Securing Sun Linux Systems: Part I, Local Access and File Systems

By Glenn Brunette, Michael Hullhorst, and Gé Weijers

Sun BluePrints™ OnLine—July 2003

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.: 817-3420-10
Revision 1.0, 6/26/03

Please
Recycle

Securing Sun Linux Systems: Part I, Local Access and File Systems

This article is the first part of a two-part series that provides recommendations for securing the Sun™ Linux 5.0 operating system. This part provides recommendations for securing local access and file systems. Part II provides recommendations for securing network access and services. The information in this article applies only to the Sun Linux 5.0 distribution, although some techniques or recommendations might apply to other Linux distributions.

It is important to address security for local access and file systems. Often, administrators are solely concerned with protecting a system from remote threats. We recommend that you have equal concern for local, authorized users who can exercise a weak configuration, either inadvertently or maliciously, and gain unauthorized privileges on a system. We highly recommend a layered approach to security where protection is implemented for both local and remote threats, resulting in a more robust security configuration.

Sun Linux 5.0 is based on the Red Hat 7.2 Linux distribution and is a flexible, general purpose operating system. To secure a Sun Linux system against unauthorized access and modification requires changes to its default configuration. Although these changes are in most cases relatively minor, we strongly recommend that you make these changes to improve the security posture of a system. The changes and recommendations in both articles address the majority of methods that intruders use to gain unauthorized or privileged access to Sun Linux systems. You should implement these changes immediately after system installation.

As with most security strategies, you must achieve a balance between system manageability and security. Some recommendations in this article might not apply to your environment and might negatively impact your ability to manage a system. You must know your system and security requirements before starting. Implementing the changes recommended in this article requires planning, testing, and documentation.

This article contains the following topics:

Installing and Patching

Since the initial release of the Sun Linux 5.0 operating system, Sun has released several updates and patches. These updates typically include security improvements as well as enhancements related to reliability, performance, and manageability. When building a Sun Linux system, be sure to use the latest software updates and patches to take advantage of these improvements.

We strongly recommend that you apply all security patches to a system immediately after installing it.

Install the System Securely

To prevent attackers from modifying a system or creating backdoors, *do not attach the system to a public network* until you have installed security patches and completed security modifications. Attackers do not need much time to exploit an unpatched out-of-the-box system.

The Sun Linux software distribution automates nearly every facet of the installation process. This benefit makes each installation repeatable and less prone to error. A side effect of this process is that you cannot select packages or clusters for installation or removal. However, after the installation is completed, you can manually add or remove packages.

Apply Patches

Immediately after installing a Sun Linux system, update it with all of the available security patches to help prevent the exploit of known attacks. You can download the software updates and security patches for Sun Linux even if you do not have a service contract.

To identify which version of Sun Linux your system has, enter the following command.

```
# cat /etc/sun-release
```

Security patches are available in two forms:

- Product Updates – Download these from `http://sunsolve.sun.com/patches/linux`. These updates are clusters of patches that address issues related to reliability, availability, security, and system management. These updates are typically downloaded and installed as a complete group.
- Individual Security Patches – Download these from `http://sunsolve.sun.com/patches/linux/security.html`. These patches address security issues in a product, tool, or function. You can download these patches in either the Red Hat package management (RPM) or Source RPM (SRPM) format. SRPMs are RPM files that contain the source code for a program instead of its compiled code (stored in the RPM). In addition, an MD5 signature is on the site for each patch, so that you can verify the integrity of downloaded files. We recommend this step to ensure that patches applied are only those provided by Sun.

Note – Apply only patches that are developed and released by Sun Microsystems, Inc. Although the distribution is based on the Red Hat Linux distribution, patches released by Red Hat should never be applied. Doing so might render the system unsupportable by Sun.

To verify that a patch was applied, use the `rpm` command. For example, to verify that the `xinetd-2.3.7-4.7x.i386.rpm` package, available from the security patch web site, is installed, use the following command.

```
# rpm -q -a | grep xinetd
xinetd-2.3.7-4.7x
```

If the `rpm` command does not return a match with the correct version, then download the patch from Sun, and install it on the system as soon as possible. If the command returns no value, then the related software package is not installed on the system and, therefore, the patch is unnecessary.

In addition to the Sun Linux security web site, Sun offers a security bulletin mailing list. This list is for administrators who want to receive security bulletins directly from the Sun Security Coordination Team. For more information on joining this list, email the Sun Security Coordination Team or submit a security alert to the following Web site:

`http://sunsolve.sun.com/pub-cgi/show.pl?target=security/sec`

Receiving and acting upon these notifications in a timely manner is essential to sustaining a strong security posture.

At this time, Sun does not provide an automated mechanism to ensure that a system is currently using the most recent patches, security or otherwise. This process must be done manually to ensure that all available and applicable patches are installed.

Note – You can use the Sun Cobalt Control Station to monitor and manage large deployments of Sun LX50 systems. For example, you can use it to apply software updates. Using this product might help simplify the patch management process for these sites. For more information, refer to `http://www.sun.com/hardware/serverappliances/controlstation`.

As with any changes made to a system's configuration, always review the impact of the resulting changes to ensure that the security posture of a system is not diminished. Ensure that previously disabled services remain disabled after patches are applied. In addition, if possible, apply patches to non-production systems first to identify the impact of the changes before implementing them on production systems.

Minimizing the Installation

It is important to reduce the Sun Linux installation down to the minimum number of packages necessary to support the applications 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.

Sun Linux uses RPM to install, upgrade, and delete packages. Each package maintains a description, file list, change log, checksum, and dependency information. Use this information to maintain and validate system integrity when adding, upgrading, or removing packages. To a limited degree, you can use the information to validate a package, before or after you install it on a system.

To remove an RPM package that is no longer needed, use the `-e` option with the `rpm` command, as in the following example:

```
# rpm -e minicom-2.00.0-3
# rpm -e glibc-2.2.5-42
error: removing these packages would break dependencies:
        glibc-common = 2.2.5-42 is needed by glibc-2.2.5-42
```

In the example, the first package `minicom-2.00.0-3` is successfully removed. The second package `glibc-common-2.2.5-42` is not removed, due to an unresolved package dependency.

Caution – When manipulating packages, take care to ensure that the RPM dependency tree is not inadvertently corrupted. We strongly recommend that you avoid using the options `--force`, `--replacepkgs`, `--replacefiles` and `--oldpackage`. Improper use of these options can cause the RPM dependency database to reflect an inaccurate state of a system.

Disabling System Services

System services are started by the `init` system. Disable services that are not necessary to system operation. For example, some services might allow a system to be compromised, due to incorrect configuration. System services under the Sun Linux OS are controlled using the `chkconfig` command, which you can use to list services available, then disable or enable them.

Note – The `chkconfig` command does not start or stop system service; it only enables or disables it from running at boot time. If you disable a system service with `chkconfig` and do not reboot the system, then you must stop the system service using the script in the `/etc/rc.d/init.d` directory.

To list existing services and their states, use the following command:

```
# /sbin/chkconfig --list
```

To disable a system service, use the following command:

```
# /sbin/chkconfig --level 0123456 <service> off
```

To enable a system service, use the following command:

```
# /sbin/chkconfig --level 0123456 <service> on
```

The previous example enables *`service`* for each of the system's seven run levels. Use only the number or numbers corresponding to the run levels at which the service should run. For example, to enable a service only for run level 5, then modify the `--level` option to include only the number 5.

For security purposes, only enable required services. The fewer services that are enabled, the less likely it is that attackers can discover ways to exploit systems.

The packages installed determine what services are enabled by default. Removing unnecessary packages disables some extraneous services. Examine the remaining services to determine their relevance to the system and the hosted applications.

Note – Be aware that installing patches and/or software packages might restore or add new entries for `init` to start. We recommend that you regularly review the services started by `init`. In particular, check for new services or changes in the status of existing services after patches or new software are installed on a system.

Verifying Integrity

After you install or upgrade a system, we strongly recommend that you verify the integrity of the Sun Linux image. You can perform this task using the commands described in the previous section, but to provide a higher degree of assurance, compare the packages on the system against a trusted source such as the Sun Linux CD-ROM distribution.

It is possible to verify whether the files installed by RPM were modified after the installation by comparing them with the original `.rpm` file. The following command compares the installed files with the original `xinetd` package.

```
# rpm --verify -p xinetd-2.3.7-4.7x.i386.rpm
```

You can use a simple shell script to validate and report on the integrity of all of the RPM packages installed on a system. This result is achieved by comparing the installed packages with their counterparts from the installation or update media.

The following shell script is an example of how to generate a usable report.

```
# !/bin/sh

INSTALLED_RPMS="‘rpm --query --all‘ | sort -u"

for pkg in ‘ls /mnt/cdrom/RedHat/RPMS/*.rpm | sort -u‘; do
   short_pkg="‘basename ${pkg} | sed ’s/.i386.rpm//g’‘"
    if [ ‘echo ${INSTALLED_RPMS} | grep -wc ${short_pkg}‘ != 0 ];
   then
      rpm --quiet --verify --package ${pkg}
      if [ $? = 0 ]; then
         result="SUCCESS"
      else
         result="FAILED"
      fi
      printf "Package Check: %-35s   RESULT: %s\n" \
         ${short_pkg} ${result}
   fi
done
```

Note – This verification method is most effective on newly installed or upgraded systems. For systems or packages that have been patched, this method only works if the packages signatures are tested against a patched, trusted copy of the package.

The following example illustrates how to verify packages against the package information stored in a system’s local RPM database. This check is similar to the `pkgchk`(1M) command in the Solaris OE.

```
# rpm -verify filesystem-2.1.6-2
# rpm -verify apache-1.3.23-11
S.5....T c /etc/rc.d/init.d/httpd
```

In the example, the integrity of the first package, `filesystem-2.1.6-2`, was successfully verified. The check failed for the second package, `apache-1.3.23-11`, when the `/etc/rc.d/init.d/httpd` was found to have been modified.

To verify all packages on a system, use the `-a` option in place of the package name.

```
# rpm -verify -a
```

This capability is not a substitute for functionality such as Tripwire. This information is used only by the RPM framework to ensure that packages are completely installed, upgraded, or removed, and that all package dependencies are properly met.

After you validate the integrity of a system, use products such as Tripwire to establish a baseline database for detecting file integrity violations. The Sun Linux distribution includes the Tripwire Open Source, Linux Edition, product originally developed by Tripwire, Inc. This tool provides data integrity assurance through the collection and management of file signatures and related data. If configured properly, this tool identifies when file system objects are changed. We recommend you consider products such as Tripwire as part of an organization's overall platform security strategy.

Note – For more information on the Tripwire Open Source, Linux Edition product, refer to the Web site `http://www.tripwire.org/`.

Other methods can provide a higher degree of assurance, but those methods are outside the scope of this article. At this time, Sun does not provide a Sun Linux equivalent to the Solaris Fingerprint Database software.

Securing the Console and Front Panel

The next task is to consider restricting access to a system's console. This task is useful if the server is located in a common area of a network operations center.

Note – These tasks do not prevent attackers with physical access from compromising systems. These methods provide incremental security, but caution must always be exercised when physical access to a system and related hardware is permitted.

This section contains the following topics:

Access and Modify BIOS Configuration

The Sun Linux 5.0 operating system is provided on the Sun LX50 system. The Sun LX50 system uses the American Megatrends, Inc. (AMI) Basic Input and Output System (BIOS). The BIOS provides security features that prevent unauthorized or accidental access to a system. When security measures are enabled, administrators and users can access the system only when they enter correct passwords. You can implement the following security measures:

- Enable an administrator password, which is used to access and configure BIOS security options
- Enable a user password, which can be granted full or limited access to BIOS
- Enable secure mode, which prevents keyboard input, front panel reset access, and power switch access.
- Enable a keyboard lockout timer, which after a time-out period, requires a password to reactivate keyboard input.
- Disable booting to alternative devices, such as diskettes and CD-ROMs

A system's BIOS performs power-on self-tests (POST), provides an interface to the hardware components on a system, and facilitates loading an operating system by locating and accessing a boot loader. In addition, the BIOS provides basic security features.

To access the BIOS configuration, press the F2 key while the initial boot screen is displayed on a console. To maneuver the BIOS menu system, follow the instructions located at the bottom of the screen.

Note – If a BIOS administrative password is defined for the system, this password must first be correctly entered before access to the BIOS configuration is granted.

If you change any of the BIOS configuration parameters, you must reboot the system for the changes to take effect.

Restrict Access to BIOS

You can set a user password, an administrator password, or both. The passwords are limited to seven alpha numeric, case-sensitive characters. By default, the passwords are not defined, and unrestricted access is granted to the BIOS for any user with physical access to a console.

Setting a user or an administrator password requires:

- Entering the password to enter BIOS setup.
- Entering the password to boot the server if "Password on Boot" is enabled.
- Entering the password to exit "Secure Mode."

Setting both passwords requires:

- Entering the password to enter BIOS setup.
  - If entering a user password, the user may not be able to change some of the BIOS options, depending upon privilege level granted.
  - If entering an administrator password, the administrator is able to enter BIOS setup and access all options.
- Entering either password to exit "Secure Mode."

Caution – With physical access to a system, BIOS passwords can be reset by changing a jumper on the motherboard.

▼ To Set an Administrator Password

1. **Enter the BIOS menu by pressing the** `F2` **key while the system's initial boot screen is displayed.**
2. **Select the** `Security` **menu tab to display the security configuration menu.**
3. **Select the** `Set Administrative Password` **option.**
4. **Enter the new administrative password.**
5. **Re-enter the new administrative password to confirm the new password.**
6. **Select the** `Exit` **menu tab, then select the** `Exit` and `Save` **option.**

   Once set, the `Administrative Password` parameter changes from `Disabled` to `Enabled`. Now the `Administrative Password` must be entered to access the BIOS configuration.

▼ To Set a User Password

1. **Enter the BIOS menu by pressing the** `F2` **key while the system's initial boot screen is displayed.**
2. **Select the** `Security` **menu tab to display the security configuration menu.**
3. **Select the** `Set User Password` **option.**
4. **Enter the new user password.**
5. **Re-enter the new user password to confirm the new password.**
6. **Select the privilege level granted to the user:**
   - `No Access` – Prevents a user from accessing the BIOS configuration. If a user is assigned to this level, the user password is used only to unlock the system when it is operating in "Secure Mode."
   - `Limited` – Allows a user to access the BIOS and to change a limited number of non-critical fields.
   - `View Only` – Allows a user to access the BIOS but in read-only mode. The user is not permitted to change any of the BIOS parameters.
   - `Full` – Allows a user to access the BIOS and change all parameters, except for the `Administrator Password`.
7. **Select the** `Exit` **menu tab, then select the** `Exit and Save` **option.**

Limit Front Panel, Keyboard, and Video Access

After setting the administrator password, limit access to the front panel, keyboard, and video. The following additional options appear on the BIOS `Security` menu:

- Secure Mode Timer – This timer is the period of inactivity in minutes before "Secure Mode" is activated and the system's keyboard and mouse are locked.
- Secure Mode Hot Key – This keyboard sequence places the system in "Secure Mode." By default, the sequence is Control-Alt-[L], which is performed by holding down the Control and Alt keys and simultaneously pressing the L key.
- Secure Mode Boot – This setting configures the BIOS to prevent the system from starting the boot process until a user or administrator password is entered. A password is required to boot from removable media such as a diskette or CD-ROM.

- Video Blanking – This setting disables the use of a video monitor when a system is in “Secure Mode.” When video blanking is off, the system displays information on the monitor even in “Secure Mode.” If a monitor is disabled in addition to the keyboard and mouse when in “Secure Mode,” video blanking should be enabled.
- Disable Power Button – This setting configures the BIOS to ignore the use of the front-panel power button. When enabled, this setting prevents a running system from being powered off using the front-panel power button.

▼ To Set Access Options

1. **Enter the BIOS menu by pressing the** `F2` **key while the system’s initial boot screen is displayed.**
2. **Select the** `Security menu` **tab to display the security configuration menu.**
3. **Select the appropriate option, and enable or disable it.**
4. **Select the** `Exit` **menu tab, then select the** `Exit and Save` **option.**

Restrict Alternate Boot Devices

You can use the Sun Linux BIOS configuration to specify the order in which devices are polled when locating an operating system. This boot device priority selects the order in which hard-drives, CD-ROM drives, and disk drives are accessed during boot processes. It is recommended that the system be configured to boot first from the local hard drive before other media. This approach can prevent a system from being compromised through a boot diskette or CD-ROM inserted during the boot process.

▼ To Set Boot Device Priority

1. **Enter the BIOS menu by pressing the** `F2` **key while the system’s initial boot screen is displayed.**
2. **Select the** `Server` **tab to display the server configuration menu.**
3. **Select the** `Boot Priority` **menu to change the default boot priority.**
4. **Select the hard drive as the first boot device.**
5. **Disable any boot devices that are not required.**
6. **Select the** `Exit` **menu tab, then select the** `Exit and Save` **option.**

Restrict Access to the LILO Boot Loader

Sun Linux uses the LILO boot loader to load the Linux kernel. LILO allows users to pass parameters to the kernel, several of which can be used to gain unrestricted access to a system (such as `single` for single-user mode). You can configure LILO to require a password before allowing access.

▼ To Configure LILO to Require a Password for Access

- **Add the following lines (see bold lines) to the `/etc/lilo.conf` file.**

```
image=/boot/vmlinuz-2.4.9-31enterprise
password=<password>
restricted
label=linux
initrd=/boot/initrd-2.4.9-31enterprise.img
append="console=ttyS1,9600 console=tty0"
read-only
root=/dev/sda3
```

In this example, the password and restriction options are added to the kernel 2.4.9-31enterprise. In practice, there exist multiple kernel definitions or image entries in the `/etc/lilo.conf`, often as a result of kernel upgrades. We recommend that you define these options for each of the kernels listed in your `/etc/lilo.conf` file.

Note – The version number `2.4.9-31enterprise` changes based on the version of the kernel running on the system.

Using the `restricted` directive in LILO allows booting of the default kernel without password verification, but requires a password if any additional arguments are added (such as `single` to boot into single-user mode) or if a kernel image other than the default is selected.

Access to the `/etc/lilo.conf` file should be restricted to only the root user, because the password contained in that file is in clear-text. Set this restriction by executing the following command.

```
# chmod go-rwx /etc/lilo.conf
```

After any modifications, you must run the command `/sbin/lilo` to propagate any changes to LILO. The following command is usually sufficient.

```
# /sbin/lilo
```

Disable Control-Alt-Delete Reboot Key Sequence

By default, a Sun Linux system reboots when the key combination Control-Alt-Delete is entered.

▼ To Disable the Control-Alt-Delete Key Sequence

1. **Comment out the following line in the** `/etc/inittab` **file:**

```
# Trap CTRL-ALT-DELETE
# ca::ctrlaltdel:/sbin/shutdown -t3 -r now
```

2. **Reload the** `inittab` **by either rebooting the system or by entering** `/sbin/telinit q`**.**

Require Single-User Mode Password

You can configure the system to prompt for a password when booted into single-user mode. Add the following line to the `/etc/inittab` file.

```
~~:S:wait:/sbin/sulogin
```

Caution – It is highly recommended that you add a password to the LILO configuration file instead of setting a password for single-user mode. Users can circumvent single-user password restrictions using the command `linux init=/bin/bash` instead of the `linux single` command at the LILO boot prompt.

Disable the Magic SysRq Key

If enabled, the SysRq key can be used for activities such as rebooting systems, inspecting memory, synchronizing disks, and killing processes. It is mainly useful to kernel developers, because it allows them to diagnose and recover a system state after problems. However, be aware that it can be used to gain unauthorized root access.

▼ To Disable the Magic SysReq Key

1. **Enter the following in the** `/etc/sysctl.conf` **file:**

```
kernel.sysrq = 0
```

2. **Reboot the system to implement this configuration change.**

Restrict Root Access to Devices

The Sun Linux operating system provides the ability to restrict from where a remote user can log into a system as root user. This restriction is an important capability to help promote accountability on a system. Typically, we recommend that administrators do not log into systems directly using a root account; instead they should log into systems using their unique account and assume root privileges as needed. Often this recommendation is combined with role-based access control capabilities such as `sudo` to further restrict what may be done with elevated privileges. By following this recommendation, actions can be better associated with specific individuals.

The `login` command is part of the authentication process to access a local Sun Linux account. Except for a root user, any user can log in to any valid device on a system, serial or virtual. A root user is not permitted to log in to any device unless the device is listed in the `/etc/securetty` file. If a root user attempts to log in to a device not listed, then the attempt fails and a failure notice is logged to the `syslog` facility.

If you need to configure the system to permit direct root login over the primary serial interface, then add the following line to the `/etc/securetty` file.

```
/dev/ttyS0
```

Note – Be sure to review the contents of the `/etc/securetty file`, removing any entries that are not required. Be careful not to remove root accounts, which would inadvertently lock root users out of the system.

Configuring the File System

The default file permissions on some files might not be adequate in all situations; therefore, configure the Sun Linux file system to provide additional protection. Also, several mount options are available to increase security, when used effectively. Sun Linux systems need some adjustment to prevent attackers from gaining superuser privileges.

This section contains the following topics:

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

The `set-user-ID` and `set-group-ID` bits (often referred to as SUID and SGID bits) on an executable file indicate to a system that the executable should operate with the privileges of the file's owner or group. For example, the effective user ID of the running program becomes that of the executable's owner, in the `set-user-ID` instance. Similarly, a `set-group-ID` file sets the running program's effective group ID to the executable's group. 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 challenges. These operational challenges can often be solved using forms of role-based access control, such as `sudo`. For more information on `sudo`, see "Limit Root Access" on page 25."

Many `set-user-ID` and `set-group-ID` commands have had flaws. Attackers have used these flaws to successfully exploit systems. When security problems are reported, Sun fixes them and provides a patch.

Attackers might use the `set-user-ID` or `set-group-ID` commands to create backdoors. One way they do this is by copying a system shell to a hidden location and adding the `set-user-ID` bit. This technique allows attackers to execute shells to gain elevated privileges (most often superuser).

▼ To Find All set-user-ID and set-group-ID Files

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

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

2. **Store the output to a file on another system.**
3. **Compare the output against the current file system from time to time, especially after applying patches, to find any unwanted additions.**

Review World-Writable File System Objects

A world-writable object is one that has the world-write permission bit set. World-writable objects are problematic because any user can modify the object. An attacker might use a world-writable file to perform a disk space-based denial of service attack on a system, or an attacker might modify the object, violating its integrity. We recommend that all world-writable objects be catalogued and tracked over a system's lifecycle. Objects that do not require this setting should have their permissions reset to a stronger value.

▼ To Find World-Writable File System Objects

- **To find all of the world-writable files and directories on a system, use the following command.**

```
# find / \(-type f -o -type d \)-perm -0002 -ls
```

Note – All world-writable directories should be configured with the "sticky" bit to prevent users from deleting files owned by other users. For more information on this capability, refer to the `chmod`(1) manual page.

Review Unowned File System Objects

Typically, file system objects stored on a system can be directly attributed to a user and group that exists either on the system or in a naming service (for example, LDAP) used by the system. It is possible for a user or group to be removed from a system, leaving file system objects in an unowned state. That is, the files and directories are now owned by a user or group that no longer exists on the system.

This situation can also occur when extracting archives (for example, `tar`) built on a different system. These programs can be configured to preserve the original object's ownership and permissions of the archived objects. These programs restore the settings without regard to whether the user or group assigned to the object actually exists on the new system.

▼ To Find Unowned File System Objects

1. **To find all file system objects that are not owned by a valid user on a system, use the command:**

```
# find / -nouser -ls
```

2. **To find all file system objects that are not owned by a valid group on a system, use the following command:**

```
# find / -nogroup -ls
```

Note – In general, all file system objects on a system should be assigned to a valid user and group. Be sure to assign a valid user or group to any files found using the previous commands.

Managing Accounts

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

This section contains the following topics:

Delete and Modify System Accounts

A default Sun Linux 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 `games`, `gopher`, and `news` and `uucp`. Some of these accounts exist to support software subsystems that are not used or are for backwards compatibility.

▼ To Delete Accounts

- **To delete accounts in `/etc/passwd` and `/etc/shadow` entries, use the `userdel` command, similar to the following example:**

```
# /usr/sbin/userdel games
```

This example removes the `/etc/passwd` and `/etc/shadow` entries for user `games`.

Except for the root account, modify any remaining system accounts for added security by locking them, setting account expiration, or setting their login shells to a restricted value.

▼ To Lock Accounts

- **To lock unused accounts that are not locked by default, use the** `passwd -1` **option. For example, if the** `uucp` **account is not used, then lock it using the following command.**

```
# passwd -l <username>
```

The password of the account is changed to a string that can never be a valid encrypted password.

▼ To Expire an Account

- **To schedule a user account to expire, use the following command:**

```
# /usr/sbin/usermod -e YYYY-MM-DD <username>
```

An expired account has similar characteristics to those of an account where the password is locked. The account can still be accessed by root via superuser, daemons can be started, and files and directories can be owned by the account. As with locked accounts, expired accounts cannot be used to login to a system or to access any of the services where authentication is required, such as Telnet, FTP, or IMAP.

To disable interactive access to a user account, use the following command.

```
# usermod -s /bin/false <username>
```

An account with a disabled shell (one that is not listed in `/etc/shells` and does not permit access to the system) only impacts applications that check for valid shells, such as Telnet, FTP, and Secure Shell. This method does not impact other applications such as IMAP and POP that by default do not check this setting.

A better alternative to using `/bin/false` as a disabled shell is to use a program that not only prevents access to the system but also logs the attempted access. The following shell script could be used to implement such functionality. This example is

based on the file /sbin/noshell program available from the Solaris™ Security Toolkit software. For more information on the Solaris Security Toolkit software, refer to http://www.sun.com/security/jass.

```
#!/bin/sh
#
trap "" 1 2 3 4 5 6 7 8 9 10 12 15 19
PATH=/bin:/usr/bin
export PATH
HNAME="‘uname -n‘"
UNAME="‘id | awk ’{ print $1 }’‘"
logger -i -p auth.crit "Unauthorized access attempt on ${HNAME} by
${UNAME}"
wait
exit
```

To change default account parameters used during new account creation (via useradd), change the following values in either the /etc/login.defs or the /etc/default/useradd file.

TABLE 1 describes the parameters in the /etc/login.defs file.

TABLE 1 /etc/login.defs File Parameters

Parameter	Description
PASS_MIN_DAYS	Minimum days allowed between password changes
PASS_MIN_LEN	Minimum acceptable password length
PASS_WARN_AGE	Days warning given before a password expires
PASS_MAX_DAYS	Maximum days a password may be used

TABLE 2 describes the parameters in the /etc/default/useradd file.

TABLE 2 /etc/default/useradd File Parameters

Parameter	Description
GROUP	Default group
HOME	Default user home location

TABLE 2 `/etc/default/useradd` File Parameters *(Continued)*

`EXPIRE`	Expiration date of an account in the format YYYY-MM-DD
`INACTIVE`	Maximum days after an unchanged and expired password that the account is locked out and can only be accessed by root
`SHELL`	Default shell

Restrict `at`, `cron`, and `batch` Command Access

The `at`, `cron`, and `batch` commands execute processes (events) at a specified future time. Access control files are stored in the `/etc` directory.

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

The allow files are checked first to determine if an account is explicitly allowed to use these facilities.

Attackers can use these commands to implement logic bombs (triggered by a system condition, logic that causes damage to data processing systems) or other programmed attacks that begin in the future. Unless administrators examine every `at`, `batch`, and `cron` event, tracking usage and abuse can be difficult. Therefore, we recommend that you restrict access to the `at`, `batch`, and `cron` commands to prevent attacks and abuse.

By default, Sun Linux includes scheduled `cron` events for the root account. Do not include the root account in the deny files, because any scheduled jobs will be prevented from running.

Add to the deny files any additional system or software-specific accounts that do not require `cron`, `batch`, or `at` access.

You might want to restrict user access to these commands as well. List individual user accounts in the deny files. To restrict all user account access, create an empty allow file, then add only the accounts that need access.

Limit Root Access

The sudo program implements a form of role based access control (RBAC). It is a commonly used tool for limiting access to privileged commands and accounts. Using sudo has the following benefits:

- sudo is configured by default to ask for a user's password
- Fewer users need to log in as privileged users and know the password
- Access to privileged commands is determined by a policy; users are given access only to the commands they need
- Access to privileged commands can be logged

Common uses for sudo are allowing system operators to make backups and create user accounts without giving operators full root access.

To create a new account, an operator enters:

```
# sudo useradd newuser
```

It is very important to ensure that mail to root is checked or forwarded. Unauthorized use of sudo generates an email to the root user and creates an entry in the system log /var/log/messages file.

```
May  4 18:25:14 scl1 sudo(pam_unix)[28769]: authentication
failure; logname=attackerid uid=0 euid=0 tty=pts/2 ruser= rhost=
user=attackerid
```

To review valid uses of sudo, refer to /var/log/secure. In the following example, the sudo command was used by the user joe to access the /bin/bash command as root.

```
May  4 18:25:28 scl1 sudo: sudoerid: TTY=pts/2 ; PWD=/home/
sudoerid/joe ; USER=root ; COMMAND=/bin/bash
```

The sudo command consults the /etc/sudoers configuration file to determine if a command is allowed. This configuration file contains access control lists (ACLs) as in the following example.

```
<user>  <hostname> = '(' <account> ')' <command> [',' <command>]
```

A rule of this form allows `user` logged in to `hostname` to run `command` as user `account`. By including the hostname in the rule, it is possible to share a single `/etc/sudoers` file between multiple machines. For more information about the format of these ACLs, refer to the `sudoers` manual page.

Use the `visudo` command to edit the `/etc/sudoers` file. This command checks the syntax of edited contents before overwriting `/etc/sudoers`. If the `/etc/sudoers` syntax is incorrect, `sudo` does not work.

To keep the description concise, you can define aliases for sets of users, hostnames, and commands. You can use the alias `ALL` in any field. In the following example, we allow a group of operators to make backups and restore them.

```
# provide access to dump and restore for backup operators
User_Alias      OPERATORS = tom, dick, harriet
Cmnd_Alias      BACKUPCMD = /sbin/dump, /sbin/restore

OPERATORS       ALL = (root) BACKUPCMD
```

Be aware that with a little creativity, any user who can make backups and restore them can gain root access, so this example is not a completely secure setup. In general, we advise that you create scripts or programs that wrap commands executed with privilege to limit the options available to its users. In addition, these scripts can sanitize the execution environment by checking or resetting variables such as a user's path to help ensure sane settings. This way, the use of privileged commands can be better controlled.

You can configure `sudo` to allow only certain parameters to be passed to privileged commands. Verify that operators cannot pass any arguments to privileged commands that have unintended consequences. The following rule allows operators to change passwords for all users except for root.

```
# User alias specification
User_Alias      OPERATORS = tom, dick, harriet

# Cmnd alias specification
Cmnd_Alias     PASSWDCMD = /usr/bin/passwd, !/usr/bin/passwd root
# Defaults specification
OPERATORS       ALL = (root) PASSWDCMD
```

The previous example prevents execution of the `sudo passwd root` command. However, it does allow execution of the `sudo passwd --stdin root` command. In this case, you can fix the problem by changing the definition of `PASSWDCMD` slightly, as in the next example.

The following example allows two classes of users (web masters and operators) to manage a system. Web masters can stop and start a web server. Operators can change forgotten user passwords (except for root passwords), start system backups, and perform web master duties.

```
# User alias specification
User_Alias      OPERATORS = tom, dick, harriet
User_Alias      WEBMASTERS = josephine, OPERATORS

# Cmnd alias specification
Cmnd_Alias    PASSWDCMD = /usr/bin/passwd [!-]*, !/usr/bin/passwd
root
Cmnd_Alias      BACKUPCMD = /usr/sbin/backup_script
Cmnd_Alias      HTTPDCTL = /etc/rc.d/init.d/httpd

# Defaults specification

Defaults        authenticate, timestamp_timeout = 60

# User privilege specification
root    ALL=(ALL) NOPASSWD: ALL

OPERATORS       ALL = (root) PASSWDCMD
OPERATORS       ALL = (root) BACKUPCMD
WEBMASTERS      ALL = (root) HTTPDCTL
```

Use the Pluggable Authentication Module (PAM)

The Pluggable Authentication Module (PAM) architecture allows administrators to control and replace the methods used for authentication. In concept, while this is similar to the PAM functionality that is available in the Solaris OE, note that there are differences.

For more information on the Solaris OE implementation of PAM, refer to the Sun BluePrints OnLine two-part article titled “Extending Authentication in the Solaris 9 OE Using Pluggable Authentication Modules (PAM).”

- Part I: `http://www.sun.com/blueprints/0902/816-7669-10.pdf`
- Part II: `http://www.sun.com/blueprints/1002/816-7670-10.pdf`

Although PAM makes authentication operations more robust, it requires that system authentication information be maintained in more places than just `/etc/passwd` and `/etc/shadow`. PAM can be expanded to support Kerberos, LDAP, OPIE, S/Key, RSA SecureID, RADIUS, TACACS+, and others. Understanding how the PAM system functions is essential for maintaining the security of a system.

All primary applications such as `login`, `su`, `ssh`, and `sudo` are PAM aware. Each application that uses PAM does so through a common API that references the general configuration file `/etc/pam.conf` or the application-specific file in the directory `/etc/pam.d/`, if it exists.

Note – If the directory `/etc/pam.d/` exists, the file `/etc/pam.conf` is ignored. The directory `/etc/pam.d/` is part of the standard Sun Linux installation.

Caution – Before editing PAM configuration files, make a backup of the files. Errors made in configuration files can prevent PAM or the service you modified from operating correctly, and might prevent privileged users from logging in. Also, you could inadvertently disable authentication and expose the system. Do not change the default ownership or file permissions of the `/etc/pam.d/` directory or its contents.

Within the PAM configuration file, define methods for authentication. Enter one line per method using the following format.

```
[service] [type] [control] <module-path> <module-arguments>
```

TABLE 3 lists the values for entering authentication methods.

TABLE 3 PAM Configuration Values

Field/Value	Description
service	This field exists only if the `/etc/pam.conf` file is used and is the name of the service defined, such as `login` or `su`. If the file is in `/etc/pam.d/` and is specific to the application, then it is not used.
type	This field is the management group and can be one of the following: • `account` - Verify that the account is valid. For example: has the account expired? • `authentication` - Verify the user's identity by using a password, smart card, or biometric device. • `password` - Maintain the authentication method. For example: if the password has expired, prompt for a new one. • `session` - Provide methods that are performed before the session is started and after the session is ended, such as setting up and reclaiming resources the user requires.
control	This field specifies what to do when a module fails its task: • `requisite` - Immediate termination of the authentication process; do not call additional modules. • `required` - If this module fails, then allow other modules to be called, but ultimately fail the authentication attempt. Each required module must succeed for the authentication to succeed. • `sufficient` - Success of the module is enough to succeed in authentication. • `optional` - Success or failure of this module is important if it is the only module in the stack associated with this `service` and `type`. More complex control syntax can be used; refer to the `pam`(8) manual page.
module-path	This path is the full file name of the authentication module called with the default location `/lib/security/`. A special module used by many configuration files is `pam_stack`. This module lets you call a service from inside the stack. This feature allows multiple services to include a system-wide configuration, so that it only needs to be maintained in one place. The most common use for this feature is the configuration file `/etc/pam.d/system-auth`.
module-arguments	Valid arguments are space-separated lists of tokens that modify behavior of the module called. They are module specific.

For additional information on PAM, refer to the Linux PAM site at `http://www.kernel.org/pub/linux/libs/pam/`, the `pam`(8) manual pages, and the PAM documentation in `/usr/share/doc/pam`.

After editing the configuration file, test the changes and make any necessary adjustments.

Note – If you lock yourself out of a system, reboot into single-user mode, then restore the configuration file from the backup.

Disabling Null Passwords

A null password allows users to log onto a system without having to first supply a valid password string. When users have null passwords, they can press the Enter key when prompted for a password and gain access to systems without a password. Obviously, this poses a significant security risk to the system and to the accountability of actions performed by users.

▼ To Disable Null Passwords

- **Make a backup of the** `/etc/pam.d/system-auth` **file, then modify the original by removing** `nullok` **from the line, resulting in the following:**

```
auth sufficient /lib/security/pam_unix.so likeauth
```

Checking Passwords Against a Dictionary

Sun Linux can be configured to verify that passwords cannot be guessed easily. On Sun Linux, this check is performed by the module `/lib/security/pam_cracklib.so`. It checks to ensure that passwords are a minimum length and verifies that a password does not occur in a dictionary.

The dictionary used by this module is located in `/usr/lib` and is in `cracklib` format. By default, each of the dictionary files is prefixed with the file name `cracklib_dict`. For more information on `cracklib` including how to add new words to the dictionary, refer to `http://www.crypticide.org/users/alecm/`.

This module has a number of parameters, the most useful of which are as follows.

TABLE 4 `pam_cracklib` Parameters

Parameter	Description
`minlen`	Specifies the minimum password length allowed for an account
`difok`	Specifies the minimum number of characters that have to differ from the previous password

▼ To Check Passwords

- **Add the following parameters to an entry in** `/etc/pam.d/system-auth`**, resulting in the following single line:**

```
password required /lib/security/pam_cracklib.so retry=3 type=
minlen=8 difok=3
```

Preventing Reuse of Old Passwords

The PAM module `pam_unix.so` can be configured to maintain a list of old passwords for every user, to prohibit the reuse of old passwords. The list `/etc/security/opasswd` is not maintained as plain text, but should be protected the same as shadow password files. This capability is often referred to as password history.

▼ To Retain a List of Passwords

- **To remember the last 15 passwords, add the following line in** `/etc/pam.d/system-auth` **file:**

```
password    sufficient    /lib/security/pam_unix.so use_authtok
md5 shadow remember=15
```

Preventing Password Guessing

The module `pam_tally` keeps track of unsuccessful login attempts, and disables user accounts (not user IDs) when a preset limit is reached. This capability is often referred to as account lockout.

▼ To Prevent Password Guessing

- **Add two entries in the** `/etc/pam.d/system-auth` **file:**

```
auth        required      /lib/security/pam_tally.so onerr=fail
no_magic_root
```

```
account     required      /lib/security/pam_tally.so deny=5
no_magic_root reset
```

Because the order of the lines in `/etc/pam.d/system-auth` is important, we list the complete file contents.

```
auth        required      /lib/security/pam_env.so
auth        required      /lib/security/pam_tally.so onerr=fail
   no_magic_root
auth        sufficient    /lib/security/pam_unix.so likeauth
auth        required      /lib/security/pam_deny.so

account     required      /lib/security/pam_unix.so
account     required      /lib/security/pam_tally.so deny=5
   no_magic_root reset

password    required      /lib/security/pam_cracklib.so retry=3
   type= minlen=8 difok=3
password    sufficient    /lib/security/pam_unix.so use_authtok
   md5 shadow remember=15
password    required      /lib/security/pam_deny.so

session     required      /lib/security/pam_limits.so
session     required      /lib/security/pam_unix.so
```

Monitoring System Activity

Examine all of the log 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.

Sun Linux includes an automatic log analysis and reporting tool named logwatch. This tool sends nightly email reports to a root user. The email address can be changed by editing the `/etc/log.d/conf/logwatch.conf` file. Logwatch is of limited use from a security perspective, because it does not constantly monitor for unusual activity.

The `syslog` daemon receives log messages from several sources and directs them to the appropriate location based on the configured facility and priority. The programmer interface `syslog()` and system command `logger` are available for creating log messages. The facility or application type and the priority are configured in the `/etc/syslog.conf` file to forward log messages to specified locations. The location can be a log file, network host, selected users, or all users.

By default, Sun Linux defines several log files in the `/etc/syslog.conf` file:

- The `/var/log/messages` log file contains a majority of the system messages.
- The `/var/log/maillog` file contains mail system messages.
- The `/var/log/secure` log file contains a majority of the security messages from `sudo` and `ssh`.

If you change the `/etc/syslog.conf` file, the `syslog` daemon must be restarted. Use the following command.

```
# killall -HUP syslogd
```

In addition to logging `syslog` events locally on each client system, Sun recommends that `syslog` events be sent to a centralized log server where logs can be more safely stored and analyzed. As an added benefit, by logging events to a central location, logs may be more readily preserved in the event that the client system is compromised.

Note that `syslog` monitoring is just a single process. Sun recommends that users protect their environments through architectures that implement defense-in-depth through mutually reinforcing, complementary security controls. The methodology for determining which controls are most appropriate to your environment and where they should be positioned in your architecture is outside the scope of this article.

Additional layered monitoring methods such as periodic-vulnerability assessments, file system integrity monitoring, and host-based intrusion detection mechanisms can greatly improve your ability to detect attempted or actual breaches of security whereas a single method might be more easily subverted.

References and Related Resources

Publications

- "Extending Authentication in the Solaris 9 OE Using Pluggable Authentication Modules (PAM)."

  Part I: `http://www.sun.com/blueprints/0902/816-7669-10.pdf`

  Part II: `http://www.sun.com/blueprints/1002/816-7670-10.pdf`

- Hatch, Brian, and Osborne, James Lee. *Hacking Linux Exposed*, Second Edition McGraw-Hill, ISBN: 0072225645, November 2002.
- Nemeth, Evi, Snyder, Garth, Seebass, Scott, and Hein, Trent R. *UNIX System Administration Handbook*, 3rd Edition, Prentice Hall PTR, ISBN: 0130206016, August 2000.
- Noordergraaf, Alex. "Solaris Operating Environment Network Settings for Security: Updated for Solaris 9 Operating Environment," Sun BluePrints OnLine, June 2003, `http://www.sun.com/solutions/blueprints/0603/816-5240.html`.
- Noordergraaf, Alex and Watson, Keith. "Solaris Operating Environment Security: Updated for the Solaris 9 Operating Environment," Sun BluePrints OnLine, December 2002, `http://www.sun.com/solutions/blueprints/1202/816-5242.pdf`.
- Reid, Jason. "Building OpenSSH - Tools and Tradeoffs," Sun BluePrints OnLine, January 2003, `http://www.sun.com/blueprints/0103/817-1307.pdf`.
- Reid, Jason. "Configuring the Secure Shell Software," Sun BluePrints OnLine, April 2003, `http://www.sun.com/blueprints/0403/817-2485.pdf`.
- Reid, Jason. "Integrating the Secure Shell Software," Sun BluePrints OnLine, May 2003, `http://www.sun.com/blueprints/0503/817-2821.pdf`.
- *Red Hat Linux 9: Red Hat Linux Security Guide*, Red Hat Inc., 2002.
- Stevens, Richard W. *TCP/IP Illustrated, Volume 1*, 1st Edition, Addison-Wesley Publishing Company, ISBN: 0201633469, January 1994.

Web Sites

Note – Sun is not responsible for the availability of third-party Web sites mentioned in this document. Sun does not endorse and is not responsible or liable for any content, advertising, products, or other material on or available from such sites or resources. Sun will not be responsible or liable for any damage or loss caused or alleged to be caused by or in connection with use of or reliance on any such content, goods, or services that are available on or through any such sites or resources.

- Center for Internet Security - Linux Benchmark `http://www.cisecurity.org/`
- 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: `http://www.sun.com/blueprints`
- TCP Wrappers tool, Wietse Venema: `ftp://ftp.porcupine.org/pub/security/index.html`

About the Authors

Glenn Brunette

Glenn Brunette is a Sun Principal Engineer with over a decade of experience in information security. Glenn works in the Sun Professional Services division as the Americas Lead Security Architect. In this role, he is responsible for the development and execution of the region's security services strategy. He works with teams throughout the Americas and the world to improve the quality and security of services delivered to Sun's customers.

Previously, Glenn worked in the North East and Financial Services Areas developing and delivering a wide array of tailored security solutions supporting the lifecycle of assessment, architecture, implementation, and management. His customers have included major financial services firms, service providers, and life sciences and government organizations. In addition to contract services, Glenn works closely with teams across Sun on the development and delivery of security strategy, methodologies, best practices, training, and tools. Glenn is a co-founder of the very

popular freeware Solaris Security Toolkit software. Glenn is a Certified Information Systems Security Professional (CISSP) and has been trained in the National Security Agency's INFOSEC Assessment Methodology (IAM).

Michael Hullhorst

Michael Hullhorst is a Staff Engineer working within Sun Microsystems as a Lead Security Architect and evangelist for the Sun Linux Security Group. The group is responsible for driving security into Sun's Linux products.

An engineer with three decades of system development experience, Michael has worked in the capacity of Director of Engineering for Progressive Systems, developing Linux based security products including the Phoenix Adaptive Firewall Appliance. Further, Michael has been an independent consultant with a wide range of experience including the development of complex applications and networks, as well as working with embedded systems. Additionally, Michael's background includes being the Director of Information Systems for a large telecommunication provisioning group.

Michael has been trained in the National Security Agency's INFOSEC Assessment Methodology (IAM).

Gé Weijers

Gé Weijers is a staff engineer working for the Sun Linux Security Group in Columbus, Ohio. He specializes in Security Engineering, the construction of systems that perform their intended function in a dependable way.

Recently Gé worked on the design and delivery of software that reduces the vulnerability of Linux-based Sun products to network-based attacks, and he worked on the development of new initiatives within Sun to improve the security of Sun products.

Prior to Sun, Gé worked for Progressive Systems, Inc. of Columbus, Ohio. The main product was the Phoenix Adaptive Firewall Appliance. He designed the cryptographic tools used to allow secure remote configuration of firewalls using a user interface based on Java™ technology.

Some of Gé's professional interests are security engineering, cryptography, protocol design, and provable security.

Ordering Sun Documents

The SunDocs[SM] program provides more than 250 manuals from Sun Microsystems, Inc. If you live in the United States, Canada, Europe, or Japan, you can purchase documentation sets or individual manuals through this program.

Accessing Sun Documentation Online

The `docs.sun.com` web site enables you to access Sun technical documentation online. You can browse the `docs.sun.com` archive or search for a specific book title or subject. The URL is `http://docs.sun.com/`

To reference Sun BluePrints OnLine articles, visit the Sun BluePrints OnLine Web site at: `http://www.sun.com/blueprints/online.html`