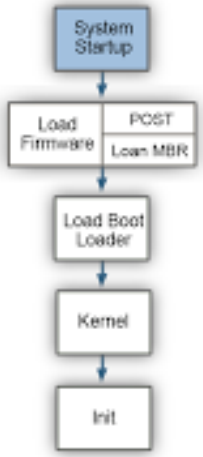
Chapter 17: Runlevels

Boot Process Review

1. The BIOS or firmware stage
   - Varies between hardware vendor
2. Bootloader stage (GRUB)
   - Discussed in last Unit
3. Kernel stage
   - Detects devices, loads drivers and starts systemd
4. The init stage
   - Handles the rest of the bootup process bringing the system to the default runlevel



Runlevels

- Different functional states of the operating system
- Uses numeric values of 0-9
- Typically only 0-6 are defined by default
- Used by both traditional init and Upstart
- Systemd uses something similar called "targets"

Standard Runlevels

- The Linux Standards Base 4.1 defines the purpose of each runlevel as follows:

Runlevel	Purpose	Systemd Target
0	Halt or shut off the system	poweroff.target
1	Single-user mode for administrative tasks	rescue.target
2	Multi-user mode without configured network interfaces or network services	multi-user.target
3	Normal startup of the system	multi-user.target
4	User-definable	multi-user.target
5	Start the system normally with a graphical display manager	graphical.target
6	Restart the system	reboot.target

Three types of "init"

- Traditional init - configured by `/etc/inittab` file
- Upstart - configured by files in the `/etc/init` directory
- Systemd - configured by files in `/etc/systemd` directory

Very rare nowadays

very common

Set default runlevel

- Systemd: use targets
  - `graphical.target` = runlevel 5
  - To set a default target, a symbolic link is created from the target definition to `/etc/systemd/system/default.target`

systemctl set-default <target-name>.target

View current target

To list the target for the next reboot:

- systemctl get-default

To list all active targets:

- systemctl list-units --type=target --state=active

To list all available targets:

- systemctl list-units --type=target

Change current state

- Additional traditional init commands:
  - `halt`
  - `poweroff`
  - `shutdown`
  - `reboot`
- With systemctl:
  - systemctl reboot
  - systemctl poweroff

  ..and a lot more...

The shutdown command

- Must specify a shutdown time:
  - `#shutdown now`
  - `#shutdown +10`
- Can provide a shutdown message to all users who are logged on before shutdown:
  - `#shutdown +15"System going down in fifteen minutes"`
- To reboot: `#shutdown -r`
- To halt: `#shutdown -h`

Notifying Users with wall

- The wall (write to all) command is used to send a message regarding a pending event to all logged on users:

```
$echo -e "The server will be offline on Saturday from\n6:00PM
   to 12:00PM for scheduled maintenance" | wall

Broadcast Message from sysadmin@localhos
        (/dev/console) at 1:56 ...

The server will be offline on Saturday from
6:00PM to 12:00PM for scheduled maintenance
```

Manually Managing Services

- Traditional init uses scripts in the `/etc/rc.d/init.d` directory
- `/etc/init.d` is a symbolic link to the `/etc/rc.d/init.d` directory
- Each script starts or stops a service or feature
- The administrator can use these scripts to manually start or stop a service

Manually Managing Services

- Start a service example:
  #/etc/rc.d/init.d/httpd start
  Starting httpd:

- Stop a service example:
  #/etc/rc.d/init.d/httpd stop
  Stopping httpd:

Manually Managing Services

- To see all possible arguments, run with no arguments:

  #/etc/init.d/httpd
  Usage: httpd {start|stop|restart|conderestart|try-restart|force-reload|reload|status|fullstatus|graceful|help|configtest}

- Note: some systems also have a `service` command:

  -**#**`service httpd start`

Typical script arguments

Argument	Function
start	If the service is not running, then attempt to start it.
stop	If the service is running, then attempt to stop it.
restart	Stop and then start the service over. If you make a major configuration change to a service, you may have to restart it to make it effective.
condrestart	Restart the service on the condition that it is currently running.
try-restart	Same as condrestart
reload	Read and load the configuration for the service. Reloading the configuration file of a service is normally a less disruptive way to make configuration changes to a service effective, but may not be successful for major changes.

Typical script arguments

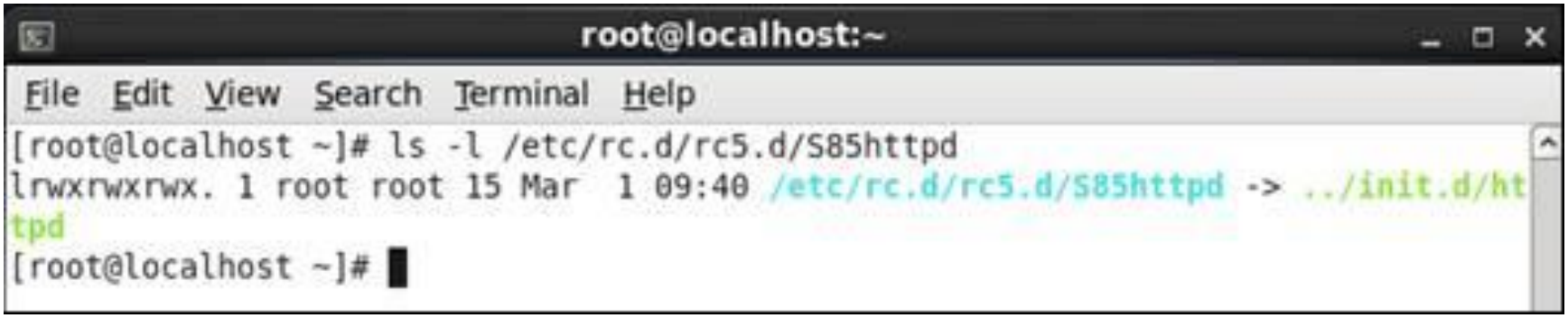
Argument	Function
status	Show whether the service is stopped or the process id (PID) if the service is running. Note: You can also use the --status-all option with the service script to see the status of all daemons.
fullstatus	For the apache web server, displays the URL /server-status.
graceful	For the apache web server, it gracefully restarts the server. If the server is not running, then it is started. Unlike a normal restart, open connections are not aborted.
help	Displays the usage of the script or, for the apache web server, the usage of the httpd daemon.
configtest	Checks the configuration files for correctness. If you modify the configuration files of some services, then you can use this to verify that the changes you made appear to be correct.

Runlevel directories

- Traditional init uses directories like `/etc/rc.d/rc3.d` and `/etc/rc.d/rc5.d` to determine what starts or stops at different runlevels
- To have a service started in a runlevel, create a symbolic link in the appropriate runlevel directory to the "init" script in the `/etc/init.d` directory

Example

- Web server will start in runlevel 5:

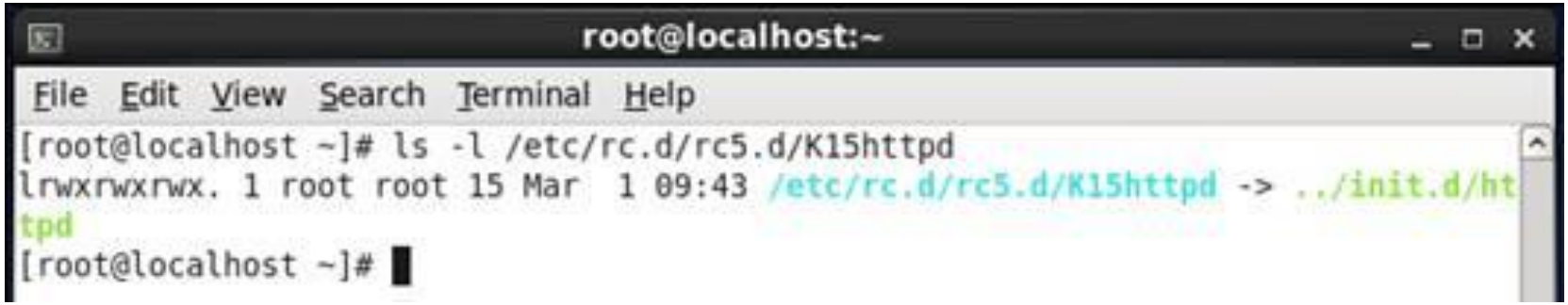


- If link didn't exist, execute:

```
#ln -s /etc/rc.d/init.d/httpd \
 /etc/rc.d/rc5.d/S85httpd
```

Example

- Web server will stop in runlevel 5:



- Look at chkconfig line of script to determine start and stop number

Start and Stop Numbers

- The "S" (start) and "K“ (kill) numbers are important
- Services need to be started and stopped in a specific order
- Look at chkconfig line of script to determine start and stop number:
  - **#**grep chkconfig /etc/init.d/httpd
    # chkconfig: - 85 15

Note: the last number of the chkconfig line is the K number and the second to last number is the S number to place on this script

The chkconfig command

- Displays which services are started and stopped at specific runlevels
  - Displays all services by default:

```
# chkconfig –list
auditd          0:off   1:off   2:on    3:on    4:on    5:on    6:off
crond           0:off   1:off   2:on    3:on    4:on    5:on    6:off
httpd           0:off   1:off   2:off   3:off   4:off   5:off   6:off
iptables        0:off   1:off   2:on    3:on    4:on    5:on    6:off
netconsole      0:off   1:off   2:off   3:off   4:off   5:off   6:off
netfs           0:off   1:off   2:off   3:on    4:on    5:on    6:off
network         0:off   1:off   2:on    3:on
```

- Can just display a single service

  #chkconfig –list http

Turning on services

- Use `chkconfig script on` to turn on service for most run levels:

```
#chkconfig httpd on
[root@localhost ~]# chkconfig --list httpd
httpd           0:off   1:off   2:on    3:on    4:on    5:on    6:off
```

- The runlevel depends on chkconfig line of script
- Use --level to specify which levels

```
# chkconfig --level 24 atd on
# chkconfig --level 35 atd off
```

Additional ckhconfig options

- Use `chkconfig script off` to turn off service
- Use `chkconfig script del` to remove service completely
- Use `chkconfig script add` to add a new service

The /etc/init Directory

- For Debian-derived Linux distributions, the `/etc/init` directory is used to store Upstart scripts
- Runlevels are slightly different:
  - 0, 1 and 6 are the same as traditional init
  - 2 is like runlevel 5
  - 3, 4 and 5 are initially the same as 2

The /etc/init Directory

- To modify runlevels, modify files in `/etc/init` directory
- Example:
  - start on runlevel [2345]
  - stop on runlevel [!2345]
- To disable a service:

```
#echo manual > \
  /etc/init/apache2.override
```

The systemctl command

- The `systemctl` command is used in systems that have Systemd
- To start a service:

```
#systemctl start httpd.service
```

- To stop a service:

```
#systemctl stop httpd.service
```

The systemctl command

- To start check the state of a service:

```
#systemctl status httpd.service
```

- To view all running services:

```
#systemctl -a
```

- To configure a service to start automatically:

```
#systemctl enable httpd.service
```