SUN BLUEPRINTS APPROVED

Solaris™ Operating Environment Security

Updated for Solaris 8 Operating Environment

By Alex Noordergraaf and Keith Watson

Sun BluePrints™ OnLine - April 2001

Sun microsystems
ENTERPRISE
ENGINEERING

`http://www.sun.com/blueprints`

Sun Microsystems, Inc.
901 San Antonio Road
Palo Alto, CA 94303 USA
650 960-1300 fax 650 969-9131

Part No.: 816-0428-10
Revision 01, April 2001

RESTRICTED RIGHTS: Use, duplication, or disclosure by the U.S. Government is subject to restrictions of FAR 52.227-14(g)(2)(6/87) and FAR 52.227-19(6/87), or DFAR 252.227-7015(b)(6/95) and DFAR 227.7202-3(a).

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.

CETTE PUBLICATION EST FOURNIE "EN L'ETAT" ET AUCUNE GARANTIE, EXPRESSE OU IMPLICITE, N'EST ACCORDEE, Y COMPRIS DES GARANTIES CONCERNANT LA VALEUR MARCHANDE, L'APTITUDE DE LA PUBLICATION A REPONDRE A UNE UTILISATION PARTICULIERE, OU LE FAIT QU'ELLE NE SOIT PAS CONTREFAISANTE DE PRODUIT DE TIERS. CE DENI DE GARANTIE NE S'APPLIQUERAIT PAS, DANS LA MESURE OU IL SERAIT TENU JURIDIQUEMENT NUL ET NON AVENU.

Please
Recycle

Adobe PostScript

Solaris™ Operating Environment Security

Updated for Solaris™ 8 Operating Environment

Update

This Sun BluePrints™ OnLine article has been updated to include changes in the Solaris™ 8 Operating Environment (Solaris OE). This article is current as of Solaris 8 OE 1/01 (update 3), which was released on January, 2001. Differences between the Solaris OE versions discussed in this article, 2.5.1 through 8, are noted where appropriate.

Overview

The Solaris Operating Environment (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. This article discusses the Solaris OE subsystems and the security issues surrounding those subsystems. In addition, this article provides recommendations on how Solaris OE subsystems should be secured.

As with any security decisions, a balance must exist between system manageability and security. Some changes in this article will not be applicable to all environments. The removal of some of the Solaris OE services mentioned in this article 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, and 8 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 these older versions.

File Systems and Local Security

It is important not to neglect the file systems and local security of a Solaris OE system. Often, administrators are greatly concerned about attackers breaking into systems remotely. There should be equal concern for local, authorized users gaining extra privileges on a system by exploiting a problem with internal system security.

Initial Installation

Building a secure Solaris OE system involves installing a new 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 “JASS” Toolkit. Sun BluePrints articles describing this security Toolkit are available at:

```
http://www.sun.com/blueprints/browsesubject.html#security
```

The “JASS” Security Toolkit itself can be downloaded from:

```
http://www.sun.com/blueprints/tools
```

“JASS” Toolkit

The goal of the “JASS” 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 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. Minimization 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 Toolkit.

The Toolkit was designed to harden systems during installation—this is achieved by using the JumpStart™ technology as a mechanism for running the Toolkit scripts. Additionally, the Toolkit can also be run outside the JumpStart framework in a standalone mode. This standalone mode allows the 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.

Solaris OE Installation

Sun works toward improving the Solaris OE with every release. Each new release includes security improvements and additional features to enhance system security. Always use the latest version of the Solaris OE that your applications will support. This article was written to Solaris 8 OE update 3 released January, 2001.

To prevent an attacker from modifying a system or creating backdoors before you have the opportunity to secure it, perform an initial Solaris OE install. Do not perform an upgrade to an existing Solaris OE system. Also, install the system from an original Sun Solaris OE CD, and do not attach the system to a "public" network until the security modifications have been made.

Partitions

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

The Solaris OE `/var` file system contains system log files, patch data, print, mail, and files for other services. The disk space required for these files will vary 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 will 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.

Additional partitions, such as `/usr` and `/opt`, may be required if the recommendations made in the Mount Options section below are followed.

Minimization

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

The November 2000 Sun BluePrints OnLine issue includes an article entitled *Solaris™ Operating Environment Minimization for Security - updated for version 8* which describes a methodology for the minimization and automation of Solaris OE installations.

Patches

Sun provides patches to the Solaris OE and unbundled software products when problems are corrected. Anyone can download the recommended, security, and Y2K patches for the Solaris OE. All other patches require a SunSpectrum^{SM} service contract. All systems should have the latest recommended, security, and Y2K patches installed. 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. These updates are only available to service contract customers.

SunSpectrum service contract customers have access to all patches, maintenance updates, and the `patchdiag` tool. `patchdiag` takes a list of current patches available from Sun and examines the local system to determine patches that have not yet been applied. It also checks for new versions of patches that have already been applied. The `patchdiag` tool should be run on systems at least once a week to determine if important patches need to be applied such as security patches.

Immediately after a Solaris OE system is installed, all recommended, security, and Y2K patches should be applied. These patches are available from the http://sunsolve.sun.com Web and FTP sites.

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 more. Be sure to examine all system `init` scripts and test all patches on non-production systems to discover any such configuration changes.

Console Security

There are several security mechanisms that Sun hardware systems provide. The OpenBoot™ PROM system on SPARC™ systems has two security modes, `command` and `full`. Failed log in attempts to the OpenBoot PROM system can be monitored. It is also possible to prevent users from using the keyboard sequence to interrupt the Solaris OE and drop to the OpenBoot PROM level.

OpenBoot PROM Security Modes

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

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

There are two security modes available. 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. `full` security mode requires operator interaction to boot the system. It will 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 `eeprom` command in the Solaris OE. Here is an example of setting the mode to `full`:

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

To set a new EEPROM password, use the following command:

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

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

These OpenBoot PROM changes can also be made while at the OpenBoot PROM level. 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 bad log in attempts have been made, use the following command:

```
# eeprom security-#badlogins
security-#badlogins=3
```

You may want to add this command to an initialization script to track password attempts. To reset the counter, use the following:

```
# eeprom security-#badlogins=0
security-#badlogins=0
```

Losing the OpenBoot PROM password requires that you 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 will no longer boot without the new password. If this happens, you must contact the SunService^{SM} organization for a new EEPROM.

Disabling Keyboard Abort

SPARC based systems can drop to the OpenBoot PROM level while the Solaris OE is running using the keyboard abort sequence. This can be disabled in Solaris 2.6 and newer OEs. This feature may be useful 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.

To disable the keyboard abort sequence change the following line from the `/etc/default/kbd` file:

```
#KEYBOARD_ABORT=enable
```

to:

```
KEYBOARD_ABORT=disable
```

Should the system hang or otherwise become unusable, it will have to be powered off to be reset. It will no longer be possible to create a crash dump from the OpenBoot PROM level on a running system for analysis.

File system

The Solaris OE file system can be configured to provide added protection. The default file permissions on some files are not adequate. There are also several mount options that increase security when used effectively. The Solaris™ Volume Management system needs some adjustment to prevent attackers from gaining superuser privileges.

Adjusting File Permissions

The Solaris OE ships with some file system permissions that should be adjusted for security reasons. Many files and directories have the group write bit set. In most instances, this permission is not necessary and should be switched off.

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

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

Please note that this tool is not supported by Sun. The fix-modes program must be compiled 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 the original state. fix-modes should be executed after all packages are installed and all patches are applied.

Sun continues to refine file permissions and group ownerships with each new Solaris OE release.

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. If the file is owned by root, an executable started by a normal user would operate with superuser privileges. This 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, this can be a useful method in solving some tricky operational problems.

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 keep a system up to date with the latest set of patches.

Attackers may also 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 allows the attacker to execute the shell to gain elevated privileges (most often superuser).

To find all the set-user-ID and set-group-ID files on a server, use the following find command:

```
# 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.

Sun has also released the Solaris Fingerprint Database. This tool enables an administrator to verify, through a crytographic 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
```

Mount Options

The Solaris OE file system partitions can be mounted with various options that enhance security. As shown in the previous section, set-user-ID files can be used by attackers to create ways to gain higher privileges. These backdoors may be hidden anywhere on the file system. While a file may have a set-user-ID bit, it will not be 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 also possible to mount a file system as in read-only mode to prevent file modification. This will also prevent an attacker from storing backdoor files or overwriting and replacing files on the 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 also required to change a nosuid file system to suid. Watch for unscheduled system reboots.

The system partitions support some of these mount options. The /usr partition can be mounted read-only. It should not be mounted nosuid since 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 modern releases of the Solaris OE. This 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
```

While the above 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.

Volume Management

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

vold 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. vold calls rmmount which determines what type of file system, if any, is on the media. If a file system is present and it is supported, rmmount mounts the file system.

If the system does not require the automatic mounting of CD-ROMs and floppy disks, Volume Management should be disabled. For example, a server does not need it, but a workstation may. Disabling this service can be accomplished by removing the Volume Management packages (SUNWvolr, SUNWvolu, and SUNWvolg).

If Volume Management is necessary, the mount options for some file systems should be modified for security. As discussed above, file systems with the suid option can be problematic. In Solaris OE versions prior release 8 the default Volume Management 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 floppy 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
```

In Solaris 8 OE these entries have been made by default. With these options, the set-user-ID bit on executables is ignored on file systems that are mounted by the Volume Management system.

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 also need to be configured to restrict user access.

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. Use the `passmgmt` command to delete accounts in `/etc/passwd` and `/etc/shadow`. Here is an example:

```
# passmgmt -d smtp
```

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

The remaining system accounts (except the root account) should also be modified for added security. System accounts listed in `/etc/passwd` have no shell listed. Those accounts also have a `NP` string (meaning “no password”) listed in the `/etc/shadow` file. By default, this is sufficient. However, some additional steps can be taken to add more security. Use the `-l` option of the `passwd` command to lock accounts. To lock the `uucp` account use the following command:

```
# passwd -l uucp
```

Also, 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 “JASS” Security Toolkit includes a shell replacement called `noshell`. When the `noshell` executable is executed (as a log in shell in `/etc/passwd`) a log entry is generated and the shell exits. This allows administrators to track unauthorized use of system accounts.

at, cron, and batch Security

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 /usr/lib/cron 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. If the file does not exist or the account is not listed in this file, the *deny* file is checked. If the account is explicitly listed in the *deny* file then access is refused. Otherwise, access is permitted. If neither the *deny* nor the *allow* files exist, then only the root account can use the at or cron system. The Solaris OE includes cron.deny and at.deny files containing some system accounts.

With the release of Solaris 8 OE, access to the cron and at commands can be controlled through the Role Based Access Control (RBAC) authorization, solaris.jobs.user. Another 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 also 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.

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.

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 means that files created by system daemons are created with permission bits that are 666 (readable and writable by all users). This can be a problem because it gives normal users permission to overwrite the contents of system files.

In Solaris 8 OE the default system `umask` has changed to 022 from the 000 in previous Solaris OE releases. The default value of 022 is defined by the `CMASK` variable in the `/etc/default/init` file. To define a different `umask` the `CMASK` definition in `/etc/default/init` must be changed.

Use the following script 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 Services

System services are started by the `init` system. Some services are not necessary to system operation and should be disabled. There are also services that 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 are listed here:

`/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 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 will discover a way to exploit the system using an enabled service.

The revision 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.

Kernel adjustments

There are several kernel adjustments that can be made to increase Solaris OE security. The /etc/system file contains kernel specific parameter adjustments. Be careful when making changes to this file. Mistakes in this file may prevent the system from booting correctly.

NFS Server

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 only process requests from these privileged ports. If the system will act as an NFS server, add the following line to the /etc/system file to any Solaris 2.5.1 OE or later

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

This change may prevent some NFS clients from operating correctly. There have been reported problems with older versions of Linux and SCO UNIX.

Executable 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. In Solaris 2.6 OE and later, some of these exploits can be avoided by making the system stack non-executable. 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 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 since the stack is not executable. Non-executable 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 which non-executable stacks cannot protect against. Always install the latest security patches. The non-executable stack feature only works on the following SPARC architectures: sun4d, sun4m, and sun4u hardware.

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

Core Files

Core files contain the memory image of an executing process that has been 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 can 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 have 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 process `core` file. This information may be used to attack the system. Add the following line to the `/etc/system` file to prevent the creation of `core` files:

```
set sys:coredumpsize = 0
```

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

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

Solaris 7 OE, 8/99 and later Solaris OE 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 also allows `set-user-ID` programs to create `core` files for debugging purposes. The `set-user-ID` feature must be used with care and should be enabled only on development and testing systems. This feature can also 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 7 include it.

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 the root causes of problems, and watching attackers. There are typically two types of log files in the Solaris OE: system log files typically managed by the `syslog` daemon, and application logs created by the application.

Log Files Managed by `syslog`

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. Uncomment the following line in /etc/syslog.conf to enable logging these messages:

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

Save the file and use the following command to force syslogd to re-read its configuration file:

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

or for Solaris 7 and 8 OE:

```
# pkill -HUP syslogd
```

All of these files should be examined regularly for errors, warnings, and signs of an attack. This task can be automated by using log analysis tools or a simple grep command.

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, but it should be created. If this file exists, the /usr/bin/login program records failed log in attempts.

All of these logs should also be monitored for problems.

Miscellaneous Configuration

The following configuration items apply to both local and remote security.

The /etc/issue File

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

Here is the legal warning found in the “JASS” 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.
```

The message of the day file (/etc/motd) can also be used to display warnings.

PAM

The Pluggable Authentication Module (PAM) architecture provides authentication, account management, session management, and password management mechanisms to applications in modular form. 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 may 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 the authentication programs.

To disable a specific log in 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 an SSH protocol system such as `ssh` or OpenSSH. Comment out the following lines in `/etc/pam.conf`:

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

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

Be careful when editing the `/etc/pam.conf` file. Errors will prevent all PAM services from operating and users will not be 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 will prevent PAM from operating and prevent users from logging in into the system.

The `login` Command

The `login` command is part of the authentication process to access a local Solaris OE account. It is used on the console and by 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 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`. The following entry in `/etc/default/login` permits only root log ins through the `ttya` serial device:

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

If direct root log ins are to be disallowed entirely, the following CONSOLE entry should be made in /etc/default/login:

```
CONSOLE=-
```

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

Network Service Security

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

In order to facilitate rapid system deployment, the Solaris OE is designed to provide unrestricted access to most installed network services by default. This allows customers to quickly integrate Solaris OE systems into the computing environment 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 discusses 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 - Updated for Solaris 8 Operating Environment* 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 must use protocols or services that are not secure. For these commonly used insecure services (such as RPC, NFS, and Trivial FTP), suggestions are given for how to improve security.

Services offered by a system should be protected by as many layers of security as possible. This protection should start at the network level. The December 2000 issue of Sun BluePrints OnLine included an article entitled *Solaris™ Operating Environment Network Settings for Security - Updated for Solaris 8 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 Service Issues

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 (i.e., encryption, strong authentication, etc) should be used whenever possible.

Available Tools

While the Solaris OE does not include mechanisms to provide protection for network services, several tools are available that are useful in securing services and systems. Well regarded open source and commercial tools allow Solaris OE administrators to protect systems throughout the enterprise. These tools address security concerns by providing the following protection: access control, logging, strong authentication, and privacy through encryption.

The SunScreen™ and SunScreen Lite software are two products from Sun Microsystems that provide network protection. Both are firewall products that can 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 can still provide adequate protection for network services. Use the SunScreen software for systems where more than two network interfaces are required.

A freeware firewall alternative is IP Filter (http://coombs.anu.edu.au/ipfilter). Versions are available for Solaris OE versions 2.3 through 8.

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.

TCP Wrappers, an open source tool developed by Wietse Venema, provides TCP level access control, logging, and DNS hostname verification. It is 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 also provides better logging and detects DNS hostname discrepancies which may indicate an attacker in progress. TCP Wrappers are fairly straight forward to deploy on Solaris OE systems.

Sun Microsystems also has a more sophisticated product which can be used to provide 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 may not be desired.

OpenSSH (an open source toolkit) and SSH (a commercial product) are both a suite of tools to replace unsafe UNIX® network commands such as `telnet`, `ftp`, `rlogin`, `rsh`, and `rcp` and securely tunnel X window network communications. Both provide strong authentication and privacy through encryption. When built with the TCP Wrappers library it also benefits from TCP Wrapper access control. Like TCP Wrappers, OpenSSH/SSH is straight forward to deploy on many systems. It is a very valuable tool simply because of the number of unsafe commands it replaces. Once deployed, the replaced network services should be disabled in favor of OpenSSH/SSH.

Telnet

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. SSH tools can also serve as an effective replacement.

If you must use a telnet daemon which does not support encryption, then One Time Passwords, host-based firewalls, or TCP Wrappers should be used to secure the connections. One Time Passwords (OTP) protect 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 used for authentication. Host-based firewalls and TCP Wrappers can be used to limit the hosts that may connect to a system. By restricting access to services based on IP addresses, a system can limit its exposure to network attacks. None of these alternatives will protect a session against being ‘hijacked’ by a malicious user. A session is hijacked when a malicious user takes over a session that was begun by an authorized user. The malicious user, in effect, takes over the session from the authorized user. Session hijacking can only be prevented through the proper use of encryption.

Remote Access Services (rsh, rlogin, and rcp)

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 (i.e., rsh, rlogin, and rcp) break both of these requirements. This is 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 the servers.

The default authentication mechanism of the r* daemons uses the hostname 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 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.

Remote Execution Service (rexec)

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 the in.rexecd entries in /etc/inetd.conf should be disabled.

FTP

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 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 which provide strong encryption and authentication. Sun Enterprise Authentication Mechanism provides a secure version of FTP and SSH provides equivalent functionality.

If FTP is required, there are two features implemented by the in.ftpd daemon which can provide additional security. The first is the /etc/ftpusers file, which is used to restrict access to the system through FTP. A default /etc/ftpusers file is included with Solaris 8 OE. 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 (i.e., 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 but root FTP access is not. This provides a backdoor for intruders which may be used to modify the systems 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.

Trivial FTP

The trivial FTP service (in.tftpd) exists to provide disk-less 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. Disk-less workstations, X-terminals, and some printers use this service to load files needed to boot. 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 will run as the user nobody and restrict 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, in.tftpd changes the root directory, using chroot(), to /tftpboot. This option should always be used when TFTP functionality is required.

inetd Managed Services

The inetd daemon controls a majority of the minor network services available on a system. It's configuration file, /etc/inetd.conf, defines what services are managed by the inetd daemon. An ideal secured server should neither have an /etc/inetd.conf nor run inetd, as the daemons started in the /etc/inetd.conf are frequently not needed. To disable a service, edit the /etc/inetd.conf file and place a comment character ("#") in front of the line containing the service definition. Once this is completed, send a HUP signal to the inetd process. This will cause it to reread its configuration file.

Of the daemons started from the /etc/inetd.conf, the remote access services FTP, TFTP, and TELNET services have already been discussed. The RPC and print services are discussed later in this article. The remaining /etc/inetd.conf entries include:

- 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 which can be used to launch attacks against the system.
- time – prints out the current time and date. Since Solaris 2.6 OE xntp functionality has been included with the Solaris OE distribution for time synchronization. The xntp daemon offers additional security and functionality improvements over rdate and time. Whenever possible xntp should be used instead of this service.
- 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 has not been affected.

For restricted access servers, all connections to services managed by inetd should be logged. This 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. The IP addresses are logged through the syslog service.

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. rpcbind maintains a table of RPC services (listed by a program number) and the network address(es) on which they listen for clients to connect. A client will first communicate with the rpcbind service to determine the network address it must use in order to contact a particular RPC service. Current RPC services can be listed using the rpcinfo command which communicates with the rpcbind service.

RPC services are used in many UNIX services including: NFS, NIS, NIS+, and Kerberos. RPC services are also used 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 the rpcbind daemon itself, but rather 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.

From `/etc/inetd.conf`:

- `testsvc`
- `sadmind`
- `rquotad`
- `rpc.rusersd`
- `rpc.sprayd`
- `rpc.rwalld`
- `rpc.rstatd`
- `rpc.rexd`
- `kcms.server`
- `ufsd`
- `cachefsd`
- `kerbd`
- `xaudio`
- `rpc.cmsd`
- `rpc.ttdbserver`
-

From `/etc/rc2.d/S71rpc`:

- `rpcbind`
- `keyserv`
- `rpc.nisd`
- `nis_cachemgr`
- `rpc.nispasswdd`

From `/etc/rc3.d/S15nfs.server`:

- `rpc.bootparamd`

On almost all servers, the RPC services in `/etc/inetd.conf` can be removed. Many applications that use RPC services add additional 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 specifically required.

The RPC daemons started in `/etc/rc2.d` and `/etc/rc3.d` are for `rpcbind`, `keyserv`, various naming services (i.e., NIS and NIS+), and are also 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 include:

- `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 (e.g., NFS, `sadmind`, NIS+) while others do not. If RPC must be used, then only those services and daemons which provide support for `AUTH_DES` should be used. This combination of RPC and `AUTH_DES` authentication is called Secure RPC. See "Bibliography" on page 36 for additional references to Secure RPC.

NFS Server

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 the following startup scripts should be disabled 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 mandate the use of the NFS server. There are several different levels of security available in the NFS server itself. In addition, careful configuration can also 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 then can be discussed here.

Automount

The automount service manages automated NFS mounts. NFS clients may 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 has been, 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 also 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. This can be done by using NFS mount options and Secure RPC.

Frequently, NFS servers allow any system to mount the filesystems they export. This incorrect and all to common practice allows attackers to mount filesystems which may contain sensitive information. If the attacker would like to modify the contents of a particular file they need only change their user ID or UID to that of the interesting file and modify its contents. This attack, and many other NFS-based attacks, can be avoided through the use of appropriate NFS exports and Secure RPC.

Sendmail

sendmail is used on a Solaris OE system to forward and receive mail from other systems. Centralized mail servers should be used to receive mail and not 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 have been developed with smaller and more robust code. 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 the following recommendations should be followed to secure it as much as possible.

Outgoing Sendmail

The sendmail daemon is not needed for email delivery to other systems. All messages that can be immediately delivered, are. Messages that cannot be immediately delivered are queued for future delivery. The sendmail daemon, if running, retries these messages again. It is recommended, for Solaris OE versions 7 and earlier, that a cron job be used to start sendmail every hour to process these undelivered messages. The following cron entry starts sendmail every hour to flush the mail queue:

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

Solaris 8 OE provides a new, undocumented way to have sendmail handle queued mail without using cron. A new default configuration file can be named /etc/default/sendmail. In this file create the following line:

```
MODE=””
```

By defining the MODE to be a null string sendmail will only process the outgoing mail queue and 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 to Solaris 8 OE.

Disable sendmail Daemon

If no sendmail functionality is required it can be disabled, in Solaris 2.6 OE and earlier Solaris OE releases, by renaming the /etc/rc2.d/S88sendmail script. Once this script is commented out sendmail will not be started during system startup. On Solaris OE versions 7 and 8 systems, it is also possible to remove all components of sendmail by removing the SUNWsndmr and SUNWsndmu packages with pkgrm.

sendmail.cf Recommendations

There is 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. Please refer to recommendations made at Sendmail Consortium in the Sendmail O'Reilly books and through the SunSolve OnLine^{SM} service.

Name Service Caching (nscd)

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. 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 has been expanded to address additional name service databases.

It is recommended that the configuration of nscd, through the /etc/nscd.conf file, be modified to cache as little data as possible. Disabling nscd entirely, by commenting out the /etc/rc2.d/S76nscd startup script, is not recommended because there may be unexpected results. Problems have been encountered when using name services such as NIS, NIS+, and even Netscape when nscd is not running.

Tuning nscd to an appropriate minimal level can address potential security issues while maintaining a robust system configuration. In particular, the configuration should be modified so that passwd, group, and Solaris 8 OE RBAC information is not cached. Depending on what parts of nscd are disabled, there may be a performance impact on systems that have many users. The nscd -g option can be used to view the current nscd configuration on a server and is a helpful resource when tuning nscd.

A sample configuration file for an `/etc/nscd.conf` supporting Solaris OE versions 2.6 and 7 with `passwd` and group caching disabled is shown below:

```
enable-cache            passwd          no
enable-cache            group           no
positive-time-to-live   hosts           3600
negative-time-to-live   hosts           5
suggested-size          hosts           211
keep-hot-count          hosts           20
old-data-ok             hosts           no
check-files             hosts           yes
```

To disable caching of the RBAC attribute databases in the Solaris 8 OE, add the following lines to the `/etc/nscd.conf` file:

```
enable-cache exec_attr no
enable-cache prof_attr no
enable-cache user_attr no
```

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 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 which 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 to be removed, the Solstice™ Print Client should also be removed. The Solstice Print Client is contained 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.

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 or not the system will be configured as a router and have ip_forwarding enabled between the different interfaces. For more information on the ip_forwarding function, refer to the Sun BluePrints article *Solaris™ Operating Environment Network Settings for Security - Updated for Solaris 8 Operating Environment*. Solaris 8 OE adds an ability to set ip_forwarding on a per-interface basis. This is also discussed in the aforementioned Sun BluePrints article.

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 have been discussed in a previous Sun BluePrints article titled *Solaris™ Operating Environment Network Settings for Security - Updated for Solaris 8 Operating Environment.*

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. For Solaris 7 OE and earlier, the following lines in /etc/init.d/inetsvc should be commented out:

```
mcastif=`/sbin/dhcpinfo 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 [ "$_INIT_NET_STRATEGY" = "dhcp" ]; then
        mcastif=‘/sbin/dhcpinfo Yiaddr‘ ||
mcastif=$_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
) &
```

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

Reducing `inetsvc`

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

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

By commenting out all of these entries, the number of active lines in the `inetsvc` file decreases from 152 to 3 lines. The following is what the resulting script looks like:

```
#!/bin/sh

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

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 a method to change these messages in an attempt to hide OS information.

To change banner messages for incoming telnet and FTP connections create the `/etc/default/telnetd` and `/etc/default/ftpd` files. Add a line similar to the following:

```
BANNER=”Generic OS”
```

Insert the appropriate message for your environment.

It is also possible to change the banner message that the `sendmail` process presents for incoming mail delivery connections. Search the `/etc/mail/sendmail.cf` file for the following line:

```
O SmtpGreetingMessage=$j Sendmail $v/$Z; $b
```

Change it to:

```
O SmtpGreetingMessage=Mail Server Ready
```

These techniques 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.

Summary

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 used to gain unauthorized or privileged access to an improperly configured system. The implementation of the changes recommended in this article require planning, testing, and documentation in order to be successful in securing a computing environment.

This Sun BluePrints article contains information updated for the Solaris 8 OE. The information contained is current as of Solaris 8 OE 1/01 (update 3).

Bibliography

AUSCERT, *UNIX Security Checklist*,
`ftp://ftp.auscert.org.au/pub/auscert/papers/unix_security_checklist`

Casper Dik, `fix-modes` tool,
`ftp://ftp.wins.uva.nl/pub/solaris/fix-modes.tar.gz`

Hal Pomeranz, *Solaris Security Step by Step*,
`http://www.sans.org/`

Jason Rhoads, *Solaris Security Guide*,
`http://www.sabernet.net/papers/Solaris.html`

Peter Baer Galvin, *The Solaris Security FAQ*,
`http://www.sunworld.com/common/security-faq.html`

Brad Powell, et al., *Titan security tool*,
`http://www.fish.com/titan/`

Alex Noordergraaf, *Solaris Operating Environment Minimization for Security: Updated for Solaris 8*, Sun BluePrints OnLine, November 2000,
`http://www.sun.com/blueprints/1100/minimize-updt1.pdf`

Alex Noordergraaf and Glenn Brunette, *JumpStart Architecture and Security Scripts for the Solaris Operating Environment - Updated for version 0.2 Part 1*, Sun BluePrints OnLine, November 2000,
`http://www.sun.com/blueprints/1100/jssec-updt1.pdf`

Alex Noordergraaf and Glenn Brunette, *JumpStart Architecture and Security Scripts for the Solaris Operating Environment - Updated for version 0.2 Part 2*, Sun BluePrints OnLine, November 2000,
`http://www.sun.com/blueprints/1100/jssec2-updt1.pdf`

Alex Noordergraaf and Glenn Brunette, *JumpStart Architecture and Security Scripts for the Solaris Operating Environment - Updated for version 0.2 Part 3*, Sun BluePrints OnLine, November 2000,
`http://www.sun.com/blueprints/1100/jssec3-updt1.pdf`

OpenSSH tool,
`http://www.openssh.com/`

Sendmail Consortium, `sendmail` configuration information,
`http://www.sendmail.org/`

Lance Spitzner, *Armoring Solaris*,
`http://www.enteract.com/~lspitz/armoring.html`

SSH Communications Security, Secure Shell (SSH) tool,
`http://www.ssh.com/`

SunScreen and SunScreen Lite,
`http://www.sun.com/security/`

Sun Enterprise Authentication Mechanism information,
`http://www.sun.com/software/solaris/ds/ds-seam`

Keith Watson, Alex Noordergraaf, *Solaris Operating Environment Network Settings for Security - Updated for Solaris 8*, Sun BluePrints OnLine, December 2000,
`http://www.sun.com/blueprints/1200/network-updt1.pdf`

Wietse Venema, TCP Wrappers tool,
`ftp://ftp.porcupine.org/pub/security/index.html`

Author's Bio: Alex Noordergraaf

Alex Noordergraaf has over nine years experience in the area of Computer and Network Security. As a Senior Staff Engineer in the Enterprise Engineering group of Sun Microsystems he is developing, documenting, and publishing security best practices through the Sun BluePrints OnLine program. He is a co-author of the freeware "JASS" Security Toolkit, which is available from http://www.sun.com/blueprints/tools.

Prior to his role in Enterprise Engineering 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. In addition to providing billable services to customers he developed and delivered an Enterprise security assessment methodology and training curriculum to be used worldwide by SunPS. His customers have included major telecommunication firms, financial institutions, ISPs, and ASPs.

Author's Bio: Keith Watson

Keith Watson has spent nearly four years at Sun working in the area of computer and network security. He is currently the product manager for core Solaris security. Previously, Keith was a member of the Global Enterprise Security Service (GESS) team in Sun Professional Services. He is also a co-developer of an enterprise network security auditing tool named the Sun Enterprise Network Security Service (SENSS). Prior to joining Sun, Keith was part of the Computer Operations, Audit, and Security Technologies (COAST) laboratory (now part of the CERIAS research center) at Purdue University.