15.1.1 The root User

Click one of the buttons to take you to that part of the video.

The root User 0:00-0:58

Every Linux system includes a default superuser account named root. This root account is very powerful. The root user account has full access to every aspect of the Linux system, and as such, it should be used with great care.

In this lesson, we're going to talk about the proper use of the root user account. We're going to also talk about options you can use to gain root level privileges, such as using the su command and also using sudo. Let's begin by discussing the proper way to use the root user account.

One of the key mistakes that I see made by many new Linux users is excessive use of the root user account. There is a time and place when you need to use the root user account.

However, most of your work on a Linux system should be done as a non-root user account. The general rule of thumb is this: you should use root only when absolutely necessary.

Security Implications of Using root 0:59-1:49

If a task can be completed as a non-root user, then that's how it should be done. Why is the proper use of the root user account of concern? Think about the risks of leaving a logged in system unattended. Imagine the havoc that an intruder could wreak if they were to happen upon an unattended system that was logged in as root.

All of the data on the system could be accessed and copied, major configuration changes could be made to the daemons running on the system, and who knows what kind of security exploits could be installed on the system?

Here's the key thing to remember, a system logged in as root represents a serious security risk. Leaving such a system unattended represents a critical security risk. Everyone, including the system administrator--that's you and me--should have a standard user account that they log in to the system with to perform their day-to-day work.

---

su Command 1:50-5:10

If you find that you need root level access while you're working on the system, you can use the su command to temporarily gain root level privileges on the system. This command allows you to change to a different user account at the shell prompt. In fact, su stands for switch user.

The syntax for using su is shown here. We enter su, followed by several options, and then we specify which user account that we want to switch to.

Here is the key thing to remember, and you probably already know this, if no user account is specified with the su command, then it assumes that you just want to switch to the root user account. In fact, that's mostly what we use su for. I actually use it very rarely only to switch to a different standard user account.

I use it probably 99.99% of the time to switch to root. Some of the options you can use with the su command are shown here. Just a plain dash (-) with nothing else specifies that we not only switch to a different user, but we also load that user's environment variables.

For example, if I were to type su- at the shell prompt, it will switch me to the root user account, and it will also load all of root's environment variables. If you need to perform root-level tasks, you're going to need root's environment variables to do so.

Otherwise, you're probably going to have trouble finding certain commands, because if you don't load root's environment variables, your previous user's environment variables will be used, and they may not have all the paths and all the information necessary to find certain commands in the file system.

You can also use the -c option, followed by a particular command. This will switch to the user account specified and also run that command. You can also use -m to switch to that user account but preserve your original user's existing environment variables.

I don't ever use that option. It doesn't even make sense, but it is there if you need to use it.

The su command is a very effective tool to use as a Linux system administrator. Your average end user should not be using the su command. In order to use the su to switch to root, you've got to have the root user's password.

Do not give your end users your root password; that's an invitation for disaster. With that said, be aware that there may be situations when there are legitimate reasons for a standard user on your system to need root-level access. In this situation, you can use sudo to provide them with very limited root-level access to the system.

For example, let's suppose you have a power user on your Linux system. This user could be a programmer, maybe a project manager, maybe a database administrator. Users in this category may frequently need to run some root-level commands, but not all of them.

In that situation, do you really want to give them your root user's password? No. Don't ever do that.

Instead, we want them to be able to run a limited number of commands with root-level privileges, but you don't want to give them full root-level access. This can be done using sudo. The sudo command allows a particular user to run a limited set of commands as the root user account.

As with su, it could actually be any other user on the system as well. It doesn't have to be root; however, nobody ever does it that way. It's most frequently used to run commands as the root user on the system.

/etc/sudoers File 5:11-9:50

The sudo command uses the sudoers configuration file, which is located in the /etc directory, to decide which user is authorized to run what command as the root user. In order to accomplish this, this file uses the aliases shown here to define who is allowed to do what.

First we have the User_Alias. This specifies the users--the user accounts--who are allowed to run commands. Then we have the Cmnd_Alias, which specifies the commands that those users are allowed to run.

We have the Host_Alias, which specifies which host the users are allowed to run those commands on. And then we have the Runas_Alias, which specifies the usernames that these commands may be run as, typically root.

In order to edit your sudoers file, you actually need to run the vi sudo command, and it has to be run as a root user. When you run this command, the /etc/sudoers file is loaded in the vi editor.

Before we go any further, I need to point out that on many distributions the sudoers file is configured by default such that users must supply the root password if they want to run sudo. Obviously, this configuration doesn't make a whole lot of sense, because it doesn't accomplish anything.

If the user already knows the root password, what's the point of configuring sudo? This functionality is configured by these two lines in the sudoers file. It even tells you right here what it does is ask for the password of the target user, such as root.

Well, we don't want that, so what we want to do is actually comment out both of these lines, first thing, in our sudoers file.

Once that's done, then you can begin your sudoers configuration in the file. The first thing you need to do is define your User_Alias, and the syntax is shown here. We enter User_Alias, and then we provide a name for our alias. It could be anything you want, but it must start with a capital letter right here.

Then we use an equals sign (=), and then we use a comma-separated list of user accounts that we want to allow to run specific commands. Then we need to use the Command_Alias to define an alias that contains the various commands that we want these users to be able to run.

The syntax is Command_Alias, and then once again the name of the alias. We can define whatever we want. It also has to start with a capital letter, equals, and then a list of commands that we want these users to be able to run as root.

You do need to specify the full path to the command that you want them to be able to run. If you want them to be able to run multiple commands, you need to separate them with commas.

The next step is to define our Host_Alias. The syntax is Host_Alias, followed by the alias name (capital letter again), equals, and then the hostname of the system that you want the users to be able to run these commands on.

Once you have these aliases all defined in the file, you then need to glue all of these aliases together. So you insert the line shown here within the file as well.

The syntax is the user alias name that you defined, followed by the host alias name that you defined, equals, followed by the name of the user that you want to be able to run the commands as, and then the name of the command alias that you defined.

We're basically saying who can run what on what systems as this user, typically root.

After you've glued everything together, you exit out of the vi editor using the exit command. The vi sudo command will check your syntax to make sure you didn't make any errors.

Once done, then the users you defined here with the User_Alias will be able to execute the commands that you specified as the user that you specified--probably root. With this configuration in place, the end user who needs to run a command as root simply types sudo at the command prompt, space, and then the name of the command that they want to run.

If that command is contained in the Command_Alias, and that user account is contained in the User_Alias, they'll be allowed to run that command as the root user. They will be prompted for a password, but they don't have to specify the root password.

Instead, they simply enter the password for their own user account. That way, you can allow them to perform a limited number of root-level tasks without having to give them the root password to the system.

Summary 9:51-10:00

That's it for this lesson. In this lesson, we talked about the proper user of the root user account. We talked about using su to switch to the root user account and then we ended this lesson by talking about the sudo command.
