Tips to keep your Ubuntu Linux server secure

As a system administrator, one of your key tasks is to handle server security issues. If your server is connected to the Internet, you should place it in the conflict zone. If it's just an internal server, you still need to handle it (maybe one way

As a system administrator, one of your key tasks is to handle server security issues. If your server is connected to the Internet, you should place it in the conflict zone. If it's just an internal server, you still need to handle (possibly randomly) dangerous objects, protesters or certain accountants who want to read the boss's secret e-mail. .

Normally Ubuntu Server is very secure. Ubuntu Server Team - the team that produces office security updates - has made the most successful molting in the history of security industry with Ubuntu Server. Ubuntu is not attached to the open port policy. This means that after you install it, to make Ubuntu desktop or server complete, no application program will accept connections to the internet by default. Like the Ubuntu desktop, Ubuntu Server uses the *sudo* mechanism in the system administrator, avoiding the use of the root account. Security updates are warranted for at least 18 months after release (some versions up to 5 years like Dapper) and are completely free.

In this section we want to discuss system file security, system resource limits, log processing, and network

security. But security in Linux is a difficult and vast topic, so we just want to provide you with some basic ways to resolve conflicts. To become a good administrator, you should consider this issue and learn more from other sources that we will provide in this article.

User account administration

Many aspects of user administration of Linux systems are implemented consistently on its distributions. Previously, Debian provided a number of utilities such as the useadd code, making it easier for you to administer. Later Ubuntu fully inherited Debian's user management model. We will not go into the details of Debian even though it is considered a standard model in user administration. If you want to find out more, check out O'Reilly's website. The problem we are concerned about is the difference of Ubuntu with the standard model: *sudo* .

Ubuntu does not allow default root, administrator, and account. It has a pretty good way to handle security benefits and some amazing discounts. It is *textualized* all in the main pages of the original file *sudo_root* .

During the installation process, when you add the user, by default the user will be placed in the admin group and can use sudo to perform system administration tasks. After adding a new username to the system, you can include them in the admin group with the command:

$ sudo adduser *username* admin

If you want to remove someone from the admin group, you simply need to set the deluser command to replace adduser .

One thing to note is that sudo not only provides root directory access. It can also control small internal permissions, such as commanding: ' *only allow this user to execute 3 commands with superuser privileges'*

The documentation describing these rights is in the 'sudoers' page but it is quite confusing. You just feel a little clearer when you read its example, this document provides most of the situations you need to use sudo. When you really understand, simply run the command:

$ visudo

Here you have to be careful. The sudoers database, located in ' */ etc / sudoers* ' cannot be opened by an editor. Because an editor cannot check the syntax.

If you mess up the sudoers database, you may have to search the data yourself and cannot become an administrator.

File system security

The file security model is standardized in most Unix-like systems and is called a POSIX model. This model has 3 extended file and directory permissions for: owners, groups and other objects. All are done the same at any Linux distribution. That is why we do not focus on analyzing this issue thoroughly. You can refer to the ' *chmod* ' and ' *chown* ' pages in the Linux help section or on the Internet.

We will now focus on secure partitioning through assembly options, an important issue to pay attention to when dealing with system security. Partitioning will have a strong impact if used appropriately. When explaining how

to partition the system, we emphasized the advantages of Linux in providing ' */ home* ', ' */ tmp* ', ' */ var* ' directories for individual partitions. These directories refer to the use of special options when merging sections into the file system.

Many assembly options are dependent file system types. But the options we consider are not this type. We have the following options:

nodev : A file system assembled with the nodev option will not allow the use or creation of special ' *device* ' files. There is no good reason to allow the file system to compile special block and character drives, thus allowing them to create hidden security risks.

nosuid Files in Unix in general and in Linux in particular can be flagged to allow someone to execute files with the rights of other people or groups, usually by system administrators. This flag is called *setuid* (*suid*) or binary *setgid bit* . It also allows executing files outside the directory containing unnecessary system binaries, reducing security. If a user is allowed to use it, he can create or retrieve a suid binary flag separately. The system can then be used effectively.

noexec : if a file system is flagged as noexec, users will not be able to run any executable program within it.

noatime This flag states that the file system does not keep the record of the last visit of the file. If used indiscriminately, it can reduce system safety. Because it limits information about security incidents. This flag also provides performance benefits for any type of use. You should use it on partitions where security is balanced with speed.

The decision to use which assembly option in which partition is a high technique. You will often have to develop references when you become more familiar with the governance mechanism. Here are the basic options you can refer to. Of course you can choose another type, but it should start with this basic type:
• / home-nosuid, nodev
• / tmp-noatime, noexec, nodev, nosuid
• / var-noexec, nodev, nosuid

Limit system resources

By default, Linux will not use any resource limits in user processes. This means that any user is free to fill the working memory on the machine, or generate infinite iterations, returning the system to a usable for a few seconds. The workaround is to set some source limits by editing the file ' */etc/security/limits.conf* ':

$ sudoedit /etc/security/limits.conf

The settings are explained in the comments inside the file. You should use at least the limit of ' *nproc* ' or ' *as / data / _memlock / rss* '.

Tip: a real-life resource limit example

We have just briefly introduced how the limits on production servers. Below is the general login server configuration of the Department of Computer Science, Harvard University, USA:

as 2097152

_____

data 131072

_____

memlock 131072

_____

rss 1013352

_____

hard nproc 128

These limits stipulate that users can use 128 processes, with the maximum address space of 2GB, the smallest data size and the locked address in memory of 128MB, saving file size limits. Maximum stay is 1GB.

System log files

As a system administrator, log log files are one of your best friends. If you follow these files regularly, carefully, you will detect errors in the system as soon as they appear. So you can solve almost any problem before they arise.

Unfortunately, the ability to care about these log files is decreasing. So administrators often only use software to perform log processes, alert them to certain events, or write their own options in a number of languages ??such as Perl and Python.

Log records are usually located in the directory ' */ var / log* '. After your server has been running for a while, you will see a lot of old log file versions are increasing in this directory. Many of them are compressed in the *gzip* compression program (with the extension ' *.gz* ').

Here are some log files to note:

1. / var / log / syslog - normal system log file
2. /var/log/auth.log - log files that authenticate the system
3. /var/log/mail.log - system log file system
4. / var / log / messages - normal log messages
5. / var / log / dmesg - kernel message cache message, usually from system startup.

Toolbox Log toolbox

When reviewing log files, there are a few selection tools that you have to use. The last part of the printed utility is by default the last ten lines of the file, which is a small option that gives information about the last time you access the log file.

$ tail / var / log / syslog

With the -f parameter, the tail is included in the form below, it will open the file and show the change on the screen to let you know.

The *z.grep* , *zcat* , *zless files* also act like the corresponding files without the 'z' at the beginning. These files are *gzip* compressed files. For example, to retrieve a list of lines in all compressed log files with the word ' *warthog*

', simply provide the following command:

$ zgrep -i warthog /var/log/*.gz

Your Toolbox toolbox handles empirical development logs and based on your references, but you should search in apt-cache log files first.

A bit of network security

Network security management is another component, provided by the operating system in a fairly wide array. There is not much difference between Ubuntun and other Linux distribution models.

The iptables command is the front end to the very powerful Linux firewall tables. Unfortunately, manipulating with iptables can be much more difficult when you're trying to set up integrated firewall policies. The command below deletes all packets coming from a bad domain:

$ sudo iptables -A INPUT -s www.slashdot.org -j DROP

The documentation, implementation methods and articles about iptables are available on the Internet in large numbers and the main page system provides detailed information on the appropriate options. You should take some time to learn about iptables because it will allow you to install security mode on any Linux mechanism and it will be easier to learn about other OS firewall systems.

The final point of attention on security

In this article we have just skimmed the surface problems of system security. Although we have tried to give you good suggestions on where to start and where you can learn more. But really, there is no perfect security system. Security does not mean completely eliminating the violations, but just making them difficult to exploit, difficult to attack. This definition can be changed easily. Because it depends on the attacker. If the attacker is a young child, living in the basement, chewing on a cold pizza, it can be frustrating, easily giving up if your security barrier is too safe. But if you are keeping expensive confidential information, no matter how secure the security system is, it can be broken someday. Because destructive activity now comes from the point of view of the attacker's price and profit.

Security is also quite obvious, as it is developed as a concept in computer science. Good security requires a deep understanding of the activities within the computer system. However, there will be no significant improvements if you have a deep understanding of it without knowing where to start. You can start today, follow the above references, and use it to improve your future security knowledge.