**Starting with RHEL7**

1. BIOS:

* The BIOS (Basic Input/Output System), performs the POST (power on self test) to detect, test and initialize system hardware components
* Loads the MBR (Master boot record).

1. Master Boot Record:

* Master Boot Record (MBR) is the first 512 bytes of the boot drive that is read into memory by the BIOS.
* The next 64 bytes contain the partition table for the disk. The last two bytes are the “Magic Number” which is used for error detection.
* MBR discovers the bootable device and loads the GRUB2 boot loader into memory and transfers control over to it

1. GRUB2 Boot Loader:

* The default bootloader program used on RHEL 7 is GRUB 2. GRUB stands for **GRand Unified Bootloader**. GRUB 2 replaces the older GRUB bootloader also called as legacy GRUB.
* The GRUB 2 configuration file is located at /boot/grub2/grub.cfg (Do not edit this file directly).
* GRUB 2 menu-configuration settings are taken from /etc/default/grub when generating grub.cfg.

[root@node1 ~]# cat /etc/default/grub

GRUB\_TIMEOUT=5

GRUB\_DISTRIBUTOR="$(sed 's, release .\*$,,g' /etc/system-release)"

GRUB\_DEFAULT=saved

GRUB\_DISABLE\_SUBMENU=true

GRUB\_TERMINAL\_OUTPUT="console"

GRUB\_CMDLINE\_LINUX="vconsole.keymap=us crashkernel=auto vconsole.font=latarcyrheb-sun16 rhgb quiet"

GRUB\_DISABLE\_RECOVERY="true"

* If changes are made to any of these parameters, you need to run grub2-mkconfig to re-generate the /boot/grub2/grub.cfg file

[root@node1 ~]# grub2-mkconfig –o /boot/grub2/grub.cfg

Generating grub configuration file ...

* GRUB2 searches the compressed kernel image file also called as vmlinuz in the /boot directory.
* GRUB2 loads the vmlinuz kernel image file into memory and extracts the contents of the initramfs image file into a temporary, memory-based file system (tmpfs)
* The initial RAM disk (initrd) is an initial root file system that is mounted before the real root file system.

**Initramfs  –  /etc/dracut.conf**

* The job of the initial RAM file system is to preload the block device modules, such as for IDE, SCSI, or RAID, so that the root file system, on which those modules normally reside, can then be accessed and mounted.
* The initramfs is bound to the kernel and the kernel mounts this initramfs as part of a two-stage boot process.
* The dracut utility creates initramfs whenever a new kernel is installed.
* lsinitrd command is used to view the contents of the image created by dracut:
* **So GRUB2 just loads and executes Kernel and initramfs images.**
* **After That Kernel starts systemd process with process ID of 1.**

**UID PID PPID C STIME TTY TIME CMD**

**root 1 0 0 12:27 ? 00:00:01 /usr/lib/systemd/systemd --switched-root --system --deserialize 21**

**SYSTEMD:**

* systemd is the ancestor of all processes on a system
* systemd reads the file linked by /etc/systemd/system/default.target (for example, /usr/lib/systemd/system/multi-user.target) to determine the default system target (equivalent to run level).

|  |
| --- |
| [root@kube-node1 ~]# systemctl status vsftpd  vsftpd.service - Vsftpd ftp daemon  Loaded: loaded (/usr/lib/systemd/system/vsftpd.service; disabled; vendor preset: disabled)  Active: inactive (dead)  [root@kube-node1 ~]# systemctl enable vsftpd  Created symlink from /etc/systemd/system/multi-user.target.wants/vsftpd.service to /usr/lib/systemd/system/vsftpd.service.  [root@kube-node1 ~]# systemctl disable vsftpd  Removed symlink /etc/systemd/system/multi-user.target.wants/vsftpd.service. |

* The system target file defines the services that systemd starts.
* systemd brings the system to the state defined by the system target, performing system initialization tasks such as:  
  1. Setting the host name  
  2. Initializing the network  
  3.Initializing the system hardware based on kernel boot arguments

1. Mounting the file systems, including virtual file systems such as the /proc file system

**SYSTEMD FEATURES:**

* **In systemd, a unit refers to any resource that the system knows how to operate on and manage. This is the primary object that the systemd tools know how to deal with. These resources are defined using configuration files called unit files.**
* **Units in some ways can be said to similar to services or jobs in other init systems.**
* **Units are the objects that systemd knows how to manage. These are basically a standardized representation of system resources that can be managed by the suite of daemons and manipulated by the provided utilities**.
* **Ideas that in other init systems may be handled with one unified service definition can be broken out into component units according to their focus. This organizes by function and allows you to easily enable, disable, or extend functionality without modifying the core behavior of a unit**.

**Types of Units**

**Systemd categories units according to the type of resource they describe. The easiest way to determine the type of a unit is with its type suffix, which is appended to the end of the resource name. The following list describes the types of units available to systemd:**

* **.service**: A service unit describes how to manage a service or application on the server. This will include how to start or stop the service, under which circumstances it should be automatically started, and the dependency and ordering information for related software.
* **.socket**: A socket unit file describes a network or IPC socket. These always have an associated .service file that will be started when activity is seen on the socket that this unit defines.
* **.device**: A unit that describes a device that has been designated as needing systemdmanagement by udev or the sysfs filesystem. Not all devices will have .device files. Some scenarios where .device units may be necessary are for ordering, mounting, and accessing the devices.

|  |
| --- |
| **Udev supplies a dynamic device directory containing only the nodes for devices which are connected to the system. It creates or removes the device node files in the /dev directory as they are plugged in or taken out** |

* **.mount**: This unit defines a mountpoint on the system to be managed by systemd. These are named after the mount path, with slashes changed to dashes. Entries within /etc/fstab can have units created automatically.
* **.automount**: An .automount unit configures a mountpoint that will be automatically mounted. These must be named after the mount point they refer to and must have a matching .mount unit to define the specifics of the mount.
* **.swap**: This unit describes swap space on the system. The name of these units must reflect the device or file path of the space.
* **.target**: A target unit is used to provide synchronization points for other units when booting up or changing states. They also can be used to bring the system to a new state. Other units specify their relation to targets to become tied to the target's operations.
* **.path**: This unit defines a path that can be used for path-based activation. By default, a .serviceunit of the same base name will be started when the path reaches the specified state. This uses inotify to monitor the path for changes.
* **.timer**: A .timer unit defines a timer that will be managed by systemd, similar to a cron job for delayed or scheduled activation. A matching unit will be started when the timer is reached.
* **.snapshot**: A .snapshot unit is created automatically by the systemctl snapshot command. It allows you to reconstruct the current state of the system after making changes. Snapshots do not survive across sessions and are used to roll back temporary states.
* **.slice**: A .slice unit is associated with Linux Control Group nodes, allowing resources to be restricted or assigned to any processes associated with the slice. The name reflects its hierarchical position within the cgroup tree. Units are placed in certain slices by default depending on their type.
* **.scope**: Scope units are created automatically by systemd from information received from its bus interfaces. These are used to manage sets of system processes that are created externally.

**# systemctl list-unit-files >> command will list all units**

**SYSTEMD Configurations Files:**

Systemd units are defined by unit configuration files located in the following directories

* /usr/lib/systemd/system

Systemd units created at runtime. This directory takes precedence over the directory with installed service units

* /etc/systemd/system

Systemd units created and managed by the system administrator. This directory takes precedence over the directory with runtime units

**SYSTEMD Targets Units:**

* OS versions prior to RHEL7 were using SysV init run levels to allow a system to be used for a specific purpose
* In RHEL 7, run levels have been replaced with systemd target units
* Target units have a .target extension
* Target units allow you to start a system with only the services that are required for a specific purpose

To list the predefined systemd run level target units

[root@node1 ~]# find /usr -name "runlevel\*.target"

/usr/lib/systemd/system/runlevel4.target

/usr/lib/systemd/system/runlevel5.target

/usr/lib/systemd/system/runlevel6.target

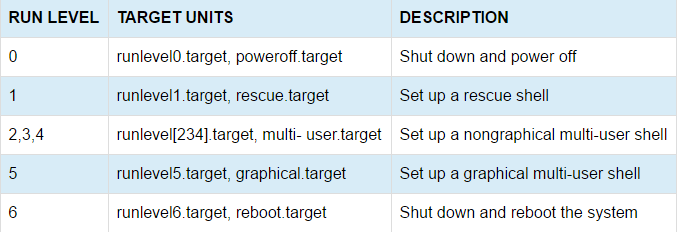
/usr/lib/systemd/system/runlevel0.target

/usr/lib/systemd/system/runlevel1.target

/usr/lib/systemd/system/runlevel2.target

/usr/lib/systemd/system/runlevel3.target

**SYSTEMD Runlevels vs Targets:**

****

[root@node1 ~]# ls -l /usr/lib/systemd/system/runlevel?.target

lrwxrwxrwx. 1 root root 15 Jun 9 13:54 /usr/lib/systemd/system/runlevel0.target -> poweroff.target

lrwxrwxrwx. 1 root root 13 Jun 9 13:54 /usr/lib/systemd/system/runlevel1.target -> rescue.target

lrwxrwxrwx. 1 root root 17 Jun 9 13:54 /usr/lib/systemd/system/runlevel2.target -> multi-user.target

lrwxrwxrwx. 1 root root 17 Jun 9 13:54 /usr/lib/systemd/system/runlevel3.target -> multi-user.target

lrwxrwxrwx. 1 root root 17 Jun 9 13:54 /usr/lib/systemd/system/runlevel4.target -> multi-user.target

lrwxrwxrwx. 1 root root 16 Jun 9 13:54 /usr/lib/systemd/system/runlevel5.target -> graphical.target

lrwxrwxrwx. 1 root root 13 Jun 9 13:54 /usr/lib/systemd/system/runlevel6.target -> reboot.target

[root@node1 ~]#

**SYSTEMD Viewing available Targets:**

root@node1 ~]# systemctl list-units --type target

UNIT LOAD ACTIVE SUB DESCRIPTION

basic.target loaded active active Basic System

bluetooth.target loaded active active Bluetooth

cryptsetup.target loaded active active Encrypted Volumes

getty.target loaded active active Login Prompts

graphical.target loaded active active Graphical Interface

local-fs-pre.target loaded active active Local File Systems (Pre)

local-fs.target loaded active active Local File Systems

multi-user.target loaded active active Multi-User System

network-online.target loaded active active Network is Online

network.target loaded active active Network

nfs-client.target loaded active active NFS client services

nss-lookup.target loaded active active Host and Network Name Lookups

nss-user-lookup.target loaded active active User and Group Name Lookups

paths.target loaded active active Paths

remote-fs-pre.target loaded active active Remote File Systems (Pre)

remote-fs.target loaded active active Remote File Systems

rpcbind.target loaded active active RPC Port Mapper

slices.target loaded active active Slices

sockets.target loaded active active Sockets

sound.target loaded active active Sound Card

swap.target loaded active active Swap

sysinit.target loaded active active System Initialization

timers.target loaded active active Timers

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.

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

To show all installed unit files use 'systemctl list-unit-files'.

* **How to change default target**

[root@node1 ~]# systemctl get-default

graphical.target

[root@node1 ~]# systemctl set-default multiuser

Failed to set default target: No such file or directory

[root@node1 ~]# systemctl set-default multi-user

Removed symlink /etc/systemd/system/default.target.

Created symlink from /etc/systemd/system/default.target to /usr/lib/systemd/system/multi-user.target.

[root@node1 ~]# systemctl get-default

multi-user.target

root@node1 ~]# systemctl --version

systemd 219

+PAM +AUDIT +SELINUX +IMA -APPARMOR +SMACK +SYSVINIT +UTMP +LIBCRYPTSETUP +GCRYPT +GNUTLS +ACL +XZ -LZ4 -SECCOMP +BLKID +ELFUTILS +KMOD +IDN

[root@node1 ~]#

* **Lists all services available in system. Service enable means will come up on reboot. Service static means cant be enable, since these are depended on other services and will come online/start auto once the service it depended is online**

[root@node1 ~]# systemctl list-unit-files --type service

UNIT FILE STATE

abrt-ccpp.service enabled

abrt-oops.service enabled

abrt-pstoreoops.service disabled

abrt-vmcore.service enabled

abrt-xorg.service enabled

abrtd.service enabled

accounts-daemon.service enabled

alsa-restore.service static

alsa-state.service static

alsa-store.service static

anaconda-direct.service static

anaconda-nm-config.service static

* **To list the active state of all loaded units, use the below command.**

root@node1 ~]# systemctl list-units --type service

UNIT LOAD ACTIVE SUB DESCRIPTION

abrt-ccpp.service loaded active exited Install ABRT coredump hook

abrt-oops.service loaded active running ABRT kernel log watcher

abrt-xorg.service loaded active running ABRT Xorg log watcher

abrtd.service loaded active running ABRT Automated Bug Reporting Tool

accounts-daemon.service loaded active running Accounts Service

alsa-state.service loaded active running Manage Sound Card State (restore and store)

atd.service loaded active running Job spooling tools

auditd.service loaded active running Security Auditing Service

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

blk-availability.service loaded active exited Availability of block devices

bluetooth.service loaded active running Bluetooth service

chronyd.service loaded active running NTP client/server

colord.service loaded active running Manage, Install and Generate Color Profiles

crond.service loaded active running Command Scheduler

cups-browsed.service loaded active running Make remote CUPS printers available locally

cups.service loaded active running CUPS Printing Service

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

* **To list the active/inactive state of all loaded units, use the below command.**

root@node1 ~]# systemctl list-units --type service -all

UNIT LOAD ACTIVE SUB DESCRIPTION

abrt-ccpp.service loaded active exited Install ABRT coredump hook

abrt-oops.service loaded active running ABRT kernel log watcher

abrt-vmcore.service loaded inactive dead Harvest vmcores for ABRT

abrt-xorg.service loaded active running ABRT Xorg log watcher

abrtd.service loaded active running ABRT Automated Bug Reporting To

accounts-daemon.service loaded active running Accounts Service

alsa-restore.service loaded inactive dead Save/Restore Sound Card State

alsa-state.service loaded active running Manage Sound Card State (restor

● apparmor.service not-found inactive dead apparmor.service

atd.service loaded active running Job spooling tools

auditd.service loaded active running Security Auditing Service

auth-rpcgss-module.service loaded inactive dead Kernel Module supporting RPCSEC

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

* **similarly we can check for other units other than service:**

root@node1 ~]# systemctl list-units --type target -all

UNIT LOAD ACTIVE SUB DESCRIPTION

basic.target loaded active active Basic System

bluetooth.target loaded active active Bluetooth

cryptsetup.target loaded active active Encrypted Volumes

emergency.target loaded inactive dead Emergency Mode

final.target loaded inactive dead Final Step

getty.target loaded active active Login Prompts

graphical.target loaded active active Graphical Interface

local-fs-pre.target loaded active active Local File Systems (Pre)

local-fs.target loaded active active Local File Systems

multi-user.target loaded active active Multi-User System

network-online.target loaded active active Network is Online

network-pre.target loaded inactive dead Network (Pre)

network.target loaded active active Network

------------------

[root@node1 ~]# systemctl list-units --type mount -all

UNIT LOAD ACTIVE SUB DESCRIPTION

-.mount loaded active mounted /

boot.mount loaded active mounted /boot

dev-hugepages.mount loaded active mounted Huge Pages File System

dev-mqueue.mount loaded active mounted POSIX Message Queue File System

proc-fs-nfsd.mount loaded active mounted NFSD configuration filesystem

proc-sys-fs-binfmt\_misc.mount loaded active mounted Arbitrary Executable File Formats File System

run-media-desingh-CentOS\x207\x20x86\_64.mount loaded active mounted /run/media/desingh/CentOS 7 x86\_64

* **To list the all sockets units on the system,**

root@node1 ~]# systemctl list-units --type socket -all

UNIT LOAD ACTIVE SUB DESCRIPTION

avahi-daemon.socket loaded active running Avahi mDNS/DNS-SD Stack Activation Socket

cups.socket loaded active running CUPS Printing Service Sockets

dbus.socket loaded active running D-Bus System Message Bus Socket

dm-event.socket loaded active listening Device-mapper event daemon FIFOs

iscsid.socket loaded active listening Open-iSCSI iscsid Socket

iscsiuio.socket loaded active listening Open-iSCSI iscsiuio Socket

lvm2-lvmetad.socket loaded active running LVM2 metadata daemon socket

lvm2-lvmpolld.socket loaded active listening LVM2 poll daemon socket

rpcbind.socket loaded active running RPCbind Server Activation Socket

* **Enable/disable start/stop a service:**

[root@node1 ~]# systemctl status httpd

● httpd.service - The Apache HTTP Server

Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled; vendor preset: **disabled**)

Active: inactive (dead)

Docs: man:httpd(8)

man:apachectl(8)

Sep 07 12:28:14 node1 systemd[1]: Starting The Apache HTTP Server...

Sep 07 12:28:22 node1 systemd[1]: Started The Apache HTTP Server.

Sep 07 13:49:03 node1 systemd[1]: Reloaded The Apache HTTP Server.

Sep 07 14:56:49 node1 systemd[1]: Stopping The Apache HTTP Server...

Sep 07 14:56:50 node1 systemd[1]: Stopped The Apache HTTP Server.

[root@node1 ~]# systemctl enable httpd

Created symlink from /etc/systemd/system/multi-user.target.wants/httpd.service to /usr/lib/systemd/system/httpd.service.

[root@node1 ~]#

[root@node1 ~]# systemctl status httpd

● httpd.service - The Apache HTTP Server

Loaded: loaded (/usr/lib/systemd/system/httpd.service; **enabled**; vendor preset: disabled)

Active: inactive (dead)

Docs: man:httpd(8)

man:apachectl(8)

Sep 07 12:28:14 node1 systemd[1]: Starting The Apache HTTP Server...

Sep 07 12:28:22 node1 systemd[1]: Started The Apache HTTP Server.

Sep 07 13:49:03 node1 systemd[1]: Reloaded The Apache HTTP Server.

Sep 07 14:56:49 node1 systemd[1]: Stopping The Apache HTTP Server...

Sep 07 14:56:50 node1 systemd[1]: Stopped The Apache HTTP Server.

[root@node1 ~]# systemctl start httpd

[root@node1 ~]#

[root@node1 ~]# systemctl status httpd

● httpd.service - The Apache HTTP Server

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

Active: active (running) since Thu 2017-09-07 14:57:42 PDT; 5s ago

Docs: man:httpd(8)

man:apachectl(8)

Main PID: 5722 (httpd)

Status: "Processing requests..."

CGroup: /system.slice/httpd.service

├─5722 /usr/sbin/httpd -DFOREGROUND

├─5725 /usr/sbin/httpd -DFOREGROUND

├─5726 /usr/sbin/httpd -DFOREGROUND

├─5727 /usr/sbin/httpd -DFOREGROUND

├─5729 /usr/sbin/httpd -DFOREGROUND

└─5730 /usr/sbin/httpd -DFOREGROUND

Sep 07 14:57:40 node1 systemd[1]: Starting The Apache HTTP Server...

Sep 07 14:57:42 node1 systemd[1]: Started The Apache HTTP Server.

[root@node1 ~]#

* **Lists dependencies for an service:**

[root@node1 ~]# systemctl list-dependencies httpd

httpd.service

● ├─-.mount

● ├─system.slice

● └─basic.target

● ├─alsa-restore.service

● ├─alsa-state.service

● ├─microcode.service

● ├─rhel-autorelabel-mark.service

● ├─rhel-autorelabel.service

● ├─rhel-configure.service

● ├─rhel-dmesg.service

● ├─rhel-loadmodules.service

● ├─selinux-policy-migrate-local-changes@targeted.service

● ├─paths.target

● ├─slices.target

● │ ├─-.slice

● │ └─system.slice

● ├─sockets.target

● │ ├─avahi-daemon.socket

● │ ├─cups.socket

● │ ├─dbus.socket

● │ ├─dm-event.socket

● │ ├─iscsid.socket

● │ ├─iscsiuio.socket

● │ ├─lvm2-lvmetad.socket

● │ ├─rpcbind.socket

● │ ├─systemd-initctl.socket

● │ ├─systemd-journald.socket

● │ ├─systemd-shutdownd.socket

● │ ├─systemd-udevd-control.socket

● │ ├─systemd-udevd-kernel.socket

● │ └─virtlogd.socket

● ├─sysinit.target

● │ ├─dev-hugepages.mount

● │ ├─dev-mqueue.mount

● │ ├─dmraid-activation.service

● │ ├─iscsi.service

● │ ├─kmod-static-nodes.service

● │ ├─lvm2-lvmetad.socket

● │ ├─lvm2-lvmpolld.socket

● │ ├─lvm2-monitor.service

● │ ├─multipathd.service

● │ ├─plymouth-read-write.service

● │ ├─plymouth-start.service

● │ ├─proc-sys-fs-binfmt\_misc.automount

● │ ├─sys-fs-fuse-connections.mount

● │ ├─sys-kernel-config.mount

● │ ├─sys-kernel-debug.mount

● │ ├─systemd-ask-password-console.path

● │ ├─systemd-binfmt.service

● │ ├─systemd-firstboot.service

● │ ├─systemd-hwdb-update.service

● │ ├─systemd-journal-catalog-update.service

● │ ├─systemd-journal-flush.service

● │ ├─systemd-journald.service

● │ ├─systemd-machine-id-commit.service

● │ ├─systemd-modules-load.service

● │ ├─systemd-random-seed.service

● │ ├─systemd-sysctl.service

● │ ├─systemd-tmpfiles-setup-dev.service

● │ ├─systemd-tmpfiles-setup.service

● │ ├─systemd-udev-trigger.service

● │ ├─systemd-udevd.service

● │ ├─systemd-update-done.service

● │ ├─systemd-update-utmp.service

● │ ├─systemd-vconsole-setup.service

● │ ├─cryptsetup.target

● │ ├─local-fs.target

● │ │ ├─-.mount

● │ │ ├─boot.mount

● │ │ ├─rhel-import-state.service

● │ │ ├─rhel-readonly.service

● │ │ ├─systemd-fsck-root.service

● │ │ └─systemd-remount-fs.service

● │ └─swap.target

● │ └─dev-disk-by\x2duuid-d37e3a9d\x2d0aca\x2d4e15\x2db005\x2dddbe620cd4b5.swap

● └─timers.target

● └─systemd-tmpfiles-clean.timer

lines 45-78/78 (END)

* **Lists dependencies for an Target:**

[root@node1 ~]# systemctl list-dependencies graphical.target

graphical.target

● ├─accounts-daemon.service

● ├─gdm.service

● ├─network.service

● ├─rtkit-daemon.service

● ├─systemd-readahead-collect.service

● ├─systemd-readahead-replay.service

● ├─systemd-update-utmp-runlevel.service

● └─multi-user.target

● ├─abrt-ccpp.service

● ├─abrt-oops.service

● ├─abrt-vmcore.service

● ├─abrt-xorg.service

● ├─abrtd.service

-------

------

* **Lists what is target dependency hierarchy : There are multiple targets through which system go while booting into default traget**

root@node1 ~]# systemctl list-dependencies graphical.target|grep target

graphical.target

● └─multi-user.target

● ├─target.service

● ├─basic.target

● │ ├─selinux-policy-migrate-local-changes@targeted.service

● │ ├─paths.target

● │ ├─slices.target

● │ ├─sockets.target

● │ ├─sysinit.target

● │ │ ├─cryptsetup.target

● │ │ ├─local-fs.target

● │ │ └─swap.target

● │ └─timers.target

● ├─getty.target

● ├─nfs-client.target

● │ └─remote-fs-pre.target

● └─remote-fs.target

● └─nfs-client.target

● └─remote-fs-pre.target

* **To get the boot process duration, type**

**[root@kube-node1 ~]# systemd-analyze**

**Startup finished in 1.702s (kernel) + 3.398s (initrd) + 3min 28.721s (userspace) = 3min 33.822s**

* **Time spent by each task during the boot process, type**

[root@kube-node1 ~]# systemd-analyze blame|more

2min 16.302s iscsi.service

14.305s kdump.service

11.886s pcsd.service

7.994s cobblerd.service

6.596s tuned.service

6.168s NetworkManager-wait-online.service

* **Get the list of the critical chain for a particular target or for any 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.**

**[**root@node1 ~]# systemd-analyze critical-chain graphical.target|grep target

graphical.target @42.291s

└─multi-user.target @42.291s

└─network.target @31.616s

└─basic.target @18.225s

└─paths.target @18.225s

└─sysinit.target @18.208s

└─local-fs.target @17.321s

└─local-fs-pre.target @13.313s

* **Get the list of the critical chain for a particular service**

[root@kube-node1 ~]# systemd-analyze critical-chain vsftpd.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.

└─network.target @27.791s

└─wpa\_supplicant.service @3min 5.004s +250ms

└─basic.target @11.418s

└─sockets.target @11.418s

└─avahi-daemon.socket @11.418s

└─sysinit.target @11.378s

└─systemd-update-utmp.service @11.303s +73ms

└─auditd.service @10.924s +377ms

└─systemd-tmpfiles-setup.service @10.785s +119ms

└─rhel-import-state.service @10.625s +158ms

└─local-fs.target @10.606s

└─run-user-0.mount @1min 30.937s

└─local-fs-pre.target @9.469s

└─lvm2-monitor.service @9.431s +36ms

└─lvm2-lvmetad.socket @2.986s

└─-.slice

* **Starting & Stopping Services Rhel7 Vs Previous Version**

|  |  |  |
| --- | --- | --- |
| **service Utility in Rhel6** | **systemctl Utility in Rhel7** | **Description** |
| #service name start | #systemctl start name | Starts a service |
| #service name stop | #systemctl stop name | Stops a service |
| #service name restart | #systemctl restart name | Restarts a service |
| #service name condrestart | #systemctl try- restart name | Restarts a service only if it is running |
| #service name reload | #systemctl reload name | Reloads a configuration |
| #service name status | #systemctl status name | Checks whether a service is running |
| #service –status- all | #systemctl list-units –type service –all | Displays the status of all services |

* **Enabling & Disabling Services Rhel7 Vs previous Versions**

|  |  |  |
| --- | --- | --- |
| **chkconfig Utility (Rhel6)** | **systemctl Utility (Rhel7)** | **Description** |
| #chkconfig name on | #systemctl enable name | Enables a service |
| #chkconfig name off | #systemctl disable name | Disables a service |
| #chkconfig –list name | #systemctl status name, systemctl is-enabled name | Checks whether a service is enabled |
| #chkconfig –list | #systemctl list-unit-files –type service | Lists all services and checks whether they are enabled |

==================================================

**Journalctl**

Journalctl is one of the new functionality to the Linux operating system by system in rhel7. The systemd brings us journal daemon called journald which enables centralized management of logs. This logs can be filtered and manipulated by utility called journalctl.

* About journalctl (journald)

The journald is service that collects and stores logs from many sources, and make indexed structured log files that are easy to interpret. Some of the sources that journald uses

* Kernel log messages, via kmsg
* Simple system log messages, via the libc syslog call
* Structured system log messages via the native Journal API
* Standard output and standard error of system services

Idea is to have all log messages centralized, no matter from which application they are coming from:

* Viewing logs: Basic command:

**[root@kube-master ~]# journalctl**

-- Logs begin at Sun 2017-08-27 19:31:31 IST, end at Wed 2017-10-25 05:00:43 IST. --

Aug 27 19:31:31 tcli systemd-journal[88]: Runtime journal is using 8.0M (max allowed 203.2M, trying to leave 304.8M free of 1.9G available → current l

Aug 27 19:31:31 tcli kernel: Initializing cgroup subsys cpuset

Aug 27 19:31:31 tcli kernel: Initializing cgroup subsys cpu

Aug 27 19:31:31 tcli kernel: Initializing cgroup subsys cpuacct

## **Filtering logs by time**

[root@kube-master ~]# journalctl --since "2017-09-10 19:00:00" --until "2017-10-10 19:05:00"

-- Logs begin at Sun 2017-08-27 19:31:31 IST, end at Wed 2017-10-25 05:10:02 IST. --

Sep 11 13:44:34 node1 systemd-journal[90]: Runtime journal is using 8.0M (max allowed 203.2M, trying to leave 304.8M free of 1.9G available → current

Sep 11 13:44:34 node1 kernel: Initializing cgroup subsys cpuset

Sep 11 13:44:34 node1 kernel: Initializing cgroup subsys cpu

Sep 11 13:44:34 node1 kernel: Initializing cgroup subsys cpuacct

Sep 11 13:44:34 node1 kernel: Linux version 3.10.0-514.21.1.el7.x86\_64 (builder@kbuilder.dev.centos.org) (gcc version 4.8.5 20150623 (Red Hat 4.8.5-11

Sep 11 13:44:34 node1 kernel: Command line: BOOT\_IMAGE=/vmlinuz-3.10.0-514.21.1.el7.x86\_64 root=UUID=e7b7948b-1210-4cc8-b3b7-b38708a7b549 ro vconsole.

Sep 11 13:44:34 node1 kernel: Disabled fast string operations

Sep 11 13:44:34 node1 kernel: e820: BIOS-provided physical RAM map:

* **Filtering messages by Units:**

[root@kube-master ~]# journalctl -u httpd.service

-- Logs begin at Sun 2017-08-27 19:31:31 IST, end at Wed 2017-10-25 05:10:02 IST. --

Aug 27 19:32:00 node1 systemd[1]: Starting The Apache HTTP Server...

Aug 27 19:32:18 node1 httpd[1369]: AH00558: httpd: Could not reliably determine the server's fully qualified domain name, using 16.151.73.67. Set the

Aug 27 19:32:28 node1 systemd[1]: Started The Apache HTTP Server.

Aug 27 20:37:12 node1 httpd[3582]: AH00558: httpd: Could not reliably determine the server's fully qualified domain name, using 16.151.73.67. Set the

Aug 27 20:37:12 node1 systemd[1]: Reloaded The Apache HTTP Server.

Aug 28 16:36:16 node1 systemd[1]: Stopping The Apache HTTP Server...

Aug 28 16:36:19 node1 systemd[1]: Stopped The Apache HTTP Server.

* **Fileting Messages with PID**

[root@kube-master ~]# journalctl \_PID=595

-- Logs begin at Sun 2017-08-27 19:31:31 IST, end at Wed 2017-10-25 05:15:59 IST. --

Oct 20 20:56:00 kube-master systemd-fsck[595]: /sbin/fsck.xfs: XFS file system.

[root@kube-master ~]#

* **Displaying Logs from the Current Boot**

[root@kube-master ~]# journalctl -b

-- Logs begin at Sun 2017-08-27 19:31:31 IST, end at Wed 2017-10-25 05:22:46 IST. --

Oct 25 00:43:10 node1 systemd-journal[88]: Runtime journal is using 8.0M (max allowed 203.2M, trying to leave 304.8M free of 1.9G available → current

Oct 25 00:43:10 node1 kernel: Initializing cgroup subsys cpuset

Oct 25 00:43:10 node1 kernel: Initializing cgroup subsys cpu

Oct 25 00:43:10 node1 kernel: Initializing cgroup subsys cpuacct

* **Journal logs file path:**

**By default logs are stored under /run/log/journal**

root@kube-node1 ~]# ls -lht /run/log/journal/

total 0

drwxr-s---+ 2 root systemd-journal 60 Oct 24 16:56 12e1f493fb784569a5663b9c61ed7af0

[root@kube-node1 ~]# ls -lht /run/log/journal/12e1f493fb784569a5663b9c61ed7af0/

total 6.2M

-rwxr-x---+ 1 root systemd-journal 6.2M Oct 24 17:01 system.journal

* **By default, Journald logs disappear after a reboot.  
  To keep Journald logs persistent on reboot. Below we can do:**

# mkdir /var/log/journal

#vi /etc/systemd/journald.conf

[Journal]

Storage=persistent

SystemMaxuse=100M ###size of log file

# systemctl restart systemd-journald

Devinder Singh Lalotra