developerWorks®

Hardening the Linux server

An introduction to GNU/Linux server security

Jeffrey Orloff

January 23, 2014
(First published December 17, 2008)

Servers — whether used for testing or production — are primary targets for attackers. By taking the proper steps, you can turn a vulnerable box into a hardened server and help thwart outside attackers. Learn how to tighten Secure Shell (SSH) sessions, configure firewall rules, and set up intrusion detection to alert you to possible attacks on your GNU/Linux® server.

Linux already claims a large share of the server market, and forecasts show that this share will increase because of the demands of cloud computing. Enterprise IT shops concerned with security need to take a look at the vulnerabilities these servers pose to the network and how these machines can be secured. This article demonstrates how to tighten Secure Shell (SSH) sessions, configure a firewall, and set up intrusion detection.

Plan the server installation

Securing the desktop and operating system

Although many core security concepts apply to both the desktop operating system and the server operating system, the ways they're secured are different. By default, a desktop operating system provides the user with an environment that can be run out of the box. Desktop operating systems are sold on the premise that they require minimal configuration and come loaded with as many applications as possible to get the user up and running. Conversely, a server's operating system should limit access to the minimal level that will allow normal functioning.

The first step in hardening a GNU/Linux server is determining the server's function, which determines the services that need to be installed on it. For example, if the server in question is used as a web server, you should install Linux, Apache, MySQL, and Perl/ PHP/ Python (LAMP) services. If the server is used for directory services, the only applications and services that should be permitted to run on it are those required for the task it's meant to perform. Nothing extra should be installed for two reasons:

- Installing extra software or running extra services creates unnecessary vulnerabilities. For example, if you run Lightweight Directory Access Protocol (LDAP) on a server for directory

services, both the operating system and LDAP must be up-to-date with security fixes and patches. If LAMP (or any other software) were installed on this server, it also would require updates and attention, even if it weren't used. Its mere existence on the server gives an attacker another avenue into your system.
- Installing extra software on a server means that someone will be tempted to use that server for something other than its intended use. Using the server for tasks other than its main task diverts resources from its primary job and exposes it to potential threats.

GUI login

Some people who rely on a GUI such as GNOME or KDE might be inclined to install a graphical login such as the GNOME Display Manager. This isn't necessary, because you can log in from the command-line interface just as easily as you can through a GUI-based login screen. The only difference is that you have to use the `sudo startx` command if you need to administer your server through a GUI.

You need to decide if you want to install a graphical user interface (GUI). GNU/Linux admins have long held a certain pride in administering their networks and servers from a command-line interface. But some systems administrators have begun administering their GNU/Linux servers through a GUI. A GUI can tax a system's resources and, because it's an extra service that isn't necessary, create vulnerabilities. However, the GUI process can be killed when it's no longer in use, and it makes certain tasks, such as working with a database, much easier for the administrator.

If you decide you want to install a GUI, the following instructions show you how to install GNOME as a desktop GUI:

1. Log in to your system. To install the GNOME core, type the following at the command prompt and press **Enter**:
```
sudo aptitude install x-window-system-core gnome-core
```
2. Type your `sudo` password, and then press **Enter**.
   You're informed about what is being installed.
3. Type `Y`, and then press **Enter**.
   This installs a scaled-down version of GNOME that keeps the features of the desktop environment to a minimum and saves system resources. To install the full-featured version of GNOME, type:
```
sudo aptitude install x-window-system-core gnome
```
4. Press **Enter** and follow the process until GNOME is installed on your system.
5. When either package is finished installing, you're still at the command prompt. To open GNOME, type `sudo startx`.

Tighten the Secure Shell protocol

SSH gives a user a connection to a remote computer; systems administrators commonly use SSH to log in to their servers from a remote computer to do maintenance and administrative tasks. Even though SSH provides a much greater level of security than the protocols it replaced, you can do some things to make it more secure.

Security by obscurity

Install Emacs

To install Emacs, use `sudo aptitude install emacs`. Now, locate the portion of the file where you set the port number. When you've found it (the default is port 22), you can change it to an arbitrary number. More than 65,000 ports are available: Choose something at the upper end of the scale, but a number you'll remember. Remember, skilled attackers know how people think. Changing the port number to 22222 or 22022 is a common mistake, so choose a number that isn't easily guessed.

One of the most common methods for hardening SSH is to change the port number used to access it. The theory is that an attacker using the default port or TCP 22 to establish a connection will be denied access because the service is running on a secure port. However, changing the port number won't prevent an attacker with a port scanner from finding the SSH port if he takes the time to scan all of the ports on your server. For this reason, many systems administrators don't bother changing the port. But this approach does prevent script kiddies from attacking SSH with automated tools dedicated to finding open TCP 22 ports, and impatient attackers may grow weary of scanning your server if they don't find SSH running in the first range of ports they scan.

To change the SSH port address, first install SSH on your server. Type the following command, and then press **Enter**:

```
sudo aptitude install openssh-server
```

Type your password. This command installs `openssh` to use for remote logins to your server.

When you have an SSH file to configure, copy the file — just in case something happens during configuration. You can always revert to the original. Then:

1. At the command line, type the following command, then press **Enter**:
   ```
   sudo cp /etc/ssh/sshd_config /ete/ssh/sshd_config.back
   ```
2. Type your password to complete the backup of this file.

Now, you need to change the permissions for the sshd_config file so you can change it. To do so:

1. Type the following command, then press **Enter**:
   ```
   sudo chmod 644 /etc/ssh/sshd_config
   ```
2. Now you can use a text editor such as Emacs or vi to change the file:
   ```
   emacs /etc/ssh/sshd_config
   ```

Leave Emacs or vi open as you make more changes to this file.

Root login permissions

The root user in all Ubuntu distributions is disabled, but you can activate this account. If you're using SSH, you should deny the root account permission to log in to the server remotely in the event that you or an attacker has activated this account. While you have the editor open, scroll down to the line that reads `PermitRootLogin`. The default is yes.

Whitelist users

Another step you can take to harden SSH on your server is to allow only certain people to use this service. This process is known as *whitelisting*. To create a whitelist, you first need the user names of the people who will be allowed to use SSH to access the server remotely. Then, perform these steps:

1. Add this line to your sshd_config file:
   ```
   # Allow only certain users
   AllowUsers username username username
   ```
   Substitute user names from your list in place of the word `username`. Alternately, you can allow groups access to SSH logins by using:
   ```
   # Allow only certain groups
   AllowGroups group group
   ```
   Again, substitute your user groups for the word `group` in the example.
2. Save your configuration file, and exit your editor.
3. Restart SSH for the changes to take effect. You don't need to shut down your computer—just type `sudo service ssh restart`.
4. Press **Enter** and provide your password.
   The service restarts and tells you `[OK]`.

You can secure SSH in other ways, although those are for more advanced users. When you've had more experience working with GNU/Linux and SSH, consider taking those steps.

Write firewall rules

You can deny access to your server through your firewall. Ubuntu Server uses a firewall called Uncomplicated FireWall (UFW), which is a management tool for `iptables`. `Iptables` filters network packets based on rules the systems administrator writes. `Iptables` can be complicated for beginners, but UFW simplifies it. Using UFW can help you harden your server, but if you write rules for `iptables`, you can fine-tune a server's security.

There was a time when the systems administrator had to install UFW using `sudo aptitude install ufw`. However, Ubuntu and many other GNU/Linux distributions come with this application installed; it's just not enabled by default. To enable UFW, type the command `sudo ufw enable` into an open terminal and press **Enter**. Now, type:

```
sudo ufw status verbose
```

Press **Enter**. The output you see should look similar to that of Listing 1:

Code output

```
youruser@yourcomputer:~$ sudo ufw status verbose
[sudo] password for youruser:
Status: active
Logging: on (low)
Default: deny (incoming), allow (outgoing)
New profiles: skip
youruser@yourcomputer:~$
```

Notice that by default, all incoming traffic is denied. You can change this to allow all incoming traffic, but that would defeat the purpose of the firewall. Instead, you can create rules that specify which traffic you'll allow by:

- Port and protocol
- IP address
- Service

Firewall rules by port and protocol

The easiest — and most common — way to write a firewall rule is to block traffic on a specific port. For instance, if you would like to allow incoming traffic on port 80, you would enter this command in the terminal:

```
sudo ufw allow 80
```

You also have the option of specifying which protocol is allowed. If no protocol is specified, then both TCP and User Datagram Protocol (UDP) are covered under the rule. However, to allow only TCP traffic on port 80, the command is:

```
sudo ufw allow 80/tcp
```

To allow incoming traffic on port 80 for UDP, the same syntax applies:

```
sudo ufw allow 80/udp
```

To deny traffic for all of these examples, enter:

```
sudo ufw deny 80
sudo ufw deny 80/tcp
sufo ufw deny 80/udp
```

Firewall rules by IP address

You can also write rules that pertain to specific IP address and even subnets. To block a specific IP address, the rule is:

```
sudo ufw deny from xxx.xxx.xxx.xxx
```

To block an entire subnet, you need to enter the entire range of addresses. For example, for this class C network, you would type:

```
sufo ufw deny from 192.168.1.0/24
```

The same syntax applies should you need to allow traffic from only one IP address or subnet — just substitute `deny` for `allow`.

Firewall rules by service

Firewall rules can also be written to govern an entire service. For example, if you want to deny SSH traffic, the rule would read:

```
sudo ufw deny ssh
```

To allow SSH traffic, state this fact by typing:

```
sudo ufw allow ssh
```

Stopping the firewall

In the event you need to undo a firewall rule that you have written, you can delete the rule as follows:

```
sudo ufw delete deny ssh
```

To stop the firewall altogether, you have the option of disabling it using:

```
sudo ufw disable
```

Disabling a firewall that runs essential services or houses sensitive data is inadvisable. Even taking down this protection for a moment could open the server and its resources to threats.

Monitoring your system

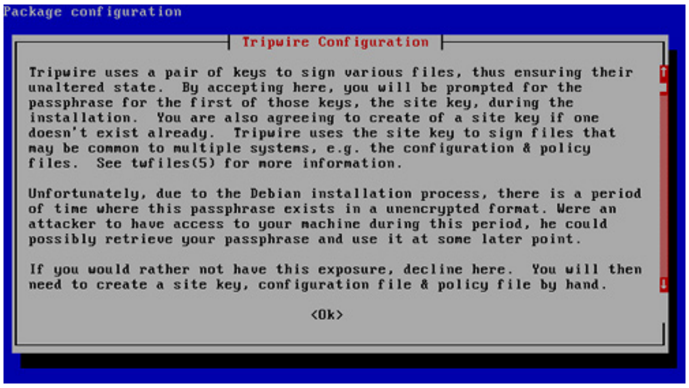
After you take steps to prevent intrusion, you need to set up a monitoring system to detect whether an attack against your server has taken place. If you're alerted to an attack, you're better prepared to handle it. Tripwire (see Related topics) alerts you to unauthorized activity that takes place with system files on your server. Use Logwatch (see Related topics) to create reports you can analyze.

Tripwire

Tripwire sets up a baseline of normal system binaries for your computer. It then reports any anomalies against this baseline through an email alert or through a log. Essentially, if the system binaries have changed, you'll know about it. If a legitimate installation causes such a change, no problem. But if the binaries are altered as a result of a Trojan horse-type installation, you have a starting point from which to research the attack and fix the problems.

1. To install and configure Tripwire through the command line, type the following command and then press **Enter**:
   ```
   sudo aptitude install tripwire
   ```
2. Choose **Yes** to all of the questions during the installation.
   You may be asked to create a passphrase. If you are, make note of it for future use.
3. When you reach the screen shown in Figure 1, Tripwire has been installed. Click **OK**.

Tripwire installation complete



```
Package configuration

                         Tripwire Configuration

  Tripwire uses a pair of keys to sign various files, thus ensuring their
  unaltered state.  By accepting here, you will be prompted for the
  passphrase for the first of those keys, the site key, during the
  installation.  You are also agreeing to create of a site key if one
  doesn't exist already.  Tripwire uses the site key to sign files that
  may be common to multiple systems, e.g. the configuration & policy
  files.  See twfiles(5) for more information.

  Unfortunately, due to the Debian installation process, there is a period
  of time where this passphrase exists in a unencrypted format. Were an
  attacker to have access to your machine during this period, he could
  possibly retrieve your passphrase and use it at some later point.

  If you would rather not have this exposure, decline here.  You will then
  need to create a site key, configuration file & policy file by hand.

                                  <Ok>
```

4. Open your text editor to make the configurations shown in Listing 2, substituting your server name for `hostname` (you'll be using Emacs here):

   Tripwire configurations

```
hostname ~ # emerge tripwire
hostname ~ # cd /etc/tripwire
hostname ~ # emacs -nw /etc/tripwire/twpol.txt
hostname ~ # emacs -nw /etc/tripwire/twcfg.txt
```

5. Enter the following command to create keys and sign the policy:

```
hostname ~ # cd /etc/tripwire ; sh ./twinstall.sh
```

6. Initialize everything, and create the database using the next command (you should be asked to supply your passphrase here):

```
hostname ~ # tripwire --init
```

When this process is complete, Tripwire has created a snapshot of your system. This baseline will be used to check whether any critical files have been changed. If they have, you'll be alerted to it.

You can run reports from Tripwire, as well. From your editor, type this command:

```
sudo twprint --print-report -r\
```

Now, your prompt changes to a single carat (`>`). At this new prompt, type:

```
/var/lib/tripwire/report/[hostname]-YYYYMMDD-HHMMSS.twr| less
```

If you don't know the exact time you ran your report, navigate to the directory /var/lib/tripwire/reports to see the complete file name.

To fine-tune the capabilities of Tripwire, you can look to `twadmin`. You can also set a `cron` job to email you a copy of this report each day or configure Tripwire to email you if an anomaly is reported.

Logwatch

Logwatch helps you monitor your system's log files. This program requires a working mail server on your network to email the logs to you. If you want to change the .conf file, you need to open /

usr/share/logwatch/default.conf/logwatch.conf and look for the line that reads `MailTo`. Change `user.name.domain.tld` to your email address.

You can install Logwatch with this command:

```
sudo aptitude install logwatch
```

To email the logs to yourself, type:

```
logwatch --mailto email@youraddress.com --range All
```

Pressing Enter sends a copy of the report to the email address specified. If you aren't running a mail server on your network but would still like to see a Logwatch report, the following command provides it on your screen:

```
logwatch --range All --archives --detail Med
```

The output spans several screens; press **Shift-Page Up** to move to the beginning of the report.

Users and groups

On a GNU/Linux system, you can organize users into groups for easy administration — but you also need to provide access to files and folders through permissions. No blanket "power user" gives users access to everything on a computer or network, as it does on a Windows operating system. The GNU/Linux system was designed to be more secure; it works off a 3x3 system for granting permissions:

Don't run as root

Never, never, never run anything as the root user in a GNU/Linux system. When you need to run something as root, use the `sudo` command. Any systems administrator can use the `sudo` command if you give him or her the password. To see how and when the `sudo` password is being used, check out /var/log/messages. Because you're looking for all uses of `sudo`, use the `grep` command to find them.

- **File permissions.** Read (r), write (w), and execute (x). Each of these permissions is also given a number: read = 4, write = 2, and execute = 1.
- **Directory-level permissions.** Enter, which gives permission to enter the directory; Show, which gives permission to see the contents of the directory; and Write, which gives permission to create a new file or subdirectory.
- **How permissions are assigned.** Permissions are assigned in three ways: by user level, group level, and other level. The user level defines the user who created the file or directory; the group level defines the group the user is in; and the other level is for any user outside of the user's group.

The user permissions are granted first: For example, r/w/x means the user can read, write, and execute the file or files in the folder. You can apply the number value to each permission. Thus, if a user can read, write, and execute, you add the corresponding numbers 4, 2, and 1 for a total of 7. Next come the group permissions. For instance, the other members of the user's group may be able to read and execute but not write. Adding up the corresponding values gives you 5. Those in the others category can only read the files, so their numerical value is 4. Thus, the permissions for the file or folder are 754.

When permissions are set to 777, everyone is given the ability to read, write, and execute. The `chmod` command changes permissions for files and directories. If you want to change ownership of a user, use the `chown` command. To change group ownership of a file or directory, use the `chgrp` command.

Encryption

Encryption is the process of scrambling data stored on a computer in a manner that makes it unreadable to anyone who doesn't possess the key to re-create the data in its original form. Data that has been encrypted can be stored on the local computer, stored on a network share, or transmitted to other users and computers.

You can encrypt an entire hard disk or the partitions of the disk. This should be done at installation. You can also secure data through encryption by creating a directory and encrypting it. For example, if you've set up a file server, you may want to encrypt a directory that holds sensitive information.

Before you go forward with protecting your data, you need to install eCryptfs from the Ubuntu repositories by typing:

```
sudo aptitude install ecryptfs-utils
```

When installed, set up a private directory where you can store your encrypted files and folders. To do so, type this command in the terminal:

```
ecryptfs-setup-private
```

You'll be asked to enter your login password and then to create a mount pass (or have one generated for you). Write down this passphrase: You'll need it to recover data manually. Log out of your computer, and then log back in. When you are logged in, any folders or files you write in `~/Private` will be encrypted.

Recovering data

In an emergency, you may need to recover your encrypted data. You can do so automatically by making sure that your hard drive is mounted, and then opening the terminal. At the prompt, run:

```
sudo encryptfs-recover-private
```

Follow the prompts, and you'll be able to access your data after it has been decrypted. Make sure you save it in another location so you can access it again.

Additional security steps

Now that you've created a solid foundation for hardening your server, take a few more steps to enhance the security measures you've put into place.

Updates

Never install updates and patches on a production server until they've been tested on a test, or development, server. Because a GUI may not be installed on your server, you have to download any updates and patches through the terminal. When you're ready to install updates, enter the command `sudo apt-get update`, and then `sudo apt-get dist-upgrade`. In some cases, you need to restart your server.

Malware

Although viruses don't pose much of a threat to the GNU/Linux server, if you run Samba to share Windows files, make sure an antivirus scanner like ClamAV is installed so infected files don't spread throughout your system. In addition to viruses, worms, Trojans, and the like, there is also the danger of a hacker installing a *rootkit* on your system and gaining root-level permissions to capture passwords, intercept traffic, and create other vulnerabilities. To combat this threat, install tools such as the Rootkit Hunter, (`rkhunter`), and `chkrootkit` on the server (see Related topics for a link to "Hardening the Linux desktop," which contains instructions).

Backup and recovery

Servers that house terabytes of information, corporate websites, or catalogs for directory services need to have a backup and recovery strategy in place. Not only does this make sense, but sometimes e-discovery laws and regulations require that you hand over information upon request. Most corporate networks can afford redundancy through multiple servers, and smaller networks can find peace of mind through virtualization and backup and recovery software.

If you're planning to run backup and recovery software from the Ubuntu repositories, Simple Backup (SBackup) is an excellent choice, because it can be run from either the command-line interface or a GUI. When backing up server data on a corporate network, it's important to store your backup files outside the server. Portable storage devices provide large amounts of storage space at reasonable prices, and they're excellent options for storing backed-up files and directories.

Passwords

As the systems administrator, you're required to set passwords for your server's root account and possibly other sensitive accounts in your organization, such as MySQL databases or FTP connections. You can't force strong passwords for your users with Ubuntu Server, but you can be sure you train users on how to create a strong password.

Network password policy

If you run directory services like OpenLDAP, you have the option of enforcing strong passwords across your network with some of the configuration options available.

Passwords should be at least eight characters long and contain at least three of the following: an uppercase letter, a lowercase letter, a number, or a symbol. One way to teach users to use strong passwords but keep them from writing down complex passwords on sticky notes is to have them use passphrases. Something like *Myf@voritecolorisBlue!* is much easier to remember than *M $iuR78$,* and both meet minimal complexity standards.

The immutable laws of security

Scott Culp of Microsoft drafted what he called the *10 Immutable Laws of Security* (see Related topics). The 10 laws for administrators are an excellent foundation for hardening any system if applied correctly. Security isn't about risk avoidance, he says, it's about risk management. Things happen. There may be a malware outbreak, your website may be attacked, or a natural disaster might strike. At some point, the security of your system will be tested. Make sure you've done everything you can to protect it, and deal with the threat in a way that keeps your server and its resources available to the users who count on it.

And remember: Throwing more technology at a security problem won't solve it. Vigilance on the part of the systems administrator, buy-in on the part of management, and acceptance on the part of users must all be in place for a security policy to work effectively.

Conclusion

The tasks outlined in this article and "Hardening the Linux desktop" should give you a solid knowledge base on the topic of system security. Keep in mind that these articles are aimed at beginners to provide a foundation for learning more about GNU/Linux security.

Related topics

- "10 Immutable Laws of Security" (Scott Culp, Microsoft, 2010): Find out more about security for users; Culp's follow-on article, 10 Immutable Laws of Security Administration, gives similar guidance for administrators.
- "Secure Linux containers cookbook" (Serge E. Hallyn, developerWorks, 2009): Learn how to strengthen lightweight containers with SELinux and Smack.
- "Secure Linux: Part 1" (Evgeny Ivashko, developerWorks, 2012): Learn about the basic milestones in the development, architecture, and operating principles of Security-Enhanced Linux.
- "Anatomy of Security-Enhanced Linux (SELinux)" (M. Tim Jones, developerWorks, 2012): Learn more about the architecture and implementation of SELinux.
- In the developerWorks Linux zone, find more resources for Linux developers (including developers who are new to Linux).
- See all Linux tips on developerWorks.
- Download VirtualBox to create a virtual machine so that you can practice with the lessons in this article.
- Learn more about Samba, which offers file and print services to Windows computers.
- Learn more about GNOME, a Linux graphical desktop.
- Learn more about GNOME Desktop Manager, which provides you with a graphical login.
- Learn more about KDE, an alternative graphical desktop to GNOME.
- Learn more about UFW from the Ubuntu wiki.
- Learn more about the Tripwire open source project.
- Learn more about Logwatch.
- Learn more about eCryptfs for securing files and folders.
- Learn more about ClamAV to help protect your server from malware.
- Learn more about how Rootkit Hunter can help secure your servers.
- Learn more about `chkrootkit`, which assists in finding rootkits that have been installed on your server.
- Learn more about iptables from the Ubuntu wiki.