# 2.7.3 Use Redirection

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

Using Redirection 0:00-0:44

In this demonstration, we're going to practice working with redirection with shell commands. When you use redirection, you take either the input, the output, or the error messages that are used by a shell command, and you redirect them to a file in the file system (or from a file in the file system in the case of the input, as we'll see in just a minute). In order to do this, I'm first going to switch to my root user account, and now I'm going to view my boot log file. The boot log file contains messages that are generated as the system boots up. It can be a valuable troubleshooting resource. I'm going to run ‘tail /var/log/boot.log'. When I do, the last few lines of the boot.log file are displayed.

---

Redirect Standard Out 0:45-3:24

Suppose we wanted to redirect the standard out from this tail command to a file in my /home directory named lastboot, so we could save it and look at it later. Well, there are actually two different ways to do this. One way is to run 'tail /var/log/boot.log', and then we use a 1 to specify that we are now about to redirect the standard out. 1 maps to the standard output of the command. Then we use a > followed by the name of the file that we want to redirect the output to: 'lastboot'. Enter. Notice that nothing was displayed on the screen. That's because the standard out was moved from the default of the screen to a file in the file system. If we do an ls command, we should see a file called lastboot right here. And if we use the cat command, we can view it, and we see that it contains the same information as we saw when we were looking at the live file, u here. The output from the command was redirected to the lastboot file. Now it's saved, and I can manipulate it and do whatever I want with it.

There is a second way to do this, and it's pretty similar to the first. The key point here is that the 1 in the redirection command is actually optional. You don't have to put it there. If you omit the 1, then the shell is going to assume that you want to write the output of the command to a file, because that's what folks do a lot. Let's run the same command again, but omit the 1. And, once again, we'll do an ls command, and we should see the lastboot file is there. And then we can use the cat command to view it. Again, it's the same as it was before.

It's important to note that whenever you do the redirection, either with or without the 1, like we did right here, if this file doesn't exist, it will create it and write the information to it. If this file already exists and already has information in it, whatever is in there will be erased and replaced with the information that's being redirected. In other words, it wipes out whatever's there and replaces it with the new information. There may be situations where you want to redirect the standard out from a command to a file without overriding the existing contents of that file.

For example, let's suppose we wanted to redirect the standard out from the tail command that we used before, and we want to write it to the lastboot file, just like we did before. But this time, we don't want to override what might already be in the lastboot file. If we just run this command again, it will override it. To prevent this, we can actually use two greater-than signs (> >). This tells the shell that instead of overriding the file, we want to just add whatever text is going to be generated by the tail command to the end of the file. Go ahead and run it. Now, if we use the cat command to view it, we see that the file is twice as long because the last command added its text to the end of the existing contents of the file. That's how we redirect the standard out from a command.

---

Redirect Standard Error 3:25-6:28

In addition to redirecting the standard out, you can also redirect the standard error of a command. This will only happen if an error message is generated for some reason. And by default, the standard error is written to the screen. For example, let's run 'cat myfiles.odt'. Hit Enter. When I do, an error is generated because that file doesn't exist in my local directory. There's no such thing as myfiles.odt, so an error message was displayed on the screen instead of the standard out.

If you wanted to redirect that error message from the screen to a file, you run the command like we just did. Now, we add the redirection information. But instead of using one or nothing, which would reference the standard out, we specify 2 and a >, specifying the standard error, and then we enter the filename we want to redirect the standard error to. Let's redirect it to a file named errorfile. Hit Enter. Notice that this time, no text was displayed on the screen. That's because this error message that was generated was redirected to the file named errorfile. Let's do an ls command to see it. Let's use cat to view it, and then you can see the error message that was generated by the cat command was written to the file.

You can actually redirect both the standard out and the standard error at the same time. To do that, let's work with our boot.log file again, and let's specify that we want the standard out to once again be written to a file in the local directory called lastboot, and we also want any error messages that might be displayed to be written to a file called booterrorfile. I'll hit Enter. Nothing is displayed on the screen because we're redirecting both the output and the error messages to a file. There actually shouldn't have been any error messages displayed because it was a successful command. We do an ls command. We see that we have the lastboot file here. We can do a cat to view it. That part worked fine. We

can also cat the booterrorfile file, which should be blank because there were no error messages displayed. But if there were, they would have been written to that file. That's how you redirect the standard out and how you redirect the standard error.

Redirect Standard In

In addition to redirecting the standard out and the standard error, you can also redirect the standard in. Essentially, this allows us to take a file in the file system and send it, as input, to a particular command. I will admit that this is not something I use a lot, but there are certain situations where it can be handy. In fact, there are several Linux commands that you will use later in this course that you have to send a file to that command for it to process.

Let's look at a very simple example. We can take the tail command, and we can send it a file to process. Normally, we would do this by typing 'tail /var/log/boot.log'. But we can also use the less-than sign (<) to send a file directly to the input of the tail command. Enter '/var/log/boot.log'. It works in exactly the same way. This file is sent to the input of the tail command. The tail command processes it and writes the standard out on the screen.

---

Summary 6:29-6:37

That's it for this demonstration. In this demo, we talked about redirection. We first looked at redirecting the standard out. We then looked at redirecting the standard error. Then we ended this lesson by looking at how to redirect the standard in.

---