by S. LEE HENRY

Monitoring the Mighty Root

Probably the single most significant security vulnerability in UNIX is the all-powerful superuser. Contrary to the limitations that ordinarily come into play in “real life,” the superuser has unlimited access in most every implementation of UNIX and requires no collaboration or approval from any other user. Ripe for abuse, this situation requires those of us with root (i.e., superuser) privilege to be both unusually ethical and unusually careful, and makes root access the Holy Grail of hackers.

Protecting against the unauthorized access and abuse of root privilege is a job that could keep any number of sysadmins busy around the clock. Just in case your site can’t afford to provide this kind of coverage, let’s consider some less aggressive things the typical site might do to limit the risk.

Changing Passwords

The root password should be changed more frequently than most user accounts. Depending on the sensitivity of the particular system and its data, this could be anywhere from every week to every couple of months. For extremely security-conscious systems, you might want to go to a onetime password system, such as S/Key or OPIE. If more than one individual needs this access, you can assign multiple user names (same UID) to the root user so there is no need for complicated coordination.

Disallow direct root login except on the console, and then only if the console is truly secure. In SunOS, you can disallow all direct root logins. This is important because if more than one person has the root password, log files will not capture who the individual logging in really is.

If you disallow root login on the console, however, be sure to have local logins (i.e., in the `/etc/passwd` file) for all sysadmins who might need access to the system. If NIS or NIS+ fail for some reason, you will need someone to be able to log in–short of far more drastic procedures.

In SunOS, the “secure” designator in the `/etc/ttytab` file permits local root login:

```
console  "/usr/etc/getty cons8"      sun      on local secure
ttya     "/usr/etc/getty std.9600"   unknown  on
```

JAMES BOZZINI

This section of the `/etc/default/login` file restricts root login to the console in Solaris–i.e., root cannot login anywhere else:

```
# If CONSOLE is set, root can only login on that device.
# Comment this line out to allow remote login by root.
#
CONSOLE=/dev/console
```

It is also a good idea to disallow root logins over the network. This will help “contain” a problem if an unauthorized person learns your root password.

Who Can su?

SunOS had the wheel group, which could be used to explicitly list those individuals allowed to execute an `su` to root. Although this feature does not exist in Solaris, you can consider limiting use of the `su` command by making it only executable by root and the sysadmin group (group 14). However, this doesn’t prevent your users from locating and downloading their own copy of `su`.

Limiting the use of trusted hosts is an extremely effective way to limit the spread of unauthorized root access. If you are in an environment with multiple sysadmins, it's a good idea to run checks of `/.rhosts` files through `cron`. It is not unusual for even fairly senior sysadmins to give a system "temporary" trust, but then forget to remove the host name from the `/.rhosts` file when it is no longer needed.

Because root on hosts listed in the `/.rhosts` file has the same authority as the local system's superuser, this privilege should be carefully monitored.

When trusted hosts are necessary, or when the benefit provided to system managers outweighs the risks of their use, the direction of trust between your systems can be extremely significant. Generally, the best approach is to have a single system that many other systems trust, rather than a more complicated relationship where host A trusts host B and host B trusts host C and host D and so on. The latter is hard to control and allows more routes to a system than you will likely want to expose it to.

NFS mounts can provide root access to mounted file systems if the remote superuser access feature is explicitly requested. Don't use this feature except when there's a real need for it. This, like trusted hosts, exposes one system to the activities of a possibly rogue root on another.

Making Routine Checks

Gauging and monitoring the status of your network with regard to root privilege is a good thing to do periodically. Here's what I suggest:

1) You might have a `cron` job that checks whether the root password has been changed in the last month and sends you mail if it has not. There are several ways this could be done. One way is to save root's "encrypted" password in a protected file along with a date and compare it to the current password.

2) You can check to be sure that the policy decisions you've made regarding direct root logins are not changed in practice by modifications to the files in question. You might do this by comparing the active file to an earlier copy, or by using software that monitors changes in system files, for example, TripWire.

3) You can routinely search for "extra" copies of `su`. Because Trojan horse `su` commands are one way that hackers might try to get the root password, or yours, for that matter, this is a good idea in general.

```
chaos# find /home -name su -ls
```

4) You can create "maps" of your trusted host matrix. Try to capture the essence of who trusts whom and pick out vulnerable systems.

```
    Host A      Host B       Host C
    ======      ======       ======
    Host B      Host A
    Host C
```

In the example above, Host A trusts B and C, C trusts no one. Although in this example, Host B doesn't trust Host C, a superuser on Host C can gain access to Host B through Host A.

5) Keep and routinely check the `sulog` to see who is using the `su` command, especially those using `su` to gain access to the root account.

This file may get very large, so it's a good idea to use a script to reduce the data to counts of each pair of real and effective users:

```
slee-root: 359 times
mrbill-root: 2 times
mrbill-gumby: 11 times
```

This type of data showing how many times each user has used `su` to become another user can be very handy.

S. Lee Henry is on the board of directors of the Sun User Group and spends most of her weekdays spreading UNIX wisdom at TASC in Reston, VA. She also is an adjunct professor for Eastern Michigan University. In her infrequent spare time, she writes short stories, sings and plays her collection of percussion instruments.