



Solaris™ Operating Environment Security

Updated for Solaris 9 Operating Environment

By Alex Noordergraaf and Keith Watson

Sun BluePrints™ OnLine—December 2002



<http://www.sun.com/blueprints>

Sun Microsystems, Inc.
4150 Network Circle
Santa Clara, CA 95045 U.S.A.
650 960-1300

Part No.: 816-5242-10
Revision 01, December 2002

Copyright 2002 Sun Microsystems, Inc. 4150 Network Circle, Santa Clara, California 95045 U.S.A. All rights reserved.

Sun Microsystems, Inc. has intellectual property rights relating to technology embodied in the product that is described in this document. In particular, and without limitation, these intellectual property rights may include one or more of the U.S. patents listed at <http://www.sun.com/patents> and one or more additional patents or pending patent applications in the U.S. and in other countries.

This product or document is protected by copyright and distributed under licenses restricting its use, copying, distribution, and decompilation. No part of this product or document may be reproduced in any form by any means without prior written authorization of Sun and its licensors, if any. Third-party software, including font technology, is copyrighted and licensed from Sun suppliers.

Parts of the product may be derived from Berkeley BSD systems, licensed from the University of California. UNIX is a registered trademark in the United States and other countries, exclusively licensed through X/Open Company, Ltd.

Sun, Sun Microsystems, the Sun logo, Sun BluePrints, Solaris, Solaris Operating Environment, JumpStart, Solaris Security Toolkit, SunSpectrum, SunService, OpenBoot, SunSolve Online, Solaris Volume Manager, Sun Fire, SunScreen, Solaris Secure Shell, Solstice DiskSuite, Sun Cluster, and Solstice Print Client are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and other countries. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. in the US and other countries. Products bearing SPARC trademarks are based upon an architecture developed by Sun Microsystems, Inc.

The OPEN LOOK and Sun™ Graphical User Interface was developed by Sun Microsystems, Inc. for its users and licensees. Sun acknowledges the pioneering efforts of Xerox in researching and developing the concept of visual or graphical user interfaces for the computer industry. Sun holds a non-exclusive license from Xerox to the Xerox Graphical User Interface, which license also covers Sun's licensees who implement OPEN LOOK GUIs and otherwise comply with Sun's written license agreements.

U.S. Government Rights—Commercial use. Government users are subject to the Sun Microsystems, Inc. standard license agreement and applicable provisions of the FAR and its supplements.

DOCUMENTATION IS PROVIDED "AS IS" AND ALL EXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS AND WARRANTIES, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT, ARE DISCLAIMED, EXCEPT TO THE EXTENT THAT SUCH DISCLAIMERS ARE HELD TO BE LEGALLY INVALID.

Copyright 2002 Sun Microsystems, Inc., 4150 Network Circle, Santa Clara, Californie 95045 Etats-Unis. Tous droits réservés.

Sun Microsystems, Inc. a les droits de propriété intellectuels relatants à la technologie incorporée dans le produit qui est décrit dans ce document. En particulier, et sans la limitation, ces droits de propriété intellectuels peuvent inclure un ou plus des brevets américains énumérés à <http://www.sun.com/patents> et un ou les brevets plus supplémentaires ou les applications de brevet en attente dans les Etats-Unis et dans les autres pays.

Ce produit ou document est protégé par un copyright et distribué avec des licences qui en restreignent l'utilisation, la copie, la distribution, et la décompilation. Aucune partie de ce produit ou document ne peut être reproduite sous aucune forme, par quelque moyen que ce soit, sans l'autorisation préalable et écrite de Sun et de ses bailleurs de licence, s'il y en a. Le logiciel détenu par des tiers, et qui comprend la technologie relative aux polices de caractères, est protégé par un copyright et licencié par des fournisseurs de Sun.

Des parties de ce produit pourront être dérivées des systèmes Berkeley BSD licenciés par l'Université de Californie. UNIX est une marque enregistrée aux Etats-Unis et dans d'autres pays et licenciée exclusivement par X/Open Company Ltd.

Sun, Sun Microsystems, le logo Sun, Sun BluePrints, Solaris, Solaris Operating Environment, JumpStart, Solaris Security Toolkit, SunSpectrum, SunService, OpenBoot, SunSolve Online, Solaris Volume Manager, Sun Fire, SunScreen, Solaris Secure Shell, Solstice DiskSuite, Sun Cluster, et Solstice Print Client sont des marques de fabrique ou des marques déposées, ou marques de service, de Sun Microsystems, Inc. aux Etats-Unis et dans d'autres pays. Toutes les marques SPARC sont utilisées sous licence et sont des marques de fabrique ou des marques déposées de SPARC International, Inc. aux Etats-Unis et dans d'autres pays. Les produits portant les marques SPARC sont basés sur une architecture développée par Sun Microsystems, Inc.

L'interface d'utilisation graphique OPEN LOOK et Sun™ a été développée par Sun Microsystems, Inc. pour ses utilisateurs et licenciés. Sun reconnaît les efforts de pionniers de Xerox pour la recherche et le développement du concept des interfaces d'utilisation visuelle ou graphique pour l'industrie de l'informatique. Sun détient une licence non exclusive de Xerox sur l'interface d'utilisation graphique Xerox, cette licence couvrant également les licenciés de Sun qui mettent en place l'interface d'utilisation graphique OPEN LOOK et qui en outre se conforment aux licences écrites de Sun.

LA DOCUMENTATION EST FOURNIE "EN L'ÉTAT" ET TOUTES AUTRES CONDITIONS, DECLARATIONS ET GARANTIES EXPRESSES OU TACITES SONT FORMELLEMENT EXCLUES, DANS LA MESURE AUTORISEE PAR LA LOI APPLICABLE, Y COMPRIS NOTAMMENT TOUTE GARANTIE IMPLICITE RELATIVE A LA QUALITE MARCHANDE, A L'APTITUDE A UNE UTILISATION PARTICULIERE OU A L'ABSENCE DE CONTREFAÇON.



Please
Recycle



Adobe PostScript

Solaris™ Operating Environment Security

Updated for Solaris 9 Operating Environment

This article provides recommendations on how to secure a Solaris™ Operating Environment (Solaris OE). Securing a Solaris OE system requires that changes be made to its default configuration. The changes outlined in this article address the majority of the methods that intruders use to gain unauthorized or privileged access to an improperly configured system. Implementing the changes recommended in this article requires planning, testing, and documentation to be successful in securing a computing environment.

This article contains the following topics:

- “Updates” on page 2
- “Introduction” on page 2
- “Using the Solaris Security Toolkit” on page 3
- “Securing File Systems and Local Access” on page 4
- “Securing Network Services” on page 33
- “References and Related Resources” on page 56
- “About the Authors” on page 57

Updates

This article was updated to include changes in the Solaris 9 OE. This article is current as of Solaris 9 OE (5/02), which was released on May, 2002. Differences between the Solaris OE versions 2.5.1 through 9 are discussed in this article and noted where appropriate.

Introduction

The Solaris OE is a flexible, general purpose operating system. Due to its general nature, changes must be made to secure the system against unauthorized access and modification.

As with any security decisions, a balance must be achieved between system manageability and security. Some recommended changes in this article may not apply to all environments. The removal of some services may negatively impact the ability to effectively maintain a system. You must know your system and security requirements before starting.

The information in this article applies to the Solaris 2.5.1, 2.6, 7, 8, and 9 OEs. Older versions of the Solaris OE may be configured in similar ways. Some investigation is necessary before making the changes suggested in this article to older versions.

Using the Solaris Security Toolkit

Many of the changes described in this article can be implemented during installation by the Solaris Security Toolkit. The goal of the Solaris Security Toolkit is to automate and simplify building secured Solaris OE systems based on the recommendations contained in this and the other security-related Sun BluePrints articles.

The Solaris™ Security Toolkit focuses on Solaris OE security modifications to harden and minimize a system. Hardening is the modification of Solaris OE configurations to improve the security of the system. Minimizing is the removal of unnecessary Solaris OE packages from the system, which reduces the number of components that have to be patched and made secure. Reducing the number of components can potentially reduce entry points to an intruder.

Note – Configuration modifications for performance enhancements and software configuration are not addressed by the Solaris Security Toolkit.

The Solaris Security Toolkit was designed to harden systems during installation—this is achieved by using the JumpStart™ technology as a mechanism for running Solaris Security Toolkit scripts.

Additionally, the Solaris Security Toolkit can be run outside the JumpStart framework in a standalone mode. The standalone mode allows the Solaris Security Toolkit to be used on systems that require security modifications or updates but cannot be taken out of service to reinstall the OS from scratch.

The Solaris Security Toolkit is available from:

<http://www.sun.com/blueprints/tools>

Sun BluePrints articles describing the Solaris Security Toolkit are available from:

<http://www.sun.com/security/blueprints>

Securing File Systems and Local Access

Often, administrators are greatly concerned about attackers breaking into systems remotely. We recommend that you have equal concern for local, authorized users gaining extra privileges on a system by exploiting a problem with internal system security.

This section contains the following topics:

- “Performing an Initial Installation” on page 4
- “Minimizing the Solaris OE Installation” on page 5
- “Securing the Console” on page 6
- “Configuring the File System” on page 10
- “Managing Accounts” on page 15
- “Securing the `init` System” on page 23
- “Making Kernel Adjustments” on page 25
- “Managing Log Files” on page 28
- “Configuring Authentication” on page 30

Performing an Initial Installation

Sun works toward improving the Solaris OE with every release. Each new release includes additional features and improvements to enhance system security. Always use the latest version of the Solaris OE that your applications support.

Building a secure Solaris OE system involves installing a system with the latest version of the Solaris OE and applying the latest patches. Many of the changes described in this article can be implemented during installation by the Solaris Security Toolkit.

To prevent an attacker from modifying a system or creating backdoors before you have the opportunity to secure the system, do not attach a system to a public network until security modifications are completed.

Deciding to either upgrade or re-install a system with the latest Solaris OE release is a challenging one from a security perspective. Ideally, all systems are installed from Sun media to avoid any potential of malicious modifications. However, it is not always appropriate to re-install the OE. When upgrading deployed systems, it is strongly recommended that the integrity of the Solaris OE image be verified, at a minimum, through a mechanism such as the Solaris Fingerprint Database.

When creating operating system file partitions, be sure to allocate adequate disk space for system directories, log files, and applications. Some server applications or services may require extra disk space or separate partitions to operate effectively without impacting other services. Typically, there at least should be partitions for the root file system (/), /var, and /var/crash.

The Solaris OE /var file system contains system log files, patch data, and files for printing, mail, and other services. The disk space required for these files varies over time. Most systems (and all servers) should maintain /var as a separate partition from the root file system.

Mail servers should maintain a large, separate /var/mail partition to contain user mail files. These extra partitions help prevent a full /var or /var/mail file system from affecting the operation of the system. Provide extra space in /var if you intend to store large log files or capture system core dumps.

Additional partitions, such as /usr and /opt, may be required if you follow the recommendations in the “Setting Mount Options” on page 12.

Minimizing the Solaris OE Installation

It is important to reduce the Solaris OE installation down to the minimum number of packages necessary to support the application being hosted. This reduction in services, libraries, and applications helps increase security by reducing the number of subsystems that must be disabled, patched, and maintained.

For detailed instructions, recommendations, and processes to minimize the Solaris OE, refer to the Sun BluePrints OnLine article “Solaris™ Operating Environment Minimization for Security: A Simple, Reproducible, and Secure Application Installation Methodology.”

Immediately after installing the Solaris OE system, apply all the patches in the Recommended and Security Patches Cluster. These patches are available from: <http://sunsolve.sun.com> Web and FTP sites.

Make sure that all systems have the latest Recommended and Security Patches Cluster installed.

Sun provides patches to the Solaris OE and unbundled software products when problems are corrected. You can download the recommended and security patches for the Solaris OE, even if you do not have a service contract. All other patches require a SunSpectrumSM service contract.

Subscribe to the Sun security bulletin mailing list to receive notification of important security-related patches. Recently, Sun started providing maintenance updates (MU) for the Solaris OE. An MU is a tested combination of patches for a specific release of

the Solaris OE that installs in one quick and easy step. Beginning with Solaris 8 OE, MU CDs are included in the media kit. For older Solaris OE versions, the updates are only available to service contract customers.

SunServiceSM provides a variety of tools to assist in patching systems. These include the PatchDiag, PatchCheck, and PatchManager. For more information about these tools, refer to SunSolve OnLine.

Care must be taken when applying patches to a system. Some patches modify the system initialization scripts and may disable security changes made to a system. Scripts that were deleted from the `init` run level directories to disable services could be replaced during the patch installation process, enabling the service once again. Be sure to examine all system `init` scripts and test all patches on non-production systems to discover any such configuration changes.

Securing the Console

There are several security mechanisms that Sun hardware systems provide for console security. The OpenBootTM PROM system on SPARCTM systems has two security modes, `command` and `full`.

Failed login attempts to the OpenBoot PROM system can be monitored. Also, it is possible to prevent users from using the keyboard sequence to interrupt the Solaris OE and drop to the OpenBoot PROM level.

This section contains the following topics:

- “Using OpenBoot PROM Security Modes” on page 6
- “Monitoring EEPROM Password Guessing” on page 8
- “Disabling Keyboard Abort” on page 9

Using OpenBoot PROM Security Modes

Sun’s SPARC hardware provides additional console security features. These features prevent EEPROM changes, hardware command execution, and system start up without the appropriate password. This password protection only works while the system is at the OpenBootTM PROM level (Solaris OE is stopped). Similar features might be available on Intel x86 hardware, however, they are not supported in the Solaris OE Intel Platform Edition.

The OpenBoot PROM password is not related to the Solaris OE root password, and it should not be set as the same password. Once set, the OpenBoot PROM password is not displayed, but can be retrieved in clear text form. When changing the

OpenBoot PROM password, the system does not ask for the old password prior to changing it to the new one. In some environments, it may make more sense to set the OpenBoot PROM password to something known to the hardware technicians.

The two security modes available are **command** and **full**.

- Unless an authorized user has the correct password, the **command** security mode prevents EEPROM changes and hardware command execution while at the OpenBoot PROM level.
- The **full** security mode provides the features of the **command** mode and, in addition, does not allow the system to boot without the correct OpenBoot PROM password. The **full** security mode requires operator interaction to boot the system. It does not boot without a password. Do not use this feature on servers or other systems that must boot quickly without manual intervention.

▼ To Set the Security Mode

- **Use the `eeeprom` command in the Solaris OE to set the security mode.**

Here is an example of setting the mode to **full**:

```
# eeeprom security-mode=full
Changing PROM password:
New password: password
Retype new password: password
```

▼ To Set a New EEPROM Password

- **Use the following command to set a new EEPROM password.**

```
# eeeprom security-password=
Changing PROM password:
New password: password
Retype new password: password
```

Be sure to include the trailing equal sign (=).

▼ To Make OpenBoot PROM Changes at the OpenBoot PROM Level

- **To make these OpenBoot PROM changes at the OpenBoot PROM level, use the following example.**

Here is an example of setting the OpenBoot PROM security mode and password while at OpenBoot PROM level:

```
ok setenv security-mode command
security-mode =          command
ok setenv security-password password
security-password =
```

The system EEPROM security mode can be disabled by setting the security mode to none.

Monitoring EEPROM Password Guessing

If someone guesses or mistypes the OpenBoot PROM password, a time-out period of ten seconds occurs and the attempt is counted.

▼ To See How Many Failed Login Attempts Were Made

1. **Use the following command to see how many failed login attempts were made.**

```
# eeeprom security-#badlogins
security-#badlogins=3
```

2. **To add this command to an initialization script to track password attempts and reset the counter, use the following command.**

```
# eeeprom security-#badlogins=0
security-#badlogins=0
```

When the security mode is enabled, losing the OpenBoot PROM password may require you to replace the EEPROM. An attacker with superuser access could set the security mode to full, set the password to random characters, and reboot the system. The system no longer boots without the new password. If this event happens, you should contact the SunService organization for assistance.

Disabling Keyboard Abort

Using the keyboard abort sequence on SPARC systems, users can drop to the OpenBoot PROM level while the Solaris OE is running. This feature can be disabled in Solaris 2.6 and newer OEs. Disabling this feature may be helpful, for example, in uncontrolled lab environments, to prevent users from bringing systems down. If OpenBoot PROM security mode `full` or `command` is enabled, the EEPROM settings cannot be altered without a password.

One mechanism for disabling the keyboard abort sequence feature is to change the following line in the `/etc/default/kbd` file from:

```
#KEYBOARD_ABORT=enable
```

to:

```
KEYBOARD_ABORT=disable
```

When you disable the keyboard abort sequence using this mechanism, it is maintained after a system reboot.

If the system hangs or become unusable, power it off to reset it. With this feature disabled, it is no longer able to create a crash dump from the OpenBoot PROM level on a running system for analysis.

Also, you can disable the keyboard abort sequence feature by using the `kdb` command directly to query and change the state of the `KEYBOARD_ABORT` option.

Configuring the File System

You can configure the Solaris OE file system to provide additional protection. The default file permissions on some files are not adequate. Also, several mount options are available to increase security, when used effectively. The Solaris™ Volume Manager system needs some adjustment to prevent attackers from gaining superuser privileges.

This section contains the following topics:

- “Adjusting File Permissions” on page 10
- “Reviewing set-user-ID and set-group-ID Files” on page 11
- “Setting Mount Options” on page 12
- “Securing Solaris Volume Manager” on page 13

Adjusting File Permissions

Solaris OE versions prior to Solaris 9 OE ship with file system permissions that need to be adjusted for security reasons. With the release of Solaris 9 OE, this adjustment is no longer necessary for the core Solaris OE packages. In Solaris 8 OE and older versions, many files and directories have the group write bit set. In most instances, this permission is not necessary and should be switched off.

Although file permission changes are not required for Solaris 9 OE and newer releases, they may be required of applications installed on top of the OE. Consequently, monitor permissions on all Solaris OE versions.

Casper Dik created a tool to adjust these permissions. The tool is called `fix-modes` and can be downloaded from:

<http://www.sun.com/blueprints/tools>

or

<ftp://ftp.wins.uva.nl/pub/solaris/fix-modes.tar.gz>

Please note that this tool is not supported by Sun. The `fix-modes` version available from `sun.com` is precompiled while the version from `uva.nl` is not. If compilation is required, it must be performed on a Solaris OE system with a C compiler. Once compiled, install the `fix-modes` files and execute it to correct file system permissions. This tool has been used in production environments for several years with no reported problems.

Be careful when installing patches and new packages. These may set permissions back to their original state. Execute the `fix-modes` tool after installing any packages or patches.

Sun continues to refine file permissions and group ownerships shipped in the Solaris OE media kits. Enhancements to `fix-modes` are implemented and tested for specialized Sun servers such as the Sun Fire 15K system controller. Periodically download the latest version of the `fix-modes` software to ensure that you have the enhancements.

Reviewing `set-user-ID` and `set-group-ID` Files

The `set-user-ID` and `set-group-ID` bits (sometimes referred to as SUID and SGID bits) on an executable file indicate to the system that the executable should operate with the privileges of the file's owner or group. In other words, the effective user ID of the running program becomes that of the executable's owner, in the `set-user-ID` instance.

A `set-group-ID` file sets the running program's effective group ID to the executable's group. This file is useful in allowing users to run some commands that gather system information or write to files not owned by the user. If the command with the `set-user-ID` and/or `set-group-ID` bit set is written correctly with security in mind, it can be a useful method for solving some tricky operational problems.

The `set-user-ID` and `set-group-ID` commands that have flaws are often used to exploit the system. The attacker uses the elevated privileges provided by the `set-user-ID` or `set-group-ID` mechanism to execute code on the program stack (a "buffer overflow" attack) or to overwrite system files. When these security problems are reported, Sun fixes them and provides a patch. This is another reason to maintain an up-to-date system with the latest set of patches.

Attackers may use the `set-user-ID` or `set-group-ID` feature to create "backdoors" into systems. One way this is done is by copying a system shell to a "hidden" location and adding the `set-user-ID` bit. This technique allows the attacker to execute the shell to gain elevated privileges (most often superuser).

▼ To Find All `set-user-ID` and `set-group-ID` Files On a Server

- **Use the following command to find all the `set-user-ID` and `set-group-ID` files on a server.**

```
# find / -type f \( -perm -u+s -o -perm -g+s \) -ls
```

Store the output to a file on another system. Compare it against the current file system from time to time and after applying patches to find any unwanted additions.

Refer to the next section, “Setting Mount Options,” for tips on preventing attackers from storing backdoor files or overwriting and replacing files on a file system.

You may want to obtain and use the Solaris Fingerprint Database. This tool enables an administrator to verify, through a cryptographic checksum, the integrity of files distributed with the Solaris OE. While useful for checking `set-user-ID` and `set-group-ID` permission, the real benefit of the Solaris Fingerprint Database is the detection of trojaned or maliciously modified executables. The Solaris Fingerprint Database does not require a service contract to access and is available from:

<http://sunsolve.sun.com>

The Sun Blueprints OnLine article “The Solaris Fingerprint Database” provides detailed examples on how to use the Fingerprint Database, descriptions of some tools to enhance its usefulness, and how a sample rootkit might be detected with the Fingerprint Database. The Sun BluePrints OnLine article is available from:

<http://www.sun.com/blueprints/0501/Fingerprint.pdf>

Setting Mount Options

The Solaris OE file system partitions can be mounted with various options that enhance security. As described in the previous section, sometimes attackers use `set-user-ID` files to create ways to gain higher privileges. These backdoors may be hidden anywhere on a file system. While a file may have a `set-user-ID` bit, it is not effective on file systems mounted with the `nosuid` option. The system ignores the `set-user-ID` bit for all files on a `nosuid` mounted file system, and programs execute with normal privilege.

It is possible to mount a file system in read-only mode to prevent file modification. This configuration prevents an attacker from storing backdoor files or overwriting and replacing files on a file system. Whenever possible, file systems should be mounted in read-only mode, and should be mounted to ignore the `set-user-ID` bit on files.

Note that these options are not complete solutions. A read-only file system can be remounted in read-write mode. The `nosuid` option can be removed. Not all file systems can be mounted in read-only mode or with `nosuid`. If a file system is remounted in read-write mode, it must be rebooted to switch back to read-only mode. A reboot is required to change a `nosuid` file system to `suid`. Watch for unscheduled system reboots.

Note – Early Solaris OE versions (for example, Solaris 2.6 and 2.5.1) are unable to remount in read-only mode a file system that is mounted read-only, then modified to read-write. These OE versions are also unable to mount a `nosuid` file system as `suid`. On these OE versions, carefully track reboots, because they may be indicative of malicious activity. Later Solaris OE versions, especially Solaris 9 OE, can do all of the previously mentioned tasks without requiring any system reboots.

Different system partitions support different mount options. The `/usr` partition can be mounted read-only. It should not be mounted `nosuid`, because there are some commands in this partition that have the `set-user-ID` bit set. The `/var` partition cannot be set to read-only, but can be set to `nosuid`. Mount all other partitions read-only and with `nosuid` whenever possible.

Contrary to suggestions in other Solaris OE security documents, it is not possible to mount the root file system (`/`) with the `nosuid` option on newer releases of the Solaris OE. This limitation is because the root file system is mounted read-only when the system boots and is later remounted read-write. When the remount occurs, the `nosuid` option is ignored.

Here is a partial `/etc/vfstab` file containing the appropriate file system options:

```
/dev/dsk/c0t3d0s0 /dev/rdsk/c0t3d0s0 /      ufs 1 no -  
/dev/dsk/c0t3d0s4 /dev/rdsk/c0t3d0s4 /usr  ufs 1 no ro  
/dev/dsk/c0t3d0s5 /dev/rdsk/c0t3d0s5 /var  ufs 1 no nosuid  
/dev/dsk/c0t3d0s6 /dev/rdsk/c0t3d0s6 /opt  ufs 2 yes nosuid,ro
```

Although these file system options significantly improve the security of a system, they may cause difficulty with some third-party applications. Thoroughly test these options before deploying them to a production system.

Securing Solaris Volume Manager

The Solaris Volume Manager system provides users an easy way to mount removable media without requiring superuser access. CD-ROMs and diskettes are mounted and unmounted automatically by the volume management system. The daemon that manages this system is called `vold`.

The `vold` daemon uses the `rmmount` command to mount the removable media device. It uses a configuration file (`/etc/rmmount.conf`) to determine the actions necessary, based on the device to be mounted. The `vold` daemon calls `rmmount`, which determines what type of file system, if any, is on the media. If a file system is present and is supported, `rmmount` mounts the file system.

If the system does not require the automatic mounting of CD-ROMs and diskettes, `vold` should be disabled. For example, a server does not need it, but a workstation may.

To disable this service, remove the Solaris Volume Manager packages `SUNWvolr`, `SUNWvolu`, and `SUNWvolg`.

If volume management is necessary, the mount options for some file systems should be modified for security. As previously mentioned, file systems with the `Suid` option can be problematic. In Solaris OE versions prior to release 8, the default Solaris Volume Manager configuration is to allow `Suid` file systems for all removable media that are capable of supporting it. In Solaris 7 OE and previous releases, anyone can insert a UFS formatted diskette containing a `set-user-ID` executable and gain control of the system. To prevent this situation, add the following lines to the end of the `/etc/rmmount.conf` file in all Solaris OE versions prior to 8:

```
mount hsfs -o nosuid
mount ufs -o nosuid
```

Solaris 8 OE and newer releases have these lines in the file by default. With these options, the `set-user-ID` bit on executables is ignored on file systems that are mounted by the Solaris Volume Manager system.

Another security issue exists when using writable automounted media such as disks, zip diskettes, and jass disks. These media types are automatically mounted as publicly readable and writable directories and files. These media types pose a significant security risk on multiuser systems because any user on the system can read, write, and modify the contents of automounted media. Care should be taken to ensure that unauthorized users cannot access or modify sensitive information through automounted writable media. Ideally, authorized users should only use and access these media types locally on a system while no other users are permitted to log into the system over the network.

Managing Accounts

Managing user and system accounts is an important aspect of Solaris OE security. Some system accounts may need to be modified or deleted. The time-based command execution system tools `cron` and `at` may need to be configured to restrict user access.

This section contains the following topics:

- “Managing System Accounts” on page 15
- “Restricting `at`, `cron`, and batch Command Access” on page 17
- “Implementing Role-Based Access Control (RBAC)” on page 18

Managing System Accounts

A default Solaris OE installation contains several accounts that either need to be deleted or modified to strengthen security. Some accounts are not necessary for normal system operation. These accounts include `smtp`, `nuucp`, and `listen`. Some of these accounts exist to support software subsystems that are not used or are for backwards compatibility.

▼ To Delete Accounts in `/etc/passwd` and `/etc/shadow`

- **Use the `userdel` command to delete accounts in `/etc/passwd` and `/etc/shadow`.**

Here is an example of the `userdel` command.

```
# userdel smtp
```

This command removes the `/etc/passwd` and `/etc/shadow` entries for `smtp`.

The remaining system accounts (except the root account) should be modified for added security. System accounts listed in `/etc/passwd` have no shell listed. Those accounts have an NP string (meaning “no password”) listed in the `/etc/shadow` file. By default, this string is sufficient. Unused accounts, which are not locked by default, should be locked with the `passwd -l` option.

▼ To Lock the uucp Account

1. Use the following command to lock the uucp account.

```
# passwd -l uucp
```

2. Use the -e option to the passwd command or edit the /etc/passwd file manually to change the default shell for those accounts to /usr/bin/true.

For example:

```
# passwd -e uucp
Old shell: /sbin/sh
New shell: /usr/bin/true
```

Administrators should monitor these system accounts for abuse. The Solaris Security Toolkit includes a shell replacement called `noshell`. When the `noshell` executable is executed (as a login shell in `/etc/passwd`) a log entry is generated and the shell exits. Administrators can track unauthorized use of system accounts.

Restricting at, cron, and batch Command Access

The `at`, `cron`, and `batch` systems execute commands at a specified future time. User submission for the `cron` system is handled by the `crontab` command. The `at` and `batch` commands are used to submit jobs to the `at` system.

Access to these commands can be restricted. The access control files are stored in the `/etc/cron.d` directory.

- The `cron.deny` and `cron.allow` files manage access to the `cron` system.
- The `at.deny` and `at.allow` files manage the access to the `at` and `batch` system.

The `allow` file is checked first to see if the account is explicitly allowed to use the system. Solaris OE versions not supporting role-based access control (RBAC) use rules to determine which accounts can access the `cron` and `at` systems. If neither the `allow` nor the `deny` file exists, then only users with the `solaris.jobs.user` authorization may use `cron` and `at`. By default, this authorization is granted to all users.

A benefit of the RBAC authorization framework, over configuring `cron` and `at` configuration files locally, is its support of name services. By centrally storing RBAC authorizations in a name service such as NIS+, server specific modifications can be avoided. Refer to the RBAC man pages for additional information on authorizations.

The `cron` and `at` systems can be problematic because commands are executed in the future. An attacker can use these systems to implement a “logic bomb” or other type of programmed attack that begins at some point in the future. Without examining every `at`, `batch`, and `cron` submission, tracking usage and abuse can be difficult.

Access should be restricted to the `at`, `batch`, and `cron` systems to prevent attacks and abuse. By default, the Solaris OE includes scheduled `cron` events for the `lp`, `adm`, and `root` accounts. These should not be included in the `deny` files.

Any additional system or software-specific accounts that do not require `cron`, `batch`, or `at` access should be added to the `deny` files.

You may want to restrict normal user access to these commands as well. Individual user accounts should be listed in the `deny` files. To restrict all user account access, create an empty `allow` file. Add only the accounts that need access to the `allow` file.

Implementing Role-Based Access Control (RBAC)

On Solaris 9 OE systems, role based access control (RBAC) provides an alternative to the standard UNIX® root account privileges with a mechanism to grant users additional privileges without root equivalent privileges. Being able to provide this capability is a very important access control commonly referred to as “least privilege,” which means that users get only the minimum privileges required to perform their tasks.

RBAC is implemented through grouping superuser-like capabilities into roles. You can create different roles for users requiring the ability to mount file systems, start privileged daemons, and other similar tasks. By providing this detailed level of system access, you can improve system security because fewer individuals have access to the root account and password. RBAC works through users logging into the system normally, then assuming special identities that permit them to perform additional tasks. Fundamentally, RBAC introduces the following three elements into the Solaris OE:

- Role – A special identity users can assume
- Authorization – A permission that can be assigned to a user or role granting access to tasks otherwise not permitted
- Rights Profile – A grouping of authorizations, commands, and additional rights that can be assigned to a user or role

Additional information on RBAC is available in the *Solaris 9 Operating Environment System Administration Guide* and in a white paper, titled “RBAC in the Solaris Operating Environment,” available on

<http://www.sun.com/software/whitepapers/wp-rbac>

The following sections provide two simple RBAC examples and procedures for implementing the examples.

▼ To Convert the Root Account to a Role

The first example illustrates how to convert the root account to a role. By doing this, additional controls can be implemented to control which accounts can have superuser privileges to root.

Note – This modification has no affect on users being able to enter the root password during a system user boot. That process is not modified in any way.

1. **Log into the system as superuser.**
2. **Verify that the following line is in the `/etc/user_attr` file:**

```
root:::auths=solaris.*,solaris.grant;profiles=All
```

3. **Add a line containing user accounts that should be able to have superuser privileges to root.**

For users Alice and Betty, you would add the following:

```
alice:::type=normal;roles=root
betty:::type=normal;roles=root
```

Now both Alice and Betty have privileges to assume the root role after it is created in the next step.

Because Alice and Betty are both local users on this system, it is possible to use the following commands to make this change, which is not necessary if the previous entries are added to the `/etc/user_attr` file:

```
# usermod -R root alice
# usermod -R root betty
```

4. **Modify the type definition of the root role as follows:**

```
root:::type=role;auths=solaris.*,solaris.grant;profiles=All
```

When the changes are made, only users Alice and Betty are able to have superuser access to the root account. In addition, the root account cannot log directly into the system regardless of the **CONSOLE** settings in `/etc/default/login`. If someone attempts to log directly into the system, the following is displayed after the `login` command:

```
arciero console login: root
Password:
Roles can only be assumed by authorized users
Login incorrect
Oct 27 20:09:14 arciero login: login account failure: Permission
denied
```

Compare this result with the following when authorized user Alice logs in to assume the root role.

```
arciero console login: alice
Password:
Last login: Sun Oct 27 20:07:59 on console
Sun Microsystems Inc.    SunOS 5.9          Generic May 2002
$ su
Password:
#
```

▼ To Restart Daemons That Require Root Privileges

The following example shows how RBAC can solve a common administrative security issue—that of restarting a daemon which requires root privileges. For this example, Apache is used. The following steps create an executable script that can be run by anyone to successfully restart Apache.

1. **At the to the end of the /etc/security/exec_attr file, add commands for restarting the Apache daemon:**

```
Apache:suser:cmd:::/etc/init.d/apache:uid=0
Apache:suser:cmd:::/usr/bin/kill:uid=0
Apache:suser:cmd:::/bin/rm:uid=0
```

2. **At the end of the /etc/security/prof_attr file, add the following:**

```
Apache::Apache Restart Profile:
```

3. **Execute the following to create a profile shell role that will be used by authorized individuals to access a superuser account and restart the daemon:**

```
# roleadd -u 1050 -g staff -P Apache -d /export/home/apacher -m -
s /bin/pfsh apacher
```

The Apache role definition sequence must be modified so that the Apache profile is listed before the default All and Basic Solaris User profiles.

4. **Log in as superuser to the apacher profile and execute the profiles command to verify that the Apache profile is listed before the default All and Basic Solaris User profiles.**
5. **If the Apache profile is not listed before the All profile, review the modifications and make the necessary adjustments.**

The output should appear similar to the following:

```
$ profiles
Apache
Basic Solaris User
All
```

6. Edit the /etc/user_attr file to add the profile to appropriate accounts.

In the following example, the apacher profile was added to the Betty user account:

```
betty:::type=normal;roles=apacher
```

At this point, the Betty user can log in as superuser to the apache profile shell and restart apache as follows:

```
arciero console login: betty
Password:
Sun Microsystems Inc.   SunOS 5.9           Generic May 2002
arciero% su - apacher
Password:
$ /etc/init.d/apache restart
```

Note – The /etc/security/policy.conf file is another important RBAC configuration file for more customized configurations. Refer to its man page for additional information.

Securing the `init` System

The Solaris OE `init` system manages system services. Some services may not be needed or should be modified to improve the security posture of a system.

This section contains the following topics:

- “Setting the System Default Umask” on page 23
- “Disabling System Services” on page 23

Setting the System Default Umask

In Solaris OE releases prior to Solaris 8 OE, the default system file mode creation mask for the Solaris OE is 000. This default means that files created by system daemons are created with permission bits that are 666 (readable and writable by all users). This default can be a problem because it gives normal users permission to overwrite the contents of system files.

The Solaris 8 OE was the first release in which the default system `umask` changed to 022 from the 000 in previous Solaris OE releases. Newer OE releases carry forward the Solaris 8 OE changes. The default value of 022 is defined by the `CMASK` variable in the `/etc/default/init` and can be modified by changing the `CMASK` value.

For Solaris OE versions prior to Solaris 8, the following script should be used to set the system `umask` to a more reasonable value:

```
echo "umask 022" > /etc/init.d/umask.sh
chmod 744 /etc/init.d/umask.sh
chgrp sys /etc/init.d/umask.sh
for d in /etc/rc?.d; do
    ln /etc/init.d/umask.sh $d/S00umask.sh
done
```

Disabling System Services

System services are started by the `init` system. Some services are not necessary to system operation and should be disabled. Also, some services may allow a system to be compromised due to incorrect configuration.

To disable services started by `init`, simply rename or delete the initialization script in the `init` system run level directory. The run level directories contain the scripts for starting or stopping services for the system run level. The system run level directories and their purpose as follows:

- `/etc/rcS.d` single user
- `/etc/rc0.d` shutdown
- `/etc/rc1.d` start
- `/etc/rc2.d` multi-user
- `/etc/rc3.d` multi-user (default)
- `/etc/rc4.d` multi-user (unused)
- `/etc/rc5.d` shutdown and power off
- `/etc/rc6.d` shutdown and reboot

These directories contain initialization scripts to start or stop services. Initialization scripts that begin with either an “S” or a “K” are executed by the `init` system. “S” scripts start services, and “K” scripts stop or “kill” services.

If you rename the scripts, make sure the name does not begin with these letters. It is recommended that an underscore (`_`) be placed at the beginning of the name. This practice makes it easy to enable services that may be needed later. For example:

```
# cd /etc/rc2.d
# mv S99dtlogin _S99dtlogin
```

For security purposes, only required services should be enabled. The fewer services that are enabled, the less likely it is that an attacker can discover a way to exploit the system using an enabled service.

The version of the Solaris OE and the packages installed determine what services are enabled by default. Removing unnecessary packages disables some extraneous services. The remaining services should be examined to determine their relevance to the system and the hosted application.

Note – Be aware that installing patches and/or software packages may restore or add new entries for `init` to start. We recommend that you regularly review which services are started by `init`.

Making Kernel Adjustments

Several kernel adjustments can be made to increase Solaris OE security. The `/etc/system` file contains kernel-specific parameter adjustments.



Caution – Be careful when making changes to this file. Mistakes in this file may prevent the system from booting. Always save a backup copy (for example, `/etc/system.save`, so that if changes render a system unbootable, you can perform a `boot -a` with the `/etc/system.save`.

This section contains the following topics:

- “Restricting NFS Server Requests” on page 25
- “Preventing Attempts to Execute Code on Stacks” on page 26
- “Managing Core Files” on page 27

Restricting NFS Server Requests

By default, the Solaris Network File System (NFS) server system accepts client NFS server requests from any port number. These requests should come from a privileged system port. The NFS server can be adjusted to process requests only from these privileged ports.

If a system serves as an NFS server, add the following line to the `/etc/system` file to any Solaris OE version 2.5.1 or newer.

```
set nfssrv:nfs_portmon = 1
```

However, this change might prevent some NFS clients from operating correctly. There are reported problems with older versions of Linux and SCO UNIX. Also, some PC-based NFS client software applications are affected by this port restriction.

Preventing Attempts to Execute Code on Stacks

Some security exploitation programs take advantage of the Solaris OE kernel executable system stack to attack the system. These attack programs attempt to overwrite parts of the program stack of a privileged program in an attempt to control it.

Methods of protecting against these attacks are as follows:

- The global `/etc/system` flag is available in the Solaris 2.6 OE `non_exec_stack`.
- In Solaris 9 OE, all system commands are linked with a map file (that is shipped in `/usr/lib/ld/map.noexstk` that turns off the exec flag.
- The SPARCv9 64-bit API prohibits the execute flag on stack pages.

Each of these is described in the following paragraphs.

In Solaris 2.6 OE and later, some of these exploits can be avoided by making the system stack nonexecutable. Add the following lines to the `/etc/system` file:

```
set noexec_user_stack = 1
set noexec_user_stack_log = 1
```

With `noexec_user_stack_log` enabled, the system logs programmatic attempts to execute code on the stack. This feature allows you to track unsuccessful exploit programs and the account which made the attempt. Here is an example of a log message from a recent Solaris OE exploitation program that was stopped by enabling this feature:

```
Nov 28 11:59:54 landreth unix: sdtcm_convert[308] attempt to
execute code on stack by uid 38918
```

This buffer overflow in `sdtcm_convert` is corrected with a patch. However, the unpatched version of the program is somewhat resistant to the attack because the stack is not executable. Nonexecutable stacks provide some added protection against vulnerabilities for which no patch is issued.

This feature does not stop all buffer overflow exploitation programs, and it does not work on Intel x86-based or older SPARC hardware. Some overflow exploitation programs work on different principles that nonexecutable stacks cannot protect against. Always install the latest security patches. The nonexecutable stack feature works only on the following SPARC architectures: sun4d, sun4m, and sun4u hardware.

Note – All 64-bit Solaris OE processes use nonexecutable stacks by default.

Managing Core Files

Core files contain the memory image of an executing process that was terminated upon receipt of a certain signal. These files (with the file name core) are often used to investigate program errors. There are two problems with them: core files consume disk space and may contain sensitive information.

The size of the core file is based on the amount of memory consumed by the process during execution. A core file can take up a great amount of file space. A system with a full root (/) file system may not perform as expected.

More importantly, the core file may contain privileged information that users should not be able to access. While running, the process may read the /etc/shadow file to check a password or load a protected configuration file. These pieces of information are normally hidden from users but may exist in the core file. This information may be used to attack the system.

For security reasons, the Solaris OE does not write core files for processes with an effective ID that is different from the real ID. This restriction means that `set-user-ID` and `set-user-ID` programs do not create core files.

If core files must be used for application debugging, clean up the old ones. From time to time, search the file system for old core files and delete them. This practice helps prevent the file system from becoming too full.

Solaris 7 OE (8/99) and later releases include a new system utility for managing core files. The `coreadm` command allows an administrator to define directories and file name patterns for core files. It allows `set-user-ID` programs to create core files for debugging purposes. For environments where centralized SYSLOG servers are deployed, we highly recommend that you use `coreadm` to configure the `coreadm.conf` file so that it logs all core dump attempts through SYSLOG.

The `set-user-ID` feature must be used with care and should be enabled only on development and testing systems. This feature can be added to older Solaris 7 OE releases with patches 106541-06 or later for SPARC and 106542-06 or later for Intel systems. All Solaris OE versions after Solaris 7 OE include it.

The following example shows a `/etc/coreadm.conf` file configured to store core files in `/var/core`, using the `set-user-ID` `coreadm` features and generating a SYSLOG message when creating a core file. Do not edit the `coreadm.conf` file manually. For detailed instructions, refer to the `coreadm` man page.

```
COREADM_GLOB_PATTERN=/var/core/core.%f.%p.%n.%u.%g.%t
COREADM_INIT_PATTERN=/var/core/core.%f.%p.%n.%u.%g.%t
COREADM_GLOB_ENABLED=yes
COREADM_PROC_NEABLE=yes
COREADM_GLOG_SETID_ENABLED=no
COREADM_PROC_SETID_ENABLED=no
COREADM_GLOB_LOG_ENABLED=yes
```

Managing Log Files

Log files are used by the system and applications to record actions, errors, warnings, and problems. They are often quite useful for investigating system quirks, discovering root causes of problems, and watching attackers. There are typically two types of log files in the Solaris OE: system log files managed by the `syslog` daemon and application logs created by an application.

This section contains the following topics:

- “Using and Monitoring SYSLOG Log Files” on page 28
- “Using and Monitoring Application Log Files” on page 29

Using and Monitoring SYSLOG Log Files

The `syslog` daemon receives log messages from several sources and directs them to the appropriate location based on the configured facility and priority. There is a programmer interface [`syslog()`] and a system command (`logger`) for creating log messages. The facility (or application type) and the priority are configured in the `/etc/syslog.conf` file to direct the log messages. The directed location can be a log file, a network host, specific users, or all users logged into the system.

By default, the Solaris OE defines two log files in the `/etc/syslog.conf` file. The `/var/adm/messages` log files contain a majority of the system messages. The `/var/log/syslog` file contains mail system messages.

A third log file is defined but commented out by default. It logs important authentication log messages to the `/var/log/authlog` file.

▼ To Enable Logging Messages

1. **Uncomment the following line in `/etc/syslog.conf` to enable logging messages:**

```
# auth.notice ifdef(`LOGHOST', /var/log/authlog, @loghost)
```

2. **Save the file and use one of the following commands to force SYSLOG to re-read its configuration file.**

- For Solaris OE versions prior to 7:

```
# kill -HUP `cat /etc/syslog.pid`
```

- For Solaris OE versions 7 and later:

```
# pkill -HUP syslogd
```

3. Examine all of these files regularly for errors, warnings, and signs of an attack.

This task can be automated by using log analysis tools or a simple `grep` command.

Using and Monitoring Application Log Files

Application log files are created and maintained by commands and tools without using the SYSLOG system. The Solaris OE includes several commands that maintain their own log files. The following is a list of some of the Solaris OE log files:

- `/var/adm/sulog` messages from `/usr/bin/su`
- `/var/adm/vold.log` messages from `/usr/sbin/vold`
- `/var/adm/wtmpx` user information from `/usr/bin/login`
- `/var/cron/log` messages from `/usr/sbin/cron`

The `/var/adm/wtmpx` file should be viewed with the `last` command.

The `/var/adm/loginlog` file does not exist in the default of the Solaris OE installation; we recommend that you create it. When this file exists, the `/usr/bin/login` program records failed login attempts.

Monitor all of these logs for problems.

Configuring Authentication

The following items apply to both local and remote authentication.

This section contains the following topics:

- “Displaying Access Warnings” on page 30
- “Using the Pluggable Authentication Module (PAM)” on page 31
- “Setting Console Access” on page 32

Displaying Access Warnings

The contents of the `/etc/issue` file are displayed on the console during login and for incoming Telnet connections. It is used to display information about the system or network. This file should contain warnings about inappropriate and unauthorized use of the system. It should warn users that their sessions and accounts may be monitored for illegal or inappropriate use. Consult your legal counsel for more information.

Here is a sample legal warning from the Solaris Security Toolkit:

```
This system is for the use of authorized users only.  
Individuals using this computer system without authority, or in  
excess of their authority, are subject to having all of their  
activities on this system monitored and recorded by system  
personnel.
```

```
In the course of monitoring individuals improperly using this  
system, or in the course of system maintenance, the activities  
of authorized users may also be monitored.
```

```
Anyone using this system expressly consents to such monitoring  
and is advised that if such monitoring reveals possible  
evidence of criminal activity, system personnel may provide the  
evidence of such monitoring to law enforcement officials.
```

You can also use the message of the day file (`/etc/motd`) to display warnings.

Using the Pluggable Authentication Module (PAM)

The Pluggable Authentication Module (PAM) framework provides a modular and flexible way to replace the authentication subsystem without having to modify or recompile the installed programs that access the authentication mechanism.

All the Solaris OE authentication applications use the PAM system to authenticate users and manage accounts. Each PAM module can be implemented as a shared library object. The configuration file for the PAM system is `/etc/pam.conf`.

The PAM system exists to provide system programmers the ability to replace the methods used to manage accounts and users. For example, it might be desirable to limit the time periods that a group of users is allowed to be logged into a system. To implement this feature, a PAM module can be written to restrict users in this way without having to replace authentication programs.

To disable a specific login method, remove or comment out its entry in the PAM configuration file. The `rlogin` and `rsh` services use inadequate authentication for security and should be replaced with a Secure Shell protocol system.

Entries in the `/etc/pam.conf` for unused or insecure services should be commented out.

For Solaris 8 OE and older versions, the following example shows the lines that are removed to disable `rlogin` and `rsh`:

```
rlogin auth sufficient /usr/lib/security/pam_rhosts_auth.so.1
rsh auth required /usr/lib/security/pam_rhosts_auth.so.1
```

For Solaris 9 OE, the entries in `/etc/pam.conf` changed. The following example shows the lines that are removed to disable `rlogin` and `rsh` in Solaris 9 OE and newer versions:

```
rlogin auth requisite pam_authok_get.so.1
rlogin auth required pam_dhkeys.so.1
rlogin auth required pam_unix_auth.so.1
rsh auth required pam_unix_auth.so.1
```

If you disable the PAM configuration for `rlogin` and `rsh` services, also remove them from the `/etc/inet/inetd.conf` file. Refer to the next section for more information.



Caution – Be careful when editing the `/etc/pam.conf` file. Errors prevent all PAM services from operating and users are not able to log in. To correct the problem, the system must be booted into single-user mode. Also, do not change the original ownership or file permissions of the `/etc/pam.conf`, because this prevents PAM from operating and prevents users from logging into the system.

Setting Console Access

The `login` command is part of the authentication process to access a local Solaris OE account. It is used on all access mechanisms using `/usr/bin/login`, such as the console and the `in.telnetd` daemon, to determine if a user may be granted access to the system. By default, the root user can only log into a Solaris OE system from the console device.

The console device is defined by the following entry in the `/etc/default/login` file:

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

When this line is commented out, the root account can log directly into the system over the network via Telnet, in addition to the console. This access is not secure and should be avoided. Do not alter the default configuration.

There are two other potential settings for `CONSOLE` entry in `/etc/default/login`.

▼ To Permit Root Logins Only Through the `ttya` Serial Device

- **Use the following entry in `/etc/default/login` file to permit root log ins only through the `ttya` serial device:**

```
CONSOLE=/dev/ttya
```

▼ To Restrict Direct Root Logins Entirely

- **Make the following `CONSOLE` entry in the `/etc/default/login` file to restrict direct root log ins entirely:**

```
CONSOLE=-
```

The recommended configuration is the default—where root logins are only permitted on the console.

Note – The Secure Shell version bundled with Solaris 9 OE does not restrict access based on the `CONSOLE` entry in the `/etc/default/login` file. Regardless of the `CONSOLE` entry, the root user may directly log in to a Solaris 9 OE system. A root user’s access to a Solaris 9 OE system is controlled through the Secure Shell configuration file.

Securing Network Services

Network services enable distributed computers and their users to communicate, access remote systems and information, transfer files, send electronic mail, print files on network printers, and manage remote systems. Multi-user operating systems such as the Solaris OE typically provide many network services.

In the standard Solaris OE configuration, even desktop systems offer some network services. Also, many third-party applications provide additional network services when deployed on the Solaris OE. These services are either necessary for the operation or management of the application (for example, VERITAS Volume Manager Storage Administrator, a web-based GUI management tool) or are essential to the service the application provides (for example, Netscape Enterprise Server, a web server). A standard Solaris OE installation with third-party applications may provide many different network services.

To facilitate rapid system deployment, by default the Solaris OE is designed to provide unrestricted access to many installed network services. This approach allows customers to quickly integrate Solaris OE systems into their computing environments with little effort and few administrative requirements. Most of the enabled network services are not necessary or even used in some environments. For security purposes, all unneeded network services should be disabled, and all required network services should be protected.

Installation and minimization of the Solaris OE are important to the security of the system. This section describes the network services provided when all Solaris OE bundled packages are installed (the Entire Distribution cluster). If a smaller installation cluster is used, some of these services are not installed. The Solaris OE Core cluster contains the fewest packages and services.

If the recommendations from the Sun BluePrints article “Solaris™ Operating Environment Minimization for Security: A Simple, Reproducible, and Secure Application Installation Methodology” are followed, then fewer network services are installed.

The network services a system provides are the entry points into that system. It is important to understand the default configuration of Solaris OE services and the methods used to disable them. Often, organizations need to use protocols or services that are not secure. For these commonly used insecure services (such as RPC, NFS, and Trivial FTP), we provide suggestions for improving security.

Services offered by a system should be protected by as many layers of security as possible. This protection starts at the network level. Refer to the Sun BluePrints OnLine article “Network Settings for Securing the Solaris Operating Environment.” It describes actual network attacks, lists available Solaris OE configuration options, and makes recommendations to provide additional protection for the ARP, ICMP, IP, TCP, and UDP protocols at the network driver layer.

Network services may be attacked in many different ways. These services may contain programming flaws, use weak or no authentication, transfer sensitive data in unencrypted format, and allow connections from any network host. These weaknesses allow a system to be compromised by an attacker.

There are some simple methods to reduce the risk of successful attacks against a system. Administrators should disable unneeded services and apply all security patches. In addition, network services with security features (for example, encryption, strong authentication, etc.) should be used whenever possible.

This section contains the following topics:

- “Using Network Security Tools” on page 35
- “Securing Telnet Connections” on page 37
- “Securing Remote Access Connections” on page 37
- “Securing Remote Execution” on page 38
- “Securing FTP and Using Alternatives” on page 38
- “Enabling TCP Wrappers” on page 39
- “Enabling Trivial FTP” on page 40
- “Securing Managed Services” on page 41
- “Securing Remote Procedure Call (RPC) Services” on page 42
- “Disabling or Securing NFS Services” on page 44
- “Disabling or Securing Automount Services” on page 45
- “Securing sendmail Services” on page 46
- “Configuring Name Service Caching” on page 49
- “Securing Print Services” on page 49
- “Configuring IP Forwarding” on page 50
- “Configuring Network Routing” on page 50
- “Disabling Multicast Routing” on page 51

- “Minimizing `inetd`” on page 53
- “Modifying Network Service Banners” on page 54

Using Network Security Tools

The Solaris 9 OE is the first Solaris OE release to bundle several tools that can provide protection for network services. These are as follows:

- Solaris Secure Shell (based on the OpenSSH source)
- Kerberos Key Distribution Center (KDC) supporting version 5
- TCP Wrappers

These integrated tools simplify the protection of network services because administrators no longer need to integrate these tools into Solaris 9 OE.

For Solaris 8 OE and previous Solaris OE releases, SunScreen™ and SunScreen Lite are two products from Sun Microsystems that provide network protection. Both are firewall products that provide network-level access control and logging. The SunScreen Lite product is a feature-reduced version of the SunScreen software that is available for the Solaris 8 OE release at no cost. The SunScreen Lite product is limited to two network interfaces, but it still provides adequate protection for network services. Use the SunScreen software for systems where more than two network interfaces are required.

Solaris 9 OE includes the entire release of SunScreen 3.2 on the Solaris OE CDs at no cost. All the capabilities available in SunScreen are enabled and available through the installation of the software.

Firewall products like these can be deployed on servers and even desktops where IP forwarding is not required but network service protection is. Massive deployments and management of firewalls on many systems can be burdensome, so plan appropriately.

Additionally, there are well-regarded open source and commercial tools that allow Solaris OE administrators to add protection to Solaris 9 OE systems and to provide protection for earlier Solaris OE releases. These tools address security concerns by providing the following protection: access control, logging, strong authentication, and privacy through encryption.

OpenSSH (an open source toolkit) and SSH (a commercial product) are both suites of tools to replace unsafe UNIX® network commands such as `telnet`, `ftp`, `rlogin`, `rsh`, and `rcp` and to provide secure tunnel X window network communications. The Secure Shell implementation in Solaris 9 OE is based on this software.

OpenSSH, SSH, and Solaris™ Secure Shell all provide strong authentication and privacy through encryption. When built with the TCP Wrappers library, OpenSSH benefits from TCP Wrapper access control, which is used with Solaris Secure Shell.

Like TCP Wrappers, OpenSSH, SSH, and Solaris Secure Shell are straightforward to deploy on many systems. They are very valuable tools simply because of the number of unsafe commands they replace. Once deployed, the replaced network services should be disabled in favor of OpenSSH, SSH, or Solaris Secure Shell.

TCP Wrappers, an open source tool developed by Wietse Venema, provides TCP level access control, logging, and DNS hostname verification. With the release of Solaris 9 OE (tcpd), this functionality is integrated into the OS. TCP Wrappers can be used to protect network services managed by `inetd`. The TCP Wrappers tool provides a flexible configuration mechanism for controlling incoming connections based on pattern matching for hostnames, DNS domains, network addresses, and NIS netgroups. The tool provides better logging and detects DNS hostname discrepancies that may indicate an attack is in progress. TCP Wrappers are fairly straightforward to deploy on Solaris OE systems.

Note – For Solaris 9 OE, the TCP Wrappers man page and the `tcpd` binary are in `/usr/sfw/man` directory, because the stability of the interfaces is external.

▼ To Access the `tcpd` Man Page

- Enter the following command to access the `tcpd` man page:

```
# man -M /usr/sfw/man tcpd
```

Sun Microsystems has a more sophisticated product that provides strong authentication and privacy for intranet network services and systems, called the Sun Enterprise™ Authentication Mechanism. It is based on MIT's Kerberos V system. The Sun product provides centralized security management and interoperates with other heterogeneous Kerberos systems. For Kerberos to be used effectively and correctly, an entire infrastructure of Kerberos components must be deployed. This infrastructure adds additional administrative overhead that might not be desired. For additional information on Sun Enterprise Authentication Mechanism or Kerberos, refer to the Sun BluePrints OnLine article "Kerberos Network Security."

Securing Telnet Connections

Telnet is a user-interactive service used to log into and access a remote system on the network. Unfortunately, this service provides little in the way of security. The only authentication information required is user name and password. Neither of these pieces of information are encrypted while in transit and are therefore vulnerable to a variety of attacks including man in the middle attack, session hijacking, and network sniffing. The Sun Enterprise Authentication Mechanism product provides a replacement `telnet` command that uses strong authentication and encryption. Tools implementing the Secure Shell protocol can also serve as an effective replacement.

If you must use a `telnet` daemon that does not support encryption, then use a product or technology such as One Time Passwords, host-based firewalls, or TCP Wrappers to secure the connections.

One Time Passwords protects against network sniffing by not transmitting the password over the network. Instead, a challenge issued by the server in combination with a secret phrase is used to generate the password for authentication.

Host-based firewalls and TCP Wrappers limit the hosts that might connect to a system. By restricting access to services based on IP addresses, a system can limit its exposure to network attacks.



Caution – None of these alternatives protect a session against being “hijacked” by a malicious user. A session is hijacked when a malicious user, in effect, takes over the session from the authorized user. Session hijacking can be prevented only through the proper use of encryption.

Securing Remote Access Connections

Access control and accountability are critical to the security of a system. Access control should involve strong authentication for system access, while accountability information should provide tracking data relative to system changes.

The standard `r*` commands (`rsh`, `rlogin`, and `rcp`) break both of these requirements, because most implementations of `r*` commands involve “zones of trust.” Within a zone of trust, all systems are trusted and no additional authentication is required. Hence, an intruder need only gain access to one server in order to gain access to all servers.

The default authentication mechanism of the `r*` daemons uses the host name or IP address of a system in combination with the user ID for authentication. No additional authentication is required. Considering the ease in which an IP address

and user ID can be stolen or misused, this default is clearly not a secure mechanism. The `r*` commands should not be used in this manner and no servers should offer the service in this manner.

One way to secure `r*` daemons is with Kerberos; the Sun Enterprise Authentication Mechanism product provides the appropriate replacement for `r*` clients and servers through Kerberos.

Securing Remote Execution

The remote execution server daemon, `in.rexecd`, is started from `/etc/inetd.conf` when a connection request is made. This daemon provides remote execution facilities based on user name and password information.

Once authenticated, the daemon executes the command passed along with the authentication information. As with the `in.telnetd` daemon, neither the user name nor password is encrypted while transmitted over the network. This exposes the `in.rexecd` daemon to the same man in the middle, session hijacking, and network sniffing attacks as the `in.telnetd` daemon. For this reason, we recommend that you disable the `in.rexecd` entries in `/etc/inetd.conf` file.

Securing FTP and Using Alternatives

The `ftp` daemon has many of the same problems as the `telnet` daemon. All authentication information transmitted over the network is in clear-text, in much the same fashion as the Telnet protocol. This default exposes the FTP protocol to many of the same attack scenarios as `telnet`, including man in the middle, session hijacking, and network sniffing. For these reasons, alternatives to FTP should be considered when FTP transport functionality is required.

There are several alternatives to FTP that provide strong encryption and authentication. Sun Enterprise Authentication Mechanism provides a mechanism to secure FTP. Another alternative is to use Solaris Secure Shell implemented in Solaris 9 OE, or OpenSSH or SSH with earlier Solaris OE releases.

Solaris 9 introduces a variety of new capabilities to the `in.ftpd` daemon based on the Washington FTP (`wu-ftp`) implementation. These may include controlling FTP access through the `/etc/ftpd/ftpaccess` file, logging all `ftp` commands to `SYSLOG`, explicit definition of the control and data ports to be used, the ability to `chroot` the FTP session, in addition to other capabilities. Refer to the various FTP man pages for additional information.

In Solaris 9 OE, all FTP capabilities can be removed from a system by removing the `SUNWftpr` and `SUNWftpu` packages.

Solaris OE versions prior to version 9 have two capabilities in the `in.ftpd` daemon that can provide additional security. The first is the `/etc/ftpusers` file, which is used to restrict access to the system through FTP.

A default `ftpusers` file is included with Solaris OE versions 8 and 9. Solaris 8 OE stores it in `/etc`, and Solaris 9 OE stores it in `/etc/ftpd`. All accounts *not* allowed to use the incoming FTP service should be specified in this file. At a minimum, this should include all system accounts (for example, `bin`, `uucp`, `smtp`, `sys`, and so forth) in addition to the root account. Only intruders and individuals attempting to gain unauthorized access use FTP with these accounts. Frequently, root access to a server over Telnet is disabled and root FTP access is not. This configuration provides a backdoor for intruders who might modify the system's configuration by uploading modified configuration files.

The second security feature of the `in.ftpd` daemon is the ability of the daemon to log the IP addresses of all connections and commands issued to the `ftp` daemon through the SYSLOG service. Logging of IP addresses is enabled with the `-l` option. Commands issued to the `ftp` daemon are logged when the `-d` option is used. By logging FTP connection requests and commands to a log server for parsing, unauthorized access attempts can be tracked and resolved.

Enabling TCP Wrappers

▼ To Enable TCP Wrappers on a Solaris 9 OE System

1. **Modify the `ENABLE_TCPWRAPPERS` in `/etc/default/inetd` as follows:**

```
ENABLE_TCPWRAPPERS=YES
```

2. **After enabling wrappers, verify that `inetd` has either restarted or sent a HUP and that services listed in `/etc/inetd.conf` can use the capabilities of TCP Wrappers.**

A simple TCP Wrapper configuration could have the following configuration in its `/etc/hosts.allow` and `/etc/hosts.deny` files:

```
# cat /etc/hosts.allow
ALL: LOCAL 10.8.10.0/255.255.255.0 10.8.11.0/255.255.255.0
in.ftpd: 10.8.31.100
in.telnetd: 192.168.254.10
sshd: 10.0.0.0/255.0.0.0 192.168.0.0/255.255.0.0

# cat /etc/hosts.deny
ALL: ALL
```

This sample configuration restricts access to all `/etc/inetd.conf` started services to the local subnet and two class C IP ranges (10.8.10.0 and 10.8.11.0). In addition, ftp, Telnet, and SSH all allow other IP addresses to connect as well. For more information on TCP Wrapper capabilities, refer to its documentation.

Enabling Trivial FTP

The trivial FTP service (`in.tftpd`) exists to provide diskless systems with a way to access files on the network. The `in.tftpd` daemon has no authentication and only allows clients to access publicly readable files in a restricted directory. Diskless workstations, X-terminals, and some printers use this service to load files needed to boot. The `in.tftpd` is managed by the `inetd` server process and is configured in `/etc/inetd.conf`. By default, it is not enabled in the Solaris OE.

If this service is necessary, it should be configured securely. The default entry in the Solaris OE `/etc/inetd.conf` is configured correctly. When enabled, `in.tftpd` runs as the user `nobody` and restricts client access to the `/tftpboot` directory (the internal default) or a specified directory. The `-S` option provides additional protection by requiring that the `/tftpboot` directory exist. If it does, the `in.tftpd` changes the root directory, using `chroot()`, to `/tftpboot`. This option should always be used when TFTP functionality is required.

Securing Managed Services

The `inetd` daemon controls a majority of the minor network services available on a system. Its configuration file, `/etc/inetd.conf`, defines what services are managed by the `inetd` daemon.

An ideally secured server should neither have an `/etc/inetd.conf` nor run `inetd`, because the daemons started in the `/etc/inetd.conf` are frequently not needed.

The Solaris 9 OE is the first Solaris OE release that adds entries only to the `/etc/inetd.conf` file when specific packages are installed. For example, the `/etc/inetd.conf` entries for FTP are added only when the `SUNWftpr` package is added. Refer to the Sun BluePrints OnLine article “Solaris™ Operating Environment Minimization for Security: A Simple, Reproducible, and Secure Application Installation Methodology.”

Solaris 9 OE implements an `/etc/default/inetd` to control the use of TCP Wrappers and connection logging. Also, connection logging can be enabled through the use of the `-t inetd` option in Solaris 9 OE, as with earlier Solaris OE releases.

Of the daemons started from the `/etc/inetd.conf`, the remote access FTP, TFTP, and Telnet services were described earlier. The RPC and print services are described later in this article. The remaining `/etc/inetd.conf` entries are as follows:

- `in.tnamed` – supports the DARPA Name Server Protocol. This daemon should be disabled.
- `in.uucpd` – supports UUCP connections over networks. This service should be disabled unless UUCP is used.
- `in.fingerd` – provides information on local system accounts. This service should be disabled unless needed.
- `systat` – provides anyone connecting to the system with the output of `ps -ef`. This service should be disabled because it provides too much system information.
- `netstat` – provides a list of current network connections via the output of the `netstat` command. This service should be disabled because it provides system information that can be used to launch attacks against the system.
- `time` – prints out the current time and date. Beginning with Solaris 2.6 OE, the `xntp` functionality is included with the Distribution cluster for time synchronization. The `xntp` daemon offers additional security and functionality improvements over `rdate` and `time`. Whenever possible, `xntp` should be used.
- `echo` – echoes back the incoming data stream. This service should be disabled.
- `discard` – discards the incoming data stream. This service should be disabled.
- `chargen` – generates a continuous stream of characters. This service should be disabled.

These entries in the `/etc/inetd.conf` file should be removed on most systems. Once removed, restart the system and test applications to verify that required functionality is not affected.

For restricted access servers, all connections to services managed by `inetd` should be logged.

- For Solaris 8 OE and older versions, this logging can be done by adding an additional option to the startup of `inetd` in `/etc/rc2.d/S72inetsvc`. By adding a `-t` option, the `inetd` daemon logs the IP address of all systems requesting `inetd` based services.
- For Solaris 9 OE and newer versions, this logging can be done by enabling the `ENABLE_CONNECTION_LOGGING` option in the `/etc/default/inetd` file.

Regardless of Solaris OE version, the IP addresses of all systems requesting services based on `inetd` are logged through the `SYSLOG` service.

▼ To Disable a Service

1. **Edit the `/etc/inetd.conf` file and place a comment character (#) in front of the line containing the service definition.**
2. **Send a HUP signal to the `inetd` process.**

This signal causes it to reread its configuration file.

Securing Remote Procedure Call (RPC) Services

The remote procedure call (RPC) mechanism provides a way for network services to communicate and make procedure calls on remote systems. When a new RPC service is started, it registers with `rpcbind`, the central RPC service agent. The `rpcbind` maintains a table of RPC services (listed by a program number) and the network address(es) on which RPC listens for clients to connect. A client first communicates with the `rpcbind` service to determine the network address it must use to contact a particular RPC service. Current RPC services can be listed using the `rpcinfo` command that communicates with the `rpcbind` service.

RPC services are used in many UNIX services including: NFS, NIS, NIS+, and Kerberos. RPC services are used also by many applications such as Solstice DiskSuite™ software, Sun™ Cluster software, and others.

When an RPC service is started, the service tells the `rpcbind` daemon the address where it is listening and the RPC program numbers it is prepared to serve. When a client wishes to make an RPC call to a given program number, it first contacts the

`rpcbind` daemon on the server machine to determine the address where RPC requests should be sent. The `rpcinfo` command can be used to determine what RPC services are registered on a host.

RPC, by itself, can be used to provide an attacker with information about a system. While this may not be ideal, the real security problem is not with the `rpcbind` daemon itself, but rather with many of the services that use RPC. Many of these services do not make use of the stronger authentication mechanisms available to them and default to weak authentication. In particular, `rpc.cmsd`, `sadmind` (running without `-S 2`), and `rpc.rexd` use weak authentication by default. Network-based attacks against these services pose a significant threat to the security of a server.

The daemons and services that use RPC on a Solaris OE system include the following:

- `cachefs`
- `kcms.server`
- `kerbd`
- `keyserv`
- `nis_cachemgr`
- `rpcbind`
- `rpc.bootparamd`
- `rpc.cmsd`
- `rpc.nisd`
- `rpc.nispasswd`
- `rpc.rexd`
- `rpc.rstatd`
- `rpc.rusersd`
- `rpc.rwalld`
- `rpc.sprayd`
- `rpc.ttdbserver`
- `rquotad`
- `sadmind`
- `testsv`
- `ufsd`
- `xaudio`

Whether these services are included in Solaris 9 OE installation depends on the packages installed. Many of these services are only installed if their related packages are installed. Solaris 9 OE no longer installs a fully populated `/etc/inetd.conf` with the seven Core Solaris OE packages.

On almost all servers, the RPC services in `/etc/inetd.conf` can be removed. Many applications that use RPC services add entries to the `/etc/inetd.conf` in addition to using one of the RPC-based daemons. The RPC services in `/etc/inetd.conf` should be removed unless required.

The RPC daemons started in `/etc/rc2.d` and `/etc/rc3.d` are for `rpcbind`, `keyserv`, various naming services (for example, NIS and NIS+), and are used by both the client and server components of NFS. The `keyserv` daemon must be run when `AUTH_DES` is used for stronger host and user authentication. The use of NIS is not recommended due to its weak security models. NIS+ provides a much more robust security model.

The RPC protocol provides support for various authentication alternatives. These are as follows:

- `AUTH_NONE` – No authentication.
- `AUTH_SYS` or `AUTH_UNIX` – Traditional UNIX-style authentication.
- `AUTH_DES` – DES encryption-based authentication.
- `AUTH_KERB` – Kerberos encryption-based authentication.

Some RPC daemons and services provide options for an administrator to specify the security model (for example, NFS, `sadmind`, NIS+) while others do not. If RPC must be used, then only those services and daemons that provide support for `AUTH_DES` should be used. This combination of RPC and `AUTH_DES` authentication is called Secure RPC. Refer to “References and Related Resources” on page 56 for additional references to Secure RPC.

Disabling or Securing NFS Services

A Solaris OE system can be either an NFS server, NFS client, both, or neither. From a security perspective, the best option is to neither provide NFS services nor accept them from any other systems.

To disable all client and server NFS daemons on Solaris 9 OE, remove the following fix packages with `pkgrm`:

- `SUNWnfscr`
- `SUNWnfscu`
- `SUNWnfscx`
- `SUNWnfssr`
- `SUNWnfssu`
- `SUNWnfssx`

On previous Solaris OE releases, the NFS services cannot be removed through `pkgrm`. Disable the following startup scripts on the system:

- `/etc/rc2.d/S73nfs.client`
- `/etc/rc3.d/S15nfs.server`

The Solaris OE uses a different set of startup files to enable NFS server or NFS client services.

Frequently, business requirements require an NFS server. Several different levels of security are available in the NFS server itself. In addition, careful configuration can greatly improve security. Here is a quick overview:

- Explicitly list hosts allowed access to NFS server directories. Do not open access to all systems.
- Export only the lowest directory necessary.
- Export read-only whenever possible.
- Use strong authentication methods such as AUTH_DES or AUTH_KERB whenever possible.

The NFS server and the various mechanisms available to secure it encompass more material than can be covered here.

Disabling or Securing Automount Services

The `automount` service manages automated NFS mounts. NFS clients might need to mount file systems from many different NFS servers. The automount service mounts file systems automatically when they are needed, and unmounts them after a specific amount of idle time.

A table used by this service defines the file system mount points, mount options, and the associated NFS servers. Also, in order to centralize the management of automount, the configuration tables can be stored in a name service such as NIS or NIS+. A kernel level service (`autofs`) interacts with the system daemon (`automountd`) to manage file system mount and unmount requests. The primary automount configuration table is stored in the `/etc/auto_master` file.

With the Solaris OE version 2.6 release, the `automount` software was, for the first time, placed in separate Solaris OE packages. By removing these packages, all automount functionality is removed from the system. The two packages that include all the automount functionality are `SUNwatfsr` and `SUNwatfsu`.

The file `/etc/auto_master` file determines the locations of all `autofs` mount points. By default, this file contains four entries:

```
# Master map for automounter
#
+auto_master
/net -hosts -nosuid
/home auto_home
/xfn -xfn
```

Ideally, `automount` should be disabled, because not only does it run as a privileged daemon, but it uses NFS and RPC. The `automount` system can be disabled by renaming `/etc/rc2.d/S74autofs`.

There are situations where the `automount` service is needed for its ability to mount and unmount file systems automatically. In particular, both NIS and NIS+ environments make extensive use of `auto_home` and `auto_master` maps to mount user home directories. In these situations, the configuration of the `auto_master` map should be carefully constructed to be as restrictive and secure as possible. You can do this by using NFS mount options and Secure RPC.

Frequently, NFS servers allow any system to mount the file systems they export. This incorrect and all too common practice allows attackers to mount file systems that might contain sensitive information. If attackers want to modify the contents of a particular file, they need only change their user ID or UID to that of the file and modify its contents. These attacks, and many other NFS-based attacks, can be avoided through the use of appropriate NFS exports, Secure RPC, or Kerberized NFS.

Securing sendmail Services

The `sendmail` daemon is used on a Solaris OE system to forward and receive mail from other systems. Centralized mail servers should be used to receive mail instead of using local servers. These local systems should, however, be able to generate mail and forward it to other servers.

Ideally, a more secure Mail Transport Agent (MTA) should be used instead of the MTA bundled with the Solaris OE. The `sendmail` daemon bundled with the Solaris OE has been subject to numerous denial of service, buffer overflow, and misconfiguration attacks. Alternative MTAs with smaller and more robust code are available. These other MTAs are more security conscious and, if configured properly, compromise the security of the server less than `sendmail`.

If `sendmail` must be used, then refer to recommendations made at SendMail Consortium, in the Sendmail O'Reilly books, and through the SunSolve OnlineSM service. There are a wide variety of `sendmail` versions in use, and there are differences in the associated `sendmail.cf` configuration files. Because of this, a sample `sendmail.cf` file is not included with this article.

Removing sendmail

If `sendmail` is not required, or only outgoing `sendmail` is required, we recommend that you remove, disable, or enable only outgoing `sendmail`. The options available vary according to the Solaris OE release you installed. TABLE 1 shows the methods available.

TABLE 1 sendmail Methods Available by Solaris OE Version

Methods	Solaris 2.6 OE and Earlier	Solaris 7 OE	Solaris 8 OE	Solaris 9 OE
Remove (pkgrm) Command	Not Available	Available	Available	Available
Disable Options	Disable startup scripts	Disable startup scripts	MODE option	Modify sendmail.cf file
Enable Outgoing Only	Add cron entry	Add cron entry	MODE option	Modify sendmail.cf file

The following subsections provide instructions and recommendations for removing or disabling sendmail.

▼ To Remove sendmail

- **If no sendmail functionality is required, remove it:**
 - For Solaris 2.6 OE and earlier releases, comment out the /etc/rc2.d/S88sendmail script. After this script is commented out, sendmail is not started during system startup.
 - For Solaris OE versions 7, 8, and 9, remove all components of sendmail by removing the SUNWSndmr and SUNWSndmu packages with the pkgrm command.

Allowing Only Outgoing sendmail

The sendmail daemon is not needed for email delivery to other systems. All messages that can be are immediately delivered. Messages that cannot be immediately delivered are queued for future delivery. The sendmail daemon, if running, retries to deliver these messages again.

▼ To Enable Only Outgoing sendmail

- **Perform the following according to which Solaris OE version is installed.**
 - For Solaris OE versions 7 and earlier, use a cron job to start sendmail every hour to process undelivered messages.

The following cron entry starts sendmail every hour to flush the mail queue:

```
0 * * * * /usr/lib/sendmail -q
```

- For Solaris 8 OE releases, use the **MODE** option. Create the following line in a new default configuration file named `/etc/default/sendmail`:

```
MODE=""
```

By defining the **MODE** to be a null string, **sendmail** processes only the outgoing mail queue and does not accept incoming connections.

An example replacement `/etc/default/sendmail` file is available from the Sun BluePrints Tools page at <http://www.sun.com/blueprints/tools>, which documents the other **sendmail** options added in Solaris 8 OE.

Note – The **MODE** option does not work with Solaris 9 OE. For previous Solaris OE versions, it was possible to start **sendmail** in queuing processing only mode. For Solaris 9 OE, it goes through an extra submission hop, sending to `localhost`. If you were to use the **MODE** option in this version, all outbound mail would stop because nothing is listening on the SMTP port on the local host. The mail would remain in the client mail queue.

- For Solaris 9 OE releases, perform the following steps to disable incoming **sendmail** and allow outgoing **sendmail**.

a. cd /usr/lib/mail/cf.

b. cp subsidiary.mc hostname.mc.

c. Edit the `hostname.mc` file, adding the following before the **MAILER() lines:**

```
DAEMON_OPTIONS( 'Name=NoMTA4, Family=inet, Addr=127.0.0.1')dnl
```

d. Enter one of the following commands:

```
/usr/ccs/bin/make hostname.cf  
or  
/usr/ccs/bin/m4 ../m4/cf.m4 hostname.mc > hostname.cf
```

e. cp hostname.cf /etc/mail/sendmail.cf.

f. Enter the restart command as `/etc/init.d/sendmail restart`.

The Solaris 9 OE **sendmail** implements RFC 2476 (Message Submission). For example, it can now listen on several different ports, notably port 587. Use of this port is enabled by default in m4-generated `.cf` files. RFC 2476 support and the use of port 587 can be disabled by adding **FEATURE('no_default_msa')** to the `sendmail.cf` file.

Configuring Name Service Caching

The name service cache daemon (`nscd`) provides caching for name service requests. It exists to provide a performance boost to pending requests and reduce name service network traffic. The `nscd` maintains cache entries for databases such as `passwd`, `group`, and `hosts`. It does not cache the shadow password file, for security reasons. All name service requests made through system library calls are routed to `nscd`. With the addition of IPv6 and RBAC in Solaris 8 OE, the `nscd` caching capability was expanded to address additional name service databases.

Because caching name service data makes spoofing attacks easier, it is recommended that the configuration of `nscd` be modified to cache as little data as possible. This task is accomplished by setting the positive `ttl` to zero in the `/etc/nscd.conf` file for the name service requests deemed vulnerable to spoofing attacks. In particular, the configuration should be modified so that `passwd`, `group`, and Solaris 8 and 9 OE RBAC information has a positive and negative `ttl` of zero.

Note – There may be a performance impact on systems that use name services intensively.

The `nscd -g` option can be used to view the current `nscd` configuration on a server and is a helpful resource when tuning `nscd`.

Disabling `nscd` entirely is not recommended, because applications make name service calls directly, which exposes various bugs in applications and name service backends.

Note – On Solaris 8 OE systems, several RBAC bugs have been encountered at sites disabling `nscd` caching through the `enable-cache passwd|group|hosts|ipnodes` mechanism. To avoid these bugs, apply the available patches that fix the bugs, or disable caching as described earlier in this article.

Securing Print Services

When a Solaris OE system is installed using the End User, Development, or Entire Distribution cluster, the line printing packages are installed. Both the client and server components for print services are enabled by default on these Solaris OE installations.

The `in.lpd` daemon is only necessary for systems that provide network print queue services.

If the system does not participate in print spooling, comment out the following line in the `/etc/inetd.conf` file to disable this service:

```
printer stream tcp nowait root /usr/lib/print/in.lpd in.lpd
```

Conversely, the `/etc/rc2.d/S80lp` script is required both for a server providing print services to other systems and a system that requires access to printers hosted by other systems. If this functionality is not required, the packages for `lp` should be removed from the system, and the `in.lpd` entry should be removed from `/etc/inetd.conf`.

The three packages for `lp` are `SUNWpsr`, `SUNWpsu`, and `SUNWlpmsg`. If all `lp`-related functionality is removed, also remove the Solstice™ Print Client. The Solstice Print Client is located in the `SUNWpcr` and `SUNWpcu` packages.

Solaris 8 OE adds the Solaris Print Manager (`printmgr(1M)`), which is a graphical printer management interface for managing both local and remote printers. The package `SUNWppm` should be removed if this functionality is not required.

Configuring IP Forwarding

During the startup phase of a Solaris OE system, the `/etc/init.d/inetinit` script evaluates the configuration of the system. It determines whether the system is configured as a router and if `ip_forwarding` is enabled between the different interfaces. Solaris 8 OE adds an ability to set `ip_forwarding` on a per-interface basis. For more information on the `ip_forwarding` function, refer to the Sun BluePrints article “Network Settings for Securing the Solaris Operating Environment.”

Configuring Network Routing

The network router (`in.routed`) and router discovery (`in.rdisc`) daemons are used by a Solaris OE system to dynamically determine network routing requirements. Both `in.routed` and `in.rdisc` functionality are addressed in the Sun BluePrints article “Network Settings for Securing the Solaris Operating Environment.”

Disabling Multicast Routing

Multicast is a method to send network data to many systems at the same time with only a single address. Unless the system must participate in a multicast application, it is recommended to disable the code that enables the multicast route assignment.

▼ To Disable the Code for Multicast Remote Assignment

- For Solaris 7 OE and earlier, comment out the following lines in `/etc/init.d/inetsvc`:

```
mcastif=`/sbin/dhccpinfo Yiaddr`  
if [ $? -ne 0 ]; then  
    mcastif=`uname -n`  
fi  
echo "Setting default interface for multicast: \c"  
/usr/sbin/route add -interface -netmask "240.0.0.0" "224.0.0.0"  
"$mcastif"
```

- For Solaris 8 OE, comment out the following lines in `/etc/init.d/inetsvc`:

```
(  
if [ "$_INIT_NET_STRATEGY" = "dhcp" ]; then  
    mcastif='/sbin/dhccpinfo Yiaddr' ||  
mcastif=$_INIT_UTS_NODENAME  
else  
    mcastif=$_INIT_UTS_NODENAME  
fi  
  
echo "Setting default IPv4 interface for multicast:" \  
    "add net 224.0/4: gateway $mcastif"  
  
/usr/sbin/route -n add -interface "224.0/4" "$mcastif" >/dev/null  
) &
```

- For Solaris 9 OE, comment out the following lines in `/etc/init.d/inetsvc`:

```
(
if [ "$_INIT_NET_STRATEGY" = "dhcp" ]; then
    mcastif='/sbin/dhcppinfo Yiaddr' ||
mcastif=$_INIT_UTS_NODENAME
else
    mcastif=$_INIT_UTS_NODENAME
fi

echo "Setting default IPv4 interface for multicast:" \
    "add net 224.0/4: gateway $mcastif"

/usr/sbin/route -n add -interface 224.0/4 -gateway "$mcastif" >/
dev/null
) &
```

When the appropriate lines are commented out, the system should be restarted.

Minimizing inetsvc

Based on the recommendations made in this article, it is possible to construct a minimized `/etc/init.d/inetsvc` file to contain only essential components. Quite a few sections of this file can be commented out including:

- DHCP support
- Use `named` startup support
- Multicast support

▼ To Minimize inetsvc

- For Solaris OE releases 7 and older, comment out all of the previously mentioned entries as follows:

```
#!/bin/sh

/usr/sbin/ifconfig -au netmask + broadcast +
/usr/sbin/inetd -s -t
```

The number of active lines in the `inetsvc` file decreases from 152 to 3 lines.

- For Solaris OE releases 8 and newer, comment out all of the previously mentioned entries as follows:

```
#!/bin/sh

/usr/sbin/ifconfig -auD4 netmask + broadcast +
/usr/sbin/inetd -s -t
```

Modifying Network Service Banners

Some Solaris OE network services provide information on the operating system version when connections are made. This information usually includes a text string indicating the name of the OS and its version. This information may be useful to attackers with exploit programs for specific OS releases. The Solaris OE provides methods to change these messages in an attempt to hide OS information.

These methods provide only minor additional security. There are methods to determine a system's operating system type and version on a network. Several network auditing tools use a technique called "TCP/IP stack fingerprinting" to determine the operating system and version.

▼ To Change Banner Messages for Incoming Telnet Connections

1. **Create an /etc/default/telnetd file.**
2. **Add a line similar to the following, as appropriate for your environment:**

```
BANNER="Authorized Use Only"
```

▼ To Change Banner Messages for Incoming FTP Connections for Solaris 8 OE and Older Versions

1. **Create an /etc/default/ftpd file.**
2. **Add a line similar to the following, as appropriate for your environment:**

```
BANNER="Authorized Use Only"
```

▼ To Change Banner Messages for Incoming FTP Connections for Solaris 9 OE

1. **Review the /etc/ftpd/ftpaccess file to determine if the following entry is present:**

```
banner=/etc/ftpd/banner.msg
```


2. If the entry is not present, add it.
3. Replace the contents of the `/etc/ftpd/banner.msg` file with a line similar to the following, as appropriate for your environment:

```
Authorized Use Only
```

4. Edit the `/etc/inetd.conf` file to add an `-a` option after the `in.ftpd` entry.

If `in.ftpd` is started with the `-a` option, it parses the `/etc/ftpd/ftpaccess` file and displays the banner message listed. By default, this is the `/etc/ftpd/banner.msg` file.

The following is an example of the modified `/etc/inetd.conf` file.

```
ftp stream tcp6 nowait root /usr/sbin/in.ftpd in.ftpd -a -l
```

▼ To Change Banner Messages for the Solaris 9 OE sendmail Daemon

- If you want to change the banner message that the `sendmail` process presents for incoming mail delivery connections, do the following:

- a. Search the `/etc/mail/sendmail.cf` file for the following line:

```
0 SmtgGreetingMessage=$j Sendmail $v/$Z; $b
```

- b. Change it to:

```
0 SmtgGreetingMessage=Mail Server Ready
```

▼ To Change Banner Messages for CDE Connections

1. Create a system warning message in, for example, the `/etc/motd` file.
2. Enter the warning message text into the file.

To improve the resulting display of the text, you may want to center the text and enter hard returns to control line length.

3. Change directory as follows:

```
cd /etc/dt/config/Xsession.d
```

4. Enter the following in your file:

```
dtstart_hello[0]="/usr/dt/bin/dthello -bground white -file /etc/  
motd &"
```

5. Log out of CDE and log back in to implement the warning message.

References and Related Resources

Publications

- AUSCERT, *UNIX Security Checklist*,
ftp://ftp.auscert.org.au/pub/auscert/papers/unix_security_checklist
- Rhoads, Jason. *Solaris Security Guide*,
<http://www.sabernet.net/papers/Solaris.html>
- Spitzner, Lance, *Armoring Solaris*,
<http://www.enteract.com/~lspitz/armoring.html>
- Galvin, Peter Baer. "The Solaris Security FAQ,"
<http://www.sunworld.com/common/security-faq.html>
- Noordergraaf, Alex. "Solaris™ Operating Environment Minimization for Security: A Simple, Reproducible, and Secure Application Installation Methodology," Sun BluePrints OnLine, November 2002.
- Noordergraaf, Alex and Brunette, Glenn. "Solaris Security Toolkit - Installation, Configuration, and Usage Guide: Updated for Version 0.3," Sun BluePrints OnLine, June 2001.
- Noordergraaf, Alex and Brunette, Glenn. "Solaris Security Toolkit - Quick Start: Updated for version 0.3," Sun BluePrints OnLine, June 2001.
- Noordergraaf, Alex and Watson, Keith. "Solaris Operating Environment Security: Updated for the Solaris 9 Operating Environment," Sun BluePrints OnLine, November 2002.
- Powell, Brad. et al., *Titan security tool*, <http://www.fish.com/titan/>
- Reid, Jason M. "Configuring OpenSSH for the Solaris Operating Environment," Sun BluePrints OnLine, January 2002

- Reid, Jason M and Watson, Keith. "Building and Deploying OpenSSH in the Solaris Operating Environment," Sun BluePrints OnLine, July 2001
- Watson, Keith and Noordergraaf, Alex. "Network Settings for Securing the Solaris Operating Environment," Sun BluePrints OnLine, December 2000.

Web Sites

- Casper Dik, fix-modes tool:
ftp://ftp.wins.uva.nl/pub/solaris/fix-modes.tar.gz
- OpenSSH tool: <http://www.openssh.com/>
- Sendmail Consortium: sendmail configuration information, <http://www.sendmail.org/>
- SSH Communications Security, Secure Shell (SSH) tool: <http://www.ssh.com/>
- Sun BluePrints Tools: <http://www.sun.com/blueprints/tools>
- SunScreen and SunScreen Lite: <http://www.sun.com/security/>
- Sun Enterprise Authentication Mechanism product information: <http://www.sun.com/software/solaris/ds/ds-seam>
- TCP Wrappers tool, Wietse Venema: <ftp://ftp.porcupine.org/pub/security/index.html>

About the Authors

Alex Noordergraaf

Alex Noordergraaf has over 10 years experience in the areas of computer and network security. As the Security Architect of the Enterprise Server Products (ESP) group at Sun Microsystems, he is responsible for providing technical leadership to define the security of Sun's next generation servers while addressing security for current products. He is the driving force behind the very popular freeware Solaris Security Toolkit. Prior to his role in ESP, he was a Senior Staff Engineer in the Enterprise Engineering (EE) group of Sun Microsystems, where he developed, documented, and published security best practices through the Sun BluePrints program. Published topics include: Sun Fire Midframe 15K system security, secure N-tier environments, Solaris OE minimization, Solaris OE network settings, and Solaris OE security. He has co-authored two Sun BluePrint books "Jumpstart Technology- Effective Use in the Solaris Operating Environment" and "Enterprise Security Solaris Operating Environment, Security Journal."

Prior to his role in EE, he was a Senior Security Architect with Sun Professional Services where he worked with many Fortune 500 companies on projects that included security assessments, architecture development, architectural reviews, and policy/procedure review and development. He developed and delivered an enterprise security assessment methodology and training curriculum to be used worldwide by SunPS. His customers included major telecommunication firms, financial institutions, ISPs, and ASPs. Before joining Sun, Alex was an independent contractor specializing in network security. His clients included BTG, Inc. and Thinking Machines Corporation.

Keith Watson

Keith Watson spent five years at Sun working in the areas of computer and network security. In 2002, he left Sun and returned to the Center for Education and Research in Information Assurance and Security (CERIAS) at Purdue University. Prior to his departure, Keith was the product manager for Solaris network security. Also, he was a member of the Global Enterprise Security Service (GESS) team in Sun Professional Services. He started at Sun as a research engineer for SunLabs, where he co-developed an enterprise network security auditing tool named the Sun Enterprise Network Security Service (SENSS).

Keith is currently a research engineer at CERIAS, where he manages advanced research projects in intrusion detection and security architecture.