4.3.2 Managing Boot Targets

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

Manage Boot Targets 0:00-0:08

In this demonstration we're going to spend some time learning how to manage Linux boot targets.

Role of Boot Targets 0:09-0:57

Understand that up until just a few years ago, most Linux distributions did not use boot targets; they used the concept of runlevels, which were managed by the init daemon. That's no longer true.

Almost all Linux distributions have shifted away from the init daemon to the systemd-daemon, and the systemd-daemon does not use the concept of runlevels. Instead, it uses the concept of boot targets, which are fairly analogous in function to the way that runlevels used to work in older versions of Linux.

Each boot target is represented by a file in the file system. I'm going to switch to my root user account here, and we're going to switch to the '/usr/lib/systemd/system' directory.

Boot Target Files 0:58-4:06

This is the directory where all of our boot target files reside and all of our boot target files have an extension of .target. We can use the 'ls' command to display all of the '*.target' files in this directory.

There are actually many different target files; we're not going to look at all of them here. We're going to focus on just a few of them that are used to configure the state of the Linux system.

The first one that you need to be familiar with is the graphical.target file. Graphical.target puts the system into a multi-user mode, meaning that multiple people can be logged in to the system at the same time and it enables networking on the system so it can connect to the network and it provides a graphical user interface.

As you can see, we are using the graphical boot target to set the system state of this system here, because we're using a graphical user interface. But you don't have to use the graphical.target; another boot target you could use is called multi-user.

Multi-user also enables networking on the system and it allows multiple users to log in to the system at the same time, but it does not use a graphical user interface; instead, it uses a text-based user interface.

Graphical.target over here is typical used on workstation systems where end users need a nice graphical user interface to work with. Whereas multi-user is commonly used on Linux server systems. Where we don't want to spend a lot of processor cycles redrawing graphical screens; we want to save those processor cycles for performing useful work and so we disable the graphical user interface, using this boot target.

There's another one that you need to be familiar with, over here, called rescue.target. Rescue.target, as its name implies, is used to repair a broken Linux system. If you use the rescue boot target, the system state will change to one where we have only a single user allowed to log in to the system--usually a root user --and networking is not enabled and it does not use a graphical interface, it uses a text-based interface.

In addition to these three boot targets, there's a couple of other special purpose boot targets that you need to be aware of. First of all reboot.target. If you use reboot.target to set the system state, as its name implies, it will simply reboot the system. There is also poweroff.target. If you use this target file to set the state of the Linux system, it shuts it down.

There are actually seven other boot target files that you can use to set the system state, and these are down here. Runlevel0 through runlevel6.

Notice that these boot targets are blue instead of white in the output of the ls command. That's because they're not really files; they're actually symbolic links that point to other files.

If we do an 'ls' command again, but use the long output format, you can see that runlevel0 points to the poweroff.target. runlevel1 points to the rescue.target. runlevel2, runlevel3, and runlevel4 all point to the multi-user.target file we just looked at. Runlevel5 points to the graphical.target, and runlevel6 points to the reboot target.

Current Boot Target 4:07-4:33

These symbolic links are here to help Linux administrators who are already familiar with the init-daemon and the concept of run level migrate to systemd and the concept of boot targets. As you can see, the equivalent of runlevel0 is poweroff. The equivalent of runlevel1 is rescue. The equivalent of runlevel2, 3, and 4 is multi-user. The equivalent of runlevel5 is graphical, and runlevel6's equivalent is reboot.

Boot Target Switching 4:34-7:01

You manage boot targets using the systemctl command. For example, if we want to see what the current state of the system is, we can use 'systemctl get -default', and as we suspected we are using graphical.target, and the reason we know that is because we're using a graphical user interface.

You can also use the systemctl command to change the state of the system to a different boot target. For example, let's suppose that we want to change to the multi-user.target, where we will have a text-based interface instead of the graphical user interface.

To do this, we enter 'systemctl isolate', and then the name of the boot target file we want to use. In this case let's change it to 'multi-user.target'. Whoops, I misspelled it-- we need a dash in there.

Notice that the system state has changed; we no longer are running a graphical user interface; we're using a text based interface. Networking still enabled, the multi-user nature of the Linux operating system has not changed either, we've just shed the graphical user interface.

I'm going to log back in as my root user. Let's switch back to the graphical boot target: 'systemctl isolate graphical.target', and as you can see we've now booted back into a full graphical environment. Let's open up our terminal window again. Switch back to our root user account.

This system is currently configured by default to boot into the graphical environment using the graphical.target file. In earlier versions of Linux with the init daemon you used the inittab file to determine what the default run level would be.

A lot of the newer distributions use systemd still include the init tab file, but as you can see it tells you right at the beginning init tab is no longer used, adding configuration information here will have no effect on your system. In essence, the init tab file is there, but it doesn't do anything.

Boot Target Default 7:02-8:40

Instead, we use a symbolic link in the '/etc/systemd/system' directory to define what our default boot target is. If I do an 'ls' command here, you can see a symbolic link right here called default.target. Let's do an 'ls -l', so we can see where this actually points. Let's clean that up a little bit by just leading with '*.target'.

Here you can see that default.target points to a particular boot target that we want to use. In order to configure what this default system state will be when the system boots up, we simply change which file this particular symbolic link points to and this is done using the systemctl command.

For example, let's say we want to modify this system such that it boots into the multi-user state without a graphical interface instead of the graphical interface. To do that, we enter 'systemctl set-default to' and then the name of the boot target we want to use as the default. In this case let's set it to 'multi-user.target'.

Notice it tells us down here that it created a symbolic link from /etc/systemd/system/default.target to usr/lib/systemd/system/multi-user.target. We can verify that by looking at the file and we see that it is now changed. Therefore, the next time I boot this system it's not going to boot into this nice graphical user interface; it is instead going to boot into the text-based user interface.

That's it for this demonstration.

Summary 8:40-8:58

In this demo we introduced you to the concept of boot targets, we talked about role of boot targets, we looked at the boot target files, we talked about how to view the current boot target, we talked about how to switch between boot targets, and then we ended this demonstration by talking about how to set the default boot target.
