the passwords of new users are encrypted with the algorithm that is associated with the new value.

When existing users change their passwords, the way their old password was encrypted affects which algorithm is used to encrypt the new password. For example, assume that the administrator has changed the password parameters to `CRYPT_ALGORITHMS_ALLOW=1,2a,md5,5,6` and `password/crypt/default=6`. The following table shows which algorithm would be used to generate the encrypted password. The password consists of *identifier=algorithm*.

Initial Password	Changed Password	Explanation
`1` = `crypt_bsdmd5`	Uses same algorithm	The `1` identifier is in the `CRYPT_ALGORITHMS_ALLOW` list. The user's password continues to be encrypted with the `crypt_bsdmd5` algorithm.
`2a` = `crypt_bsdbf`	Uses same algorithm	The `2a` identifier is in the `CRYPT_ALGORITHMS_ALLOW` list. Therefore, the new password is encrypted with the `crypt_bsbdf` algorithm.
`md5` = `crypt_md5`	Uses same algorithm	The `md5` identifier is in the `CRYPT_ALGORITHMS_ALLOW` list. Therefore, the new password is encrypted with the `crypt_md5` algorithm.
`5` = `crypt_sha256`	Uses same algorithm	The `5` identifier is in the `CRYPT_ALGORITHMS_ALLOW` list. Therefore, the new password continues to be encrypted with the `crypt_sha256` algorithm.
`6` = `crypt_sha512`	Uses same algorithm	The `6` identifier is s the value of `CRYPT_DEFAULT`. Therefore, the new password continues to be encrypted with the `crypt_sha512` algorithm.
`__unix__` = `crypt_unix`	Uses `crypt_sha512` algorithm	The `__unix__` identifier is not in the `CRYPT_ALGORITHMS_ALLOW` list. Therefore, the `crypt_unix` algorithm cannot be used. The new password is encrypted with the `CRYPT_DEFAULT` algorithm.

For more information about configuring the algorithm choices, see the `account-policy`(8S) man page. To specify password encryption algorithms, see Changing the Default Algorithm for Password Encryption.

Special System Accounts

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

> **Note:**
>
> UUCP was removed in the Oracle Solaris 11.4 SRU 12 release.

> **Caution:**
>
> Never change the password setting of a system account. System accounts from Oracle Solaris are delivered in a safe and secure state. Do not revise or create system files with a UID that is 101 or less.

The following table lists some system accounts and their uses. The system accounts perform special functions. Each account on this list has a UID that is less than 100. For a full listing of system files, use the command `logins -s`.

Table 1-2 Selected System Accounts and Their Uses

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

Two-Factor Authentication With OTP and Smart Cards

Oracle Solaris supports smart cards and one-time passwords (OTP). These technologies require the user to provide two forms of identification. The first form is the UNIX user name and password. The second is a smart card and PIN, or a mobile authenticator and a OTP. See Chapter 3, Using Smart Cards for Multifactor Authentication in Oracle Solaris in *Managing Authentication in Oracle Solaris 11.4* and Chapter 4, Using One-Time Passwords for Multifactor Authentication in Oracle Solaris in *Managing Authentication in Oracle Solaris 11.4*.

Remote Logins

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

By default, remote logins cannot gain control or read certain system devices, such as the system mouse, keyboard, frame buffer, or audio device. For more information, see the `logindevperm`(5) man page.

Controlling Access to System Resources

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

Using the Secure by Default Configuration

By default, when Oracle Solaris is installed, a large set of network services are disabled. This configuration is called "Secure by Default" (SBD). With SBD, the only network service that accepts network requests is the `sshd` daemon. All other network services are disabled or handle local requests only. To enable individual network services, such as `ftp`, you use the Service Management Facility (SMF) feature of Oracle Solaris. For more information, see the `smf`(7) man page.

Preventing Intentional Misuse of System Resources

Malware can target memory, process heaps, buffers, and other vulnerable areas in the kernel. Oracle Solaris provides a Security Extensions Framework to protect applications from malware. Security extensions provide system-level security defenses, although applications can explicitly opt in and opt out of each particular defense.

As the name implies, the Security Extensions Framework is designed to counter malicious attacks on hardware or kernel software by extending security to specific kernel objects and hardware objects. In all cases, the framework allows the administrator to decide whether the system's environment is enough at risk to warrant enabling the extension to mitigate attacks on the at-risk objects. Note that some mitigations by the framework could result in slower performance.

For more information, see Protecting Against Malware With Security Extensions and the `sxadm`(8) man page.

Limiting and Monitoring Superuser Access

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

Configuring Role-Based Access Control to Replace Superuser

Role-based access control (RBAC), a feature of Oracle Solaris, distributes the capabilities of superuser to administrative roles. Roles get these capabilities through bundles called rights profiles. Regular users can also be assigned rights profiles.

Superuser, the `root` user, has access to every resource in the system. With RBAC, you can replace many of `root`'s responsibilities with a set of roles with discrete powers. For example, you can set up one role to handle user account creation and another role to handle system file modification. Although you might not modify the `root` account, you can leave the account

as a role, then not assign the role. This strategy effectively removes `root` access to the system.

Each role requires that a known user log in with her or his user name and password. After logging in, the user then assumes the role with a specific role password. If you assign rights profiles directly to a user, the user must open a profile shell to gain administrative powers. For more information about RBAC, see User Rights Management in *Securing Users and Processes in Oracle Solaris 11.4*.

Preventing Unintentional Misuse of System Resources

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

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

Setting the `PATH` Variable

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

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

Assigning a Restricted Shell to Users

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

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

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

The restricted shell enables you to limit a user's ability to stray into system files. The shell creates a limited environment for a user who needs to perform specific tasks. The restricted shell is not completely secure, however, and is intended only to keep unskilled users from inadvertently doing damage.

For information about the restricted shell, use the `man -s8 rsh` command to see the `rsh`(8) man page.

Restricting Access to Data in Files

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

You might want to allow some users to read some files, and give other users permission to change or delete some files. You might have some data that you do not want anyone else to see. Chapter 1, Controlling Access to Files in *Securing Files and Verifying File Integrity in Oracle Solaris 11.4* discusses how to set file permissions.

Restricting `setuid` Executable Files

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

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

For more information, see Protecting Executable Files From Compromising Security in *Securing Files and Verifying File Integrity in Oracle Solaris 11.4*. For procedures, see Protecting Against Programs With Security Risk in *Securing Files and Verifying File Integrity in Oracle Solaris 11.4*.

Using Resource Management Features

Oracle Solaris software provides sophisticated resource management features. Using these features, you can allocate, schedule, monitor, and cap resource use by applications in a server consolidation environment. The resource controls framework enables you to set constraints on system resources that are consumed by processes. Such constraints help to prevent denial-of-service attacks by a script that attempts to flood a system's resources.

With these resource management features, you can designate resources for particular projects. You can also dynamically adjust the resources that are available. For more information, see *Administering Resource Management in Oracle Solaris 11.4*.

Using Oracle Solaris Zones

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

Oracle Solaris zones are ideal for environments that place several applications on a single server. For more information, see *Introduction to Oracle Solaris Zones*.

Auditing System Use

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

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

With this kind of knowledge, you can use the available tools to audit system use and monitor the activities of individual users. Monitoring is very useful when a breach in security is suspected. For more information about the audit service, see Chapter 1, About Auditing in Oracle Solaris in *Managing Auditing in Oracle Solaris 11.4*.

Monitoring Compliance

As a system administrator, you need to monitor that your systems comply with site security requirements. The `compliance` command can verify that one or more systems comply with the security profile that is set for those systems. For more information, see *Oracle Solaris 11.4 Compliance Guide*. Compliance offers a more complete check than BART, as described in Monitoring File Integrity.

Controlling Access to Files

Oracle Solaris is a multiuser environment in which all the users who are logged in to a system can read files that belong to other users. With the appropriate file permissions, users can also use files that belong to other users. For more discussion, see Chapter 1, Controlling Access to Files in *Securing Files and Verifying File Integrity in Oracle Solaris 11.4*. For step-by-step instructions on setting appropriate permissions on files, see Protecting Files in *Securing Files and Verifying File Integrity in Oracle Solaris 11.4*.

Encrypting Files on Disk

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

For more information about ZFS file systems, see Encrypting ZFS File Systems in *Managing ZFS File Systems in Oracle Solaris 11.4*.

The Cryptographic Framework provides `digest`, `mac`, and `encrypt` commands. Regular users can use these commands to protect files and directories. For more information, see Chapter 1, About Cryptographic Providers in Oracle Solaris in *Managing Encryption and Certificates in Oracle Solaris 11.4*.

Using Access Control Lists

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

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

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

To protect ZFS files with access control lists (ACLs), see Setting ACLs on ZFS Files in *Securing Files and Verifying File Integrity in Oracle Solaris 11.4*. For information about using ACLs on legacy file systems, see Using Access Control Lists to Protect UFS Files in *Securing Files and Verifying File Integrity in Oracle Solaris 11.4*.

Sharing Files Across Systems

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

When you create an NFS share of a ZFS file system, the file system is permanently shared until you remove the share. SMF automatically manages the share when the system is rebooted. For more information, see Oracle Solaris ZFS Features in *Managing ZFS File Systems in Oracle Solaris 11.4*.

Restricting `root` Access to Shared Files

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

An NFS server can grant `root` access to a shared file system on a per-host basis. To grant these privileges, use the `root=` *hostname* option to the `share` command. You should use this option with care. For a discussion of security options with NFS, see Chapter 5, Commands for Managing Network File Systems in *Managing Network File Systems in Oracle Solaris 11.4*.

Assigning Labels to Files

Oracle Solaris enables you to configure systems that enforce company security policy in software by using labels. You can use provided labels or customize the labels to display your organization's security phrases, such as Confidential - Internal Only. An Oracle Solaris label policy enables you to assign these labels to existing or new file systems that contain sensitive data, and assign a set of trusted users the ability to access the files based on the users' clearances.

For information, see Chapter 3, Labeling Files for Data Loss Protection in *Securing Files and Verifying File Integrity in Oracle Solaris 11.4*.

For other file security options, see Controlling Access to Files.

Monitoring File Integrity

As a system administrator, you need assurance that the files that were installed on the systems that you administer have not changed in unexpected ways. In large installations, a comparison and reporting tool about the software stack on each of your systems enables you to track your systems. The Basic Audit Reporting Tool (BART) enables you to comprehensively validate systems by performing file-level checks of one or more systems over time. Changes in a BART manifest across systems, or for one system over time, can validate the integrity of your systems. BART provides manifest creation, manifest comparison, and rules for scripting reports. For more information, see Chapter 4, Verifying File Integrity by Using BART in *Securing Files and Verifying File Integrity in Oracle Solaris 11.4*.

Controlling Access to Devices

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

Oracle Solaris software provides several methods of controlling access to devices.

- **Set device policy –** You can require that the process that is accessing a particular device be run with a set of privileges. Processes without those privileges cannot use the device. At boot time, Oracle Solaris software configures device policy. Third-party drivers can be configured with device policy during installation. After installation, you as the administrator can add device policy to a device.
- **Make devices allocatable –** You can require that a user must allocate a device before use. Allocation restricts the use of a device to one user at a time. You can further require that the user be authorized to use the device.
- **Prevent devices from being used –** You can prevent the use of a device, such as a microphone, by any user on a computer system. For example, a computer kiosk might be a good candidate for making certain devices unavailable for use.
- **Confine a device to a particular zone –** You can assign the use of a device to a non-global zone. For more information, see Device Use in Non-Global Zones in *Creating and Using Oracle Solaris Zones*.

Device Policy

The device policy mechanism enables you to specify that processes that open a device require certain privileges. Devices that are protected by device policy can only be accessed by processes that are running with the privileges that the device policy specifies. Oracle Solaris provides default device policy. For example, network interfaces such as `bge0` require that the processes that access the interface be running with the `net_rawaccess` privilege. The requirement is enforced in the kernel. For more information about privileges, see Process Rights Management in *Securing Users and Processes in Oracle Solaris 11.4*.

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

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

For more information about device policy, see the following:

- Configuring Device Policy Task Map
- Device Policy Commands
- Privileges and Devices in *Securing Users and Processes in Oracle Solaris 11.4*

Device Allocation

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

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

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

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

For more information about device allocation, see the following:

- Managing Device Allocation Task Map
- Device Allocation
- Device Allocation Commands

Controlling Network Access

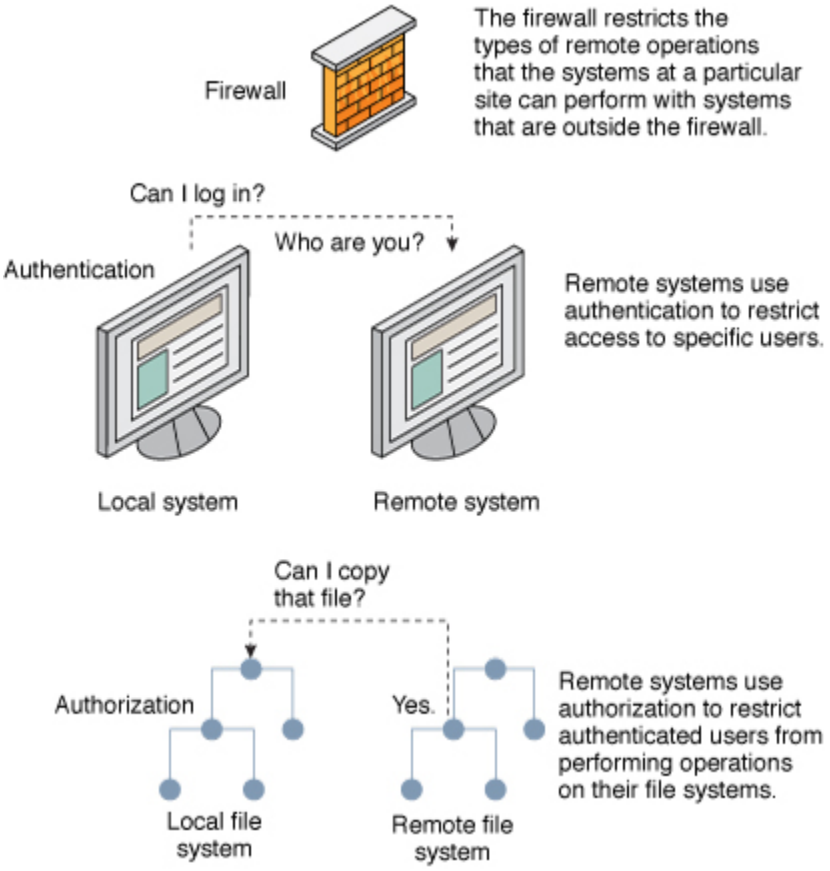
Computers are often part of a network of computers that allows connected computers to exchange information. Networked computers can access data and other resources from other computers on the network. Although computer networks create a powerful and sophisticated computing environment, networks also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

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

Network Security Mechanisms

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

Security Restrictions for Remote Operations



Authentication and Authorization for Remote Access

Authentication is a way to control access when users try to access a remote system. Authentication can be set up at both the system level and the network level. After a user has gained access to a remote system, *authorization* is a way to restrict operations that the user can perform. The following table lists the services that provide authentication and authorization.

Table 1-3 Authentication Services for Remote Access

Service	Description	For More Information
IPsec	IPsec provides host-based and certificate-based authentication and network traffic encryption.	Chapter 6, About IP Security Architecture in *Securing the Network in Oracle Solaris 11.4*
Kerberos	Kerberos uses encryption to authenticate and authorize a user who is logging in to the system.	For an example, see How the Kerberos Service Works in *Managing Kerberos in Oracle Solaris 11.4*.
LDAP	The LDAP directory service can provide both authentication and authorization at the network level.	*Working With Oracle Solaris 11.4 Directory and Naming Services: DNS and NIS*
Remote login commands	Remote login commands enable users to log in to a remote system over the network and use its resources. The `ssh` command is enabled by default. If you are a trusted host, authentication is automatic. Otherwise, you are asked to authenticate yourself.	Chapter 3, Accessing Remote Systems in *Managing Remote Systems in Oracle Solaris 11.4*
SASL	The Simple Authentication and Security Layer (SASL) is a framework that provides authentication and optional security services to network protocols. Plugins enable you to choose an appropriate authentication protocol.	About SASL in *Managing Authentication in Oracle Solaris 11.4*
Secure NFS	MIT Kerberos V supports a secure NFS environment by protecting communications for integrity, privacy, and authentication.	*Managing Kerberos in Oracle Solaris 11.4*
Secure Shell	Secure Shell encrypts network traffic over an unsecured network. Secure Shell provides authentication by the use of passwords, public keys, or both.	About Secure Shell in *Managing Secure Shell Access in Oracle Solaris 11.4*

The Oracle Solaris *privileged port* mechanism can protect Secure Shell communication. A privileged port is assigned a port number less than `1024`. After a client system has authenticated the client's credential, the client builds a connection to the server by using the privileged port. The server then verifies the client credential by examining the connection's port number.

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

```
"Weak Authentication
NFS request from unprivileged port"
```

Firewall Systems

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

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

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

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

> **Caution:**
>
> Even if you maintain strict and rigidly enforced security on the firewall, if you relax security on other systems on the network, an intruder who can break into your firewall system can then gain access to all the other systems on the internal network.

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

IPsec and the Packet Filter feature of Oracle Solaris can provide firewall protection. For more information about protecting network traffic, see *Securing the Network in Oracle Solaris 11.4*.

Encryption and Firewall Systems

Unauthorized users from outside a network can corrupt or destroy the data in packets by capturing the packets before they reach their destination and injecting arbitrary data into the contents before sending the packets back on their original course. This procedure is called *packet smashing*.

On a local area network, packet smashing is impossible because packets reach all systems, including the server, at the same time. Packet smashing is possible on a gateway, however, so make sure that all gateways on the network are protected.

The most dangerous attacks affect the integrity of the data. Such attacks involve changing the contents of the packets or impersonating a user.

Other attacks might involve eavesdropping but do not compromise data integrity or impersonate a user. An eavesdropper records conversations for later replay. Although eavesdropping attacks do not attack data integrity, the attacks do affect privacy. You can protect the privacy of sensitive information by encrypting data that goes over the network.

- To encrypt remote operations over an insecure network, see Chapter 1, Using Secure Shell in *Managing Secure Shell Access in Oracle Solaris 11.4*.
- To encrypt and authenticate data across a network, see Chapter 1, Kerberos on Oracle Solaris in *Managing Kerberos in Oracle Solaris 11.4*.
- To encrypt IP datagrams, see Chapter 6, About IP Security Architecture in *Securing the Network in Oracle Solaris 11.4*.

Reporting Security Problems

If you experience a suspected major enterprise security breach, you can contact the Computer Emergency Response Team/Coordination Center (CERT/CC). CERT/CC is a Defense Advanced Research Projects Agency (DARPA) funded project that is located at the Software Engineering Institute at Carnegie Mellon University. This agency can assist you with any security problems you are having. This agency can also direct you to other Computer Emergency Response Teams that might be more appropriate for your particular needs. For current contact information, consult the Carnegie Mellon CERT Division (https://www.sei.cmu.edu/about/divisions/cert/index.cfm) web site.

2

Protecting Oracle Solaris System Integrity

Oracle Solaris systems can be protected from unauthorized kernel modules, Trojan applications, and other threats being loaded on the system. This chapter describes security features in Oracle Solaris that provide protection from such threats and maintain system integrity as a whole.

The chapter covers the following topics:

Using Verified Boot

Malicious programs can pass information to third parties as well as alter the behavior of Oracle Solaris. Although third-party modules are typically non-malicious, they might violate policies that control site changes. Therefore, the system also needs protection from unauthorized installation of these modules.

Verified boot in Oracle Solaris secures a system's boot process. You must enable this feature, which protects the system from threats such as the following:

- Corruption of kernel modules
- Insertion or substitution of malicious programs that masquerade as legitimate kernel modules, such as Trojan viruses, spyware, and rootkits
- Installation of unauthorized third-party kernel modules

A firmware upgrade may be required to use verified boot. For information, see SPARC: Firmware Upgrade for Verified Boot.

You can enable verified boot in the following configurations and using the following tools:

- Oracle Solaris SPARC systems – Refer to Policy for Verified Boot.
- UEFI Secure Boot (BIOS menu) for x86 – Refer to your platform's instructions about Secure Boot configuration.
- Oracle Solaris Kernel Zones – Refer to Using Verified Boot to Secure an Oracle Solaris Kernel Zone in *Creating and Using Oracle Solaris Kernel Zones*.
- Logical Domains (LDOM) – Refer to Using Verified Boot in *Oracle VM Server for SPARC 3.6 Administration Guide*.
- Oracle Integrated Lights Out Manager (ILOM) – Refer to Configuring SPARC Verified Boot Properties in *Oracle ILOM Administrator's Guide for Configuration and Maintenance Firmware Release 3.2.x*.

SPARC: Firmware Upgrade for Verified Boot

SPARC firmware is installed at the factory. For some SPARC platforms, an expanded bootblock is incompatible with verified boot functionality. If the verified boot level is set to `enforce`, the system will not boot, and messages similar to the following display:

```
WARNING: Total size of bootblk fcode greater than expected
FATAL: Bootblk fcode extraction failed
Missing cmn-xxx[ caused cmn-append with 'verified boot policy =
enforce, halting boot' argument to fail
```

If the verified boot level is set to `warning`, messages similar the following display:

```
WARNING: Bootblk fcode extraction failed
WARNING: Signature verification of boot-archive bootblk failed
```

Update the firmware on SPARC systems where verified boot is enabled as follows:

- T5-Series, M5-Series, and M6-Series – Upgrade to firmware 9.6.5 or later.
- T7-Series and M7-Series – Upgrade to firmware 9.7.1 or later.
- Fujitsu M10 – Upgrade to XCP 2320 or later.

To update firmware, see the `fwupdate`(1M) man page.

Verified Boot and ELF Signatures

In Oracle Solaris, boot verification is performed by means of `elfsign` signatures or keys. At the factory, Oracle Solaris kernel modules are signed with these keys. Because of their file format, these modules are also called ELF objects. The signature is created by using the SHA-256 checksums of selected ELF records in an object file. The SHA-256 checksums are signed with a RSA-2048 private and public key pair. The public key is distributed from the `/etc/certs/elfsign` directory while the private key is not distributed.

All keys are stored in the system's pre-boot environment, which is the software or firmware that runs prior to the booting of Oracle Solaris. The firmware loads and boots `platform/.../unix`.

The pre-boot environment differs for each category of SPARC systems, as follows:

- SPARC systems with verified boot support in their Oracle Integrated Lights Out Manager (ILOM) – Keys and configuration settings are stored in ILOM.

  Because Oracle ILOM is outside the operating system's file system, verified boot configuration is protected from tampering by users of the operating system, including those with administrator (root) privileges. Thus, verified boot in this category of systems is more secure.

  You must ensure that access to ILOM is secure to prevent unauthorized changes to the verified boot configuration. For more information about securing ILOM, refer to the documentation at System Management and Diagnostics Documentation (http://www.oracle.com/goto/ilom/docs).
- SPARC M5-Series, SPARC M6-Series, and SPARC T5-Series – Configuration settings are stored in the system's ILOM. The SPARC firmware sends the configuration information to Oracle Solaris.

- Fujitsu SPARC M12 and Fujitsu M10 systems – Configuration settings are stored in the system's XSCF. The Fujitsu SPARC M12 and Fujitsu M10 XSCF firmware send the configuration information, such as policies for verified boot and enabling certificates, to Oracle Solaris. OpenBoot (OBP) reads this configuration information before booting the Oracle Solaris system.

  All XCP firmware on Fujitsu SPARC M12 systems supports verified boot. For more information about configuring verified boot, refer to the following guides:

  – *Fujitsu SPARC M12 and M10/SPARC M10 System Operation and Administration Guide*
  – *Fujitsu M10/SPARC M10 Systems Product Notes* – For the XCP firmware version that supports verified boot on Fujitsu M10 systems

Verification Sequence During System Boot

Verified boot automates the verification of the `elfsign` signatures of Oracle Solaris kernel modules. With verified boot, the administrator can create a verifiable chain of trust in the boot process beginning from system reset through the completion of the boot process.

During a system boot, each block of code that is started in the boot process verifies the next block that needs to be loaded. The sequence of verification and loading continues until the last kernel module is loaded.

When a power cycle is subsequently performed on the system, a new sequence of verification begins. The administrator can also configure verified boot to take the appropriate action in the event of verification failure.

The following illustrates the boot flow of Oracle Solaris on a SPARC system:

```
Firmware -> Bootblock -> /platform/.../unix -> genunix -> other kernel modules
```

The firmware verifies and then loads the Oracle Solaris `/platform/.../unix` module, the initial Oracle Solaris module. In turn, the Oracle Solaris kernel runtime loader `krtld`, which is part of the `unix` module, verifies and loads the generic UNIX (`genunix`) module and subsequent modules.

Policy for Verified Boot

In this release, verified boot has only one policy property: `boot_policy`. The `boot_policy` property manages verified boot behavior when loading kernel modules during the boot process.

On legacy SPARC systems and x86 systems, the `boot_policy` property is defined in the `/etc/system` file. On SPARC systems with Oracle ILOM verified boot support, `boot_policy` is a property of ILOM in `/HOST`*n*`/verified_boot`, where *n* is the physical domain (PDomain) number.

The `boot_policy` property can be configured with one of the following values:

- `none` – No boot verification is performed. This is the default.
- `warning` – The `elfsign` signature of each kernel module is verified before the module is loaded. If verification fails on a module, the module is still loaded. The discrepancies are recorded on the system console or, if available, in the system log. By default, the log is `/var/adm/messages`.

- `enforce` – The `elfsign` signature of each kernel module is verified before the module is loaded. If verification fails on a module, the module is not loaded. The discrepancies are recorded on the system console or, if available, in the system log. By default, the log is `/var/adm/messages`.

> **Note:**
>
> By default, any logical domain that was created on an Oracle VM Server for SPARC version earlier than 3.4 sets `boot-policy=warning`. If the kernel module is unsigned or corrupted, this setting results in warning messages being issued while the domain boots after an update to the server.

Public Key Certificates for Verified Boot

Verified boot uses public key certificates from the following sources:

- `/etc/certs/elfsign` directory

  If your image contains ELF objects that are signed by a third-party vendor, you must add the vendor's certificate to this directory.
- Kernel Zones, as added by the `zoneadm` command
- UEFI Secure Boot (BIOS menu) for x86
- Oracle ILOM for SPARC

  Oracle ILOM for SPARC that supports verified boot provides a preinstalled verified boot certificate file, `/etc/certs/elfsign/ORCLS11SE`. The certificate contains the RSA public key that is used to verify the `elfsign` signatures in ELF objects that Oracle Solaris signed. All certificates are loaded and managed on each individual PDomain.
- ILOM syntax varies according to hardware platform and firmware version. To configure certificates using Oracle ILOM, review Configuring SPARC Verified Boot Properties in *Oracle® ILOM Administrator's Guide for Configuration and Maintenance Firmware Release 3.2.x*.

You can also manually verify a kernel module's signature. Manual verification can be useful during diagnostics to confirm that the signature is present and correct.

Example 2-1 Manually Verifying a Kernel Module's Signature

Use the `elfsign verify -v` *kernel_module* command syntax as follows:

```
$ elfsign verify -v /kernel/misc/sparcv9/bignum
elfsign: verification of /kernel/misc/sparcv9/bignum passed.
Elfsign signature format: rsa_sha256
Signer: O=Oracle Corporation, OU=Corporate Object Signing, OU=Solaris Signed
Execution, CN=Solaris 11
```

Using Trusted Platform Module

Trusted Platform Module (TPM) refers to the device as well as the implementation by which encrypted configuration information specific to the system is stored. The information serves as metrics against which processes are measured during system

boot. TPM serves as a secure hardware keystore for RSA keys that can be accessed with PKCS #11 libraries and with the `pktool` command.

The following components implement TPM in Oracle Solaris:

- The TPM device driver communicates with the TPM device.
- The Trusted Computing Group (TCG) Software Stack, or TSS, functions as the communication channel with the TPM device by means of the `tcsd` daemon.
- The PKCS #11 libraries implement a hardware token or provider that uses the TPM to generate keys and perform sensitive operations. The provider protects all private data objects by encrypting them with keys that can be used only inside the TPM device. The PKCS #11 libraries adhere to the following standard: RSA Security Inc. PKCS #11 Cryptographic Token Interface (Cryptoki).
- The `tpmadm` command is used to administer the TPM-related aspects for verification of the boot process.

  For more details, see the `tpmadm`(8) man page.

The platform owner must initialize TPM by setting an owner password which is used to authorize privileged operations. The platform owner, also called the TPM owner, differs from the traditional superuser in two ways:

- To access TPM functions, process privilege is irrelevant. Privileged operations require knowledge of the owner password regardless of the privilege level of the calling process.
- The TPM owner cannot override access controls for data protected by TPM keys. The owner can effectively destroy data by reinitializing the TPM. However, the owner cannot access data that has been encrypted with TPM keys which are owned by other users.

Trusted Platform Module, together with the other measures described in this guide, secures the system from unauthorized access by users or applications.

Initializing and Backing Up TPM on Oracle Solaris Systems

This section contains procedures for initializing TPM on Oracle Solaris systems and for backing up TPM data and keys. The procedures vary between SPARC and x86 systems. However, to initialize TPM, certain prerequisites are common for both platforms.

- The TPM device `/dev/tpm` must be installed on the system.
- TPM must be using TCG Trusted Platform Module specification Version 1.2, otherwise known as ISO/IEC 11889-1:2009. Refer to the specification published in https://trustedcomputinggroup.org//tpm-main-specification/.
- The following Oracle Solaris TPM packages must be installed:
  - Trusted Platform Module driver (`driver/crypto/TPM`)
  - TrouSerS TCG software (`library/security/trousers`)

  To install these packages, use the following commands:

```
# pkg install driver/crypto/tpm
# pkg install library/security/trousers
```

How to Check Whether the TPM Device Is Recognized by the Operating System

Use this procedure to determine whether Oracle Solaris recognizes the installed TPM device. This procedure applies to both SPARC and x86 systems.

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*.

- **In a terminal window, issue the following command:**

  ```
  # prtconf -v |grep tpm
  ```

  If the TPM device is recognized, the command generates output similar to the following:

  ```
  # prtconf -v |grep tpm
  tpm, instance #0
  dev_path=/pci@0,0/isa@lf/tpm@0,fed40000:tpm
  dev_link=/dev/tpm
  ```

  If no output is generated, then TPM might be disabled. For information about how to enable the device, see either How to Initialize TPM Using the Oracle ILOM Interface (SPARC Only) or How to Initialize TPM Using BIOS (x86 Only) depending on your system's platform.

  > **Note:**
  >
  > As an alternative, you can also use the `ls` command to obtain the same information. However, the output would contain less information than the `prtconf` command provides.
  >
  > ```
  > # ls -l /dev/tpm
  > lrwxrwxrwx  1 root root  44 May 22 2012 /dev/tpm ->
  > ../devices/pci@0,0/isa@lf/tpm@0,fed40000:tpm
  > ```

How to Initialize TPM Using the Oracle ILOM Interface (SPARC Only)

On SPARC systems, you use both the system's ILOM and Oracle Solaris interfaces to initialize TPM.

This procedure includes instructions for backing up the TPM data and keys.

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. **At the ILOM prompt, stop the host system.**

   - For single-host servers:

     ```
     -> stop /System
     ```

   - For multidomain servers:

     ```
     -> stop /Servers/PDomains/PDomain_n/HOST
     ```

   Stopping the server can take some time. You must wait until the host console displays the following message before proceeding to the next step.

   ```
   -> SP NOTICE: Host is off
   ```

> **Note:**
>
> Add the `-f|force` option to stop the host system only if the preceding step does not shut down the host.

2. **Activate TPM.**

   Activate TPM with one of the following sets of commands depending on the SPARC system.

   - On SPARC M5-Series servers and SPARC T5-Series servers, use the following command:

     ```
     -> set /HOST/tpm mode=activated
     ```

   - On SPARC M5-32 Series servers, use the following command:

     ```
     -> set /HOST0/tpm mode=activated
     ```

   - On SPARC T4 servers, use the following commands:

     ```
     -> set /HOST/tpm enable=true activate=true
     -> show /HOST/tpm
     ```

3. **At the Oracle Solaris prompt, initialize TPM.**

   Initializing TPM causes you to become a TPM owner and requires you to assign an owner password, also called the Owner PIN.

   ```
   # tpmadm init
   TPM Owner PIN:
   Confirm TPM Owner PIN
   ```

4. **Verify the status of TPM.**

   ```
   # tpmadm status
   TPM Version: 1.2 (ATML Rev: 13.9, SpecLevel: 2, ErrataRev: 1)
   TPM resources
   Contexts: 16/16 available
   Sessions: 2/3 available
   Auth Sessions: 2/3 available
   Loaded Keys: 18/21 available
   Platform Configuration Registers (24)
   PCR 0: E1 EE 40 D8 66 28 A9 08 B6 22 8E AF DC 3C BC 23 71 15 49 31
   PCR 1: 5B 93 BB A0 A6 64 A7 10 52 59 4A 70 95 B2 07 75 77 03 45 0B
   PCR 2: 5B 93 BB A0 A6 64 A7 10 52 59 4A 70 95 B2 07 75 77 03 45 0B
   PCR 3: 5B 93 BB A0 A6 64 A7 10 52 59 4A 70 95 B2 07 75 77 03 45 0B
   PCR 4: AF 98 77 B8 72 82 94 7D BE 09 25 10 2E 60 F9 60 80 1E E6 7C
   PCR 5: E1 AA 8C DF 53 A4 23 BF DB 2F 4F 0F F2 90 A5 45 21 D8 BF 27
   PCR 6: 5B 93 BB A0 A6 64 A7 10 52 59 4A 70 95 B2 07 75 77 03 45 0B
   PCR 7: 5B 93 BB A0 A6 64 A7 10 52 59 4A 70 95 B2 07 75 77 03 45 0B
   PCR 8: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
   PCR 9: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
   PCR 10: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
   PCR 11: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
   PCR 12: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
   PCR 13: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
   PCR 14: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
   PCR 15: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
   PCR 16: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
   PCR 17: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
   PCR 18: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
   ```

```
PCR 19: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
PCR 20: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
PCR 21: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
PCR 22: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
PCR 23: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

5. **Back up TPM data and keys for future use during system migration or hardware replacement.**

   - **For multidomain systems with Oracle Solaris installed, enable failover of the SP board that contains the TPM.**

     ```
     # tpmadm failover
     Enter TPM Owner PIN:
     Enter PIN for the migration key:
     Confirm PIN for the migration key:
     ```

     > **Note:**
     >
     > The TPM owner PIN is the PIN used when TPM was initialized.

     Make a note of the PIN you supply for the migration key, so you can use that PIN to backup and restore the TPM keystore for future system migrations or hardware replacements. For more information, see SPARC: TPM Failover Option and the `tpmadm`(8) man page.

   - **For all other platforms, perform a manual backup of TPM data and keys. For instructions, see How to Back Up TPM Data and Keys (SPARC Only).**

6. **(Optional) Enable the TPM crypto provider.**

   > **Note:**
   >
   > The TPM crypto provider is slower than Oracle Solaris. Perform this step only if you want TPM to perform cryptographic operations.

```
# cryptoadm install provider='/usr/lib/security/$ISA/pkcs11_tpm.so'
# cryptoadm list -mv provider='/usr/lib/security/$ISA/pkcs11_tpm.so'
```

How to Back Up TPM Data and Keys (SPARC Only)

After you boot the system for the first time, you should back up the TPM data and keys so that they could be used during future system migrations or hardware replacements.

For multi-domain systems with Oracle Solaris installed, use the `tpmadm failover` command to specify that TPM data and keys are automatically backed up to the Standby SP on the server. You can use the backed-up TPM data and keys on the new SP for a system migration or hardware replacement. For instructions, see the backup step in How to Initialize TPM Using BIOS (x86 Only).

For all other platforms, use the following procedure to manually back up TPM data and keys for use during a system migration or hardware replacement.

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. **In a terminal window, ensure that TPM is enabled.**

   ```
   # tpmadm status
   ```

   If the system notes that no TPM owner is installed, TPM is not initialized. Do not proceed.

2. **Back up the migration data using the ID of the storage root key (SRK).**

   ```
   #  tpmadm migrate export 00000000-0000-0000-0000-00000000000b
   ```

   If the key requires authorization, the system will prompt you for a key password. You will also be prompted for the migration key password.

3. **Verify that the data has been backed up by locating the migration files in `/var/tpm/system`.**

   ```
   # ls -l /var/tpm/system/tpm-migration.*
   -rw-------   1 root  root  563 July 21 10:45 /var/tpm/system/tpm-migration.dat
   -r--------   1 root  root  766 July 21 10:36 /var/tpm/system/tpm-migration.key
   ```

How to Initialize TPM Using BIOS (x86 Only)

On x86 systems, you perform steps on the system's BIOS before initializing the service using Oracle Solaris.

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. **In a terminal window, reboot the system.**

   ```
   # reboot -p
   ```

2. **While the system is booting, press F2 to access the BIOS menu.**
3. **Using BIOS menu options, configure TPM.**
   a. **Navigate to Advanced → Trusted Computing.**
   b. **Set TPM by specifying values for the following menu items.**

      ```
      TCG/TPM Support [Yes]
      Execute TPM Command [Enabled]
      ```

   c. **Press the Esc key to exit the BIOS menu.**
   d. **Choose Save Changes and Exit.**
   e. **To proceed with the boot process, choose Ok.**
4. **After the boot process is completed, enable the `tcsd` daemon.**

   ```
   # svcadm enable -s svc:/application/security/tcsd
   ```

5. **Initialize TPM.**

   Initializing TPM causes you to become a TPM owner and requires you to assign an owner password.

   ```
   # tpmadm init
   TPM Owner PIN:
   Confirm TPM Owner PIN
   ```

6. **Verify the status of TPM.**

```
# tpmadm status
TPM Version: 1.2 (ATML Rev: 13.9, SpecLevel: 2, ErrataRev: 1)
TPM resources
Contexts: 16/16 available
Sessions: 2/3 available
Auth Sessions: 2/3 available
Loaded Keys: 18/21 available
Platform Configuration Registers (24)
PCR 0: E1 EE 40 D8 66 28 A9 08 B6 22 8E AF DC 3C BC 23 71 15 49 31
PCR 1: 5B 93 BB A0 A6 64 A7 10 52 59 4A 70 95 B2 07 75 77 03 45 0B
PCR 2: 5B 93 BB A0 A6 64 A7 10 52 59 4A 70 95 B2 07 75 77 03 45 0B
PCR 3: 5B 93 BB A0 A6 64 A7 10 52 59 4A 70 95 B2 07 75 77 03 45 0B
PCR 4: AF 98 77 B8 72 82 94 7D BE 09 25 10 2E 60 F9 60 80 1E E6 7C
PCR 5: E1 AA 8C DF 53 A4 23 BF DB 2F 4F 0F F2 90 A5 45 21 D8 BF 27
PCR 6: 5B 93 BB A0 A6 64 A7 10 52 59 4A 70 95 B2 07 75 77 03 45 0B
PCR 7: 5B 93 BB A0 A6 64 A7 10 52 59 4A 70 95 B2 07 75 77 03 45 0B
PCR 8: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
PCR 9: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
PCR 10: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
PCR 11: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
PCR 12: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
PCR 13: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
PCR 14: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
PCR 15: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
PCR 16: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
PCR 17: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
PCR 18: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
PCR 19: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
PCR 20: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
PCR 21: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
PCR 22: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
PCR 23: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

7. **(Optional) Enable the TPM crypto provider.**

> **Note:**
>
> The TPM crypto provider is slower than Oracle Solaris. Perform this step only if you want TPM to perform cryptographic operations.

```
# cryptoadm install provider='/usr/lib/security/$ISA/pkcs11_tpm.so'
# cryptoadm list -mv provider='/usr/lib/security/$ISA/pkcs11_tpm.so'
```

How to Enable PKCS #11 Consumers to Use TPM as a Secure Keystore

To perform this procedure, you must install and enable TPM on the system. Ensure that the `tcsd` daemon is also running.

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) If the TPM PKCS #11 token provider has not been installed, install that provider.**

> **Note:**
>
> To see if this step needed, check that the `pkcs11_tpm.so` provider is included when you run the `cryptoadm list` command.

```
# pkg install pkcs11_tpm
# cryptoadm install provider='/usr/lib/security/$ISA/pkcs11_tpm.so'
```

2. **Verify that the TPM device is installed.**

```
# ls -alF /dev/tpm
lrwxrwxrwx 1 root 39 Dec 27 2011 /dev/tpm -> ../devices/pci@0,0/isa@1/
tpm@1,1670:tpm
```

3. **Enable the `tcsd` daemon.**

```
# svcadm enable tcsd
```

4. **(Optional) If no TPM owner has been installed, initialize the TPM.**

> **Note:**
>
> To see if this step is needed, run the `tpmadm status` command.

```
# tpmadm init
```

5. **Initialize the personal TPM-protected token storage area.**

```
$ pktool inittoken currlabel=TPM
```

> **Note:**
>
> All individual users must perform this step.

6. **Set the token PIN for the security officer.**

```
$ pktool setpin token=tpm/TPM usertype=so
```

7. **Set the user's PIN.**

```
$ pktool setpin token=tpm/TPM
```

8. **Generate keys and certificates that use the TPM device by specifying the token name that was used when the token was initialized.**

```
$ pktool gencert token=tpm/TPM -i
$ pktool list token=tpm/TPM
```

Any existing applications that already use the Cryptographic Framework in `libpkcs11` can use the TPM token for their operations by making the applications select the TPM token device for the sessions.

Example 2-2 Enabling PKCS #11 Consumers to Use TPM

In this example, the TPM token is first assigned a new name. All subsequent actions on the token refer to the new name.

```
$ pktool inittoken currlabel=TPM newlabel=JanDoeTPM
$ pktool setpin token=tpm/JanDoeTPM so
$ pktool gencert token=tpm/JanDoeTPM -i
$ pktool list token=tpm/JanDoeTPM
```

Troubleshooting TPM

This section covers the following:

- Monitoring TPM Status
- SPARC: TPM Failover Option
- SPARC: Migrating or Restoring TPM Data and Keys

Monitoring TPM Status

Use the commands described in this section to monitor different operating components that enable you to successfully use TPM and troubleshoot TPM problems.

- To verify that the `tcsd` daemon is running:

```
# svcs tcsd
STATE      STIME      FMRI
online     Nov_07     svc:/application/security/tcsd:default
```

- To ensure that the TPM device is installed:

```
# ls -alF /dev/tpm
lrwxrwxrwx 1 root 39 Dec 27 2011 /dev/tpm -> ../devices/pci@0,0/isa@1/
tpm@1,1670:tpm
```

- To verify that the TSS software package is installed:

```
# pkg info trousers
             Name: library/security/trousers
          Summary: TrouSerS TCG software to access a TPM device
      Description: The TrouSerS library provides a software stack from the
                   Trusted Computer Group (TCG) that accesses a Trusted
Platform
                   Module (TPM) hardware device.
         Category: System/Security
            State: Installed
        Publisher: solaris
          Version: 0.3.13
           Branch: 11.4.27.0.0.76.0
   Packaging Date: Sun Jul 19 01:41:46 2020
Last Install Time: Sun Jul 19 21:38:20 2020
             Size: 3.85 MB
             FMRI: pkg://solaris/library/security/
trousers@0.3.13-11.4.27.0.0.76.0:20200719T014146Z
      Project URL: http://trousers.sourceforge.net/
       Source URL: https://sourceforge.net/projects/trousers/files/trousers/
0.3.13/trousers-0.3.13.tar.gz
```

- To check the current status of TPM:
  - The following output means that TPM is not initialized.

```
# tpmadm status
TPM Version: 1.2 (STM  Rev: 13.12, SpecLevel: 2, ErrataRev: 3)
No TPM owner installed.
```

- The following output means that the `tcsd` service needs to be started by using the `svcadm enable tcsd` command.

```
# tpmadm status
Connect context: Communication failure (TSS.TSS_E_COMM_FAILURE 0x3011).
Make sure the tcsd service "svc:/application/security/tcsd" is running.
```

- The following output means that TPM is initialized.

```
# tpmadm status
TPM Version: 1.2 (IFX Rev: 3.16, SpecLevel: 2, ErrataRev: 2)
TPM resources
        Contexts: 32/32 available
        Sessions: 20/20 available
        Authentication Sessions: 20/20 available
        Loaded Keys: 8/10 available
Platform Configuration Registers (24)
        PCR 0:  D1 8A 59 A6 64 6C 38 D7 01 14 F6 F5 05 77 2B 2C AA 4A AC 7F
        PCR 1:  AE 00 DE C4 9F 35 C6 A4 1B 5D E7 7D 57 73 87 2C B2 B9 F2 79
        PCR 2:  3C 80 7F A0 CE 0D 71 47 3D BB 27 62 B8 26 81 23 F6 37 C1 4C
        PCR 3:  3A 3F 78 0F 11 A4 B4 99 69 FC AA 80 CD 6E 39 57 C3 3B 22 75
        PCR 4:  67 36 B9 7C 15 A0 1E 59 5A E5 83 F7 D5 B4 60 16 FB F3 9F 07
        PCR 5:  A0 AD 25 17 E3 1A 35 7D 70 2B 46 3C 2D 82 6A 64 8A DE 82 5A
        PCR 6:  3A 3F 78 0F 11 A4 B4 99 69 FC AA 80 CD 6E 39 57 C3 3B 22 75
        PCR 7:  3A 3F 78 0F 11 A4 B4 99 69 FC AA 80 CD 6E 39 57 C3 3B 22 75
        PCR 8:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
        PCR 9:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
        PCR 10: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
        PCR 11: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
        PCR 12: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
        PCR 13: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
        PCR 14: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
        PCR 15: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
        PCR 16: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
        PCR 17: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
        PCR 18: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
        PCR 19: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
        PCR 20: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
        PCR 21: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
        PCR 22: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
        PCR 23: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

- To clear TPM as a requirement after TPM was previously reinitialized.
  - At the Oracle Solaris prompt:

```
# tpmadm clear owner
```

  - At the ILOM prompt:

```
-> stop /SYS
-> set /HOST/tpm forceclear=true
-> start /SYS
```

SPARC: TPM Failover Option

SPARC multi-domain servers that have Oracle Solaris 11.4 installed have the ability to fail over the SP/SPP board that contains the TPM. You can enable TPM failover by using the `-failover` option of the `tpmadm` command.

The `-failover` option prompts for the TPM Owner PIN and a new PIN for the Migration Key. These settings will be used to backup and restore the TPM keystore in case the TPM chip fails over to a new TPM chip on another SPARC SP/SPP board.

For instructions, see the backup step in How to Initialize TPM Using the Oracle ILOM Interface (SPARC Only). See also the `tpmadm`(8) man page.

SPARC: Migrating or Restoring TPM Data and Keys

SPARC multi-domain servers that have Oracle Solaris 11.4 installed can, if the `-failover` option was previously enabled, fail over the SP/SPP board that contains the TPM. See SPARC: TPM Failover Option.

All other platforms must have had a manual backup created. See How to Back Up TPM Data and Keys (SPARC Only). If a manual backup was created, you can use the following procedure to install the backup of the TPM data and keys on a new SP.

How to Migrate or Restore TPM Data and Keys (SPARC Only)

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. **Migrate the TPM data and keys.**

```
#  tpmadm migrate import
```

2. **Verify that the data has migrated.**

```
# tpmadm keyinfo
    [SYSTEM] 00000000-0000-0000-0000-000000000001 (loaded)
        [SYSTEM] 00000000-0000-0000-0000-00000000000b
         [USER] bc25ec53-239e-6ae8-f888-9e46d8f8f40f
           [USER] f5cc255c-2bd5-cb2d-e961-874f82dad286
```

Using ILOM to Prevent Access to USB Ports

Oracle ILOM is a system management firmware that is preinstalled on some SPARC platforms, such as the T7 platform. Oracle ILOM enables you to actively manage and monitor components installed in your server.

> **Note:**
>
> The following procedure is not fully supported on all platforms. Check your platform manual to determine whether this option is available.

How to Disable USB Ports by Using ILOM

This task uses Oracle ILOM tools. You must be able to access the ILOM in order to use this procedure. For further information, see the Oracle® Integrated Lights Out Manager Documentation Library.

1. **In ILOM, set the USB control to a disabled state.**

```
->set /SYS/MB/USB_CTRL requested_config_state=Disabled
```

2. **To make the setting take effect, initiate a DC power cycle.**

```
->stop /SYS
->start /SYS
```

Protecting Against Malware With Security Extensions

Oracle Solaris protects address space, process heap, and the process stack through the Security Extensions Framework. Security extensions are enabled by default for kernel processes, such as Kerberos daemons.

The Security Extensions Framework also protects platforms that run Oracle Solaris from various speculative execution vulnerabilities. These platform security extensions are managed slightly differently from the heap and stack extensions. See Protecting Platforms Against Speculative Execution Attacks.

Security extensions protect selected application binaries in Oracle Solaris. For example, the Apache HTTP Server, DHCP, Secure Shell, and `sendmail` are protected by security extensions. To see whether a binary is protected by security extensions, see Determining Whether a Binary is Protected by a Security Extension.

You can use the framework's `sxadm` command to enable and disable security extensions for selected binaries and to manage their properties.

The security extensions configurations for a binary include:

- **Disabled –** The security extension is disabled for all binaries.
- **Tagged binaries –** The security extension is controlled by the tag that is coded in the binaries.
- **Enabled –** The security extension is enabled for all binaries, except for those that are explicitly tagged to disable it.

You can use the framework's `sxadm status` command to show the current status of the extension. The flags that indicate the status are:

- `c` – Indicates a **c**onfigurable security extension
- `k` – Indicates a **k**ernel security extension
- `p` – Indicates that changes are **p**ending which will be active after a reboot
- `r` – Indicates that a **r**eboot is requred after configuration changes
- `u` – Indicates a **u**serland security extension

`sxadm` includes a debugging interface, `sxadm exec`, which executes a specific program with a given security extension enabled or disabled for that single execution.

Randomizing the Layout of the Address Space

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

Zones inherit this randomized layout for their processes. Because ASLR might not be optimal for all binaries, its use is configurable at the zone level and at the binary level.

The default Oracle Solaris value for ASLR is `tagged-binaries`. Many binaries in Oracle Solaris are tagged to use ASLR.

You must assume the `root` role to run the `sxadm` command. For examples and information, see the `sxadm`(8) man page. For developer information, see *Developer's Guide to Oracle Solaris 11.4 Security*.

Protecting the Process Heap and Executable Stacks From Compromise

A common method of computer attack is to place malicious code in memory and then jump to that code. Such attacks rely on segments that are both writable and executable. You can use the `nxheap` and `nxstack` security extensions to systematically make the stack and heap of all Oracle Solaris processes non-executable. The `nxstack` security extension replaces the `noexec_user_stack` system variable.

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

64-bit processes always have non-executable stacks. By default, 32-bit SPARC processes have executable stacks. The `nxstack` security extension, which is enabled by default, prevents the stacks of 32-bit processes from being executable. Programs that attempt to execute code on their stack are sent a `SIGSEGV` signal. This signal usually results in the program terminating with a core dump.

A log is written by default. The log is useful for identifying valid programs that depend upon executable stacks that have been prevented from correct operation by setting the `nxstack` security extension. Even when messages are not being logged, the `SIGSEGV` signal can continue to cause the executing program to terminate with a core dump. See How to Prevent the Execution of Malicious Code From the Process Stack and Process Heap and the `sxadm`(8) man page.

Programs can explicitly mark or prevent stack execution. The `mprotect`() function in programs explicitly marks the stack as executable. For more information, see the `mprotect`(2) man page. A program compiled with `-z nxstack=enable` makes the stack non-executable regardless of the system-wide setting.

The heap is memory set aside for dynamic allocation. It is reclaimed when the application, that is, the process, exits. Removing execute permission from the process heap prevents the storing of malicious code in the heap. Most programs function correctly without executing code on the heap.

The `nxheap` security extension is enabled by default, as are logs. For examples and information, see How to Prevent the Execution of Malicious Code From the Process Stack and Process Heap and the `sxadm`(8) man page.

`nxstack` and `noexec_user_stack` Compatibility

The `noexec_user_stack` and `noexec_user_stack_log` system variables are deprecated. However, if the variables remain in the `/etc/system` file, the protection of executable stacks is ensured by the following enforcement:

- If `noexec_user_stack` is set to `1`, the value of `nxstack` remains enabled for all processes.
- If `noexec_user_stack` is set to `0`, the value of `nxstack` becomes `tagged-files`.
- If `noexec_user_stack_log` is set to `1`, log files of error messages are kept.
- If `noexec_user_stack_log` is set to `0`, log files of error messages are not kept.

How to Prevent the Execution of Malicious Code From the Process Stack and Process Heap

For a description of the security risks of 32-bit executable stacks, see Protecting the Process Heap and Executable Stacks From Compromise.

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. **View the status of the `nxstack` and `nxheap` security extensions.**

   For example, on a SPARC platform that supports ADI, the output is similar to the following:

   ```
   $ sxadm status
   EXTENSION           STATUS                      FLAGS
   adiheap             enabled (tagged-files)      u-c--
   adistack            enabled (tagged-files)      u-c--
   ...
   ```

   For parseable output, specify the parameters with the `-po` options:

   ```
   $ sxadm status -po extension,status,configuration
   aslr:enabled.tagged-files:enabled.default
   nxheap:enabled.tagged-files:enabled.default
   nxstack:enabled.all:enabled.default
   ...
   ```

2. **Return the `nxheap` and `nxstack` security extensions to the default.**

   If the `nxheap` or `nxstack` security extensions show values other than the default values, delete the customizations. In the example output, you would run the following commands:

   ```
   # sxadm delcust nxheap
   # sxadm status
   aslr                enabled (tagged-files)      u-c--
   nxheap              enabled (tagged-files)      u-c--
   nxstack             enabled (all)               u-c--
   ```

   The logs for `nxheap` and `nxstack` are stored in the `/var/adm/messages` file.

3. **(Optional) To disable the logging of error messages, disable the `log` property.**

   ```
   # sxadm set log=disable nxheap
   # sxadm set log=disable nxstack
   # sxadm get log
   EXTENSION           PROPERTY                    VALUE
   ...
   nxstack             log                         disable
   nxheap              log                         disable
   ```

Troubleshooting: If your `nxstack` setting is ignored, remove the `noexec_user_stack` and `noexec_user_stack_log` system variables from the `/etc/system` file. Then, enable the `nxstack` security extension again.

If you disable `noexec_user_stack` in the `/etc/system` file but do not remove the entry, binaries that are tagged continue to be protected. This `tagged-files` configuration allows binaries that can only succeed when their stack is executable to succeed, while protecting most executable stacks from malicious code. For more information, see nxstack and noexec_user_stack Compatibility.

Preventing Process Heap Corruption Using `adiheap`

The SPARC M7, SPARC M8, SPARC S7, SPARC T7, and SPARC T8 processors support Application Data Integrity (ADI), a hardware feature that helps prevent software exploitation threats such as linear buffer overflows and stray pointer dereferences.

The `adiheap` security extension enables ADI usage by the `malloc` family of functions in the `libc` and `libumem` libraries. `adiheap` provides a reliable defense against linear buffer overflows and a valid mitigation against use-after-free issues. `adiheap` can also uncover subtle, latent bugs that have not yet affected the binary behavior, but can be triggered by an otherwise innocuous code change.

On systems where ADI is not supported or when `adiheap` is disabled, `malloc` memory allocation is unaffected. See the "`ADI INTERACTION`" section of the `malloc`(3C) man page.

`adiheap` is not enabled by default. Applications can enable `adiheap` through binary tagging, supported by the following linker option:

`-z sx=adiheap[=`*`enable`*`|`*`disable`*`]`

The following output indicates that ADI protections are available on your hardware:

```
$ sxadm status
EXTENSION           STATUS                      FLAGS
adiheap             enabled (tagged-files)      u-c--
adistack            enabled (tagged-files)      u-c--
...
```

> **Note:**
>
> In the `sxadm` command, the `model=all` property value is currently not allowed for `adiheap`.

ADI-Based Stack Protection Using `adistack`

The SPARC M7, SPARC M8, SPARC S7, SPARC T7, and SPARC T8 processors support Application Data Integrity (ADI), a hardware feature that helps prevent software exploitation threats such as linear buffer overflows and stray pointer dereferences.

The `adistack` security extension manages and supports ADI-based detection of stack buffer overflows. These overflows can overwrite the register save area of a stack frame in a SPARC 64-bit user process. When active, `adistack`, together with ADI, leverages SPARC register window spill and fill handling to detect this type of buffer overflow and generate SEGV signals.

`adistack` is not enabled by default. Applications can enable `adistack` through binary tagging, supported by the following linker option:

```
-z sx=adistack[=enable|disable]
```

For more information about this option, see Compiling an Application With adistack Enabled, Specifying Per-Object Security Extensions and the `ld`(1) man page.

> **Note:**
>
> The `model=all` property is not supported for `adistack` because some applications or interpreted languages may directly read or modify their own stack. In particular, C++ and Java applications do not yet support `adistack`.

For more information, see Protecting Against Malware With Security Extensions and the `sxadm`(8) man page.

Example 2-3 Compiling an Application With `adistack` Enabled

Using a basic makefile with the linker option, this example specifies `adistack` as a per-object security extension with the extension mode set to `enable`.

Specify a makefile rule that describes a dependency and how to resolve it as follows:

```
CFLAGS=-m64 -O
LDFLAGS = -z sx=adistack=enable

prog: prog.o
    ${CC} ${CFLAGS} ${LDFLAGS} -o prog prog.o
```

In this example:

- The `CFLAGS` variable specifies that the object created will be a 64-bit object.
- The `LDFLAGS` variable enables `adistack` as a security extension for the object created.
- The prog binary depends on the existence of the `prog.o` object file. `prog.o` depends on `prog.c`.

Using the preceding makefile rule, you can execute the following command to fulfill the dependencies and create the `prog` binary with `adistack` enabled:

```
$ make
```

Without writing the makefile rule, you would execute the following commands to enable `adistack`:

```
$ cc -m64 -O  -c  prog.c
$ cc -m64 -O -z sx=adiheap=enable -o prog prog.o
```

For more information, see the `make`(1S) and the `cc`(1) man pages.

KADI for Kernel Debugging on ADI Systems

KADI is designed for use on test and development systems only. KADI catches memory corruption in the kernel heap on SPARC systems that implement Application Data Integrity

(ADI). KADI provides protection against buffer overflows, use-after-frees, use-after-reallocs, and double frees. Because it may decrease system stability, do not enable KADI on production systems.

KADI is suitable for developers of device drivers and other kernel modules for use on the following SPARC servers:

SPARC M7
SPARC M8
SPARC S7
SPARC T7
SPARC T8

KADI is enabled on debug kernels. To enable it for developers on non-debug kernels, run the following command:

```
$ sxadm enable kadi
```

You must reboot the Oracle Solaris system for the change to take effect.

To check the status of KADI on a system, type `sxadm status`. For example:

```
$ sxadm status
EXTENSION           STATUS                      FLAGS
adiheap             enabled (tagged-files)      u-c--
adistack            enabled (tagged-files)      u-c--
aslr                enabled (tagged-files)      u-c--
hw_bti              enabled                     ---r-
hw_ssb              not supported               -----
kadi                disabled                    -kcr-
kpti                enabled                     -----
mds_no              enabled                     -----
nxheap              enabled (tagged-files)      u-c--
nxstack             enabled (all)               u-c--
rdcl_no             enabled                     -----
```

For more information, see Using KADI in *Writing Device Drivers in Oracle Solaris 11.4* and the `sxadm`(8) man page.

Enabling Inheritance of Security Extension Status

The `-i` option to the `sxadm exec` command enables or disables the inheritance of a security extension's configuration.

Example 2-4 Illustrating Security Extension Inheritance

This example illustrates inheriting and not inheriting the configuration of the ASLR security extension.

1. Disable ASLR only for this shell.

   ```
   # sxadm exec -s aslr=disable /bin/bash
   ```

2. Verify the randomized heap address from this shell.

   The repeated `pmap self` command from this shell shows a randomized heap address.

3. ```
   # pmap self | grep heap
   00000054BF32E000 8K rw---i- [ heap ]
   00000054BF330000 64K rw---i- [ heap ]
   ```
   ```

```
# pmap self | grep heap
0000005B50708000 32K rw---i- [ heap ]
# pmap self | grep heap
000000A48D30E000 8K rw---i- [ heap ]
000000A48D310000 64K rw---i- [ heap ]
```

4. Run a shell with ASLR disabled for the shell and any descendant by using the `-i` option.

```
# sxadm exec -i -s aslr=disable /bin/bash
```

5. The repeated `pmap self` command from this shell shows a constant heap address with ASLR disabled:

```
# pmap self | grep heap
0000000080000000         64K rw---i-    [ heap ]
# pmap self | grep heap
0000000080000000         64K rw---i-    [ heap ]
# pmap self | grep heap
0000000080000000         64K rw---i-    [ heap ]
```

Example 2-5 Determining Whether a Binary is Protected by a Security Extension

You can use the `elfdump -d` command to determine whether a particular binary is compiled with security extensions. The security extensions must be enabled on the system for the binary to be protected. The following output indicates the extensions that are enabled, disabled, and not supported on this hardware:

```
$ sxadm status
EXTENSION           STATUS                      FLAGS
adiheap             enabled (tagged-files)      u-c--
adistack            enabled (tagged-files)      u-c--
aslr                enabled (tagged-files)      u-c--
hw_bti              enabled                     ---r-
hw_ssb              not supported               -----
kadi                disabled                    -kcr-
nxheap              enabled (tagged-files)      u-c--
nxstack             enabled (all)               u-c--
```

The following output shows that the `cat` command and the `ipsecconf` command are protected by security extensions.

```
$ elfdump -d /bin/cat | grep SUNW_SX
     [33]  SUNW_SX_ASLR     0x2        ENABLE
     [34]  SUNW_SX_NXHEAP   0x2        ENABLE
     [35]  SUNW_SX_NXSTACK  0x2        ENABLE
$ elfdump -d /usr/sbin/ipsecconf | grep SUNW_SX
     [35]  SUNW_SX_ASLR     0x2        ENABLE
     [36]  SUNW_SX_NXHEAP   0x2        ENABLE
     [37]  SUNW_SX_NXSTACK  0x2        ENABLE
```

The following output shows that the userland binaries for DHCP and Secure Shell are protected by security extensions.

```
$ elfdump -d /usr/sbin/dhcpagent | grep SUNW_SX
     [52]  SUNW_SX_ASLR     0x2        ENABLE
     [53]  SUNW_SX_NXHEAP   0x2        ENABLE
     [54]  SUNW_SX_NXSTACK  0x2        ENABLE
$ elfdump -d /usr/bin/ssh | grep SUNW_SX
     [43]  SUNW_SX_ASLR     0x2        ENABLE
     [44]  SUNW_SX_NXHEAP   0x2        ENABLE
     [45]  SUNW_SX_NXSTACK  0x2        ENABLE
```

Specifying Per-Object Security Extensions

Administrators can use the `ld -z` command to specify per-object security extensions. In this release, a revised command option provides a consistent method for specifying various security extensions as follows:

```
ld -z sx=extension[mode],...
```

Replace the *extension* variable with the name of a security extension. Replace the *mode* variable with either `enable` or `disable`. If the mode is omitted, the extension is enabled. You can specify the following security extensions:

aslr
Specifies the Address Space Layout Randomization behavior for a process.

nxheap
Specifies a non-executable heap requirement for a process.

nxstack
Specifies a non-executable stack requirement for a process.

adiheap
Specifies an Application Data Integrity (ADI) requirement for memory allocators within a process.

adistack
Specifies an Application Data Integrity (ADI) stack protection requirement for a process.
See this option used in Compiling an Application With adistack Enabled. For further information, see the `ld`(1) man page and *Oracle Solaris 11.4 Linkers and Libraries Guide*.

Protecting Platforms Against Speculative Execution Attacks

The Security Extensions Framework also provides platform-specific security extensions that protect Oracle Solaris from various speculative execution vulnerabilities. Some of these security extensions are always enabled, some can be enabled or disabled, and a few can be managed per process, similar to the extensions in the preceding section. You use the `sxadm` command to manage the extensions that can be configured. See the `sxadm`(8) man page.

> **Note:**
>
> If you enable or disable mitigations that are set at boot time, you must reboot the system for the changes to take effect.

Security Extensions Protection on the SPARC Platform

All SPARC mitigations display in the output of the `sxadm status` command, but some are not configurable. The following mitigations are configurable:

HW_BTI
Hardware BTI Mitigation (`HW_BTI`) mitigates Branch Target Injection, Spectre Variant 2 (https://nvd.nist.gov/vuln/detail/CVE-2017-5715). `HW_BTI` is not enabled by default. You must reboot after enabling or disabling it for the changes to take effect. When it is enabled, application performance can slow.

SSBD
Speculative Store Bypass Disable (`SSBD`) mitigates CVE-2018-3639 (https://nvd.nist.gov/vuln/detail/CVE-2018-3639). It restricts loads from speculating around older stores, which mostly affects interpreters such as the JVM and Javascript engines. `SSBD` is enabled by default on systems where it is required and supported. When it is enabled, application performance can slow.

> **Note:**
>
> The `SSBD` mitigation is implemented differently on the x86 platform. See `SSBD` in Security Extensions Protection on the x86 Platform.

> **Tip:**
>
> Use the `sxadm status` command to display the current status of SPARC mitigations. To change the status, use the ILOM interface, as shown in Setting Host Control and Boot Properties on SPARC Host Server in *Oracle ILOM Administrator's Guide for Configuration and Maintenance Firmware Release 4.0.x*.

Security Extensions Protection on the x86 Platform

Oracle Solaris provides several speculative execution mitigations for x86 systems. Use the `sxadm` command to configure them.

> **Note:**
>
> If you enable or disable mitigations that are set at boot time, you must reboot the system for the changes to take effect.

IBPB
Indirect Branch Prediction Barrier (`IBPB`) mitigates Branch Target Injection, Spectre Variant 2 (https://nvd.nist.gov/vuln/detail/CVE-2017-5715). It is used in the kernel to guarantee that older indirect branches cannot influence predictions of indirect branches in the future. It is enabled by default on systems where it is required and supported. When it is enabled, application performance can slow.

IBRS
Indirect Branch Restricted Speculation (`IBRS`) mitigates Branch Target Injection, Spectre Variant 2 (https://nvd.nist.gov/vuln/detail/CVE-2017-5715). At every entry into the kernel,

`IBRS` restricts the speculation of indirect branches. It is enabled by default on systems where it is required and supported. When it is enabled, application performance can slow.

`IF_PSCHANGE_MC_NO`
Machine Check Error on Page Size Change (`IF_PSCHANGE_MC_NO`) is a read-only extension that mitigates CVE-2018-12207 (https://nvid.nist.gov/vuln/detail/CVE-2018-12207). It is enabled by default on systems where it is supported.

`KPTI`
Kernel Page Table Isolation (`KPTI`) is a software workaround for Meltdown vulnerability (https://nvd.nist.gov/vuln/detail/CVE-2017-5754). It is enabled by default on systems where it is supported.

`L1DF`
Level 1 Data Cache Flush (`L1DF`) mitigates CVE-2018-3646 (https://nvd.nist.gov/vuln/detail/CVE-2018-3646). It flushes sensitive data from the L1D cache to prevent an untrusted guest virtual machine from inferring data from other guest virtual machines. This flush is performed every time the host system enters a virtual machine (VM entry). `L1DF` is enabled by default on systems where it is required and supported.

> **Note:**
>
> Although `L1DF` is needed only when running non-trusted kernel zones, full mitigation also requires disabling hyper-threading (HT).

`MD_CLEAR`
Microarchitectural Data Sampling Avoidance Mitigation (`MD_CLEAR`) mitigates the Microarchitectural Data Sampling (MDS) series of vulnerabilities. The vulnerabilities are:

- Microarchitectural Data Sampling Uncacheable Memory (MDSUM) (https://nvd.nist.gov/vuln/detail/CVE-2019-11091)
- Microarchitectural Store Buffer Data Sampling (MSBDS) (https://nvd.nist.gov/vuln/detail/CVE-2018-12126)
- Microarchitectural Load Port Data Sampling (MLPDS) (https://nvd.nist.gov/vuln/detail/CVE-2018-12127)
- Microarchitectural Fill Buffer Data Sampling (MFBDS) (https://nvd.nist.gov/vuln/detail/CVE-2018-12130)

`MD_CLEAR` overwrites the store and fill buffers on the logical processors that are affected by MDS. It is enabled by default on systems where `MD_CLEAR` is required and supported.

> **Note:**
>
> Full mitigation of `MD_CLEAR` also requires disabling hyper-threading (HT).

`MDS_NO`
Microarchitectural Data Sampling Hardware Avoidance Mitigation (`MDS_NO`) is a read-only extension that is only enabled if the CPU is *not* vulnerable to the

Microarchitectural Data Sampling (MDS) series of vulnerabilities that the `MD_CLEAR` extension mitigates in software.

> **Note:**
>
> When `MDS_NO` is enabled, `MD_CLEAR` is enabled read-only.

`RDCL_NO`
Rogue Data Cache Avoidance Mitigation (`RDCL_NO`) mitigates CVE-2017-5754 Version 2.2 (https://nvd.nist.gov/vuln/detail/CVE-2017-5754?cpeVersion=2.2) and CVE-2018-3646 (https://nvd.nist.gov/vuln/detail/CVE-2018-3646). It prevents unauthorized disclosure of information to an attacker with local user access through a side-channel analysis of the data cache. `RDCL_NO` is read-only, and enabled by default on systems where it is supported.

> **Note:**
>
> When `RDCL_NO` is enabled, `L1DF` is also enabled read-only.

`RSBS`
Return Stack Buffer Speculation (`RSBS`) counters Spectre RSB (https://nvd.nist.gov/vuln/detail/CVE-2017-5715) by making several consecutive calls and returns for every context switch. `RSBS` is enabled by default on systems where it is required and supported.

`SMAP`
Supervisor Mode Access Prevention (`SMAP`) prevents supervisor mode execution of text that is mapped in userland. It is enabled by default when it is supported by the hardware. Certain applications or drivers can fail when `SMAP` is enabled.

`SSBD`
Speculative Store Bypass Disable (`SSBD`) mitigates CVE-2018-3639 (https://nvd.nist.gov/vuln/detail/CVE-2018-3639). It restricts loads from speculating around older stores, which mostly affects interpreters such as the JVM and Javascript engines. `SSBD` is not enabled at boot time.
Similar to the ASLR and ADI security extensions, this extension can be enabled on individual binaries. Such configuration changes do not require a reboot. For examples of how to do this, see Compiling an Application With adistack Enabled, Illustrating Security Extension Inheritance, and the `sxadm`(8) man page.

> **Note:**
>
> The `SSBD` mitigation is implemented differently on the SPARC platform. See `SSBD` in Security Extensions Protection on the SPARC Platform.

`TAA_NO`
`TAA_NO` is a read-only extension that mitigates the TSX Asynchronous Abort (TAA) (https://nvid.nist.gov/vuln/detail/CVE-2019-11135) vulnerability. It is enabled by default only when the CPU supports the Intel TSX feature and is not vulnerable to the TAA vulnerability due to a hardware mitigation. Otherwise, the extension is in the `not supported` state.

TSX_DISABLE

TSX_DISABLE is a read-only extension that mitigates the TSX Asynchronous Abort (TAA) (https://nvid.nist.gov/vuln/detail/CVE-2019-11135) vulnerability by using a control register to disable TSX.

If the TAA_NO, TSX_DISABLE, and MDS_NO extensions are in the not supported state, you might be able to mitigate TAA by enabling the MD_CLEAR extension, if not enabled already, and then rebooting the system.

If the MDS_NO extension is in the enabled state, and if both the TAA_NO and TSX_DISABLE extensions are in the not supported state, you cannot mitigate the TAA vulnerability until after you perform a microcode update.

The following table shows the minimum microcode version for each Intel Xeon CPU that contains the mitigation for the TAA vulnerability. Each table entry lists information about an Intel Xeon CPU including the CPU name and code name, its CPU identifier, and its minimum microcode version.

Intel CPU (Code Name)	CPU Identifier	Minimum Microcode Version
E7 v3 (Haswell-EX)	306F4	0x00000016
E5 v4 (Broadwell-EP)	406F1	0x0B000038
Scalable Processor (Skylake-SP)	50654	0x02000065
Scalable Processor (Cascade Lake-SP)	50657	0x0500002C

See the following information about updating the system firmware on your x86 systems:

- Update the microcode by updating the system firmware to the latest version. See Firmware Resources (https://www.oracle.com/servers/technologies/firmware-resources.html).
- Install firmware updates on Oracle x86 systems. See Oracle x86 Servers Administration, Diagnostics, and Applications Documentation (https://docs.oracle.com/cd/E23161_01/).
- Download the latest Oracle x86 system firmware version. See Welcome to the Server System Firmware Release Hub (https://www.oracle.com/servers/technologies/firmware.html).
- Obtain information about updating system firmware on non-Oracle x86 systems by referring to your vendor's documentation.

UMIP

User-Mode Instruction Prevention (UMIP) is a mechanism on Intel CPUs that restricts the execution of specific instructions if the CPU is running outside of its highest privileged mode (e.g., running in user mode). This is a security feature to prevent potential manipulation of system software data structures by malicious userland applications.

It is enabled by default when it is supported by the hardware. A reboot is required after enabling or disabling UMIP for the changes to take effect.

3
Controlling Access to Systems

This chapter describes the procedures for controlling who can access Oracle Solaris systems.

The chapter covers the following topics:

- Securing Logins and Passwords
- Changing the Default Algorithm for Password Encryption
- Monitoring and Restricting root Access
- Controlling Access to System Hardware

For overview information about system security, see Managing Computer System Security.

Securing Logins and Passwords

To guard access to your systems, you can limit remote logins, require users to have passwords, and require the `root` account to have a complex password. To manage user access, you can display a security message to users, monitor failed access attempts, and disable logins temporarily.

The following task map points to procedures that monitor user logins and that disable user logins.

Table 3-1 Securing Logins and Passwords Task Map

Task	Description	For Instructions
Inform users of site security at login.	Displays a text message on the login screen with site security information.	How to Place a Security Message in Banner Files
Display the user's login status.	Lists extensive information about the user's login account, such as full name and password aging information.	How to Display the User's Login Status
Find users who do not have passwords.	Finds only those users whose accounts do not require a password.	How to Display Users Without Passwords
Disable logins temporarily.	Denies user logins to a computer system as part of system shutdown or routine maintenance.	How to Temporarily Disable User Logins

How to Place a Security Message in Banner Files

Use this procedure to create security messages in two banner files that reflect your site's security policy. The `/etc/issue` file displays before authentication, while the `/etc/motd` file displays after authentication.

> **Note:**
>
> The sample messages in this procedure do not satisfy U.S. government requirements and likely do not satisfy your security policy. Consult with your company's legal counsel about the content of the security message.

You must become an administrator who is assigned the Administrator Message Edit rights profile. For more information, see Using Your Assigned Administrative Rights in *Securing Users and Processes in Oracle Solaris 11.4*.

1. **Create the `/etc/issue` file and add a security message.**

   ```
   # pfedit /etc/issue
   ALERT   ALERT   ALERT   ALERT   ALERT

   This system is available to authorized users only.

   If you are an authorized user, continue.

   Your actions are monitored, and can be recorded.
   ```

   The `login` command displays the contents of `/etc/issue` before authentication, as do the `ssh`, `graphical-login/gdm`, and FTP services.

   For more information, see the `issue`(5) and `pfedit`(8) man pages.

2. **Add a security message to the `/etc/motd` file.**

   ```
   # pfedit /etc/motd
   This system serves authorized users only. Activity is monitored and reported.
   ```

   In Oracle Solaris, the user's initial shell displays the contents of the `/etc/motd` file.

How to Display the User's Login Status

To use the `logins` command, you must become an administrator who is assigned either the User Management or the User Security rights profile. By default, the `root` role has this authorization. For more information, see Using Your Assigned Administrative Rights in *Securing Users and Processes in Oracle Solaris 11.4*.

- **Display a user's login status by using the `logins` command.**

  ```
  # logins -x -l username
  ```

  `-x`
  Displays an extended set of login status information.

  `-l` *username*
  Displays the login status for the specified user. The variable *username* is a user's login name. Multiple login names are separated by commas.

  The `logins` command uses the appropriate password database to obtain a user's login status. The database can be the local `/etc/passwd` file, or a password

database for the naming service. For more information, see the logins(8) man page.

Example 3-1 Displaying a User's Login Status

In the following example, the login status for the user jdoe is displayed.

```
# logins -x -l jdoe
jdoe       500     staff           10   Jaylee Jaye Doe
/home/jdoe
/bin/bash
PS 010103 10 7 -1
```

jdoe
Identifies the user's login name.

500
Identifies the user ID (UID).

staff
Identifies the user's primary group.

10
Identifies the group ID (GID).

Jaylee Jaye Doe
Identifies the comment.

/home/jdoe
Identifies the user's home directory.

/bin/bash
Identifies the login shell.

PS 010170 10 7 -1
Specifies the password aging information:

- Last date that the password was changed
- Number of days that are required between changes
- Number of days before a change is required
- Warning period

How to Display Users Without Passwords

To use the logins command, you must become an administrator who is assigned either the User Management or the User Security rights profile. By default, the root role has this authorization. For more information, see Using Your Assigned Administrative Rights in *Securing Users and Processes in Oracle Solaris 11.4*.

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

  ```
  # logins -p
  ```

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

Example 3-2 Displaying Accounts Without Passwords

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

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

How to Temporarily Disable User Logins

Temporarily disable user logins during system shutdown or routine maintenance.

> **Note:**
>
> This procedure does not affect all users. The following can continue to log in to the system despite the presence of the `/etc/nologin` file created by this procedure.
>
> - Superuser
> - Users who are assigned the `root` role
> - Users who are assigned the `solaris.system.maintenance` authorization

For more information, see the `nologin`(5) man page.

You must become an administrator who is assigned the `solaris.admin.edit/etc/nologin` authorization. By default, the `root` role has this authorization. For more information, see Using Your Assigned Administrative Rights in *Securing Users and Processes in Oracle Solaris 11.4*.

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

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

   For an example of using the `solaris.admin.edit/etc/nologin` authorization, see Disabling User Logins.
2. **Include a message about system availability.**
3. **Close and save the file.**

Example 3-3 Disabling User Logins

In this example, a user is authorized to write the notification of system unavailability.

```
$ pfedit /etc/nologin
***No logins permitted.***

***The system will be unavailable until 12 noon.***
```

Changing the Default Algorithm for Password Encryption

The default `crypt_sha256` algorithm is represented by the value `5`. To switch to another algorithm, assign a different identifier. For a list of password encryption algorithms and their corresponding identifiers, see Password Hashing Algorithms.

> **Note:**
>
> Use FIPS 140-2 approved algorithms when possible. For a list of FIPS 140-2 approved algorithms, see FIPS 140-2 Algorithm Lists and Certificate References for Oracle Solaris Systems in *Using a FIPS 140-2 Enabled System in Oracle Solaris 11.4*.

Note that the new algorithm applies only to password encryption for new users. For existing users, the previous algorithm remains operative if it remains defined in the `CRYPT_ALGORITHMS_ALLOW` parameter and is not `unix`. To see how encryption is implemented in this case, see Password Hashes Configuration. To include existing users under the new password encryption algorithm, remove the previous algorithm from the `CRYPT_ALGORITHMS_ALLOW` parameter as well.

For more information about configuring the algorithm choices, see the `policy.conf`(5) man page.

> **Note:**
>
> The procedures and examples in this section do not work if you are using the `account-policy` service. If you have enabled this service, see Modifying Rights System-Wide As SMF Properties in *Securing Users and Processes in Oracle Solaris 11.4* for how to modify the security attributes that you used to modify by editing the `policy.conf` file.

How to Specify an Algorithm for Password Encryption

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. **In the `/etc/security/policy.conf` file, specify the identifier for your chosen encryption algorithm as the value for the `CRYPT_DEFAULT` variable.**
2. **(Optional) Comment the file to explain your choice.**

   For example:

```
# cat  /etc/security/policy.conf
…
# Sets the SHA256 (5) algorithm as default.
# SHA256 supports 255-character passwords.
# Passwords previously encrypted with MD5 (1) will be encrypted
# with SHA256 (5) when users change their passwords.
```

```
#CRYPT_DEFAULT=1
CRYPT_DEFAULT=5
```

In this example, the new value of `CRYPT_DEFAULT` is `5`, which is `SHA256`, the SHA256 algorithm. SHA stands for Secure Hash Algorithm. This algorithm is a member of the SHA-2 family. SHA256 supports 255-character passwords.

3. **(Optional) Remove the previous algorithm from the `CRYPT_ALGORITHM_ALLOWED` to make the new algorithm apply to existing users.**

   For example, to ensure that the SHA256 algorithm also applies to existing users, the `CRYPT_ALGORITHM_ALLOWED` should exclude the prior identifier for MD5, `1`.

> **Note:**
>
> In addition, to promote FIPS 140-2 security, exclude the Blowfish algorithm (2a) from the entry.

```
CRYPT_ALGORITHMS_ALLOW=5,6
```

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

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

```
CRYPT_ALGORITHMS_ALLOW=1,2a,5
#CRYPT_ALGORITHMS_DEPRECATE=__unix__,md5,6
CRYPT_DEFAULT=5
```

How to Specify a New Password Algorithm for an NIS Domain

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

> **Note:**
>
> If you are using the `account-policy` SMF stencil and the `config/etc_default_passwd` property is enabled, you must change the corresponding SMF property on every system that will use this new algorithm. For examples, 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.

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. **Specify the password encryption algorithm in the `/etc/security/policy.conf` file on the NIS client.**
2. **Copy the modified `/etc/security/policy.conf` file to every client system in the NIS domain.**
3. **To minimize confusion, copy the modified `/etc/security/policy.conf` file to the NIS root server and to the slave servers.**

How to Specify a New Password Algorithm for an LDAP Domain

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

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

   > **Note:**
   >
   > If you are using the `account-policy` SMF stencil and the `config/etc_default_passwd` property is enabled, you must change the corresponding SMF property on every system that will use this new algorithm. For examples, 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.

2. **Copy the modified `policy.conf` file to every client system in the LDAP domain.**
3. **Ensure that the client's `/etc/pam.conf` file does not use a `pam_ldap` module.**

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

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

   When users in the LDAP domain change their passwords, the LDAP client consults its local algorithms configuration in the `/etc/security/policy.conf` file. The LDAP

client system encrypts the password. Then, the client sends the encrypted password, with a `{crypt}` tag, to the server. The tag tells the server that the password is already encrypted. The password is then stored, as is, on the server. For authentication, the client retrieves the stored password from the server. The client then compares the stored password with the encrypted version that the client has just generated from the user's typed password.

> **Note:**
>
> To take advantage of password policy controls on the LDAP server, use the `server_policy` option with the `pam_authtok_store` entries in the `pam.conf` file. Passwords are then encrypted on the LDAP server. For the procedure, see Chapter 4, Setting Up an Oracle Unified Directory Server or OpenLDAP Server in *Working With Oracle Solaris 11.4 Directory and Naming Services: LDAP*.

Monitoring and Restricting `root` Access

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

How to Monitor Who Is Using the `su` Command

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

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

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

> **Note:**
>
> If you are using the `account-policy` SMF stencil and the `config/etc_default_passwd` property is enabled, you must change the corresponding SMF property on every system that will use this new algorithm. For examples, 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.

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*.

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

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

```
SU 01/12 14:56 + pts/0 jdoe-root
SU 01/12 14:57 + pts/0 jdoe-root
```

The entries display the following information:

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

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

Entries similar to the following can indicate that the `su` command was not invoked on the command line: `SU 10/10 10:20 + ??? root-oracle`. A Trusted Extensions user might have switched to the `oracle` role by using a GUI.

How to Restrict and Monitor `root` Logins

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

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. **View the CONSOLE entry in the `/etc/default/login` file.**

   > **Note:**
   >
   > If you are using the `account-policy` SMF stencil and the `config/etc_default_login` property is enabled, you must view and change the `login_policy/root_login_device` property. For examples, 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.

   ```
   CONSOLE=/dev/console
   ```

   By default, the console device is set to `/dev/console`. With this setting, `root` can log in to the console. `root` cannot log in remotely.

2. **Verify that root cannot log in remotely.**

   From a remote system, try to log in as `root`.

   ```
   system2 $ ssh -l root system1
   Password: Type root password of system1
   Password:
   Password:
   Permission denied (gssapi-keyex,gssapi-with-mic,publickey,keyboard-interactive).
   ```

In the default configuration, `root` is a role, and roles cannot log in. Also, in the default configuration the `ssh` protocol prevents `root` user login.

3. **Monitor attempts to become `root`.**

   By default, attempts to become `root` are printed to the console by the `SYSLOG` utility.

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

   b. **In another window, use the `su` command to become `root`.**

   ```
   $ su -
   Password: Type root password
   #
   ```

   A message is printed on the terminal console.

   ```
   Sep 7 13:22:57 system1 su: 'su root' succeeded for jdoe on /dev/pts/6
   ```

Example 3-5 Logging `root` Access Attempts

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

```
# CONSOLE determines whether attempts to su to root should be logged
# to the named device
#
CONSOLE=/dev/console
```

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

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

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

If the console displays an entry similar to `Last login: Thu Sep 7 15:13:11 2017 from system2`, then the system is configured to permit remote `root` logins. To prevent remote `root` access, change the `#CONSOLE=/dev/console` entry to `CONSOLE=/dev/console` in the `/etc/default/login` file. To find out how to return the `ssh` protocol to the default, see the `sshd_config` (5) man page.

Controlling Access to System Hardware

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

To protect the BIOS, consult the vendor documentation. To require encryption and verification when booting, see Using Trusted Platform Module and Using Verified Boot.

How to Require a Password for SPARC Hardware Access

You must become an administrator who is assigned the Device Security, Maintenance and Repair, or System Administrator rights profile. For more information, see Using

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

1. **In a terminal window, enable the PROM security mode.**

```
# eeprom security-mode=command

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

Choose the value `command` or `full`. For more details, see the `eeprom`(8) man page.

If, when you type the preceding command, you are not prompted for a PROM password, the system already has a PROM password.

2. **(Optional) Change the PROM password.**

> **Caution:**
>
> Do not forget the PROM password. The hardware is unusable without this password.

```
# eeprom security-password=
Press Return
Changing PROM password:
New password: <Type password>
Retype new password: <Retype password>
```

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

How to Disable a System's Abort Sequence

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

You must become an administrator who is assigned the `solaris.admin.edit/etc/default/kbd` authorization. By default, the `root` role has this authorization. For more information, see Using Your Assigned Administrative Rights in *Securing Users and Processes in Oracle Solaris 11.4*.

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

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

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

```
#KEYBOARD_ABORT=enable
KEYBOARD_ABORT=disable
```

2. **Update the keyboard defaults.**

```
# kbd -i
```

4
Controlling Access to Devices

This chapter provides step-by-step instructions for protecting devices that are attached to a system, in addition to a reference section. The chapter covers the following topics:

- Configuring Device Policy
- Managing Device Allocation
- Allocating Devices
- Device Protection Reference

For overview information about device protection, see Controlling Access to Devices.

Configuring Device Policy

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

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

Table 4-1 Configuring Device Policy Task Map

Task	Description	For Instructions
View the device policy for the devices on your system.	Lists the devices and their device policy.	How to View Device Policy
Audit changes in device policy.	Records changes in device policy in the audit trail.	How to Audit Changes in Device Policy
Access `/dev/arp`.	Gets Oracle Solaris IP MIB-II information.	How to Retrieve IP MIB-II Information From a /dev/* Device

How to View Device Policy

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

```
$ getdevpolicy | more
DEFAULT
read_priv_set=none
write_priv_set=none
ip:*
read_priv_set=net_rawaccess
write_priv_set=net_rawaccess
…
```

Example 4-1 Viewing the Device Policy for a Specific Device

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

```
$ getdevpolicy /dev/allkmem /dev/ipsecesp /dev/net0
/dev/allkmem
read_priv_set=all
write_priv_set=all
/dev/ipsecesp
read_priv_set=sys_net_config
write_priv_set=sys_net_config
/dev/bge
read_priv_set=net_rawaccess
write_priv_set=net_rawaccess
```

How to Audit Changes in Device Policy

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

You must become an administrator who is assigned the Audit Configuration rights profile. For more information, see Using Your Assigned Administrative Rights in *Securing Users and Processes in Oracle Solaris 11.4*.

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

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

  For detailed instructions, see How to Preselect Audit Classes in *Managing Auditing in Oracle Solaris 11.4*.

How to Retrieve IP MIB-II Information From a `/dev/*` Device

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

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

   ```
   $ getdevpolicy /dev/ip /dev/arp
   /dev/ip
   read_priv_set=net_rawaccess
   write_priv_set=net_rawaccess
   /dev/arp
   read_priv_set=none
   write_priv_set=none
   ```

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

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

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

Managing Device Allocation

When the `pkg:/system/device-allocation` package is on the system, device allocation can be controlled. Device allocation is commonly implemented at sites that

require an additional layer of device security. Typically, users must have authorization to access allocatable devices.

The following task map points to procedures and command options that enable, configure, and troubleshoot device allocation. Device allocation is not enabled by default. After device allocation is enabled, see Allocating Devices for instructions on allocating devices.

Table 4-2 Managing Device Allocation Task Map

Task	Description	For Instructions
Make a device allocatable. Disable device allocation.	Enables a device to be allocated to one user at a time. Removes allocation restrictions from all devices.	Enabling or Disabled Device Allocation
Authorize users to allocate a device.	Assigns device allocation authorizations to users.	How to Authorize Users to Allocate a Device
View the allocatable devices on your system.	Lists the devices that are allocatable, and the state of the device.	Viewing Allocation Information About a Device
Forcibly allocate or deallocate a device.	Allocates or deallocates a device to a user who has an immediate need.	Forcibly Allocating or Deallocating a Device
Change the allocation properties of a device.	Changes the requirements for allocating a device.	Changing Which Devices Can Be Allocated
Audit device allocation.	Records device allocation in the audit trail	Auditing Device Allocation
Create a device-clean script.	Purges data from a physical device.	Writing New Device-Clean Scripts

Enabling or Disabled Device Allocation

> **Note:**
>
> If Trusted Extensions is installed and enabled on the system, then the `svc:/system/device/allocate` package has already been installed and enabled.

You must become an administrator who is assigned the Device Security rights profile to perform these actions. For more information, see Using Your Assigned Administrative Rights in *Securing Users and Processes in Oracle Solaris 11.4*.

To enable the device allocation service and verify that the service is enabled:

```
# svcadm enable svc:/system/device/allocate
svcadm: Pattern 'svc:/system/device/allocate' doesn't match any instances
# pkg install system/device-allocation
...
# svcs -x allocate
svc:/system/device/allocate:default (device allocation)
State: online since September 10, 2016 01:10:11 PM PDT
```

```
See: allocate(1)
See: deallocate(1)
See: list_devices(1)
See: device_allocate(8)
See: mkdevalloc(8)
See: mkdevmaps(8)
See: dminfo(8)
See: device_maps(5)
See: /var/svc/log/system-device-allocate:default.log
Impact: None.
```

To disable the device allocation service:

```
# svcadm disable device/allocate
```

Authorizing Users to Allocate a Device

A system administrator can enable users to allocate devices.

How to Authorize Users to Allocate a Device

You must become an administrator who is assigned the User Security rights profile. Your rights profiles must include the `solaris.auth.delegate` authorization. For more information, see Using Your Assigned Administrative Rights in *Securing Users and Processes in Oracle Solaris 11.4*.

1. **Create a rights profile that contains the appropriate authorization and commands.**

   Typically, you would create a rights profile that includes the `solaris.device.allocate` authorization. Follow the instructions in How to Create a Rights Profile in *Securing Users and Processes in Oracle Solaris 11.4*. Give the rights profile appropriate properties, such as the following:

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

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

   Follow the instructions in Assigning Rights to Users in *Securing Users and Processes in Oracle Solaris 11.4*. Use the following role properties as a guide:

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

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

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

Additional Action: Teach the users how to use device allocation.

For examples of allocating removable media, see How to Allocate a Device.

Viewing Allocation Information About a Device

You must become an administrator who is assigned the Device Security rights profile to display this information. For more information, see Using Your Assigned Administrative Rights in *Securing Users and Processes in Oracle Solaris 11.4*.

To display information about allocatable devices on your system:

```
# list_devices device-name
```

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

- `audio`[*n*] – Microphone and speaker.
- `rmdisk`[*n*] – Removable media device, such as a USB flash drive.
- `sr`[*n*] – CD-ROM drive.
- `st`[*n*] – Tape drive.

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

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

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

Forcibly Allocating or Deallocating a Device

You can forcibly allocate or deallocate a device.

You must become an administrator who is assigned the `solaris.device.revoke` authorization to perform these actions. For more information, see Using Your Assigned Administrative Rights in *Securing Users and Processes in Oracle Solaris 11.4*.

Determine whether you have the appropriate authorizations in your role.

```
$ auths
solaris.device.allocate solaris.device.revoke
```

Forcibly Allocating a Device

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

Use the `allocate -U` command to forcibly allocate the device to the user who needs the device. In this example, a USB flash drive is forcibly allocated to the user `jdoe`.

```
$ allocate -U jdoe
```

Forcibly Deallocating a Device

Because devices that a user has allocated are not automatically deallocated when the process terminates or when the user logs out, you might have to use forcible deallocation when a user has forgotten to deallocate a device.

Use the `deallocate -f` command to forcibly deallocate the device as follows:

```
$ deallocate -f /dev/lp/printer-1
```

In this example, a printer is forcibly deallocated so it is available for allocation by another user.

Changing Which Devices Can Be Allocated

Device allocation must be enabled for this task to succeed. To enable device allocation, see Enabling or Disabled Device Allocation. You must assume the `root` role.

To change which devices can be allocated, change the fifth field in the device entry in the `device_allocate` file to specify whether authorization is required, or specify the `solaris.device.allocate` authorization.

```
audio;audio;reserved;reserved;solaris.device.allocate;/etc/security/lib/
audio_clean
fd0;fd;reserved;reserved;solaris.device.allocate;/etc/security/lib/fd_clean
sr0;sr;reserved;reserved;solaris.device.allocate;/etc/security/lib/sr_clean
```

where `solaris.device.allocate` indicates that a user must have the `solaris.device.allocate` authorization to use the device.

Example 4-2 Permitting Any User to Allocate a Device

In the following example, any user on the system can allocate any device. The fifth field in every device entry in the `device_allocate` file has been changed to an "at" sign (`@`).

```
# pfedit /etc/security/device_allocate
audio;audio;reserved;reserved;@;/etc/security/lib/audio_clean
fd0;fd;reserved;reserved;@;/etc/security/lib/fd_clean
sr0;sr;reserved;reserved;@;/etc/security/lib/sr_clean
…
```

Example 4-3 Preventing Some Peripheral Devices From Being Used

In the following example, the audio device cannot be used. The fifth field in the audio device entry in the `device_allocate` file has been changed to an asterisk (`*`).

```
# pfedit /etc/security/device_allocate
audio;audio;reserved;reserved;*;/etc/security/lib/audio_clean
fd0;fd;reserved;reserved;solaris device.allocate;/etc/security/lib/fd_clean
sr0;sr;reserved;reserved;solaris device.allocate;/etc/security/lib/sr_clean
…
```

Example 4-4 Preventing All Peripheral Devices From Being Used

In the following example, no peripheral device can be used. The fifth field in every device entry in the `device_allocate` file has been changed to an asterisk (`*`).

```
# pfedit /etc/security/device_allocate
audio;audio;reserved;reserved;*;/etc/security/lib/audio_clean
fd0;fd;reserved;reserved;*;/etc/security/lib/fd_clean
sr0;sr;reserved;reserved;*;/etc/security/lib/sr_clean
…
```

Auditing Device Allocation

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

> **Note:**
>
> You must become an administrator who is assigned the Audit Configuration rights profile. For more information, see Using Your Assigned Administrative Rights in *Securing Users and Processes in Oracle Solaris 11.4*.

You can preselect the `ot` audit class as follows:

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

For detailed instructions, see How to Preselect Audit Classes in *Managing Auditing in Oracle Solaris 11.4*.

Allocating Devices

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

How to Allocate a Device

Device allocation must be enabled, as described in Enabling or Disabled Device Allocation. However, if Trusted Extensions is installed and enabled on the system, then device allocation is already enabled. Device allocation is usually enabled.

> **Note:**
>
> If authorization is required, the user must have the authorization.

1. **Allocate the device.**

   Specify the device by device name.

   ```
   $ allocate device-name
   ```

2. **Verify that the device is allocated by repeating the command.**

   ```
   $ allocate device-name

   allocate. Device already allocated.
   ```

Example 4-5 Allocating a Printer

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

```
$ allocate /dev/lp/printer-1
```

For an example of forcible deallocation, see Forcibly Allocating or Deallocating a Device.

Example 4-6 Allocating a USB Flash Drive

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

```
$ allocate rmdisk1
```

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

How to Mount an Allocated Device

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

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

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

   ```
   $ su - role-name

   Password: <Type role-name password>
   $
   ```

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

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

   ```
   $ mkdir mount-point ; chmod 700 mount-point
   ```

3. **List the allocatable devices.**

   ```
   $ list_devices -l
   List of allocatable devices
   ```

4. **Allocate the device.**

   Specify the device by device name.

   ```
   $ allocate device-name
   ```

5. **Mount the device.**

   ```
   $ mount -o ro -F filesystem-type device-path mount-point
   ```

   `-o ro`

   Indicates that the device is to be mounted read-only. Use `-o rw` to make the device writable.

   `-F` filesystem-type

   Indicates the file system format of the device. Typically, a CD-ROM is formatted with an HSFS file system.

device-path
Indicates the path to the device. The output of the `list_devices -l` command includes the *device-path*.

mount-point
Indicates the mount point that you created in Step 2.

Example 4-7 Allocating a CD-ROM Drive

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

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

Troubleshooting: If the `mount` command cannot mount the device, the `mount: insufficient privileges` error message is displayed: Check the following:

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

Contact your administrator if you still cannot mount the allocated device. See How to Troubleshoot Rights Assignments in *Securing Users and Processes in Oracle Solaris 11.4* is a starting point.

How to Deallocate a Device

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

You must have allocated the device. For information, see How to Allocate a Device.

1. **If the device is mounted, unmount the device.**

   ```
   $ cd $HOME
   $ umount
   mount-point
   ```

2. **Deallocate the device.**

   ```
   $ deallocate device-name
   ```

Example 4-8 Deallocating a Microphone

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

```
$ whoami
jdoe
$ deallocate audio0
```

Example 4-9 Deallocating a CD-ROM Drive

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

```
$ whoami
devicealloc
$ cd /home/devicealloc
$ umount /home/devicealloc/mymnt
$ ls /home/devicealloc/mymnt
$
$ deallocate sr0
/dev/sr0:       326o
/dev/rsr0:      326o
…
sr_clean: Media in sr0 is ready.  Please, label and store safely.
```

Device Protection Reference

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

Device Policy Commands

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

The following table lists the device management commands.

Table 4-3 Device Management Commands

Command	Purpose
`add_drv`(8)	Adds a new device driver to a running system. Contains options to add device policy to the new device. Typically, this command is called in a script when a device driver is being installed.
`devfsadm`(8)	Administers devices and device drivers on a running system. Also loads device policy. The `devfsadm` command enables the cleanup of dangling `/dev` links to disk, tape, port, audio, and pseudo devices. Devices for a named driver can also be reconfigured.
`getdevpolicy`(8)	Displays the policy associated with one or more devices. This command can be run by any user.
`rem_drv`(8)	Removes a device or device driver.
`update_drv`(8)	Updates the attributes of an existing device driver. Contains options to update the device policy for the device. Typically, this command is called in a script when a device driver is being installed.

Device Allocation

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

Components of Device Allocation

The components of the device allocation mechanism are as follows:

- The `svc:/system/device/allocate` service. For more information, see the `smf`(7) man page and the man pages for the device allocation commands.
- The `allocate`, `deallocate`, `dminfo`, and `list_devices` commands. For more information, see Device Allocation Commands.
- The Device Management and Device Security rights profiles. For more information, see Device Allocation Rights Profiles.
- Device-clean scripts for each allocatable device.

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

- The `/etc/security/device_allocate` file. For more information, see the `device_allocate`(5) man page.
- The `/etc/security/device_maps` file. For more information, see the `device_maps`(5) man page.
- A lock file, in the `/etc/security/dev` directory, for each allocatable device.
- The changed attributes of the lock files that are associated with each allocatable device.

Device Allocation Service

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

Device Allocation Rights Profiles

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

These rights profiles include the following authorizations:

- `solaris.device.allocate` – Required to allocate a device
- `solaris.device.cdrw` – Required to read and write a CD-ROM
- `solaris.device.config` – Required to configure the attributes of a device
- `solaris.device.mount.alloptions.fixed` – Required to specify mount options when mounting a fixed device
- `solaris.device.mount.alloptions.removable` – Required to specify mount options when mounting a removable device
- `solaris.device.mount.fixed` – Required to mount a fixed device
- `solaris.device.mount.removable` – Required to mount a removable device

- `solaris.device.revoke` – Required to revoke or reclaim a device

Device Allocation Commands

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

Table 4-4 Device Allocation Commands

Man Page for Command	Purpose
`allocate`(1)	Reserves an allocatable device for use by one user. By default, a user must have the `solaris.device.allocate` authorization to allocate a device. You can modify the `device_allocate` file to not require user authorization. Then, any user on the system can request the device to be allocated for use.
`deallocate`(1)	Removes the allocation reservation from a device.
`dminfo`(8)	Searches for an allocatable device by device type, by device name, and by full path name.
`list_devices`(1)	Lists the status of allocatable devices. Lists all the device-special files that are associated with any device that is listed in the `device_maps` file. With the `-U` option, lists the devices that are allocatable or allocated to the specified user ID. This option allows you to check which devices are allocatable or allocated to another user. You must have the `solaris.device.revoke` authorization.

Authorizations for the Allocation Commands

By default, users must have the `solaris.device.allocate` authorization to reserve an allocatable device. To create a rights profile to include the `solaris.device.allocate` authorization, see How to Authorize Users to Allocate a Device.

Administrators must have the `solaris.device.revoke` authorization to change the allocation state of any device. For example, the `-U` option of the `allocate` and `list_devices` commands, and the `-F` option of the `deallocate` command require the `solaris.device.revoke` authorization.

For more information, see Selected Commands That Require Authorizations in *Securing Users and Processes in Oracle Solaris 11.4*.

Allocate Error State

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

The `deallocate` command with the `-F` option forces deallocation. Or, you can use `allocate -U` to assign the device to a user. Once the device is allocated, you can

investigate any error messages that appear. After any problems with the device are corrected, you can forcibly deallocate it.

`device_maps` File

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

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

Each device is represented by a one-line entry in the following format:

device-name:*device-type*:*device-list*

Example 4-10 Sample `device_maps` Entry

The following example shows an entry in a `device_maps` file.

```
audio0:\
audio:\
/dev/audio /dev/audioctl /dev/dsp /dev/dsp0 /dev/mixer0 /dev/sound/0
/dev/sound/0ctl /dev/sound/audio810\:0mixer /dev/sound/audio810\:0dsp
/dev/sound/audio810\:0 /dev/sound/audio810\:0ctl
```

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

device-name
Specifies the name of the device. For a list of current device names, see Viewing Allocation Information About a Device.

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

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

`device_allocate` File

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

An entry in the `device_allocate` file does not mean that the device is allocatable, unless the entry specifically states that the device is allocatable.

In the `device_allocate` file, each device is represented by a one-line entry in the following format:

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

The following example shows a sample `device_allocate` file.

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

Note the asterisk (*) in the fifth field of the `audio` device entry.

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

device-name
Specifies the name of the device. For a list of current device names, see Viewing Allocation Information About a Device.

device-type
Specifies the generic device type. The generic name is the name for the class of devices, such as `st`, `fd`, and `sr`. The *device-type* field logically groups related devices. When you make a device allocatable, retrieve the device name from the *device-type* field in the `device_maps` file.

`reserved`
Oracle reserves the two fields that are marked `reserved` for future use.

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

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

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

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

You can decide to accept the default devices and their defined characteristics. After you install a new device, you can modify the entries. Any device that needs to be allocated before use must be defined in the `device_allocate` and `device_maps` files for that device's system. Currently, cartridge tape drives, CD-ROM drives,

removable media devices, and audio chips are considered allocatable. These device types have device-clean scripts.

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

Device-Clean Scripts

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

Device-Clean Script for Tapes

The `st_clean` device-clean script supports three tape devices:

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

The `st_clean` script uses the `rewoffl` option to the `mt` command to clean up the device. For more information, see the `mt`(1) man page. If the script runs during system boot, the script queries the device to determine whether the device is online. If the device is online, the script determines whether the device has media in it. The ¼-inch tape devices that have media in them are placed in the allocate error state. The allocate error state forces the administrator to manually clean up the device.

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

Device-Clean Scripts for CD-ROM Drives

The **sr_clean**device-clean script is provided for CD-ROM drives:

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

Device-Clean Script for Audio

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

Writing New Device-Clean Scripts

If you add more allocatable devices to the system, you might need to create your own device-clean scripts. The `deallocate` command passes a parameter to the device-clean scripts. The parameter, which is shown here, is a string that contains the device name. For more information, see the `device_allocate`(5) man page.

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

Device-clean scripts must return "`0`" for success and greater than "`0`" for failure. The options `-I`, `-f`, and `-S` determine the running mode of the script:

`-I`
Needed during system boot only. All output must go to the system console. Failure or inability to forcibly eject the media must put the device in the allocate error state.

`-i`
Similar to the `-I` option, except that output is suppressed.

`-f`
For forced cleanup. The option is interactive and assumes that the user is available to respond to prompts. A script with this option must attempt to complete the cleanup if one part of the cleanup fails.

`-S`
Standard cleanup. The option is interactive and assumes that the user is available to respond to prompts.

5

Scanning for Viruses

This chapter provides information about using antivirus software, and covers the following topics:

About Virus Scanning

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

The `vscan` service can be configured to use multiple scan engines. It is recommended that the `vscan` service use a minimum of two scan engines. The requests for virus scans are distributed among all available scan engines.

> **Note:**
>
> Oracle Solaris 11.4 does not endorse or support any third-party product that uses kernel modules on Oracle Solaris 11.4 systems to scan for viruses, scan for malware, or enforce a security policy.

About the `vscan` Service

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

The following describes the virus scanning process:

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

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

Using the `vscan` Service

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

- The virus scanning package is installed.
- At least one scan engine is installed and configured.
- The files reside on a file system that supports virus scanning.
- Virus scanning is enabled on the file system.
- The `vscan` service is enabled.
- The `vscan` service is configured to scan files of the specified file type.

The following table points to the tasks you perform to set up the vscan service.

Task	Description	For Instructions
Install a scan engine.	Installs and configures one or more of the supported third-party products listed in Antivirus Scan Engine Software.	See the product documentation.
Install the `virus-scan` IPS package.	Installs the virus scanning package.	How to Install Virus Scanning Software
Enable the file system to allow virus scans.	Enables virus scans on a ZFS file system. By default, scans are disabled.	How to Enable Virus Scanning on a File System
Enable the `vscan` service.	Starts the scan service.	How to Enable the vscan Service
Add a scan engine to the `vscan` service.	Includes specific scan engines in the `vscan` service.	How to Add a Scan Engine
Configure the `vscan` service.	Views and changes vscan properties.	How to View vscan Properties How to Change vscan Properties
Configure the `vscan` service for specific file types.	Specifies the file types to include and exclude in a scan.	How to Exclude Files From Virus Scans

How to Install Virus Scanning Software

You must become an administrator who is assigned the Software Installation rights profile. For more information, see Using Your Assigned Administrative Rights in *Securing Users and Processes in Oracle Solaris 11.4*.

1. **Install the virus scanning package.**

```
$ pkg install virus-scan
```

2. **(Optional) Confirm the installation.**

```
$ pkg list virus-scan
NAME (PUBLISHER)                          VERSION                    IFO
service/storage/virus-scan                0.11.4.0.............      i--

$ pkg verify -v virus-scan
PACKAGE                                                             STATUS
pkg://solaris/service/storage/virus-scan                              OK
```

How to Enable Virus Scanning on a File System

Use the file system command to allow virus scans of files. For example, to include a ZFS file system in a virus scan, use the `zfs`(8) command.

The ZFS file system allows some administrative tasks to be delegated to specific users. For more information about delegated administration, see Chapter 9, Oracle Solaris ZFS Delegated Administration in *Managing ZFS File Systems in Oracle Solaris 11.4*.

You must become an administrator who is assigned the ZFS File System Management or the ZFS Storage Management rights profile. For more information, see Using Your Assigned Administrative Rights in *Securing Users and Processes in Oracle Solaris 11.4*.

- **Enable virus scanning on a ZFS file system, for example, `pool/volumes/vol1`.**

```
$ zfs set vscan=on path/pool/volumes/vol1
```

How to Enable the `vscan` Service

You must become an administrator who is assigned the VSCAN Management rights profile. For more information, see Using Your Assigned Administrative Rights in *Securing Users and Processes in Oracle Solaris 11.4*.

- **Enable the virus scanning service.**

```
$ svcadm enable vscan
```

How to Add a Scan Engine

You must become an administrator who is assigned the VSCAN Management rights profile. For more information, see Using Your Assigned Administrative Rights in *Securing Users and Processes in Oracle Solaris 11.4*.

- **To add a scan engine to the `vscan` service with default properties, type:**

```
$ vscanadm add-engine engine-ID
```

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

How to View `vscan` Properties

You must become an administrator who is assigned the VSCAN Management rights profile. For more information, see Using Your Assigned Administrative Rights in *Securing Users and Processes in Oracle Solaris 11.4*.

- **View the properties of the `vscan` service, of all scan engines, or of a specific scan engine.**
  - **To view the properties of a particular scan engine, type:**

    ```
    $ vscanadm get-engine engine-ID
    ```
  - **To view the properties of all scan engines, type:**

    ```
    $ vscanadm get-engine
    ```
  - **To view one of the properties of the `vscan` service, type:**

    ```
    $ vscanadm get -p property
    ```

    where *property* is one of the parameters described in the man page for the `vscanadm`(8) command.

    For example, if you want to see the maximum size of a file that can be scanned, type:

    ```
    $ vscanadm get max-size
    ```

How to Change `vscan` Properties

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

You must become an administrator who is assigned the VSCAN Management rights profile. For more information, see Using Your Assigned Administrative Rights in *Securing Users and Processes in Oracle Solaris 11.4*.

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

   ```
   $ vscanadm show
   max-size=1GB
   max-size-action=allow
   timeout=30
   ...
   ```

2. **Set the `timeout` value.**

   If the scanning time exceeds the timeout, access to the file is denied.

   ```
   $ vscanadm set -p timeout=60
   ```

3. **Set the maximum size for virus scans to, for example, 128 megabytes.**

   ```
   $ vscanadm set -p max-size=128M
   ```

4. **Specify that access is denied to any file that is not scanned due to its size.**

   ```
   $ vscanadm set -p max-size-action=deny
   ```

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

How to Exclude Files From Virus Scans

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

You must become an administrator who is assigned the VSCAN Management rights profile. For more information, see Using Your Assigned Administrative Rights in *Securing Users and Processes in Oracle Solaris 11.4*.

1. **View the list of all file types that are included in the virus scan.**

   ```
   $ vscanadm get -p types
   ```

2. **Specify the types of files to be scanned for viruses:**

   - **Exclude a specific file type, for example the JPEG type, from the virus scan.**

     ```
     $ vscanadm set -p types=-jpg,+*
     ```

   - **Include a specific file type, for example executable files, in the virus scan.**

     ```
     $ vscanadm set -p types=+exe,-*
     ```

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

6
Systems and Devices Glossary

These glossary entries cover words that can be ambiguous because they are used differently in different parts of the operating system, or have meanings in Oracle Solaris that are distinct from other operating systems.

device allocation

Device protection for your systems at the user level. Device allocation enforces the exclusive use of a device by one user at a time. Device data is purged before device reuse. Authorizations can be used to limit who is permitted to allocate a device.

device policy

Device protection for your systems at the kernel level. Device policy is implemented as two sets of privileges on a device. One set of privileges controls read access to the device. The second set of privileges controls write access to the device. See also policy.

least privilege

A security model which gives a specified process only a subset of superuser powers. The least privilege model assigns enough privilege to regular users that they can perform personal administrative tasks, such as mount file systems and change the ownership of files. On the other hand, processes run with just those privileges that they need to complete the task, rather than with the full power of superuser, that is, all privileges. Damage due to programming errors like buffer overflows can be contained to a non-root user, which has no access to critical abilities like reading or writing protected system files or halting the system.

password policy

The encryption algorithms that can be used to generate passwords. Can also refer to more general issues around passwords, such as how often the passwords must be changed, how many password attempts are permitted, and other security considerations. Security policy requires passwords. Password policy might require passwords to be encrypted with the AES algorithm, and might make further requirements related to password strength.

policy

Generally, a plan or course of action that influences or determines decisions and actions. For computer systems, policy typically means security policy. Your site's security policy is the set of rules that define the sensitivity of the information that is being processed and the measures that are used to protect the information from unauthorized access. For example, security policy might require that systems be audited, that system devices must be allocated for use, and that passwords be changed every six weeks.

For the implementation of policy in specific areas of the Oracle Solaris OS, see device policy and password policy.

privilege

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

2. A discrete right on a process in an Oracle Solaris system. Privileges offer a finer-grained control of processes than does `root`. Privileges are defined and enforced in the kernel. Privileges are also called *process privileges* or *kernel privileges*. For a full description of privileges, see the `privileges`(7) man page.

privilege model

A stricter model of security on a computer system than the superuser model. In the privilege model, processes require privilege to run. Administration of the system can be divided into discrete parts that are based on the privileges that administrators have in their processes. Privileges can be assigned to an administrator's login process. Or, privileges can be assigned to be in effect for certain commands only.

rights

An alternative to the all-or-nothing superuser model. User rights management and process rights management enable an organization to divide up superuser's privileges and assign them to users or roles. Rights in Oracle Solaris are implemented as kernel privileges, authorizations, and the ability to run a process as a specific UID or GID. Rights can be collected in a rights profile.

rights profile

Also referred to as a profile. A collection of security overrides that can be assigned to a role or user. A rights profile can include authorizations, privileges, commands with security attributes, and other rights profiles that are called supplementary profiles.

Secure Shell

A special protocol for secure remote login and other secure network services over an insecure network.

security attributes

Overrides to security policy that enable an administrative command to succeed when the command is run by a user other than superuser. In the superuser model, the `setuid root` and `setgid` programs are security attributes. When these attributes are applied to a command, the command succeeds no matter who runs the command. In the privilege model, kernel privileges and other rights replace `setuid root` programs as security attributes. The privilege model is compatible with the superuser

model, in that the privilege model also recognizes the `setuid` and `setgid` programs as security attributes.

security policy

See policy.

superuser model

The typical UNIX model of security on a computer system. In the superuser model, an administrator has all-or-nothing control of the system. Typically, to administer the system, a user becomes superuser (`root`) and can do all administrative activities.

trusted users

Users whom you have decided can perform administrative tasks at some level of trust. Typically, administrators create logins for trusted users first and assign administrative rights that match the users' level of trust and ability. These users then help configure and maintain the system. Also called *privileged users*.

Index

Symbols

Numerics

A

B

C

D

E

F

G

H

I

K

L

M

N

O

P

R

S

T

U

V

X

Z