

SysVinit - Directory Structures

Generally, you will find in the **/etc** directory some symlinks to stuff that is actually in the **/etc/rc.d/** directory. This can cause some confusion, since we have some other symlink stuff for backward compatibility for systems that once supported Upstart but no longer do so. This writing ignores all of that and sticks to CentOS 5.5

/etc/init.d is a symlink to the directory **/etc/rc.d/init.d** and the same with **/etc/rc#.d** linking to **/etc/rc.d/rc#.d**, also the same with scripts **rc**, **rc.local** and **rc.sysinit**, who's actual locations is also in the **/etc/rc.d/** directory as well. Even though you will often see these in **/etc**. Here is where they actually live:

```
/etc/rc.d/init.d/  
/etc/rc.d/rc0.d/  
/etc/rc.d/rc1.d/  
...and so on....  
/etc/rc.d/rc5.d/  
/etc/rc.d/rc6.d/  
/etc/rc.d/rc  
/etc/rc.d/rc.local  
/etc/rc.d/rc.sysinit
```

Service's scripts are in /etc/rc.d/init.d/ (often accessible via the symlink **/etc/init.d/**)

- Each service managed by SystemVinit needs a script in **/etc/rc.d/init.d/**
- Common elements to the scripts in **/etc/rc.d/init.d/<servicename>** are the top several lines, beginning in a prelude declaring the script processor (as in **#!/bin/bash**); followed by a line with name and brief description; another with **chkconfig** default runlevels the service should be started, and the start and stop priority levels.
- If default is to not be started in any runlevels, a "-" should be used in place of the runlevels list.
- Another entry contains service description (used by **ntsysv**)
- Finally the general functions container for **init.d** scripts is defined (usually **/etc/init.d/functions**), followed by lines setting and **ENVVARS** and functions for the service, (much like in **bashrc** does for it's purpose).

For example, the beginning of **/etc/rc.d/init.d/kudzu** has these line common to SysVinit scripts:

```
#!/bin/bash  
# kudzu          This scripts runs the kudzu hardware probe.  
# chkconfig: 345 05 95  
# description:   This runs the hardware probe, and optionally configures \  
#               changed hardware.  
# Source function library.  
. /etc/init.d/functions
```

Says that the script should be started in levels 3, 4, and 5, start priority 5, stop priority 95

/etc/rc.d/init.d directory contents:

```
/etc/rc.d/init.d/acpid  
/etc/rc.d/init.d/anacron  
...  
/etc/rc.d/init.d/ypbind  
/etc/rc.d/init.d/yum-updatesd
```

Files in the **/etc/rc.d/rc#.d** directories are symlinks to the actual scripts for all of SysVinit's managed programs in **/etc/rc.d/init.d**. For example, **/etc/rc.d/rc0.d/K99cpuspeed** links to **/etc/rc.d/init.d/cpuspeed**

With those links, the naming convention of K or S means "kill" or "start" and the number (like 99) indicates the numerical order that it is executed in that runlevel's directory, when that runlevel starts. This way, it is directed that things are stopped and started in the proper order.

As an example, here is a sample of some filenames in **/etc/rc.d/rc3.d/**

K88wpa_supplicant	S02lvm2-monitor
K89netplugd	S04readahead_early
K89rdisc	S05kudzu
K91capi	S08ip6tables
K99readahead_later	S08iptables
S00microcode_ctl	S08mcstrans

rc.local is to execute commands during the startup without needing symlinks. "Local system initialization script"
S99local -> softlink for /etc/rc.local in 2,3,4 and 5 runlevels
You can optionally have a similar shutdown items script in **/etc/rc.d/rc.local_shutdown**
rc.sysinit seems to be redhat specific and is executed very early in the process while rc.local is executed later.
rc is typically not used by linux distributions but is used in BSD

The rc stands for "run commands"; runcom (as in .cshrc or /etc/rc) comes from the runcom facility from the MIT CTSS system, ca. 1965. From Kernighan and Ritchie, as told to Vicki Brown: "There was a facility that would execute a bunch of commands stored in a file; it was called runcom for "run commands", and the file began to be called "a runcom". rc in Unix is a fossil from that usage."

The idea of having the command processing shell be an ordinary slave program came from the Multics design, and a predecessor program on CTSS by Louis Pouzin called RUNCOM. The first time I remember the name "shell" for this function was in a Multics design document by Doug Eastwood (of BTL). Commands that return a value into the command line were called "evaluated commands" in the original Multics shell, which used square brackets where Unix uses backticks.

/etc/inittab Main config file for SysVinit

- Specifies runlevels, scripts to run when certain runlevels are selected, and items to respawn (getty).
- Syntax for an entry in inittab: id:runlevels:action:process.
- First is a unique arbitrary identifier, second indicates what runlevels invoke the command, third is how to handle this entry (like execute command once or respawn whenever it exits, fourth is the command and it's arguments
- x:5:respawn:/etc/X11/prefdm -nodaemon Runlevel 5, specify default login screen for X11
- 3:2345:respawn:/sbin/mingetty tty3 Virtual terminal 3, available for runlevels 2 through 5

Sample /etc/inittab (truncated):

```
# Set the default runlevel to three - points to line below "l3:3:wait:/etc/rc.d/rc 3"
id:3:initdefault:
```

```
# Execute /etc/rc.d/rc.sysinit when the system boots
# starts network, establishes mounted systems, starts SELinux, encryption
si:S:sysinit:/etc/rc.d/rc.sysinit
```

```
# Run /etc/rc.d/rc with the runlevel as an argument - e.g., a 5 points it to /etc/rc5.d/
# Runlevels are designated in /etc/rc.d/
l0:0:wait:/etc/rc.d/rc 0
l1:1:wait:/etc/rc.d/rc 1
...
l5:5:wait:/etc/rc.d/rc 5
l6:6:wait:/etc/rc.d/rc 6
```

```
# Executed when we press ctrl-alt-delete
ca::ctrlaltdel:/sbin/shutdown -t3 -rf now
```

```
# Startagetty for virtual consoles 1 through 6
c1:12345:respawn:/sbin/agetty 38400 tty1
c2:12345:respawn:/sbin/agetty 38400 tty2
...
c6:45:respawn:/sbin/agetty 38400 tty6
```

Default runlevel is determined, then scripts in appropriate **/etc/rc.d/rcX/** directory are run.

When SysVinit is instructed to change runlevels, it reads inittab for what **/etc/rc.d** directory belongs to that runlevel.

The **rc.d** directory contains the daemon scripts which run at boot and when switching runlevels.

Contents in **/etc/rc.d/rcX/** directories are just symlinks to the files in **/etc/rc.d/init.d/** and are named to either start with an S (start) or a K (kill), in order of the number to be processed

For example: take a symlink **S45dhcpd** in **/etc/rc.d/rc3/** - This means the **/etc/rc.d/init.d/dhcpd** script will be 45th in order to start for that runlevel directory containing this symlink- in this case runlevel 3.

Some types of Linux using SysVinit don't even use this system of symlinks. Slackware uses something similar to BSD where all directives for a runlevel are only put in a runlevel script.

Runlevel Service Management Tools -SysVinit initscript utilities

service <servicename> [start | stop | restart | status | list] Activate, etc., a daemon in current runlevel

OR Go to **/etc/rc.d/init.d/** directory and type **./<servicename> start**.

init OR **telinit [0-6]** switches to the specified runlevel

runlevel tells you your runlevel- returns two numbers - 3 5 means that current runlevel is 5 and previous was 3.

chkconfig --list gives you this type of output:

```
[root@localhost rc.d]# chkconfig --list
NetworkManager 0:off 1:off 2:off 3:off 4:off 5:off 6:off
acpid           0:off 1:off 2:on  3:on  4:on  5:on  6:off
anacron         0:off 1:off 2:on  3:on  4:on  5:on  6:off
apmd            0:off 1:off 2:on  3:on  4:on  5:on  6:off
atd             0:off 1:off 2:off 3:on  4:on  5:on  6:off
auditd          0:off 1:off 2:on  3:on  4:on  5:on  6:off
autofs          0:off 1:off 2:off 3:on  4:on  5:on  6:off
avahi-daemon    0:off 1:off 2:off 3:on  4:on  5:on  6:off
```

chkconfig --list <servicename> will output the same on one line for that one service

chkconfig --list | grep <servicename> might be more helpful to list multiple matches (e.g. "avahi-" services)

chkconfig --level 35 <servicename> off | on | reset | resetpriorities - affects service in runlevels 3 and 4

chkconfig --level <servicename> on -turns service on in levels 2-5, if 'off' affects 0-6

chkconfig --add OR **--del** adds or removes scripts from **/etc/rc.d/init.d/**

chkconfig --override <servicename> - files in **/etc/chkconfig.d/<servicename>** can override init service scripts

ntsysv is just a Red Hat TUI interface to turn off and on services in the currently active runlevel. Debian has **rcconf**

ntsysv --runlevel 35 (OR **--level 35**) manages services on levels 3 and 5

redhat-config-services or **system-config-services** - graphical Services Configuration Tool

/sbin/telinit is linked to **/sbin/init** - takes a one-character argument (0-6 for switching runlevels, s/S/1 all work for single user mode, U/u to resart current runlevel init scrips without checking inittab; Q/q to do so forcing checking inittab. The **init** binary checks if it is **init** or **telinit** by looking at its process id; the real **init**'s process id is always 1.

- **ls /etc/init.d/** will also list all of the currently available service files

- Scripts to stop and start processes can be used as an alternative to running **kill**.

- neither **ntsysv** or **chkconfig** starts or stops services- only dictates runlevel. The service command does that.

- **xinetd** services are immediately affected by **ntsysv**, unlike others

SysVinit Runlevel	Systemd Target	Description
0	<i>poweroff.target</i>	<i>Halts the system</i>
1	<i>rescue.target</i>	<i>Single-user mode (everything mounted, minimal services)</i>
2	<i>multi-user.target</i>	<i>Multiuser mode without networking</i>
3	<i>multi-user.target</i>	<i>Multiuser mode with networking</i>
4	<i>multi-user.target</i>	<i>User configurable</i>
5	<i>graphical.target</i>	<i>Used for the GUI (X11 multiuser mode)</i>
6	<i>reboot.target</i>	<i>Reboots the system</i>

systemd's emergency.target

- Is like **init=/bin/sh** on the kernel command line

- Has no corresponding **sysvinit** runlevel- would just boot your machine to a shell with really nothing started.

- You get a shell, but almost nothing else (except for **systemd** in the background)

- No services are started, no mount points mounted, no sockets established.

- Useful for running specific scripts which could then be started independently.

- Allows booting bit-by-bit, starting the various services and other units step-by-step manually.

In **SysVinit**, services can define arbitrary commands. Examples would be service **iptables** **panic**, or service **httpd** **graceful**. Native **systemd** services do not have this ability. Any service that defines an additional command in this way would need to define some other, service-specific, way to accomplish this task when writing a native **systemd** service definition. Check the package-specific release notes for any services that may have done this.

Example systemd Unit Script Contents

It is *not* advised to edit unit scripts in **/usr/lib/systemd/system/** so they remain default/as-installed by packages. For custom changes, make copy to edit in **/etc/systemd/system** - this directory overrides those defaults for you.

Example Service Unit Script - /usr/lib/systemd/system/httpd.service contents:

```
[Unit]
Description=The Apache HTTP Server
After= network.target remote-fs.target nss-lookup.target

[Service]
Type=notify
EnvironmentFile=/etc/sysconfig/httpd
ExecStart=/usr/sbin/httpd $OPTIONS -DFOREGROUND
ExecReload=/usr/sbin/httpd $OPTIONS -k graceful
ExecStop=/bin/kill -WINCH ${MAINPID}
KillSignal=SIGCONT
PrivateTmp=true

[Install]
WantedBy=multi-user.target
```

Example Target Unit Script - /usr/lib/systemd/system/multi-user.target contents:

```
[Unit]
Description= Multi-User System
Documentation=man:systemd.special(7)
Requires=basic.target
Conflicts=rescue.service rescue.target
After=basic.target rescue.service rescue.target
AllowIsolate=yes
[Install]
Alias=default.target
```

*"After" is what is loaded after this finishes activating
"Requires" should be what needs to be loaded before*

Example Custom Mount Scripts - /etc/systemd/system/lvdisk.mount and lvdisk.automount

Needs a mount file and automount file with a matching name (mydisk5.automount for mydisk5.mount)
This is the way future versions of RHEL will likely do automount instead of the old /etc/fstab method

vim /etc/systemd/system/lvdisk.mount

```
[Unit]
Description= Example test mount
[Mount]
what = /dev/vgdisk/lvdisk
where = /lvdisk
type = xfs
[Install]
WantedBy=multi-user.target
```

vim /etc/systemd/system/lvdisk.automount

```
[Unit]
Description= Example test automount
[Automount]
where = /lvdisk
[Install]
WantedBy=multi-user.target
```

Test it out - systemctl enable lvdisk.automount; systemctl start lvdisk.automount; mount | grep lvdisk

Contents of the systemd Package

Installed programs: bootctl, busctl, coredumpctl, halt, hostnamectl, init, journalctl, kernel-install, localectl, loginctl, machinectl, networkctl, poweroff, reboot, runlevel, shutdown, systemctl, systemd-analyze, systemd-ask-password, systemd-cat, systemd-cgls, systemd-cgtop, systemd-delta, systemd-detect-virt, systemd-escape, systemd-hwdb, systemd-inhibit, systemd-machine-id-setup, systemd-mount, systemd-notify, systemd-nspawn, systemd-path, systemd-resolve, systemd-run, systemd-socket-activate, systemd-stdio-bridge, systemd-tmpfiles, systemd-tty-ask-password-agent, telinit, timedatectl, and udevadm

Installed libraries: libnss_myhostname.so.2, libnss_mymachines.so.2, libnss_resolve.so.2, libnss_systemd.so.2, libsystemd.so, libsystemd-shared-231.so, and libudev.so

Installed directories: /etc/binfmt.d, /etc/init.d, /etc/kernel, /etc/modules-load.d, /etc/sysctl.d, /etc/systemd, /etc/tmpfiles.d, /etc/udev, /etc/xdg/systemd, /lib/systemd, /lib/udev, /usr/include/systemd, /usr/lib/binfmt.d, /usr/lib/kernel, /usr/lib/modules-load.d, /usr/lib/sysctl.d, /usr/lib/systemd, /usr/lib/tmpfiles.d, /usr/share/doc/systemd-234, /usr/share/factory, /usr/share/systemd, /var/lib/systemd, and /var/log/journal

bootctl	Query the firmware and boot manager settings
busctl	Review logs and monitor the D-Bus bus
coredumpctl	Retrieve coredumps from the systemd Journal
halt	Normally invokes shutdown with the -h option, except when already in run-level 0, then it tells the kernel to halt the system; it notes in the file /var/log/wtmp that the system is being brought down
hostnamectl	Query and change the system hostname and related settings
init	The first process to be started when the kernel has initialized the hardware which takes over the boot process and starts all the processes it is instructed to
journalctl	Query the contents of the systemd Journal
kernel-install	Add and remove kernel and initramfs images to and from /boot
localectl	Query and change the system locale and keyboard layout settings
loginctl	Review logs and control the state of the systemd Login Manager
machinectl	Review logs and control the state of the systemd Virtual Machine and Container Registration Manager
networkctl	Review logs and state of the network links as seen by systemd-networkd
poweroff	Tells the kernel to halt the system and switch off the computer (see halt)
reboot	Tells the kernel to reboot the system (see halt)
runlevel	Reports the previous and the current run-level, as noted in the last run-level record in /var/run/utmp
shutdown	Brings the system down in a secure way, signaling all processes and notifying all logged-in users
systemctl	Review logs and control the state of the systemd system and service manager
systemd-analyze	Determine system boot-up performance of the current boot
systemd-ask-password	Query a system password or passphrase from the user, using a question message specified on the command line
systemd-cat	Connect STDOUT and STDERR of a process with the Journal
systemd-cgls	Recursively shows the contents of the selected Linux control group hierarchy in a tree
systemd-cgtop	Shows the top control groups of the local Linux control group hierarchy, ordered by their CPU, memory and disk I/O load
systemd-delta	Identify and compare configuration files in /etc that override default counterparts in /usr
systemd-detect-virt	Detects execution in a virtualized environment
systemd-escape	Escape strings for inclusion in systemd unit names
systemd-hwdb	Manage hardware database (hwdb)
systemd-inhibit	Execute a program with a shutdown, sleep or idle inhibitor lock taken

systemd-machine-id-setup	Used by system installer tools to initialize the machine ID stored in /etc/machine-id at install time with a randomly generated ID
systemd-mount	A tool to temporarily mount or auto-mount a drive.
systemd-notify	Used by daemon scripts to notify the init system about status changes
systemd-nspawn	Run a command or OS in a light-weight namespace container
systemd-path	Query system and user paths
systemd-resolve	Resolve domain names, IPV4 and IPV6 addresses, DNS resource records, and services
systemd-run	Create and start a transient .service or a .scope unit and run the specified command in it
systemd-socket-activate	A tool to listen on socket devices and launch a process upon connection.
systemd-tmpfiles	Creates, deletes and cleans up volatile and temporary files and directories, based on the configuration file format and location specified in tmpfiles.d directories
systemd-tty-ask-password-agent	Used to list or process pending systemd password requests
telinit	Tells init which run-level to change to
timedatectl	Query and change the system clock and its settings
udevadm	Generic Udev administration tool: controls the udevd daemon, provides info from the Udev database, monitors uevents, waits for uevents to finish, tests Udev configuration, and triggers uevents for a given device
libsystemd	systemd utility library
libudev	A library to access Udev device information

-- from <http://www.linuxfromscratch.org/lfs/view/systemd/chapter06/systemd.html>

You may have noticed that installed items listed contains telinit, and an /etc/init.d directory. According to the Fedora Wiki, the 'service' and 'chkconfig' commands will (surprisingly) mostly continue to work as expected in the systemd world. Presumably, this would be for backward compatibility support for old scripts, etc.

Target unit directories hold symlinks to the real unit files like this:	/etc/systemd/system/XXXXXX.target.wants/bluetooth.service
Those symlinks point to the actual service (etc) unit files that reside here:	/usr/lib/systemd/system/bluetooth.service
The default.target file here is a target/runlevel symlink:	/etc/systemd/system/default.target
And the default.target symlink points to the actual target here:	/usr/lib/systemd/system/XXXXXX.target

Running **systemctl isolate graphical.target** will not affect the default.target symlink, and merely switches the current runlevel (use **set-default**).

Running **systemctl disable myservice** basically does the same as **rm '/etc/systemd/system/multi-user.target.wants/service.myservice'**

Running **systemctl enable myservice** basically does the same as

ln -s '/usr/lib/systemd/system/myservice.service' '/etc/systemd/system/multi-user.target.wants/service.myservice'

The "**target.wants**" directories in **/usr/lib/systemd/system/** hold symlinks to the corresponding runlevel's unit files just like init's **/etc/rc.d/rc#.d/**

A target is itself a unit file, manages other unit files. Defaults are multi-user.target, graphical.target, rescue.target, emergency.target, poweroff.target, and reboot.target

Systemd Command Examples (verbose version)

Is systemd installed on your system?

systemd-run --version

systemd 215 +PAM +AUDIT +SELINUX +IMA +SYSVINIT +LIBCRYPTSETUP +GCRYPT +ACL +XZ -SECCOMP -
APPARMOR

Where the binaries and libraries of systemd and systemctl?

whereis systemd

systemd: /usr/lib/systemd /etc/systemd /usr/share/systemd /usr/share/man/man1/systemd.1.gz

whereis systemctl

systemctl: /usr/bin/systemctl /usr/share/man/man1/systemctl.1.gz

Is it running?

ps -eaf | grep [s]ystemd

(-e) all Processes, (-a) all processes except session leaders, (-f) full format listing

```
root      1    0  0 16:27 ?        00:00:00 /usr/lib/systemd/systemd --switched-root --system --deserialize 23
root     444    1  0 16:27 ?        00:00:00 /usr/lib/systemd/systemd-journald
root     469    1  0 16:27 ?        00:00:00 /usr/lib/systemd/systemd-udev
root     555    1  0 16:27 ?        00:00:00 /usr/lib/systemd/systemd-logind
dbus     556    1  0 16:27 ?        00:00:00 /bin/dbus-daemon --system --address=systemd: --nofork --nopidfile --
systemd-activation
```

Check if a Unit (cron.service) is enabled or not?

systemctl is-enabled crond.service

enabled

Check whether a Unit or Service is running or not?

systemctl status firewalld.service

firewalld.service - firewalld - dynamic firewall daemon

Loaded: loaded (/usr/lib/systemd/system/firewalld.service; enabled)

Active: active (running) since Tue 2015-04-28 16:27:55 IST; 34min ago

Main PID: 549 (firewalld)

CGroup: /system.slice/firewalld.service

└─549 /usr/bin/python -Es /usr/sbin/firewalld --nofork --nopid

Apr 28 16:27:51 domain2 systemd[1]: Starting firewalld - dynamic firewall daemon...

List all the available units.

systemctl list-unit-files

UNIT FILE	STATE
proc-sys-fs-binfmt_misc.automount	static
dev-hugepages.mount	static
dev-mqueue.mount	static
proc-sys-fs-binfmt_misc.mount	static
sys-fs-fuse-connections.mount	static
sys-kernel-config.mount	static
sys-kernel-debug.mount	static
tmp.mount	disabled
brandbot.path	disabled

List all running units.

systemctl list-units

UNIT	LOAD	ACTIVE	SUB	DESCRIPTION
proc-sys-fs-binfmt_misc.automount	loaded	active	waiting	Arbitrary Executable File Formats File System
sys-devices-pc...0-1:0:0:0-block-sr0.device	loaded	active	plugged	VBOX_CD-ROM
sys-devices-pc...:00:03.0-net-enp0s3.device	loaded	active	plugged	PRO/1000 MT Desktop Adapter
sys-devices-pc...00:05.0-sound-card0.device	loaded	active	plugged	82801AA AC'97 Audio Controller
sys-devices-pc...:0:0-block-sda-sda1.device	loaded	active	plugged	VBOX_HARDDISK
sys-devices-pc...:0:0-block-sda-sda2.device	loaded	active	plugged	LVM PV Qzyo3l-qYaL-uRUa-Cjuk-pljo-qKtX
sys-devices-pc...0-2:0:0:0-block-sda.device	loaded	active	plugged	VBOX_HARDDISK

List all failed units.

systemctl --failed

UNIT	LOAD	ACTIVE	SUB	DESCRIPTION
------	------	--------	-----	-------------

kdump.service	loaded	failed	failed	Crash recovery kernel arming
---------------	--------	--------	--------	------------------------------

LOAD = Reflects whether the unit definition was properly loaded.

ACTIVE = The high-level unit activation state, i.e. generalization of SUB.

SUB = The low-level unit activation state, values depend on unit type.

1 loaded units listed. Pass --all to see loaded but inactive units, too.

Analyze the systemd boot process.

systemd-analyze

Startup finished in 487ms (kernel) + 2.776s (initrd) + 20.229s (userspace) = 23.493s

Analyze time taken by each process at boot.

systemd-analyze blame

8.565s mariadb.service

7.991s webmin.service

6.095s postfix.service

4.311s httpd.service

3.926s firewalld.service

3.780s kdump.service

3.238s tuned.service

1.712s network.service

1.394s lvm2-monitor.service

1.126s systemd-logind.service

Analyze critical chain at boot.

Accepts services (.service), mount point (.mount), sockets (.socket), devices (.device) as units.

systemd-analyze critical-chain

The time after the unit is active or started is printed after the "@" character.

The time the unit takes to start is printed after the "+" character.

multi-user.target @20.222s

└─mariadb.service @11.657s +8.565s

└─network.target @11.168s

└─network.service @9.456s +1.712s

└─NetworkManager.service @8.858s +596ms

└─firewalld.service @4.931s +3.926s

└─basic.target @4.916s

└─sockets.target @4.916s

└─dbus.socket @4.916s

└─sysinit.target @4.905s

└─systemd-update-utmp.service @4.864s +39ms

└─auditd.service @4.563s +301ms

└─systemd-tmpfiles-setup.service @4.485s +69ms

└─rhel-import-state.service @4.342s +142ms

└─local-fs.target @4.324s

└─boot.mount @4.286s +31ms

└─systemd-fsck@dev-disk-by\x2duuid-79f594ad\x2da332\x2d4730\x2dbb5f\x2d85d19608096

└─dev-disk-by\x2duuid-79f594ad\x2da332\x2d4730\x2dbb5f\x2d85d196080964.device @4

How to "active" a service and enable or disable a service at boot time (auto start service at boot).

systemctl [is-active | enable | disable] httpd.service

How to mask (making it impossible to start) or unmask a service (httpd.service).

systemctl [mask | unmask] httpd.service

List all services (including enabled and disabled).

systemctl list-unit-files --type=service

UNIT	FILE	STATE
------	------	-------

arp-ethers.service		disabled
--------------------	--	----------


```

auditd.service                enabled
autovt@.service               disabled
blk-availability.service       disabled
brandbot.service              static
collectd.service              disabled
console-getty.service          disabled
console-shell.service          disabled
cpupower.service              disabled
crond.service                  enabled
dbus-org.fedoraproject.FirewallD1.service enabled
....

```

Start, restart, stop, reload and check the status of a service (httpd.service)

When we use commands like start, restart, stop and reload with systemctl, we will not get any output on the terminal, only status command will print the output.

systemctl [start | restart | stop | reload | status] httpd.service

httpd.service - The Apache HTTP Server

Loaded: loaded (/usr/lib/systemd/system/httpd.service; enabled)

Active: active (running) since Tue 2015-04-28 17:21:30 IST; 6s ago

Process: 2876 ExecStop=/bin/kill -WINCH \${MAINPID} (code=exited, status=0/SUCCESS)

Main PID: 2881 (httpd)

Status: "Processing requests..."

CGroup: /system.slice/httpd.service

└─2881 /usr/sbin/httpd -DFOREGROUND

└─2888 /usr/sbin/httpd -DFOREGROUND

Apr 28 17:21:30 domain2 systemd[1]: Starting The Apache HTTP Server...

Apr 28 17:21:30 domain2 httpd[2881]: AH00558: httpd: Could not reliably determine the server's fully q...ssage

Apr 28 17:21:30 domain2 systemd[1]: Started The Apache HTTP Server.

How to a Kill a service using systemctl command.

systemctl kill httpd

systemctl status httpd

httpd.service - The Apache HTTP Server

Loaded: loaded (/usr/lib/systemd/system/httpd.service; enabled)

Active: failed (Result: exit-code) since Tue 2015-04-28 18:01:42 IST; 28min ago

Main PID: 2881 (code=exited, status=0/SUCCESS)

Status: "Total requests: 0; Current requests/sec: 0; Current traffic: 0 B/sec

Apr 28 17:37:29 domain2 systemd[1]: httpd.service: Got notification message from PID 2881, but recepti...bled.

Apr 28 18:01:42 domain2 systemd[1]: httpd.service: control process exited, code=exited status=226

Apr 28 18:01:42 domain2 systemd[1]: Unit httpd.service entered failed state.

Control System Runlevels

How to start system rescue mode.

systemctl rescue

Broadcast message from root@domain2 on pts/0 (Wed 2015-04-29 11:31:18 IST):

The system is going down to rescue mode NOW!

How to enter into emergency mode.

systemctl emergency

Welcome to emergency mode! After logging in, type "journalctl -xb" to view system logs, "systemctl reboot" to reboot, "systemctl default" to try again to boot into default mode.

List current run levels in use.

systemctl get-default

multi-user.target

Start Runlevel 5 aka graphical mode.

systemctl isolate runlevel5.target

OR

systemctl isolate graphical.target

Start Runlevel 3 aka multiuser mode (commandline).

systemctl isolate runlevel3.target

OR

systemctl isolate multiuser.target

Set multiusermode or graphical mode as default runlevel.

systemctl set-default runlevel3.target

OR

systemctl set-default runlevel5.target

Reboot, halt, suspend, hibernate or put system in hybrid-sleep.

systemctl [reboot | halt | suspend | hibernate | hybrid-sleep]

Runlevels refresh:

Runlevel 0 : Shut down and Power off the system

Runlevel 1 : Rescue/Maintenance Mode

Runlevel 3 : multiuser, no-graphic system

Runlevel 4 : multiuser, no-graphic system

Runlevel 5 : multiuser, graphical system

Runlevel 6 : Shutdown and Reboot the machine

Control and Manage Mount Points using Systemctl

List all system mount points.

systemctl list-unit-files --type=mount

UNIT FILE	STATE
dev-hugepages.mount	static

...

sys-kernel-debug.mount	static
------------------------	--------

tmp.mount	disabled
-----------	----------

Mount, unmount, remount, reload system mount points; check the status of mount points

systemctl [start | restart | stop | reload | status] tmp.mount

Active, enable or disable a mount point at boot time (auto mount at system boot).

systemctl [is-active | enable | disable] tmp.mount

Mask (making it impossible to start) or unmask a mount points in Linux.

systemctl [mask | unmask] tmp.mount

Control and Manage Sockets using Systemctl

List all available system sockets.

systemctl list-unit-files --type=socket

UNIT FILE	STATE
dbus.socket	static
dm-event.socket	enabled

...

Start, restart, stop, reload and check the status of a socket (example: cups.socket) in Linux.

systemctl [start | restart | stop | reload | status] cups.socket

Active a socket and enable or disable at boot time (auto start socket at system boot).

systemctl [is-active | enable | disable] cups.socket

Mask (making it impossible to start) or unmask a socket (cups.socket).

systemctl [mask | unmask] cups.socket

In -s '/dev/null' '/etc/systemd/system/cups.socket'

CPU Utilization (Shares) of a Service

Get the current CPU Shares of a Service (say httpd).

systemctl show -p CPUShares httpd.service

CPUShares=1024

Default- each service has a CPUShare = 1024. You may increase/decrease CPU share of a process:

systemctl set-property httpd.service CPUShares=2000

systemctl show -p CPUShares httpd.service

CPUShares=2000

When you set CPUShare for a service, a directory with the name of service is created (httpd.service.d) which contains a file 90-CPUShares.conf which contains the CPUShare Limit information. You may view the file as:

vi /etc/systemd/system/httpd.service.d/90-CPUShares.conf

[Service]

CPUShares=2000

Check all the configuration details of a service.

systemctl show httpd

Id=httpd.service

Names=httpd.service

Requires=basic.target

Wants=system.slice

WantedBy=multi-user.target

Conflicts=shutdown.target

Before=shutdown.target multi-user.target

After=network.target remote-fs.target nss-lookup.target systemd-journald.socket basic.target system.slice

Description=The Apache HTTP Server

LoadState=loaded

ActiveState=active

SubState=running

FragmentPath=/usr/lib/systemd/system/httpd.service

....

Analyze critical chain for a services(httpd).

systemd-analyze critical-chain httpd.service

The time after the unit is active or started is printed after the "@" character.

The time the unit takes to start is printed after the "+" character.

httpd.service +142ms

└─network.target @11.168s

└─network.service @9.456s +1.712s

└─NetworkManager.service @8.858s +596ms

└─firewalld.service @4.931s +3.926s

└─basic.target @4.916s

└─sockets.target @4.916s

└─dbus.socket @4.916s

└─sysinit.target @4.905s

└─systemd-update-utmp.service @4.864s +39ms

└─auditd.service @4.563s +301ms

└─systemd-tmpfiles-setup.service @4.485s +69ms

└─rhel-import-state.service @4.342s +142ms

└─local-fs.target @4.324s

└─boot.mount @4.286s +31ms

└─systemd-fsck@dev-disk-by\x2duuid-

79f594ad\x2da332\x2d4730\x2dbb5f\x2d85d196080964.service @4.092s +149ms

└─dev-disk-by\x2duuid-79f594ad\x2da332\x2d4730\x2dbb5f\x2d85d196080964.device @4.092s

Get a list of dependencies for a service (httpd).

systemctl list-dependencies httpd.service

httpd.service

└─system.slice

└─basic.target

└─firewalld.service

└─microcode.service

└─rhel-autorelabel-mark.service

```

├─rhel-autorelabel.service
├─rhel-configure.service
├─rhel-dmesg.service
├─rhel-loadmodules.service
├─paths.target
├─slices.target
├─┬─.slice
│   └─system.slice
├─sockets.target
└─┬─dbus.socket

```

from <http://www.freedesktop.org/software/systemd/man/systemctl.html>:

Shows units required and wanted by the specified unit. This recursively lists units following the Requires=, RequiresOverridable=, Requisite=, RequisiteOverridable=, Wants=, BindsTo= dependencies. If no unit is specified, default.target is implied.

By default, only target units are recursively expanded. When --all is passed, all other units are recursively expanded as well.

List control groups hierarchically.

systemd-cgls

```

├─1 /usr/lib/systemd/systemd --switched-root --system --deserialize 23
├─user.slice
│   └─user-0.slice
│       └─session-1.scope
│           ├──2498 sshd: root@pts/0
│           ├──2500 -bash
│           ├──4521 systemd-cgls
│           └─4522 systemd-cgls
├─system.slice
│   ├──httpd.service
│   │   ├──4440 /usr/sbin/httpd -DFOREGROUND
│   │   ├──4442 /usr/sbin/httpd -DFOREGROUND
│   │   ├──4443 /usr/sbin/httpd -DFOREGROUND
│   │   ├──4444 /usr/sbin/httpd -DFOREGROUND
│   │   ├──4445 /usr/sbin/httpd -DFOREGROUND
│   │   └─4446 /usr/sbin/httpd -DFOREGROUND
│   └─polkit.service
│       └─721 /usr/lib/polkit-1/polkitd --no-debug
....

```

List control group according to CPU, memory, Input and Output.

systemd-cgtop

Path	Tasks	%CPU	Memory	Input/s	Output/s
/	83	1.0	437.8M	-	-
/system.slice	-	0.1	-	-	-
/system.slice/mariadb.service	2	0.1	-	-	-
/system.slice/tuned.service	1	0.0	-	-	-
/system.slice/httpd.service	6	0.0	-	-	-
/system.slice/NetworkManager.service	1	-	-	-	-
/system.slice/atop.service	1	-	-	-	-
/system.slice/atopacct.service	1	-	-	-	-
/system.slice/auditd.service	1	-	-	-	-
/system.slice/crond.service	1	-	-	-	-
/system.slice/dbus.service	1	-	-	-	-
/system.slice/firewalld.service	1	-	-	-	-
/system.slice/lvm2-lvmetad.service	1	-	-	-	-
/system.slice/polkit.service	1	-	-	-	-
/system.slice/postfix.service	3	-	-	-	-