Securing Systems and Attached Devices in Oracle® Solaris 11.3

Part No: E54828
November 2016

ORACLE

Securing Systems and Attached Devices in Oracle Solaris 11.3

Part No: E54828

Copyright © 2002, 2016, Oracle and/or its affiliates. All rights reserved.

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

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

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

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

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

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

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

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

Documentation Accessibility

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

Access to Oracle Support

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

Référence: E54828

Copyright © 2002, 2016, Oracle et/ou ses affiliés. Tous droits réservés.

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

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

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

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

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

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

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

Ce logiciel ou matériel et la documentation qui l'accompagne peuvent fournir des informations ou des liens donnant accès à des contenus, des produits et des services émanant de tiers. Oracle Corporation et ses affiliés déclinent toute responsabilité ou garantie expresse quant aux contenus, produits ou services émanant de tiers, sauf mention contraire stipulée dans un contrat entre vous et Oracle. En aucun cas, Oracle Corporation et ses affiliés ne sauraient être tenus pour responsables des pertes subies, des coûts occasionnés ou des dommages causés par l'accès à des contenus, produits ou services tiers, ou à leur utilisation, sauf mention contraire stipulée dans un contrat entre vous et Oracle.

Accessibilité de la documentation

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

Accès aux services de support Oracle

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

Contents

Using This Documentation

Securing Systems and Attached Devices in Oracle® Solaris 11.3 explains how to protext and monitor your Oracle Solaris system from unauthorized access.

- **Overview** – Describes different methods of securing systems and devices from unauthorized access.
- **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=E53394`.

Feedback

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

♦ ♦ ♦ CHAPTER 1

Managing Computer System Security

Keeping a computer system's information secure 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 11.3

The following new features in this release secure systems and attached devices from unauthorized access.

- "Verified Boot has One Policy: `boot_policy`" on page 11
- "Protecting the Process Heap and Executable Stacks From Compromise" on page 31
- Configuring verified boot not only on systems, but on Kernel Zones. See "Using Verified Boot to Secure an Oracle Solaris Kernel Zone" in *Creating and Using Oracle Solaris Kernel Zones*.

Verified Boot has One Policy: `boot_policy`

Starting in the Oracle Solaris 11.3 release, verified boot has only one policy value: `boot_policy`. SPARC Firmware 9.5.0 and above support only one policy setting, `boot_policy`.

The `boot_policy` value now controls all checking including checking the `bootblk` and the loading of all kernel modules, including `unix` and `genunix`. The `module_policy` value is ignored. In order to best transition from Solaris 11.2 and older SPARC Firmware, the `boot_policy` and `module policy` values should always be the same.

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

For further information, see "Policies for Verified Boot" on page 36.

Controlling Access to a Computer System

In the workplace, all computers that are connected to a network 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" on page 60. To protect x86 hardware, consult the vendor documentation.

Maintaining Login Control

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(1M)` man page. For information about naming services and directory services, see *Working With Oracle Solaris 11.3 Directory and Naming Services: DNS and NIS* and *Working With Oracle Solaris 11.3 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 Kerberos and Other Authentication Services in Oracle Solaris 11.3*.

Sophisticated authentication and authorization mechanisms are available on Oracle Solaris systems. For a discussion of authentication and authorization mechanisms at the network level, see "Authentication and Authorization for Remote Access" on page 28.

Managing Password Information

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

Passwords are initially created when you set up a user account. To maintain security on user accounts, you can set up password aging to force users to routinely change their passwords. You can also disable a user account by locking the password. For detailed information about administering passwords, see Chapter 1, "About User Accounts and User Environments" in *Managing User Accounts and User Environments in Oracle Solaris 11.3* and the `passwd(1)` man page.

Local Passwords

If your network uses local files to authenticate users, the password information is kept in the system's `/etc/passwd` and `/etc/shadow` files. The user 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 11.3 Directory and Naming Services: LDAP*.

Password Encryption

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

Note - To be FIPS 140-approved, use the SHA algorithms. For information, see "passwd Command as a FIPS 140-2 Consumer" in *Using a FIPS 140 Enabled System in Oracle Solaris 11.3*.

Password Algorithm Identifiers

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

Note - Use FIPS-approved algorithms when possible. For lists of FIPS-approved algorithms, see "FIPS 140-2 Algorithm Lists and Certificate References for Oracle Solaris Systems" in *Using a FIPS 140 Enabled System in Oracle Solaris 11.3*.

TABLE 1 Password Encryption Algorithms

Identifier	Description	Algorithm Man Page
`1`	The MD5 algorithm that is compatible with MD5 algorithms on BSD and Linux systems.	`crypt_bsdmd5`(5)
`2a`	The Blowfish algorithm that is compatible with the Blowfish algorithm on BSD systems. **Note -** To promote FIPS 140 security, remove the Blowfish algorithm (2a) from the `CRYPT_ALGORITHMS_ALLOW=2a,5,6` entry in the `/etc/security/policy.conf` file.	`crypt_bsdbf`(5)
`md5`	The Sun MD5 algorithm, which is considered stronger than the BSD and Linux version of MD5.	`crypt_sunmd5`(5)
`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`(5)
`6`	The SHA512 algorithm.	`crypt_sha512`(5)
`__unix__`	Deprecated. The traditional UNIX encryption algorithm. This algorithm can be of use when connecting to old systems.	`crypt_unix`(5)

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" on page 54.

Algorithms Configuration in the `policy.conf` File

The default algorithms configuration in the `policy.conf` file is as follows:

```
#
…
# crypt(3c) Algorithms Configuration
#
# CRYPT_ALGORITHMS_ALLOW specifies the algorithms that are allowed
to
# be used for new passwords.  This is enforced only in crypt_gensalt(3c).
#
CRYPT_ALGORITHMS_ALLOW=1,2a,md5,5,6

# To deprecate use of the traditional unix algorithm, uncomment below
# and change CRYPT_DEFAULT= to another algorithm.  For example,
# CRYPT_DEFAULT=1 for BSD/Linux MD5.
#
#CRYPT_ALGORITHMS_DEPRECATE=__unix__

# The Oracle Solaris default is a SHA256 based algorithm.  To revert to
# the policy present in Solaris releases set CRYPT_DEFAULT=__unix__,
# which is not listed in crypt.conf(4) since it is internal to libc.
#
CRYPT_DEFAULT=5
…
```

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

When existing users change their passwords, the way their old password was encrypted affects which algorithm is used to encrypt the new password. For example, assume that `CRYPT_ALGORITHMS_ALLOW=1,2a,md5,5,6`, and `CRYPT_DEFAULT=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.

Initial Password	Changed Password	Explanation
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 policy.conf(4) man page. To specify password encryption algorithms, see "Changing the Default Algorithm for Password Encryption" on page 54.

Special System Accounts

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

Caution - Never change the password setting of a system account. System accounts from Oracle Solaris are delivered in a safe and secure state. 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 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-to-UNIX copy program.
nuucp	9	Used by remote systems to log in to the system and start file transfers.

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 Table 4, "Securing Logins and Passwords Task Map," on page 51.

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

Controlling Access to Devices

Peripheral devices that are attached to a computer system pose a security risk. Microphones can pick up conversations and transmit them to remote systems. CD-ROMs can leave their information behind for reading by the next user of the CD-ROM device. Printers can be accessed remotely. Devices that are integral to the system, 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*. For a more general discussion of devices and zones, see "/dev File System in Non-Global Zones" in *Oracle Solaris Zones Configuration Resources*.

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 11.3*.

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:

- Table 5, "Configuring Device Policy Task Map," on page 63
- "Device Policy Commands" on page 74
- "Privileges and Devices" in *Securing Users and Processes in Oracle Solaris 11.3*

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" on page 81.

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:

- Table 6, "Managing Device Allocation Task Map," on page 65
- "Device Allocation" on page 75
- "Device Allocation Commands" on page 76

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.

Address Space Layout Randomization

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

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

The three ASLR configurations are:

- **Disabled –** ASLR is disabled for all binaries.
- **Tagged binaries –** ASLR is controlled by the tag that is coded in the binaries.

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

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

Limiting and Monitoring Superuser Access

Your system requires a `root` password for superuser access. In the default configuration, a user cannot remotely log in to a system as `root`. When logging in remotely, 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" on page 57.

Configuring Role-Based Access Control to Replace Superuser

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

Each role requires that a known user log in with her or his user name and password. After logging in, the user then assumes the role with a specific role password. For more information about RBAC, see "User Rights Management" in *Securing Users and Processes in Oracle Solaris 11.3*.

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 -s1m rsh` command to see the `rsh`(1M) man page.

Restricting Access to Data in Files

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

You might want to allow some users to read some files, and give other users permission to change or delete some files. You might have some data that you do not want anyone else to see. Chapter 1, "Controlling Access to Files" in *Securing Files and Verifying File Integrity in Oracle Solaris 11.3* 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 11.3*. For procedures, see "Protecting Against Programs With Security Risk" in *Securing Files and Verifying File Integrity in Oracle Solaris 11.3*.

Using the Secure by Default Configuration

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

Using Resource Management Features

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

With these resource management features, you can designate resources for particular projects. You can also dynamically adjust the resources that are available. For more information, see *Administering Resource Management in Oracle Solaris 11.3*.

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 network server. For more information, see *Introduction to Oracle Solaris Zones*.

Monitoring Use of System Resources

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 11.3*.

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 2, "Verifying File Integrity by Using BART" in *Securing Files and Verifying File Integrity in Oracle Solaris 11.3*.

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 11.3*. For step-by-step instructions on setting appropriate permissions on files, see "Protecting Files" in *Securing Files and Verifying File Integrity in Oracle Solaris 11.3*.

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 11.3*.

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, "Cryptographic Framework" in *Managing Encryption and Certificates in Oracle Solaris 11.3*.

Using Access Control Lists

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

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

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

To protect ZFS files with access control lists (ACLs), see Chapter 9, "Using ACLs and Attributes to Protect Oracle Solaris ZFS Files" in *Managing ZFS File Systems in Oracle Solaris 11.3*. 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 11.3*.

Sharing Files Across Computer 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 11.3*.

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 system 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 11.3*.

Controlling Network Access

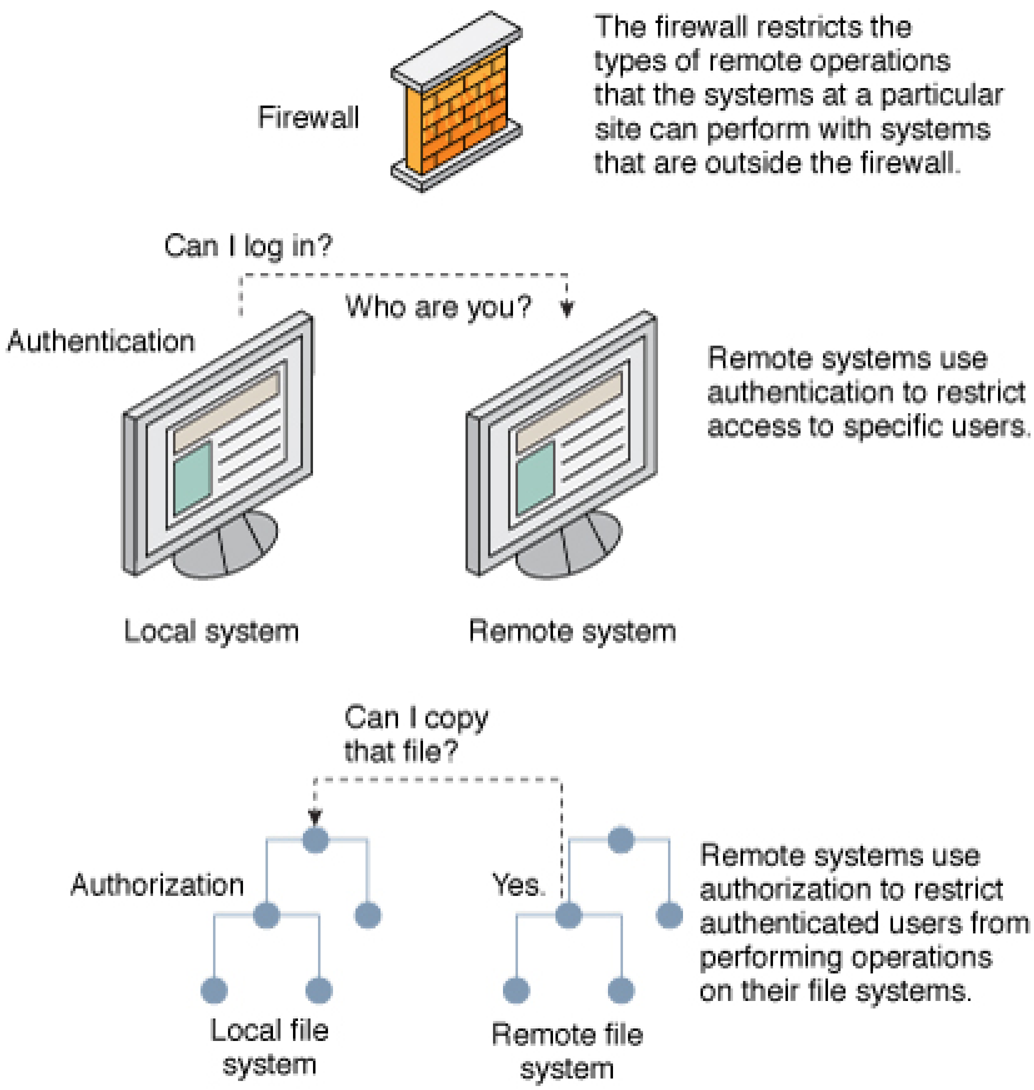
Computers are often part of a network of computers that allows connected computers to exchange information. Networked computers can access data and other resources from other computers on the network. Although computer networks create a powerful and sophisticated computing environment, networks also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

FIGURE 1 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 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 8, "About IP Security Architecture" in *Securing the Network in Oracle Solaris 11.3*
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 and Other Authentication Services in Oracle Solaris 11.3*.
LDAP	The LDAP directory service can provide both authentication and authorization at the network level.	*Working With Oracle Solaris 11.3 Directory and Naming Services: DNS and NIS*
Remote login commands	The remote login commands enable users to log in to a remote system over the network and use its resources. Some of the remote login commands are `rlogin`, `rcp`, and `ftp`. If you are a trusted host, authentication is automatic. Otherwise, you are asked to authenticate yourself.	Chapter 3, "Accessing Remote Systems" in *Managing Remote Systems in Oracle Solaris 11.3*
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 Kerberos and Other Authentication Services in Oracle Solaris 11.3*
Secure RPC	Secure RPC improves the security of network environments by authenticating users who make requests on remote systems. You can use either a UNIX, DES, or Kerberos authentication mechanism for Secure RPC.	"About Secure RPC" in *Managing Kerberos and Other Authentication Services in Oracle Solaris 11.3*
Secure NFS	Secure RPC can also be used to provide additional security in an NFS environment. An NFS environment with secure RPC is called Secure NFS.	"NFS Services and Secure RPC" in *Managing Kerberos and Other Authentication Services in Oracle Solaris 11.3*
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 11.3*

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

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

```
“Weak Authentication
NFS request from unprivileged port”
```

Firewall Systems

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

A firewall acts as a gateway and as a barrier. As a gateway, it passes data between the networks. As a barrier, it blocks the free passage of data to and from the network. A user on the internal network must log in to the firewall system to access 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 network servers.

IPsec and the IP Filter feature of Oracle Solaris can provide firewall protection. For more information about protecting network traffic, see *Securing the Network in Oracle Solaris 11.3*.

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 network 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 11.3*.
- To encrypt and authenticate data across a network, see Chapter 2, "About the Kerberos Service" in *Managing Kerberos and Other Authentication Services in Oracle Solaris 11.3*.
- To encrypt IP datagrams, see Chapter 8, "About IP Security Architecture" in *Securing the Network in Oracle Solaris 11.3*.

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 CERT/CC (`http://www.cert.org/contact_cert/`) web site.

♦ ♦ ♦ CHAPTER 2

Protecting Oracle Solaris Systems 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:

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" on page 32 and the sxadm(1M) 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" on page 32 and the sxadm(1M) 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" on page 31.

Before You Begin You must become an administrator who is assigned the `sys_config` privilege. The `root` role has that privilege by default. For more information, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.3*.

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

   The `-p` option provides parseable output. For example:

```
$ sxadm info
EXTENSION           STATUS                       CONFIGURATION
aslr                enabled (tagged-files)       default (default)
nxstack             enabled (all)                enabled (default)
nxheap              disabled                     disabled

$ sxadm info -p
aslr:enabled.tagged-files:enabled.default
nxstack:enabled.all:enabled.all
nxheap:disabled:disabled
```

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 info
EXTENSION           STATUS                       CONFIGURATION
aslr                enabled (tagged-files)       default (default)
nxstack             enabled (all)                enabled (default)
nxheap              enabled (tagged-files)       enabled (default)
```

   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" on page 32.

Using Verified Boot

Verified boot in Oracle Solaris secures a system's boot process. The feature 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

In Oracle Solaris, verified boot is configurable not only on systems but also on Kernel Zones. For more information, refer to "Using Verified Boot to Secure an Oracle Solaris Kernel Zone" in *Creating and Using Oracle Solaris Kernel Zones*.

Verified boot is also configurable in LDOMS. For more information, refer to "Using Verified Boot" in "Oracle VM Server for SPARC 3.4 Administration Guide".

Note - By default, any domain created by using a version of Oracle VM Server for SPARC earlier than version 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 Oracle VM Server for SPARC update.

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 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 in the `/etc/certs` file 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 systems, as follows:

- SPARC systems with verified boot support in their Oracle Integrated Lights Out Manager (ILOM) - Keys and configuration settings are stored in Oracle 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 Oracle ILOM is secure to prevent unauthorized changes to the verified boot configuration. For more information about securing Oracle ILOM, refer to the documentation at `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 Oracle ILOM. The SPARC firmware sends the configuration information to Oracle Solaris.
- Fujitsu M10 systems - Configuration settings are stored in the system's XSCF. The Fujitsu M10 XSCF firmware sends the configuration information, such as policies for verified boot and enabling certificates, to Oracle Solaris.

  For more information about configuring the verified boot, refer to *Fujitsu M10/SPARC M10 Systems System Operation and Administration Guide*. For the XCP firmware version that supports Fujitsu M10 systems verified boot, see the latest version of *Fujitsu M10/SPARC M10 Systems Product Notes*.

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.

Consider the boot flow of Oracle Solaris on a SPARC system:

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

SPARC firmware is installed at the factory. You can also update the firmware by using the `fwupdate` utility. The firmware verifies, and then loads, the Oracle Solaris `/platform/.../unix` module, which is the initial Oracle Solaris module. In turn, the Oracle Solaris kernel runtime loader `krtld`, which is part of the module, verifies and loads the generic UNIX (`genunix`) module and subsequent modules.

Policies for Verified Boot

Starting in the Oracle Solaris 11.3 release, verified boot has only one policy value: `boot_policy`. See "Verified Boot has One Policy: `boot_policy`" on page 11.

The `boot_policy` variable manages verified boot behavior when loading kernel modules during the boot process.

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

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

- `none` - No boot verification is performed. By default, `boot_policy` is not configured and therefore verified boot is disabled.
- `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`.

In addition to configuring boot_policy, you also specify `elfsign` X.509 public key certificates on the system. Similar to the modules, you specify the certificates by either using a variable or defining an Oracle ILOM property. See "Managing Certificates on Systems With Oracle ILOM Verified Boot Support" on page 36.

On systems with Oracle ILOM that supports verified boot, a preinstalled verified boot certificate file, `/etc/certs/ORCLS11SE`, is provided as part of Oracle ILOM.

The certificate contains the RSA public key that is used to verify the `elfsign` signatures in ELF objects. However, you can install a company-provided certificate to replace `/etc/certs/ORCLS11SE`. All certificates are loaded and managed on each individual PDomain.

Managing Certificates on Systems With Oracle ILOM Verified Boot Support

Use the following options to manage the system's verified boot certificates.

- To replace the preinstalled certificate with a certificate provided by the user:

```
--> load /HOSTx/verified_boot/cert -source ftp://server/filename
```

- To save a copy of the current certificate to the location that is provided by the user:

```
--> dump /HOSTn/verified_boot/cert -dest ftp://server/filename
```

- To remove any user-installed certificate and revert to the system's preinstalled certificate:

```
--> reset /HOSTn/verified_boot/cert
```

Enabling Verified Boot

By default, verified boot is disabled on systems. The procedures to enable the feature differ depending on your system. To enable the feature, use the procedure in this section that applies to your system.

▼ SPARC: How to Enable Verified Boot on SPARC Systems With Oracle ILOM Verified Boot Support

For SPARC systems with Oracle ILOM verified boot support, the verified boot properties are in `/HOSTn/verified_boot`, where *n* is the PDomain number, such as `HOST0`, `HOST1`, and so on.

Note - Some SPARC systems have only one physical domain, `/HOST`, while others have multiple physical domains. This procedure assumes that you are using a system with multiple physical domains and refers to a physical domain as `/HOSTn`. For security features that are specific to your system, refer to your system's security manual.

---

1. **(Optional) Determine whether your system supports verified boot.**

   ```
   # show /HOSTn/verified_boot
   show: Invalid target /HOST/verified_boot
   ```

   You can use the `fwupdate` to update the system's Oracle ILOM firmware.

2. **As an administrator, log in to the Oracle ILOM user interface.**

   ```
   % ssh root@ILOM
   ```

   where *ILOM* can be either the Oracle ILOM service processor IP address or the chassis-monitoring module IP address.

3. **Configure the verified boot properties.**

   ```
   --> set /HOSTn/verified_boot boot_policy=warning
   ```

Note - Specify either `warning` or `enforce` for each property. The properties can have differing configurations. For an explanation of these policy configurations, see "Policies for Verified Boot" on page 36.

If the boot policy is configured with `enforce` and discrepancies in the UNIX or `genunix` modules are detected, the system does not boot. Instead, the system reverts to OpenBoot PROM (OBP).

---

4. **Specify the certificate that you want to use in place of the certificate that is provided with the system.**

```
--> load /HOSTn/verified_boot/cert -source FTP-location
```

where *FTP-location* refers to the FTP server and file name that stores the certificate. *FTP-location* must be in the URL format `ftp://`*server*`/`*filename*.

5. **(Optional) Display the verified boot configuration.**

```
--> show /HOSTn/verified_boot
/HOST0
Properties:
boot_policy = warningcert = ftp://server/filename
```

6. **Bring the operating system to the OBP prompt.**

```
# halt
```

Alternately, type `init 0`.

7. **For the changes to be effective, at OBP, disable `nvramrc` and reset the OBP.**

```
OK> setenv use-nvramrc? false
OK> reset-all
```

Note - If `use-nvramrc? = true`, the operating system will fail to boot and will return to the OBP prompt.

Alternately, reset the system using a DC power cycle.

▼ How to Manually Verify the `elfsign` Signature

Verified boot is an automatic mechanism that provides a quick and efficient way to ensure the integrity of the boot process. However, you can still verify a kernel module's signature manually.

- **Use the `elfsign` command syntax as follows:**

  $ **elfsign verify -v** *kernel_module*

  For example:

```
$ elfsign verify -v /kernel/misc/sparcv9/cardbus
elfsign: verification of /kernel/misc/sparcv9/cardbus passed.
format: rsa_sha1.
signer: O=Oracle Corporation, OU=Corporate Object Signing, \
               OU=Solaris Signed Execution, CN=Solaris 11
```

About 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 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`(1M) man page.

The Trusted 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 differ 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 `http://www.trustedcomputinggroup.org/resources/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.

- **On 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 the device might be disabled. For information about how to enable the device, see either "How to Initialize TPM Using the Oracle ILOM Interface" on page 41 or "How to Initialize TPM Using BIOS" on page 44 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 what is provided by the `prtconf` syntax.

```
# ls -l /dev/tpm
lrwxrwxrwx  1 root root  44 May 22 2012 /dev/tpm ->
../devices/pci@0,0/isa@lf/tpm@0,fed40000:tpm
```

SPARC: How to Initialize TPM Using the Oracle ILOM Interface

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

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

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

   - **For single-host network 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 host does not shut down using the step above.

   ---

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 /HOSTn/tpm mode=activated
```

Where *n* is an instance number, for example, `HOST0/tpm`.

- 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 11.3 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 "TPM Failover Option" on page 49 and the `tpadm`(1M) 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" on page 43.**

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'
```

▼ SPARC: How to Back Up TPM Data and Keys

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 multidomain systems with Oracle Solaris 11.3 installed, use the `tpmadm failover` command to specify that TPM data and keys are automatically backed up to the Standby SP on the network 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 the Oracle ILOM Interface" on page 41.

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.

1. **Log in and assume the root role.**

   For information, see "User and Process Rights Provide an Alternative to the Superuser Model" in *Securing Users and Processes in Oracle Solaris 11.3*.

2. **At the Solaris prompt, ensure that TPM is enabled.**

   ```
   # tpmadm status
   ```

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

3. **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.

4. **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
   ```

▼ x86: How to Initialize TPM Using BIOS

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

1. **At the Oracle Solaris prompt, 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. Therefore, 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

Before You Begin To perform this procedure, you will need to install and enable TPM on the system. Ensure that the `tcsd` daemon is also running.

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 1 Enabling PKCS #11 Consumers to Use TPM

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

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

Troubleshooting TPM

This section covers the following:

- “Monitoring TPM Status” on page 47
- “TPM Failover Option” on page 49
- “Migrating or Restoring TPM Data and Keys” on page 50

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: 0.3.6
Build Release: 5.11
Branch: 0.175.1.0.0.24.0
Packaging Date: September 4, 2012 05:28:21 PM
Size: 3.65 MB
FMRI: pkg://solaris/library/security/
trousers@0.3.6,5.11-0.175.1.0.0.24.0:20120904T1728212
```

- 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 Oracle ILOM prompt:

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

SPARC: TPM Failover Option

SPARC multidomain servers that have Oracle Solaris 11.3 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 `tpadm` 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" on page 41. See, also, the `tpadm`(1M) man page.

SPARC: Migrating or Restoring TPM Data and Keys

SPARC multidomain servers that have Oracle Solaris 11.3 installed can, if the `-failover` option was previously enabled, fail over the SP/SPP board that contains the TPM. See "TPM Failover Option" on page 49.

All other platforms must have had a manual backup created. See "How to Back Up TPM Data and Keys" on page 43. 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.

▼ SPARC: How to Migrate or Restore TPM Data and Keys

1. **Log in and assume the root role.**

   For information, see "User and Process Rights Provide an Alternative to the Superuser Model" in *Securing Users and Processes in Oracle Solaris 11.3*.

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

   ```
   # tpmadm migrate import
   ```

3. **Verify the data has been 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
   ```

CHAPTER 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" on page 51
- "Changing the Default Algorithm for Password Encryption" on page 54
- "Monitoring and Restricting `root` Access" on page 57
- "Controlling Access to System Hardware" on page 60

For overview information about system security, see Chapter 1, "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 4 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" in *Oracle Solaris 11 Security and Hardening Guidelines*
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" on page 52
Find users who do not have passwords.	Finds only those users whose accounts do not require a password.	"How to Display Users Without Passwords" on page 53

Task	Description	For Instructions
Disable logins temporarily.	Denies user logins to a computer system as part of system shutdown or routine maintenance.	"How to Temporarily Disable User Logins" on page 53

How to Display the User's Login Status

Before You Begin 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 11.3*.

- **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(1M)` man page.

Example 2 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

Before You Begin 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 11.3*.

● **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 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(4)` man page.

Before You Begin 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 11.3*.

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

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

For an example of using the `solaris.admin.edit/etc/nologin` authorization, see Example 4, "Disabling User Logins," on page 54.

2. **Include a message about system availability.**

3. **Close and save the file.**

Example 4 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 in the policy.conf file.To switch to another algorithm, assign a different identifier. For a list of password encryption algorithms and their corresponding identifiers, see Table 1, "Password Encryption Algorithms," on page 15.

Note - Whenever possible, use FIPS-approved algorithms. See "FIPS 140-2 Algorithm Lists and Certificate References for Oracle Solaris Systems" in *Using a FIPS 140 Enabled System in Oracle Solaris 11.3* for lists of FIPS-approved algorithms and non-approved algorithms.

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 "Algorithms Configuration in the `policy.conf` File" on page 16. 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(4)` man page.

▼ How to Specify an Algorithm for Password Encryption

Before You Begin You must assume the `root` role. For more information, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.3*.

1. **In the `/etc/security/polic.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 security, exclude the Blowfish algorithm (2a) from the entry.

```
CRYPT_ALGORITHMS_ALLOW=5,6
```

Example 5 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.

Before You Begin You must assume the `root` role. For more information, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.3*.

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.

Before You Begin You must assume the `root` role. For more information, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.3*.

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

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

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

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

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

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

Note - To take advantage of password policy controls on the LDAP server, use the `server_policy` option with the `pam_authtok_store` entries in the `pam.conf` file. Passwords are then encrypted on the LDAP server. For the procedure, see Chapter 4, "Setting Up the Oracle Directory Server Enterprise Edition With LDAP Clients" in *Working With Oracle Solaris 11.3 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
```

Before You Begin You must assume the `root` role. For more information, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.3*.

- **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.

Before You Begin You must assume the `root` role. For more information, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.3*.

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

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

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

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

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

```
mach2 % ssh -l root mach1
Password:       <Type root password of mach1>
Password:
Password:
Permission denied (gssapi-keyex,gssapi-with-mic,publickey,keyboard-interactive).
```

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

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

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

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

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

```
% su -
Password:       <Type root password>
#
```

A message is printed on the terminal console.

```
Sep 7 13:22:57 mach1 su: 'su root' succeeded for jdoe on /dev/pts/6
```

Example 6 Logging `root` Access Attempts

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

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

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

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

Troubleshooting

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

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

▼ How to Require a Password for SPARC Hardware Access

Before You Begin

You must become an administrator who is assigned the Device Security, Maintenance and Repair, or System Administrator rights profile. For more information, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.3*.

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`(1M) man page.

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

2. **(Optional) 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.

Before You Begin You must become an administrator who is assigned the `solaris.admin.edit/etc/default/kbd` authorization. By default, the `root` role has this authorization. For more information, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.3*.

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
```

CHAPTER 4

Controlling Access to Devices

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

For overview information about device protection, see "Controlling Access to Devices" on page 18.

Configuring Device Policy

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

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

TABLE 5 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" on page 64
Audit changes in device policy.	Records changes in device policy in the audit trail.	"How to Audit Changes in Device Policy" on page 64
Access `/dev/arp`.	Gets Oracle Solaris IP MIB-II information.	"How to Retrieve IP MIB-II Information From a `/dev/*` Device" on page 65

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 7 Viewing the Device Policy for a Specific Device

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

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

How to Audit Changes in Device Policy

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

Before You Begin You must become an administrator who is assigned the Audit Configuration rights profile. For more information, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.3*.

- **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 11.3*.

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

Device allocation is commonly implemented at sites that require an additional layer of device security for the computer systems. 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" on page 71 for instructions on allocating devices.

TABLE 6 Managing Device Allocation Task Map

Task	Description	For Instructions
Make a device allocatable.	Enables a device to be allocated to one user at a time.	"Enabling or Disabled Device Allocation" on page 66

Task	Description	For Instructions
Disable device allocation.	Removes allocation restrictions from all devices.	
Authorize users to allocate a device.	Assigns device allocation authorizations to users.	"How to Authorize Users to Allocate a Device" on page 67
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" on page 68
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" on page 68
Change the allocation properties of a device.	Changes the requirements for allocating a device.	"Changing Which Devices Can Be Allocated" on page 69
Audit device allocation.	Records device allocation in the audit trail	"Auditing Device Allocation" on page 70
Create a device-clean script.	Purges data from a physical device.	"Writing New Device-Clean Scripts" on page 81

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 11.3*.

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

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

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 for their computer systems.

▼ How to Authorize Users to Allocate a Device

Before You Begin You must become an administrator who is assigned the User Security rights profile. Your rights profiles must include the `solaris.auth.delegate` authorization. For more information, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.3*.

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 11.3*. 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 11.3*. 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.**

Next Steps Teach the users how to use device allocation for their systems.

For examples of allocating removable media, see "How to Allocate a Device" on page 71.

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 11.3*.

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 on your system.

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 11.3*.

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" on page 66. 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 8 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 9 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 10 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 11.3*.

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 11.3*.

Allocating Devices

On your computer systems, you can use device allocation to reserve 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 on their systems.

▼ How to Allocate a Device

Before You Begin Device allocation must be enabled, as described in "Enabling or Disabled Device Allocation" on page 66, 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 11 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" on page 68.

Example 12 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.

Before You Begin You have allocated the device. You are assigned the privileges that are required for mounting the device, as described in "How to Authorize Users to Allocate a Device" on page 67.

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

```
% su - role-name
Password:       &lt;Type role-name password&gt;
$
```

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 13 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 11.3* is a starting point.

▼ How to Deallocate a Device

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

Before You Begin You must have allocated the device. For information, see "How to Allocate a Device" on page 71.

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

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

2. **Deallocate the device.**

```
$ deallocate device-name
```

Example 14 Deallocating a Microphone

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

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

Example 15 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 systems 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 a system's 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 7 Device Management Commands

Command	Purpose
add_drv(1M)	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(1M)	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(1M)	Displays the policy associated with one or more devices. This command can be run by any user.
rem_drv(1M)	Removes a device or device driver.
update_drv(1M)	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 on your systems.

Components of Device Allocation

The components of the device allocation mechanism are as follows:

- The `svc:/system/device/allocate` service. For more information, see the smf(5) man page and the man pages for the device allocation commands.
- The `allocate`, `deallocate`, `dminfo`, and `list_devices` commands. For more information, see "Device Allocation Commands" on page 76.
- The Device Management and Device Security rights profiles. For more information, see "Device Allocation Rights Profiles" on page 76.
- 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(4) man page.
- The `/etc/security/device_maps` file. For more information, see the device_maps(4) man page.
- A lock file, in the `/etc/security/dev` directory, for each allocatable device.
- The changed attributes of the lock files that are associated with each allocatable device.

Device Allocation Service

The `svc:/system/device/allocate` service controls device allocation for your systems. 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 8 Device Allocation Commands

Man Page for Command	Purpose
`allocate(1)`	Reserves an allocatable device for use by one user on a system. 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(1M)`	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 on a system.

Man Page for Command	Purpose
	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 on a system. To create a rights profile to include the `solaris.device.allocate` authorization, see "How to Authorize Users to Allocate a Device" on page 67.

Administrators must have the `solaris.device.revoke` authorization to change the allocation state of any device on a system. 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 11.3*.

Allocate Error State

On your systems, 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 16 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" on page 68.

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

On your systems, 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" on page 68.

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

On your systems, 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(4)` 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.

CHAPTER 5

Virus Scanning Service

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 system, 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. Best practice is to use a minimum of two scan engines. The requests for virus scans are distributed among all available scan engines.

The `vscanadm show` command lists scan engines configured on the system.

```
# vscanadm show
max-size=1GB max-size-action=allow
types=+*
no scan engines configured
```

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.

When a user opens a file from the client system, the virus scanning process operates as follows:

1. 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 scanning is not necessary, then the process ends and the user is permitted to access the file
2. If scanning is necessary, the file is transferred to the scan engine.

   If the transfer is successful, then the engine scans the file using the current virus definitions to determine whether the file is infected.

   If the transfer fails, the process continues as follows:
   - The file is transferred to the next available scan engine that can perform the file scanning.
   - If no alternative engines exist or are available, virus scanning is considered failed and access to the file might be denied.
3. If no virus is detected, the file is tagged with a scan stamp and the client is permitted to access the file.

   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.

Using the `vscan` Service

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

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

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

Task	Description	For Instructions
Install a scan engine.	Installs and configures one or more of the supported third-party antivirus products in Oracle Solaris.	See the product documentation.
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" on page 85
Enable the `vscan` service.	Starts the scan service.	"How to Enable the `vscan` Service" on page 86
Add a scan engine to the `vscan` service.	Includes specific scan engines in the `vscan` service.	"How to Add a Scan Engine" on page 86
Configure the `vscan` service.	Views and changes `vscan` properties.	"How to View Vscan Properties" on page 86 "How to Limit the Size of Scanned Files" on page 87
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" on page 87

▼ How to Enable Virus Scanning on a File System

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

The ZFS file system allows some administrative tasks to be delegated to specific users. For more information about delegated administration, see Chapter 10, "Oracle Solaris ZFS Delegated Administration" in *Managing ZFS File Systems in Oracle Solaris 11.3*.

Before You Begin You must become an administrator who is assigned the ZFS File System Management or the ZFS Storage Management rights profile. For more information, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.3*.

- **Enable virus scanning on a ZFS file system.**

  ```
  # zfs set vscan=on zfs-file-system
  ```

  For example, if the ZFS file system is *path*`/pool/volumes/vol1`, then type the following command:

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

How to Enable the `vscan` Service

Before You Begin You must become an administrator who is assigned the VSCAN Management rights profile. For more information, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.3*.

- **Enable the virus scanning service.**

  ```
  # svcadm enable vscan
  ```

How to Add a Scan Engine

Before You Begin You must become an administrator who is assigned the VSCAN Management rights profile. For more information, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.3*.

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

  ```
  # vscanadm add-engine engineID
  ```

  For more information, see the `vscanadm(1M)` man page.

How to View Vscan Properties

Before You Begin You must become an administrator who is assigned the VSCAN Management rights profile. For more information, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.3*.

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

    ```
    # vscanadm get-engine engineID
    ```

  - **To view the properties of all scan engines, type:**

    ```
    # vscanadm get-engine
    ```

  - **To view one of the properties of the `vscan` service, type:**

    ```
    # vscanadm get -p property
    ```

where *property* is one of the parameters described in the man page for the `vscanadm`(1M) command.

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

```
# vscanadm get max-size
```

▼ How to Limit the Size of Scanned Files

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

Before You Begin You must become an administrator who is assigned the VSCAN Management rights profile. For more information, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.3*.

1. **View the current properties.**

   ```
   # vscanadm show
   ```

2. **Set the maximum size for virus scans.**

   For example, to set a limit of 128 megabytes:

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

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

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

   For more information, see the `vscanadm`(1M) man page.

▼ How to Exclude Files From Virus Scans

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

Before You Begin You must become an administrator who is assigned the VSCAN Management rights profile. For more information, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.3*.

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

   ```
   # vscanadm get -p types
   ```

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

   For example:

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

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

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

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

   For more information, see the `vscanadm`(1M) man page.

Glossary

Access Control List (ACL)
An access control list (ACL) provides finer-grained file security than traditional UNIX file protection provides. For example, an ACL enables you to allow group read access to a file, while allowing only one member of that group to write to the file.

algorithm
A cryptographic algorithm. This is an established, recursive computational procedure that encrypts or hashes input.

asynchronous audit event
Asynchronous events are the minority of system events. These events are not associated with any process, so no process is available to be blocked and later woken up. Initial system boot and PROM enter and exit events are examples of asynchronous events.

audit trail
The collection of all audit files from all host systems.

authenticated rights profile
A rights profile that requires the assigned user or role to type a password before executing an operation from the profile. This behavior is similar to `sudo` behavior. The length of time that the password is valid is configurable.

authentication
The process of verifying the claimed identity of a principal.

authorization
1. In Kerberos, the process of determining if a principal can use a service, which objects the principal is allowed to access, and the type of access that is allowed for each object.

2. In user rights management, a right that can be assigned to a role or user (or embedded in a rights profile) for performing a class of operations that are otherwise prohibited by security policy. Authorizations are enforced at the user application level, not in the kernel.

basic set
The set of privileges that are assigned to a user's process at login. On an unmodified system, each user's initial inheritable set equals the basic set at login.

Blowfish
A symmetric block cipher algorithm that takes a variable-length key from 32 bits to 448 bits. Its author, Bruce Schneier, claims that Blowfish is optimized for applications where the key does not change often.

client
Narrowly, a process that makes use of a network service on behalf of a user; for example, an application that uses `rlogin`. In some cases, a network server can itself be a client of some other server or service.

More broadly, a host system that a) receives a Kerberos credential, and b) makes use of a service that is provided by a server.

Informally, a principal that makes use of a service.

consumer In the Cryptographic Framework feature of Oracle Solaris, a consumer is a user of the cryptographic services that come from providers. Consumers can be applications, end users, or kernel operations. Kerberos, IKE, and IPsec are examples of consumers. For examples of providers, see provider.

credential An information package that includes a ticket and a matching session key. Used to authenticate the identity of a principal. See also session key.

cryptographic algorithm See algorithm.

DES Data Encryption Standard. A symmetric-key encryption method developed in 1975 and standardized by ANSI in 1981 as ANSI X.3.92. DES uses a 56-bit key.

device allocation Device protection 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.

digest See message digest.

effective set The set of privileges that are currently in effect on a process.

FQDN Fully qualified domain name. For example, `central.example.com` (as opposed to simply `denver`).

hardware provider In the Cryptographic Framework feature of Oracle Solaris, a device driver and its hardware accelerator. Hardware providers offload expensive cryptographic operations from the computer system, thus freeing CPU resources for other uses. See also provider.

host A system that is accessible over a network.

host principal A particular instance of a service principal in which the principal (signified by the primary name `host`) is set up to provide a range of network services, such as `ftp`, `rcp`, or `rlogin`. An example of a host principal is `host/central.example.com@EXAMPLE.COM`.

inheritable set The set of privileges that a process can inherit across a call to `exec`.

instance In the context of network communications, an instance is the second part of a principal name. An instance qualifies the principal's primary. In the case of a service principal, the instance is required. The instance is the host's fully qualified domain name, as in `host/central.example.com`. For user principals, an instance is optional. Note, however, that `jdoe` and `jdoe/admin` are unique principals. See also primary and user principal.

integrity A security service that, in addition to user authentication, provides for the validity of transmitted data through cryptographic checksumming. See also authentication and privacy.

Kerberos An authentication service, the protocol that is used by that service, or the code that is used to implement that service.

The Kerberos implementation in Oracle Solaris that is closely based on Kerberos V5 implementation.

While technically different, “Kerberos” and “Kerberos V5” are often used interchangeably in the Kerberos documentation.

Kerberos (also spelled Cerberus) was a fierce, three-headed mastiff who guarded the gates of Hades in Greek mythology.

key 1. Generally, one of two main types of keys:

- A *symmetric key* – An encryption key that is identical to the decryption key. Symmetric keys are used to encrypt files.
- An *asymmetric key* or *public key* – A key that is used in public key algorithms, such as Diffie-Hellman or RSA. Public keys include a private key that is known only by one user, a public key that is used by the network server or general resource, and a private-public key pair that combines the two. A private key is also called a *secret* key. The public key is also called a *shared* key or *common* key.

2. An entry (principal name) in a keytab file. See also keytab file.

3. In Kerberos, an encryption key, of which there are three types:

- A *private key* – An encryption key that is shared by a principal and the KDC, and distributed outside the bounds of the system. See also private key.
- A *service key* – This key serves the same purpose as the private key, but is used by Kerberos servers and services. See also service key.
- A *session key* – A temporary encryption key that is used between two principals, with a lifetime limited to the duration of a single login session. See also session key.

keystore A keystore holds passwords, passphrases, certificates, and other authentication objects for retrieval by applications. A keystore can be specific to a technology, or a location that several applications use.

keytab file A key table file that contains one or more keys (principals). A host system or service uses a keytab file in the much the same way that a user uses a password.

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.

MD5 An iterative cryptographic hash function that is used for message authentication, including digital signatures. The function was developed in 1991 by Rivest. Its use is deprecated.

mechanism 1. A software package that specifies cryptographic techniques to achieve data authentication or confidentiality. Examples: Kerberos V5, Diffie-Hellman public key.

2. In the Cryptographic Framework feature of Oracle Solaris, an implementation of an algorithm for a particular purpose. For example, a DES mechanism that is applied to authentication, such as CKM_DES_MAC, is a separate mechanism from a DES mechanism that is applied to encryption, CKM_DES_CBC_PAD.

message digest A message digest is a hash value that is computed from a message. The hash value almost uniquely identifies the message. A digest is useful for verifying the integrity of a file.

nonattributable audit event An audit event whose initiator cannot be determined, such as the AUE_BOOT event.

PAM Pluggable Authentication Module. A framework that allows for multiple authentication mechanisms to be used without having to recompile the services that use them. PAM enables Kerberos session initialization at login.

passphrase A phrase that is used to verify that a private key was created by the passphrase user. A good passphrase is 10-30 characters long, mixes alphabetic and numeric characters, and avoids simple prose and simple names. You are prompted for the passphrase to authenticate use of the private key to encrypt and decrypt communications.

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

permitted set The set of privileges that are available for use by a process.

policy Generally, a plan or course of action that influences or determines decisions and actions. For computer systems, policy typically means security policy. Your site's security policy is the set of rules that define the sensitivity of the information that is being processed and the measures that are used to protect the information from unauthorized access. For example, security policy might require that systems be audited, that 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 policy in the Cryptographic Framework, device policy, password policy, and rights policy.

policy in the Cryptographic Framework In the Cryptographic Framework feature of Oracle Solaris, policy is the disabling of existing cryptographic mechanisms. The mechanisms then cannot be used. Policy in the Cryptographic Framework might prevent the use of a particular mechanism, such as `CKM_DES_CBC`, from a provider, such as DES.

primary The first part of a principal name. See also instance.

principal 1. A uniquely named client/user or server/service instance that participates in a network communication. Kerberos transactions involve interactions between principals (service principals and user principals) or between principals and KDCs. In other words, a principal is a unique entity to which Kerberos can assign tickets. See also user principal.

privacy A security service, in which transmitted data is encrypted before being sent. Privacy also includes data integrity and user authentication. See also authentication, integrity, and service.

private key A key that is given to each user principal, and known only to the user of the principal and to the KDC. For user principals, the key is based on the user's password. See also key.

private-key encryption In private-key encryption, the sender and receiver use the same key for encryption. See also public-key encryption.

privilege 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(5)` 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.

privilege set A collection of privileges. Every process has four sets of privileges that determine whether a process can use a particular privilege. See effective set set, permitted set set, and inheritable set set.

Also, the basic set set of privileges is the collection of privileges that are assigned to a user's process at login.

privilege-aware Programs, scripts, and commands that turn on and off the use of privilege in their code. In a production environment, the privileges that are turned on must be supplied to the process, for example, by requiring users of the program to use a rights profile that adds the privileges to the program. For a full description of privileges, see the `privileges(5)` man page.

privileged application An application that can override system controls. The application checks for security attributes, such as specific UIDs, GIDs, authorizations, or privileges.

privileged user A user who is assigned rights beyond the rights of regular user on a computer system. See also trusted users.

profile shell In rights management, a shell that enables a role (or user) to run from the command line any privileged applications that are assigned to the role's rights profiles. The profile shell versions correspond to the available shells on the system, such as the `pfbash` version of `bash`.

provider In the Cryptographic Framework feature of Oracle Solaris, a cryptographic service that is provided to consumers. PKCS #11 libraries, kernel cryptographic modules, and hardware accelerators are examples of providers. Providers plug in to the Cryptographic Framework, so are also called *plugins*. For examples of consumers, see consumer.

public-key encryption An encryption scheme in which each user has two keys, one public key and one private key. In public-key encryption, the sender uses the receiver's public key to encrypt the message, and the receiver uses a private key to decrypt it. The Kerberos service is a private-key system. See also private-key encryption.

RBAC Role-based access control, the user rights management feature of Oracle Solaris. See rights.

RBAC policy See rights policy.

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 and a role.

rights policy The security policy that is associated with a command. Currently, `solaris` is the valid policy for Oracle Solaris. The `solaris` policy recognizes privileges and extended privilege policy, authorizations, and `setuid` security attributes.

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.

role A special identity for running privileged applications that only assigned users can assume.

RSA A method for obtaining digital signatures and public key cryptosystems. The method was first described in 1978 by its developers, Rivest, Shamir, and Adleman.

scan engine A third-party application, residing on an external host system, that examines a file for known viruses.

secret key See private key.

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 mechanism See mechanism.

security policy See policy.

security service See service.

server A principal that provides a resource to network clients. For example, if you `ssh` to the system `central.example.com`, then that system is the network server that provides the `ssh` service.

service 1. A resource that is provided to network clients, often by more than one network server. For example, if you `rlogin` to the system `central.example.com`, then that system is the server that provides the `rlogin` service.

2. A security service (either integrity or privacy) that provides a level of protection beyond authentication. See also integrity and privacy.

service key An encryption key that is shared by a service principal and the KDC, and is distributed outside the bounds of the system. See also key.

session key A key that is generated by the authentication service or the ticket-granting service. A session key is generated to provide secure transactions between a network client and a service. The lifetime of a session key is limited to a single login session. See also key.

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.

synchronous audit event The majority of audit events. These events are associated with a process in the system. A non-attributable event that is associated with a process is a synchronous event, such as a failed login.

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*.

user principal A principal that is attributed to a particular user. A user principal's primary name is a user name, and its optional instance is a name that is used to described the intended use of the corresponding credentials (for example, `jdoe` or `jdoe/admin`). Also known as a user instance.

Index

O

P

R

S