Tips 1:

Starting and stopping services

Command:

o start a systemd service, executing instructions in the service's unit file, use the start command. If you are running as a non-root user, you will have to use sudo since this will affect the state of the operating system:

$ sudo systemctl start application.service

As we mentioned above, systemd knows to look for *.service files for service management commands, so the command could just as easily be typed like this:

$sudo systemctl start application

If you are unsure whether the service has the functionality to reload its configuration, you can issue the reload-or-restart command. This will reload the configuration in-place if available. Otherwise, it will restart the service so the new configuration is picked up:

$ sudo systemctl reload-or-restart application.service

Tis 2:

Enabling and disabling services

To start a service at boot, use the enable command:

- sudo systemctl enable application.service

This will create a symbolic link from the system's copy of the service file (usually in /lib/systemd/system or /etc/systemd/system) into the location on disk where systemd looks for autostart files (usually /etc/systemd/system/some_target.target.wants. We will go over what a target is later in this guide).

To disable the service from starting automatically, you can type:

- sudo systemctl disable application.service

This will remove the symbolic link that indicated that the service should be started automatically.Keep in mind that enabling a service does not start it in the current session. If you wish to start the service and also enable it at boot, you will have to issue both the start and enable commands.

Tips 3:

Checking the Status of Services

To check the status of a service on your system, you can use the status command:

- systemctl status application.service

This will provide you with the service state, the cgroup hierarchy, and the first few log lines.

For instance, when checking the status of an Nginx server, you may see output like this:

There are also methods for checking for specific states. For instance, to check to see if a unit is currently active (running), you can use the is-active command:

- systemctl is-active application.service

This will return the current unit state, which is usually active or inactive. The exit code will be “0” if it is active, making the result simpler to parse in shell scripts.

To see if the unit is enabled, you can use the is-enabled command:

- systemctl is-enabled application.service

This will output whether the service is enabled or disabled and will again set the exit code to “0” or “1” depending on the answer to the command question.

A third check is whether the unit is in a failed state. This indicates that there was a problem starting the unit in question:

- systemctl is-failed application.service

This will return active if it is running properly or failed if an error occurred. If the unit was intentionally stopped, it may return unknown or inactive. An exit status of “0” indicates that a failure occurred and an exit status of “1” indicates any other status.

System State Overview

The commands so far have been useful for managing single services, but they are not very helpful for exploring the current state of the system. There are a number of systemctl commands that provide this information.

Tips 4:

Listing Current Units

To see a list of all of the active units that systemd knows about, we can use the list-units command:

- systemctl list-units

This will show you a list of all of the units that systemd currently has active on the system. The output will look something like this:

```
Output

UNIT                         LOAD   ACTIVE SUB     DESCRIPTION

atd.service                    loaded active running ATD daemon

avahi-daemon.service                 loaded active running Avahi mDNS/DNS-SD Stack

dbus.service                    loaded active running D-Bus System Message Bus

dcron.service                   loaded active running Periodic Command Scheduler

dkms.service                    loaded active exited  Dynamic Kernel Modules System

getty@tty1.service                  loaded active running Getty on tty1

. . .
```

The output has the following columns:

- **UNIT**: The systemd unit name
- **LOAD**: Whether the unit's configuration has been parsed by systemd. The configuration of loaded units is kept in memory.
- **ACTIVE**: A summary state about whether the unit is active. This is usually a fairly basic way to tell if the unit has started successfully or not.
- **SUB**: This is a lower-level state that indicates more detailed information about the unit. This often varies by unit type, state, and the actual method in which the unit runs.
- **DESCRIPTION**: A short textual description of what the unit is/does.

Since the list-units command shows only active units by default, all of the entries above will show loaded in the LOAD column and active in the ACTIVE column. This display is actually the default behavior of systemctl when called without additional commands, so you will see the same thing if you call systemctl with no arguments:

- Systemctl

We can tell systemctl to output different information by adding additional flags. For instance, to see all of the units that systemd has loaded (or attempted to load), regardless of whether they are currently active, you can use the --all flag, like this:

- systemctl list-units --all

This will show any unit that systemd loaded or attempted to load, regardless of its current state on the system. Some units become inactive after running, and some units that systemd attempted to load may have not been found on disk.

You can use other flags to filter these results. For example, we can use the --state= flag to indicate the LOAD, ACTIVE, or SUB states that we wish to see. You will have to keep the --all flag so that systemctl allows non-active units to be displayed:

- systemctl list-units --all --state=inactive

Another common filter is the --type= filter. We can tell systemctl to only display units of the type we are interested in. For example, to see only active service units, we can use:

- systemctl list-units --type=service