4.3.1 systemd Boot Targets

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

systemd Boot Targets 0:00-1:32

In this lesson, we're going to look at the concept of boot targets that are used with systemd.

What exactly is a boot target? It sounds kind of complicated. But if you look at it, it actually isn't. A boot target simply represents one of several different modes that your Linux system can run within. Be aware that in earlier versions of Linux that used the init daemon instead of systemd, we used run levels instead of boot targets. The names are different, but the way they work--the concepts themselves--are actually pretty similar.

On a systemd system, a boot target is a mode that the system can boot into. If you're familiar with Windows systems, you know that you can actually boot a Windows system into a different startup mode. You know that during the boot process, you can press the F8 key multiple times until the boot menu is displayed, and then you can select one of several different startup modes.

For example, if you're having trouble with a particular device driver on your Windows system, you can boot it into safe mode, which will just load the basic operating system and only a minimal set of drivers so that you can troubleshoot the driver that's causing the problems. Linux boot targets are really similar.

Like Windows startup modes, Linux boot targets allow you to specify the mode in which you want your system to run. You can also specify a default boot target that your system will boot into each time it's turned on. You can even dynamically switch between boot targets while the system is running. You can't do that with Windows.

Linux Boot Targets 1:33-4:39

Each boot target available on your Linux system is represented by a file in this directory: /usr/lib/systemd/system. They all have an extension of dot target (.target). A list of them is shown here.

First of all, we have poweroff.target, which actually halts the system. We also have rescue.target, which boots the system into a single user mode. Basically, it's used for rescuing a system that's really having a lot of problems. We also have multi-user.target. This provides a multi-user text-based interface--basically, a non-graphical interface. This one is commonly used with server systems, where we don't need a graphical user interface.

We also have graphical.target for systems that do need a graphical user interface, such as a desktop system. This provides a multi-user environment with a graphical user interface. Graphical target usually provides all the services that are provided with multi-user.target, but it also uses a graphical login instead of a text-based login. It allows multiple users to be logged into the system at the same time, and it provides, as we said, a graphical user interface. There's also a target called reboot.target, which simply reboots the system. Then, finally, we have emergency.target, which is used when things are really bad, the system is running very poorly, and you need to create an emergency shell so you can fix the problem.

Now, because init-based Linux systems have been around for decades and everybody is used to working with init, systemd, in order to provide some degree of backwards compatibility, provides several additional boot targets that use the run level syntax, as you can see right here. Basically, this is provided for us old-timers who are just used to init run levels.

First of all, we have runlevel0.target, which is equivalent to poweroff.target that halts the system. We have runlevel1.target, which is equivalent to rescue.target, which boots the system into single user mode. We have runlevel2, runlevel3, and runlevel4.target. These are equivalent to multi-user.target. It provides, basically, a server-type interface. It's a multi-user system, but does not have a graphical user interface configured. Everything's done from the command line.

runlevel5.target is equivalent to graphical.target. This creates a desktop environment where we have a multi-user system and a graphical user interface and a graphical logon. Then, finally, we have runlevel6.target, which is equivalent to reboot.target, which reboots the system.

If you were to go into the /usr/live/system.do/system directory and run the ls -l command, you would actually see that each one of these files here is, simply, a symbolic link that points to one of these files. That's why we say this boot target is equivalent to this boot target, because it is. It's a symbolic link to the actual systemd boot target.

Switching Boot Targets 4:40-6:13

Here's the key thing that you need to understand, especially if you're coming from an older Linux distribution that uses init. On a Linux distribution that uses systemd, you use this command, right here, to manage system services and to manage boot targets. It's the systemctl command. Instead of using init or chkconfig or service, as we did with older Linux distributions, you now use systemctl in order to manage most aspects of a systemd system.

For example, if we have a Linux system running, and we want to change our boot target—"essentially, change the equivalent of changing our run level on older systems--then you would run the system control command, and then you would point to the boot target that you want to switch to using the isolate option.

For example, if I wanted to switch to the systemd equivalent of run level five on my system, I would enter 'systemctl isolate runlevel5.target' at the shell prompt as my root user. Or—"because, remember, run level five is equivalent to graphical target (in fact it's a symbolic link to graphical target), I could also run 'systemctl isolate graphical.target'.

It's important to remember that if you change boot targets on the system while it's running--we call that changing boot targets on the fly--then systemd will only change the current boot target. As long as my system stays running, it will stay in that boot target. But if I reboot the system, then it will boot back into the default boot target.

Viewing the Current Boot Target 6:14-7:48

With older Linux systems that used init, we use the init tab file to set that default run level the system booted into whenever it was powered on. That's not the case with a system-based distribution. Instead, a system-based distribution will use the /etc/systemd/system/default.target file to control the default boot target.

Here's the thing that trips folks up if they're coming to a newer Linux distribution that uses systemd from an older distribution that used init. Remember, we said just a second ago that on older init-based systems, we used a /etc/inittab file to set the default run level. When you look at the /etc directory on a modern Linux distribution that uses systemd, you're still going to see this file, the inittab file.

A lot of folks make the mistake of thinking that that's the file that they use to set the current boot target. It's not. You can do anything you want in this inittab file, and it does not affect the way the system boots up. In fact, if you open up the inittab file on a system-based system, you'll see a big warning. And it's saying, basically, this file is here for backwards compatibility purposes. It doesn't actually do anything. Don't put anything in here.

Even though you might see this file on a systemd distribution, remember that it doesn't do anything. This is the file that you should use. In fact, this file is really, again, just a symbolic link that points over to the appropriate boot target file that we looked at.

Default Boot Target 7:49-8:48

When you change the boot target on the system, all you do is change which file this symbolic link, right here, points to.

If you want to just view what your current boot target is, use this command that you see here: you type 'systemctl get-default', and the command will return what your current boot target is.

On the other hand, if you want to modify the default boot target, then what we need to do is change the target file that the default.target symbolic link points to. You do this using the 'systemctl set-default' command.

For example, if I wanted to set the default boot target of my system to graphical mode--say it's a desktop system, I want a nice graphical user interface--then I would use either of these two commands. I would run either a 'systemctl set-default graphical.target', or I would set 'systemctl set-default runlevel5.target'.

Summary 8:49-9:03

That's it for this lesson. In this lesson, we introduced you to the concept of a boot target under systemd. We first talked about what a boot target is. Then we reviewed several of the key boot targets that are used on a Linux system. Then we talked about how to change boot targets. And then we finally ended this lesson by discussing how to set the default boot target.
