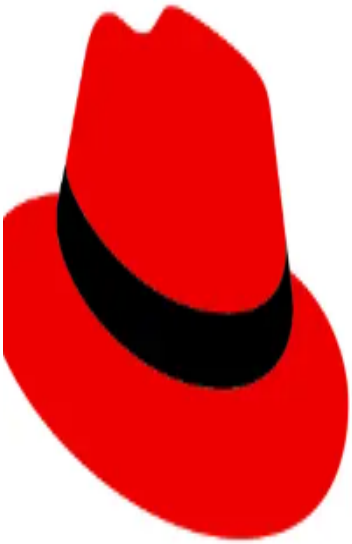

Red Hat Enterprise Linux 9

RHCSA-SA1 LAB-Book

Chapter 9 - Controlling Services and Daemons

systemd: System start up and server processes are managed by the systemd System and Service Manager.

This program provides a method for activating system resources, server daemons, and other processes, both time and on a running system.

Daemons: Processes that wait or run in the background performing various tasks.

Generally, daemons start automatically at boot time and continue to run until shutdown or until they are manually stopped. By convention, the names of many daemons' programs end in the letter "d".

Socket: For connection, a daemon used a socket. This is the primary communication channel with local or remote clients. Sockets may be created by daemons or may be separated from the daemon and be created by another process, such as systemd. The socket is passed to the daemon when a connection is established by the client.

Service: A service often refers to one or more daemons, but starting or stopping a service may instead make a one-time change to the state of the system, which does not involve leaving a daemon process running afterward (called one-shot).

History:

For many years, process ID of Linux and UNIX systems has been the init process. This process was responsible for activating others services on the system and is the origin of the term "init system".

In Red Hat Enterprise, Linux 7, process ID is systemd, the new inti system. A few of the new features provided by systemd include:

Parallelization capabilities, which increase the boot speed of a system.

On – demand starting pf a daemon without requiring a separate service.

Automatically service dependency management, which can prevent long timeouts, such as by not starting a network service when the network is not available.

A method of tracking related processes together by using Linux control groups.

systemctl and systemd units:

The systemctl command is used to manage different types of systemd objects, called units. A list of available unit types can be displayed with systemctl -t help.

IMP: The systemctl may abbreviate or “ellipsize” unite names, process tree entries, and unit descriptions unless run with the -l option.

Some common unit types are listed below:

Service units have a. service extension and represent system services. This type of unit is used to start frequently accessed daemons, such as a web server.

Socket units have a. socket extension and represent inter-process communication (IPC) sockets. Control of the socket will be passed to a daemon or newly started service when a client connection is made. Socket units are used to delay the start of a service at boot time and to start less frequently used services on demand. These are similar in principle to service which use the ximetd surperserver to start on demand.

Service States:

The status of a service can be viewed with systemctl status name type. If the unit type is not provided, systemctl will show the status of a service unit, if one exists.

Several keyboard indicating the state of the service can be found in the status output:

Keyword	Description
Loaded	Unit configuration file has been processed.
Active (running)	Running with one or more continuing processes.
Active (exited)	Successfully completed a one -time configuration.
Active (waiting)	Running but waiting for an event.
Inactive	Not running.
Enabled	Will be started at boot time.
Disabled	Will not be started at boot time.
Static	Can not be enabled, but may be started by an enabled unit automatically.

Practical on listing unit files with systemctl:

1. Query the state of all units to verify a system start-up.

 systemctl.

2. Query the state of only the service units.

 systemctl –type=service

3. Investigate any units which are in a failed or maintenance state. Optionally, add the -l option to show the full output.

 systemctl status rmgd.service -l

4. The status argument may also be used to determine if a particular unit is active and show if the unit is enabled to start at bottom time. Alternative commands can be also easily showing the active and enabled states:

- systemctl is-active sshd.
- systemctl is-enabled sshd.

5. List the active state of loaded units. Optionally, limit the type of unit. The –all option will add inactive units.

- systemctl list-units – -type=service
- systemctl list-units – - type=service - - all

6. View the enabled and disabled settings for all units. Optionally, limit the type of unit.

- systemctl list-unit-files - -type=service.

7. View only failed services.

- systemctl - - failed - - type=service

<u>Practical on identify the status of systemd units:</u>

1. List all service units on the system.

- systemctl list-units - - type=service

2. List all socket units, active and inactive, on the system.

- Systemctl list-units - -type=socket - - all

3. Explore the status of the chronyd service. This service is used for network time synchronization (NTP).

a) Display the status of the chronyd service. Note the

process ID of any active daemons.

- ➔systemctl status chronyd.

b) Confirm that the listed daemons are running.

- ➔ps -p PID.

4. Explorer the status of the sshd service. This service is used for secure encrypted communication between systems.

a) Determine if the sshd service is enabled to start at system boot.

- ➔systemctl is-enabled sshd

b) Determine if the sshd service is active without displaying all of the status information.

- ➔systemctl is-active sshd.

c) Display the status of the sshd service.

- ➔systemctl status sshd.

5. List the enabled or disabled states of all service units.

➔systemctl list-unit-files - -type=service.

<u>Controlling System Services:</u>

<u>Starting and stopping system daemons on a running system:</u>

Changing to a configuration file or other updates to a service may require that the service be restarted.

A service that is no longer used may be stopped before removing the software.

A service that is no frequently used may be manually started by an administrator only when it is needed.

Practical are as below:

1. View the status of a service.

 ➔systemctl status sshd. Service

2. Verify that the process is running.

 ➔ps -up PID

3. Stop the service and verify the status.

 ➔systemctl stop sshd. service

 s➔systemctl status sshd.service

4. Start the service and view the status. The process ID has charged.

 ➔systemctl start sshd. service

 ➔systemctl status sshd. service

5. Stop then start, the service in a single command.

 ➔systemctl restart sshd. service

 ➔systemctl status sshd. service

6. Issue instructions for a service to read and reload to configuration file without a complete stop and start. The process ID will not change.

➔systemctl reload sshd. service

➔systemctl status sshd. service

Unit Dependencies:

Services may be started as dependencies of other services.

If a socket unit is enabled and the service unit with the same name is not, the service will automatically be started when a request is made on the network socket.

Services may also be triggered by path units when a file system condition is met.

To completely stop printing services on system, stop all three units. Disabling the service will disable the dependencies.

For example, a file placed into the print spool directory will cause the cups print service to be started if it not running.

Practical:

systemctl stop cups.service

systemctl list – dependencies UNIT:

This command can be used to print out a tree of what other units must be started if the specified unit is started.

Depending on the exact dependency, the other unit may need to be running before or after the specified unit starts.

The - - reverse option to this command will show what units need to have the specified unit started in order to run.

Enabling system daemons to start or stop at boot:

Starting a service on a running system does not guarantee that the service will be started when the system reboots.

Similarly, stopping a service on a running system will not keep it from starting again when the system reboots.

Services are started at boot time when links are created in the appropriate systemd configuration directories.

These links are created and removed with systemctl command.

Practical:

1. View the status of a service.

 ➔systemctl status sshd.service

2. Disable the service and verify the status. Note that disabling a service does not stop the service.

 ➔systemctl disable sshd.service

 ➔systemctl status sshd.service

3. Enable the service and verify the status.

 ➔systemctl enable sshd. service

 ➔systemctl is-enabled sshd. Service

SUMMARY OF systemctl:

Service can be started and stopped on a running system and enabled or disabled for automatic start at boot time

TASK	COMMAND
View detailed information about a unit state.	systemctl status UNIT
Stop a service on a running system.	systemctl stop UNIT
Start a service on a running system.	systemctl start UNIT
Restart a service on a running system.	systemctl restart UNIT
Reload configuration file of a running service	systemctl reload UNIT
Completely disabled a service from being started, both manually and a boot.	systemctl mask UNIT
Make a masked service available.	systemctl unmask UNIT
Configure a service to start at boot time.	Systemctl enable UNIT
Disable a service from starting a boot time.	systemctl disable UNIT.
List units which are required and wanted by the specified unit.	systemctl list-dependencies UNIT.

Practical on using systemctl to manage services:

1. Observe the results of systemctl restart and systemctl reload commons.

a. Display the status of the sshd service. Note the process ID of the daemon.

- ➔systemctl status sshd.

b. Restart the sshd service and view the status. The process ID of the daemon has changed.

- ➔sudo systemctl restart sshd.
- ➔systemctl status sshd.

c. Reload the sshd service and view the status. The process

ID of the daemon has not changed and connections have not been interrupted.

- ➔sudo systemctl reload sshd.
- ➔systemctl status sshd.

2. Verify that the chronyd service is running,

- ➔systemctl status chronyd

3. Stop the chronyd service and view the status.

- ➔sudo systemctl stop chronyd
- ➔systemctl status chronyd

4. Determine if the chronyd service enabled to start at system boot.

- ➔systemctl is-enabled chronyd.

5. Reboot the system, then view the status of the chronyd service.

- ➔systemctl status chronyd.

6. Disable the chronyd service so that it does not start at system boot then view the status of the service.

- ➔sudo systemctl disable chronyd
- ➔systemctl status chronyd

7. Reboot the system then view the status of the chronyd service.

 ➔systemctl status chronyd.

END