8.8.1 Ownership

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

Ownership 0:00-0:21

In this lesson, we're going to discuss file and directory ownership within the Linux file system. To effectively control who is allowed to do what with the files and directories in the file system, you first have to consider who owns each file and directory. So let's begin this lesson by discussing how ownership works.

How Ownership Works 0:22-2:20

Understand that anytime a user creates a new file or a new directory in the Linux file system, that user's account is assigned to be that file or directory's owner by default.

Now by default, the owner of a directory in the Linux file system will automatically receive read, write, and execute permissions to that directory, which basically allows them to do whatever they want to do in that directory as that directory's owner.

Likewise, the owner of a file in the Linux system will receive read and write permissions to that file by default. For example, suppose I log into my Linux system, and then I open up my LibreOffice word processing application and I create a file named schedule.odt in my home directory.

Because I created this file, the owner of this file is me, as it shows right here, which is actually the rtracy user. Because the rtracy user account created this file, the rtracy user is automatically assigned ownership of schedule.odt.

In this screen, what we did is right-click on this file in the graphical user interface of my Fedora system and then I selected Properties, and then I selected the Permissions tab right here, and here I can see who the file's owner is.

Be aware that there are actually 2 different owners for this file. The first one is the name of the user who owns the file. Again, that's me, rtracy.

But, in addition, there's also a group that owns the file as well, and by default it will be the primary group of whatever user created the file in the first place. In this case, my default group in the system is the rtracy group, because that's the primary group that the rtracy user belongs to.

Therefore, the owner of the file is the rtracy user and the owning group of the file is the rtracy group.

Ownership List 2:21-3:26

You can also view file and directory ownership from the command line using the ls -l command. In this example I ran the command in my user's home directory. Notice that the third column in the output displays the name of the file or directory's owner. In this case, they are all owned by rtracy.

The fourth column over here, displays the name of the group that owns the file. Now, all of them are owned by the rtracy group, except for one. The runme file is owned by a different group. It's owned by a group named users.

Now understand that file and directory ownership isn't fixed. Even though ownership is automatically assigned whenever a file directory is created, you can modify it if you want to.

You can specify a different user, or a different group, or both as the owner of a particular file or directory in the file system, but to do so, you need to keep this in mind.

In order to change the user who owns the file, you have to be logged in as root. Only root is allowed to do this. But, to change the group that owns the file, you have to be logged in as either root or as the user who currently owns the file.

---

chown 3:27-6:23

There are a couple of different utilities that you can use to do this. The first on is the chown utility which stands for change owner. It can be used the change the user or the group that owns a particular file or directory and the syntax is shown here.

You enter chown followed by the name or the user of the group that you want to change ownership to, followed by the file or directory in the file system whose ownership you want to change. For example, suppose I have a file named resources.txt. It's located in the /tmp directory, and currently that file is owned by the root user.

Well, I want to change the ownership of that file, from root to the rtracy user. To do this, I would enter chown and then the name of the user that I want to change ownership to, in this case rtracy, and then the name of the file.

If we do an ls command down here, ls -l, we see that rtracy is now the user that owns the resources.txt file. But, notice that the name of the group that owns the file was not changed.

The group named root. Which is the primary group associated with the root user, still owns this file. This was assigned when the file was originally created by the root user.

Well, suppose I want to change the group that owns the file to my primary group from the root user's primary group. Now my primary group is the rtracy group, therefore, I would enter chown and then I would enter .rtracy and then the name of the file that I want to modify.

This dot right here is very important because essentially what it does is tell the chown command that the entity that follows is not a username, but is a group name. When we run the command, we see that the name of the owning group is now changed from root to rtracy.

Be aware that I could've actually accomplished both tasks with one single command. You can change the owning user and group at the same time by simply specifying the name of the user first then the period and then the name of the group that you want to change ownership to, and then the name of the file. It does it all at once.

Also be aware that you can use the -R option with the chown command. In the examples that we've shown right here, we're modifying just one single file at a time and that's fine if you have just one single file to modify. But, if you have a whole bunch of files that you need to modify and you have maybe even subdirectories containing files whose ownership you need to modify, -R saves a ton of time.

Basically, this tells chown to change ownership recursively. In which case, it will burrow down through all of the subdirectories of whatever path you specify and apply the ownership change that you specify to all of those files, all at once.

---

chgrp 6:24-6:50

In addition to the chown command, there is another command you can use to manage ownership. It's the chgrp command, and as its name implies, it's used only to change the name of the group that owns the file or directory.

It cannot be used to modify the user that owns the file or directory. The syntax is pretty straight forward. You enter chgrp, the name of the group that you want to change ownership to, followed by the name of the file that you want to modify.

Summary 6:51-7:04

That's it for this lesson. In this lesson, we discussed file and directory ownership. We first discussed how ownership works. We also discussed how to change the owning user and the owning group, using chown. Then we ended this lesson by talking about how to use chgrp to change just the owning group.
