


CentOS 7 and RHEL 7 Boot Process Overview and Troubleshooting

 [vultr.com/docs/centos-7-and-rhel-7-boot-process-overview-and-troubleshooting](https://www.vultr.com/docs/centos-7-and-rhel-7-boot-process-overview-and-troubleshooting)

This article describes the boot process for CentOS/RHEL 7.x systems. While it may remain similar to previous releases, with RHEL 7, systemd is being introduced. In addition to the boot process, I will provide troubleshooting tips and tricks along the way.

You need to understand the boot process before you can actively troubleshoot a problem on boot. Review the steps below until you are familiar with the process.

High level overview

- Power + post.
- Firmware device search.
- Firmware reads bootloader.
- Boot loader loads config (grub2).
- Boot loader loads kernel and initramfs.
- Boot loader passes control to the kernel.
- Kernel initializes hardware + executes `/sbin/init` as pid 1.
- Systemd executes all initrd targets (mounts filesystem on `/sysroot`).
- Kernel root FS switched from initramfs root (`/sysroot`) to system rootfs (`/`) and systemd re-executes as system version.
- Systemd looks for default target and starts/stops units as configured while automatically solving dependencies and login page appears.

For more information on the boot process, refer to the official OS documentation for your system.

Systemd targets

Targets are basically dependency checks. They have a "before" and "after" configuration for exactly what services are required to meet that target. For example: `arp.etherenet.service` , `firewalld.service` , and so forth need to be started and working before `network.target` can be reached. If it is not reached, services such as `httpd` , `nfs` , and `ldap` cannot be started. There are 4 targets that can be set in RHEL/CentOS 7.

- `graphical.target` (GUI interface)
- `multi-user.target` (multi user mode, text based login)
- `rescue.target` (sulogin prompt, basic system initialization)
- `emergency.target` (sulogin prompt, initramfs pivot complete and system root mounted on `/` as read only)

To view the current default boot target, use the following:

```
systemctl get-default
```

Keep in mind, you can change this at run-time by isolating the target. This will start/stop all services associated with the new target, so use caution (see `systemctl isolate new.target`).

Single user mode

There are times when you will need to boot into single user mode to fix an issue with the operating system. For this example, I will show you how to use the `rescue.target` which is "single user mode" on RHEL/CentOS 7.

1. Interrupt the grub2 menu by pressing "e" to edit when prompted with the grub menu.
2. Find the line that specifies the kernel version (**vmlinuz**) and append the following to it:
`systemd.unit=rescue.target`
3. Press "Ctrl+x" to start.
4. You will then be prompted with the root password to continue, once you exit the rescue shell, the boot process will continue to load your default target.

Recovering the root password

This process is a little different than what we have used in past releases, but it's a simple task and requires very few steps to do so. If you need to recover any credentials, you can use this method for gaining access to a VM. You can still boot off of a live CD, mount the root filesystem, and edit the password, but that method is dated and requires more effort.

1. Reboot the system.
2. Interrupt the grub2 menu by pressing "e" to edit when prompted with the grub menu.
3. Move the cursor to the end of the line that specifies the kernel (**vmlinuz**). You may want to remove all other consoles other than TTY0, however this step may not be necessary in your environment.
4. Append `rd.break` (**no quotes**) which will break the boot process just before the control is handed from initramfs to the actual system.
5. Ctrl+x to boot.

At this point, a root shell is presented with the root filesystem mounted in read-only mode on `/sysroot`. We will need to remount it with write privileges.

Remount `/sysroot`.

```
# mount -o remount,rw /sysroot
```

Switch to a chroot jail.

```
# chroot /sysroot
```

Change the password for the user in which we have outdated credentials.

```
# passwd <username>
```

If you are using SELinux, you should consider re-labeling all files before continuing the boot process. This part can be skipped if you are not utilizing SELinux.

```
# touch /.autorelabel
```

Exit twice and the system will cleanly boot from the point we interrupted it.

Reviewing logs from previous boots

It can be useful to view logs of previous failed boot attempts. If the journald logs have been made persistent (normally stored in memory and released on boot) this can be done with the `journalctl` tool. Follow these steps if you need to setup persistent boot logging.

As root, create the log file for this information to be stored.

```
# mkdir -p 2775 /var/log/journal && chown :systemd-journal /var/log/journal
# systemctl restart systemd-journald
```

To inspect the logs of a previous boot, use the `-b` option with `journalctl`. Without any arguments, `-b` will filter output only to messages pertaining to the last boot. A negative number to this argument will filter on previous boots. For example:

```
# journalctl -b-1 -p err
```

This will show you the error logs from the boot that occurred before the most recent. You should change the numerical value to reflect the boot you need to view.

Repairing disk and filesystem errors

One of the most common boot time errors is a misconfigured `/etc/fstab` file. You **CANNOT** use the `rescue.target` to fix an `/etc/fstab` error. Most of these issues will require us to use the `emergency.target` since "rescue" requires a more functional system.

The following are examples of problems that require the `emergency.target` :

1. Corrupt file system.
2. Non-existent UUID in `/etc/fstab` .
3. Non-existent mount point in `/etc/fstab` .
4. Incorrect mount option in `/etc/fstab` .

Important: After editing the `/etc/fstab` file in emergency mode, you must run the following for safety measures:

```
# systemctl daemon-reload
```

Here is a walkthrough example. We are going to boot into emergency mode to remove a false entry in `/etc/fstab`.

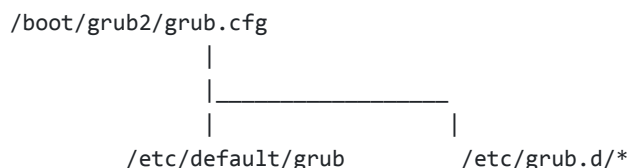
1. Interrupt the grub2 menu by pressing "e" to edit when prompted with the grub menu.
2. Find the line that specifies the kernel version (**vmlinuz**) and append the following to it:
`systemd.unit=emergency.target`
3. Press "Ctrl+x" to boot.
4. You will be prompted with the root password to continue.
5. Remount `/` so that we can make changes to the `fstab` file: `# mount -oremount,rw /`
6. We can use the `mount` command to see which entry is causing the error: `# mount -a`
7. Remove the offending entry from the `fstab` file.
8. Use `mount -a` again to make sure the error has been resolved.
9. Use `systemctl daemon-reload` as I had mentioned earlier to reload all unit files, and recreate the entire dependency tree.

Once you exit the emergency shell, the system will finish booting from the emergency target, you will then be able to continue as usual from that point. This example was just used to show you the process of using the emergency target to make persistent changes to files on the system.

Boot loader issues with Grub 2

The `/boot/grub2/grub.cfg` file is the main configuration file. **DO NOT** ever edit this file manually. Instead, use `grub2-mkconfig` to generate the new grub2 config using a set of *different* configuration files and the list of the installed kernels. The `grub2-mkconfig` command will look at `/etc/default/grub` for options such as the default menu timeout and kernel command line to use, then use a set of scripts in `/etc/grub.d/` to generate the resulting configuration file.

Here is an textual diagram of this relationship.



Important: To edit the main grub.cfg file, you will need to make the desired changes to `/etc/default/grub` and to files in `/etc/grub.d/` and then create a new `grub.cfg` by running:

```
# grub2-mkconfig > /boot/grub2/grub.cfg
```

Troubleshooting grub

It is important to understand the syntax of the `/boot/grub2/grub.cfg` file before troubleshooting.

- First, bootable entries are encoded inside 'menuentry' blocks. In these blocks, `linux16` and `initrd16` lines point to the kernel to be loaded from disk (along with the kernel command line) and the initramfs to be loaded. During interactive editing at boot, tab is used to find these lines.
- The "set root" lines inside those blocks do not point to the root file system for the RHEL/CentOS 7 system, but instead point to the file system from which grub2 should load the kernel and initramfs files. The syntax is `harddrive.partition` where `hd0` is the first hard drive in the system and `hd1` is the second. The partitions are indicated as `msdos1` for the first MBR partition or `gpt1` for the first GPT partition.

Example from `/boot/grub2/grub.cfg` :

```
### BEGIN /etc/grub.d/10_linux ###
menuentry 'CentOS Linux (3.10.0-514.26.2.el7.x86_64) 7 (Core)' --class centos --class gnu-linux --
class gnu --class os --unrestricted $menuentry_id_option 'gnulinux-3.10.0-514.el7.x86_64-advanced-
a2531d12-46f8-4a0f-8a5c-b48d6ef71275' {
    load_video
    set gfxpayload=keep
    insmod gzio
    insmod part_msdos
    insmod ext2
    set root='hd0,msdos1'
    if [ x$feature_platform_search_hint = xy ]; then
        search --no-floppy --fs-uuid --set=root --hint='hd0,msdos1' 123455ae-46f8-4a0f-8a5c-
b48d6ef71275
    else
        search --no-floppy --fs-uuid --set=root 123455ae-46f8-4a0f-8a5c-b48d6ef71275
    fi
}
```

If you need to re-install the bootloader on a device, use the following command.

```
# grub2-install <device>
```

Fixing a broken grub installation

For cases when the system will not boot after reaching the grub2 menu.

- You should start by editing the grub menu and searching for syntax errors. If you find one, correct it and get into the system to make persistent changes to fix the problem.
- If you cannot find any errors, refer to the above section where we boot into the emergency target. You will need to remount root (`/`) again.
- View the current grub2 configuration with the following command: `# grub2-mkconfig`
- If you do not see any errors, its likely that someone edited the `/boot/grub2/grub.cfg` file. Do not modify this file. Rebuild the config with the following command: `# grub2-mkconfig > /boot/grub2/grub.cfg`

Once you have rebuilt the grub config, you should be able to reboot without having any issues.

