What is systemd?

- Replaces init
  - Literally!

```bash
[root@rhel7 ~]# ls -al /sbin/init
lrwxrwxrwx. 1 root root 22 Jan 27 13:43 /sbin/init -> ../lib/systemd/systemd
```

- First process to start and last to stop
- Parent process of all other processes
- Manages services \textit{and} other resources
What was init again?

- init – System V UNIX origins in 1970s
- Process for starting system:
  - BIOS/UEFI → Bootloader → Kernel → init
- init is the parent of all processes
- Creates processes from scripts stored in /etc/inittab
- “Modern” init scripts are stored in /etc/init.d and called from /etc/rc*
Why replace System V init?

- init scripts!
  - Old, poorly maintained
  - Lack of standardization
  - Difficult / impossible to analyze (by humans and/or computers)
- Single threaded
- Unable to represent complex relationships
. /etc/rc.d/init.d/functions
if [ -f /etc/sysconfig/httpd ]; then
    . /etc/sysconfig/httpd
fi
HTTPD_LANG=${HTTPD_LANG-"C"}
INITLOG_ARGS=""
apachectl=/usr/sbin/apachectl
httpd=${HTTPD-/usr/sbin/httpd}
prog=httpd
pidfile=${PIDFILE-/var/run/httpd/httpd.pid}
lockfile=${LOCKFILE-/var/lock/subsys/httpd}
RETVAL=0
STOP_TIMEOUT=${STOP_TIMEOUT-10}
start() {
    echo -n "$"Starting $prog: 
    LANG=$HTTPD_LANG daemon --pidfile=${pidfile} $httpd $OPTIONS
    RETVAL=$?
    echo
    [ $RETVAL = 0 ] && touch ${lockfile}
    return $RETVAL
}
stop() {
    echo -n "$"Stopping $ prog: 
    killproc -p ${pidfile} -d ${STOP_TIMEOUT} $httpd
    RETVAL=$?
    echo
    [ $RETVAL = 0 ] && rm -f ${lockfile} ${pidfile}
}
reload() {
    echo -n "Reloading $prog: 
    if ! LANG=$HTTPD_LANG $httpd $OPTIONS -t >&/dev/null; then
        RETVAL=6
        echo "$not reloading due to configuration syntax error"
        failure "$not reloading $httpd due to configuration syntax error"
    else
        LSB=1 killproc -p ${pidfile} $httpd -HUP
        RETVAL=\$
        if [ $RETVAL -eq 7 ]; then
            failure "$httpd shutdown"
        fi
    fi
    echo
}

case "$1" in
    start)
        start
    ;;
    stop)
        stop
    ;;
    status)
        status -p ${pidfile} $httpd
        RETVAL=\$
    ;;

restart)
    stop
    start
    ;;
_condrestart|try-restart)
    if status -p ${pidfile} $httpd >&/dev/null; then
        stop
        start
    fi
    ;;
_force-reload|reload)
    reload
    ;;
graceful|help|configtest|fullstatus)
    $apachectl $@
    RETVAL=$?
    ;;
*)
    echo "$Usage: $prog
{start|stop|restart|condrestart|try-restart|force-reload|reload|status|fullstatus|graceful|help|configtest}"
    RETVAL=2
esac
exit $RETVAL
[Unit]
Description=The Apache HTTP Server
After=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=/usr/sbin/httpd $OPTIONS -k graceful-stop
KillSignal=SIGCONT
PrivateTmp=true

[Install]
WantedBy=multi-user.target
So long, and thanks for all the fish

RIP
Sys V
init
1970-2011
Adoption

• Default init in
  - Fedora 15 – May 2011
  - Arch – October 2012
  - Red Hat – June 2014
  - SUSE – October 2014
  - Ubuntu & Debian – April 2015
systemd - System & Service Manager
systemd Overview

- Controls More than Services
- Dependency Control
- Tracks and Restarts Services
- Service Activation
- Faster Start Up and Shutdown
- Improved Resource Management
- Better Logging, Debugging and Profiling
- Backwards compatible
- Easier to learn
### systemd Units

Controls more than services, it controls all resources on the system - referred to as units.

#### Examples of Units:

<table>
<thead>
<tr>
<th>Services</th>
<th>Sockets</th>
<th>Mounts</th>
</tr>
</thead>
<tbody>
<tr>
<td>Targets</td>
<td>Swap</td>
<td>and more...</td>
</tr>
</tbody>
</table>

Units are defined using Unit Files

- Naming convention is name.unit_type
systemd Unit Files

- Maintainer files: /usr/lib/systemd/system
- Administrator files: /etc/systemd/system
- Non-persistent, runtime data: /run/systemd
- Drop-ins: /etc/systemd/system/[name.type].d/*.conf

**Note:** unit files under /etc will take precedence over /usr

Don't forget `systemctl daemon-reload` when modifying units.
Common Unit File Options

- Description=Unit description
- Documentation=Documentation links
- Requires=Additional units required
- Before/After=Unit must start Before/After
- Wants=Weaker Requires
- Conflicts=Units cannot co-exist
- WantedBy/RequiredBy=Set other units requirement

- Lots of great detail in the RHEL 7 System Administrator's Guide
Service Activation

- Start up services when needed
  - Save resources
  - Increased reliability
  - Transparent to client

- Activation by Socket, Device, Path, Bus, and Timer

- Recommended to convert xinetd services to units
Improved Resource Management

- Services labeled and isolated with Cgroups
- More control than nice alone
- Can properly kill/restart entire service chain
- Can configure multiple instances for a single service
- Can balance by shares or by hard limits
Kill/Restart Cleanly

- Tracked in the kernel
- Knows all children
- Don’t need to rely on a potentially misbehaving process to hopefully kill its children
Auto-Restarting

- It’s paying attention!
- Reality: software does crash occasionally
- Reduces need for manual intervention
- Socket stays open, only lose that single transaction
systemd: Managing Services

With init:
$ service unit {start,stop,restart,reload}

With systemd:
$ systemctl {start,stop,restart,reload} unit1 [unit2 ...]

- Allows multiple services to be acted on simultaneously
- Assumes .service as unit type
- Tab completion works great with systemctl
  • Install bash-completion
systemctl vs service

[root@rhel6 ~]# service hellod start
Starting Hell
[root@rhel6 ~]
hellod (pid
[root@rhel6 ~]
Stopping Hell
[root@rhel6 ~]

[root@rhel7 ~]
[root@rhel7 ~]
hellod.service
  Loaded: loaded
  Active: active (running)
  Process: 3
  Main PID: 3
  CGroup: /sys/fs/cgroup/systemd/hellod.service

Feb 17 18:29 [root@rhel7 ~]
Feb 17 18:29 [root@rhel7 ~]#
systemctl vs service

• List services:

[root@rhel6 ~]# service --status-all
abrt-ccpp hook is installed
abrt daemon (pid 1652) is running...
abrt-dump-oops is stopped
acpid (pid 1440) is running...
atd (pid 1675) is running...
auditd (pid 1106) is running...
automount (pid 1518) is running...
certmonger (pid 1704) is running...
Stopped
cgred is stopped

[root@rhel7 ~]# systemctl --type service --state active
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
abrt daemon.service loaded active running ABRT Automated Bug Reporting
accounts-daemon.service loaded active running Accounts Service
alsa-state.service loaded active running Manage Sound Card State (res
Managing Services: Enable / Disable

With init:

$ chkconfig unit {on, off}

With systemctl:

$ systemctl {enable, disable, mask, unmask} unit [unit...]

mask — “This will link these units to /dev/null, making it impossible to start them. This is a stronger version of disable, since it prohibits all kinds of activation of the unit, including manual activation. Use this option with care.”
Systemctl vs chkconfig

List all services:

```
[root@rhel6 ~]# chkconfig --list
abrt-ccpp  0:off  1:off  2:off  3:on  4:off  5:on  6:off
abrttd     0:off  1:off  2:off  3:on  4:off  5:on  6:off
acpid      0:off  1:off  2:on   3:on  4:on   5:on  6:off
atd        0:off  1:off  2:off  3:on  4:off  5:on  6:off
auditd     0:off  1:off  2:on  3:on  4:off  5:on  6:off
autofs     0:off  1:off  2:off  3:on  4:off  5:on  6:off
blk-availability 0:off  1:on  2:off  3:on  4:off  5:off  6:off
certmonger 0:off  1:off  2:off  3:on  4:off  5:on  6:off
```

```
[root@rhel7 ~]# 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
abrttd.service   enabled
accounts-daemon.service enabled
alsa-restore.service static
alsa-state.service  static
```
# systemctl

Lots of options...

```
[root@rhel7 ~]# systemctl
cancel           is-active            reload-or-restart
condreload       is-enabled           reload-or-try-restart
condrestart      is-failed           rescue
condstop         isolate              reset-failed
daemon-reexec    kexec                restart
daemon-reload    kill                 set-default
default          link                 set-environment
delete           list-dependencies    set-property
disable          list-jobs            show
emergency        list-sockets        show-environment
enable           list-unit-files     snapshot
exit             list-units           start
force-reload     mask                 status
get-default      poweroff             stop
halt             preset               suspend
help             reboot               try-restart
hibernate        reenable            unmask
hybrid-sleep     reload               unset-environment
```
Lots of new commands...

<table>
<thead>
<tr>
<th>Command</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>systemctl-analyze</td>
<td>systemd-logind</td>
</tr>
<tr>
<td>systemctl-ask-password</td>
<td>systemd-machine-id-setup</td>
</tr>
<tr>
<td>systemctl-cat</td>
<td>systemd-notify</td>
</tr>
<tr>
<td>systemctl-cgls</td>
<td>systemd-nsspawn</td>
</tr>
<tr>
<td>systemctl-cgtop</td>
<td>systemd-run</td>
</tr>
<tr>
<td>systemctl-coredumpctl</td>
<td>systemd-stdio-bridge</td>
</tr>
<tr>
<td>systemctl-delta</td>
<td>systemd-sysv-convert</td>
</tr>
<tr>
<td>systemctl-detect-virt</td>
<td>systemd-tmpfiles</td>
</tr>
<tr>
<td>systemctl-inhibit</td>
<td>systemd-tty-ask-password-agent</td>
</tr>
</tbody>
</table>
**systemd Dependencies**

- Define order and requirements for each unit
- Example: `nfs-lock.service`
  
  Requires=`rpcbind.service network.target`
  
  After=`network.target named.service rpcbind.service`
  
  Before=`remote-fs-pre.target`

- No more semi-arbitrary 00-99 ASCII order loading
Parallel, Not Serial

- Allows for Faster Start Up and Shutdown
- Efficiently Use System Resources

[root@rhel7 ~]# systemctl-analyze plot > /tmp/boot_plot.svg
Boot Process

- Boot path determined by default.target

Let’s track it backwards!

```
[root@rhel7 ~]# systemctl get-default
graphical.target
```

```
[root@rhel7 ~]# grep -v '^#' /usr/lib/systemd/system/graphical.target
[Unit]
Description=Graphical Interface
Documentation=man:systemd.special(7)
Requires=multi-user.target
After=multi-user.target
Conflicts=rescue.target
Wants=display-manager.service
AllowIsolate=yes
```
Boot Process

- graphical.target requires multi-user.target...

```
[root@rhel7 ~]# grep -v '^#' /usr/lib/systemd/system/multi-user.target

[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
```
Boot Process

• Which requires basic.target...

```
[root@rhel7 ~]# grep -v '^#' /usr/lib/systemd/system/basic.target

[Unit]
Description=Basic System
Documentation=man:systemd.special(7)
Requires=sysinit.target
Wants=sockets.target timers.target paths.target slices.target
After=sysinit.target sockets.target timers.target paths.target slices.target
```

• Which requires sysinit.target...

```
[root@rhel7 ~]# grep -v '^#' /usr/lib/systemd/system/sysinit.target

[Unit]
Description=System Initialization
Documentation=man:systemd.special(7)
Conflicts=emergency.service emergency.target
Wants=local-fs.target swap.target
After=local-fs.target swap.target emergency.service emergency.target
```
Boot Process

Which wants local-fs-pre.target and swap.target...

```
[root@rhel7 ~]# grep -v '^#' /usr/lib/systemd/system/local-fs-pre.target

[Unit]
Description=Local File Systems (Pre)
Documentation=man:systemd.special(7)
RefuseManualStart=yes
```

```
[root@rhel7 ~]# grep -v '^#' /usr/lib/systemd/system/swap.target

[Unit]
Description=Swap
Documentation=man:systemd.special(7)
```

• End of the line!
Targets then loaded from the beginning..

But, how does this work for starting individual services?
Boot Process – Services/Units

- Target “Wants” Directories:
  
  /usr/lib/systemd/system/<name>.target.wants/
  /etc/systemd/system/<name>.target.wants/

- Files are symlinks to actual unit files
- Empty target wants directories are placeholders
Example for multi-user.target.wants:

```
[root@rhel7 ~]# ls /usr/lib/systemd/system/multi-user.target.wants
bradbot.path  Plymouth-quit.service  systemd-logind.service
dbus.service  Plymouth-quit-wait.service  systemd-user-sessions.service
getty.target  systemd-ask-password-wall.path
[root@rhel7 ~]# ls /etc/systemd/system/multi-user.target.wants
abrt-ccpp.service  hypervkvpd.service  postfix.service
abrttd.service  hypervvssd.service  remote-fs.target
abrt-oops.service  irqbalance.service  rhsmcertd.service
abrt-vmcore.service  kdump.service  rngd.service
abrt-xorg.service  ksm.service  rsyslog.service
atd.service  ksmtuned.service  smartd.service
auditd.service  libstoragemgmt.service  sshd.service
avahi-daemon.service  libvirtd.service  sysstat.service
chronyd.service  mariadb.service  tuned.service
crond.service  mdmonitor.service  vmtoolsd.service
cups.path  ModemManager.service
httpd.service  NetworkManager.service
```
Exploring dependencies

List all services by target:

[root@rhel7 ~]# systemctl list-dependencies multi-user.target --no-pager
multi-user.target
  - abrt-ccpp.service
  - abrt-oops.service
  - abrt-vmcore.service
  - basic.target
    - alsa-restore.service
    - alsa-state.service
    - paths.target
    - slices.target
      - .slice
        - system.slice
      - sockets.target
        - avahi-daemon.socket
        - cups.socket
      - timers.target
        - systemd-tmpfiles-clean.timer
    - getty.target
      - getty@tty1.service
      - remote-fs.target
Analyzing Boot

• Each unit is tracked during start up

[root@rhel7 ~]# systemctl-analyze blame --no-pager
  2.598s mariadb.service
  1.459s kdump.service
  868ms Plymouth-quit-wait.service
  867ms postfix.service
  510ms firewalld.service
  397ms network.service
  380ms httpd.service
  347ms boot.mount
  311ms tuned.service
  245ms lvm2-monitor.service
  237ms libvirtd.service
  232ms accounts-daemon.service
  203ms systemd-vconsole-setup.service
  203ms ModemManager.service
  168ms avahi-daemon.service
  167ms systemd-logind.service
  156ms rtkit-daemon.service
  127ms chronyd.service
Targets are the new Runlevels

Targets != Runlevels – some equivalency

<table>
<thead>
<tr>
<th>Traditional Runlevel</th>
<th>Equivalent Target</th>
<th>Symlink Target</th>
</tr>
</thead>
<tbody>
<tr>
<td>Runlevel 0</td>
<td>poweroff.target</td>
<td>runlevel0.target</td>
</tr>
<tr>
<td>Runlevel 1</td>
<td>rescue.target</td>
<td>runlevel1.target</td>
</tr>
<tr>
<td>Runlevel 2</td>
<td>multi-user.target</td>
<td>runlevel2.target</td>
</tr>
<tr>
<td>Runlevel 3</td>
<td>multi-user.target</td>
<td>runlevel3.target</td>
</tr>
<tr>
<td>Runlevel 4</td>
<td>multi-user.target</td>
<td>runlevel4.target</td>
</tr>
<tr>
<td>Runlevel 5</td>
<td>graphical.target</td>
<td>runlevel5.target</td>
</tr>
<tr>
<td>Runlevel 6</td>
<td>reboot.target</td>
<td>runlevel6.target</td>
</tr>
</tbody>
</table>

• Targets can and will contain other targets
## Common Targets

<table>
<thead>
<tr>
<th>Target</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>graphical.target</td>
<td>Supports multiple users, graphical and text-based logins</td>
</tr>
<tr>
<td>multi-user.target</td>
<td>Supports multiple users, text-based logins only</td>
</tr>
<tr>
<td>rescue.target</td>
<td>Single user, local file systems mounted and basic system initialization completed, networking is not activated</td>
</tr>
<tr>
<td>emergency.target</td>
<td>Single user, root file system is mounted read-only, only a few essential services are started, networking is not activated</td>
</tr>
</tbody>
</table>

- Rescue and Emergency require root password!
Working with Targets

Viewing the default target:

```
[root@rhel7 ~]# systemctl get-default
multi-user.target
```

Setting default target:

```
[root@rhel7 ~]# systemctl set-default graphical.target
rm '/etc/systemd/system/default.target'
ln -s '/usr/lib/systemd/system/graphical.target' '/etc/systemd/system/default.target'
```

Default target is just a symlink:

```
[root@rhel7 ~]# ls -al /etc/systemd/system/default.target
lrwxrwxrwx. 1 root root 40 Feb 22 21:17 /etc/systemd/system/default.target -> /usr/lib/systemd/system/graphical.target
```

```
[root@rhel7 ~]#"
Changing currently loaded target:

[root@rhel7 ~]# systemctl isolate graphical.target
[root@rhel7 ~]#

Changing to rescue mode:

[root@rhel7 ~]# systemctl rescue

Broadcast message from mruzicka@rhel7.mruzicka on pts/0 (Sat 2015-02-14 19:48:43 EST):

The system is going down to rescue mode NOW!

Changing to emergency mode without sending message:

[root@rhel7 ~]# systemctl --no-wall emergency
Working with Targets

View list of currently loaded targets:

```
[root@rhel7 ~]# systemctl list-units --type target
```

Results pipe to less by default: (can use --no-pager)

<table>
<thead>
<tr>
<th>UNIT</th>
<th>LOAD</th>
<th>ACTIVE</th>
<th>SUB</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>basic.target</td>
<td>loaded</td>
<td>active</td>
<td>active</td>
<td>Basic System</td>
</tr>
<tr>
<td>cryptsetup.target</td>
<td>loaded</td>
<td>active</td>
<td>active</td>
<td>Encrypted Volumes</td>
</tr>
<tr>
<td>getty.target</td>
<td>loaded</td>
<td>active</td>
<td>active</td>
<td>Login Prompts</td>
</tr>
<tr>
<td>local-fs-pre.target</td>
<td>loaded</td>
<td>active</td>
<td>active</td>
<td>Local File Systems (Pre)</td>
</tr>
<tr>
<td>local-fs.target</td>
<td>loaded</td>
<td>active</td>
<td>active</td>
<td>Local File Systems</td>
</tr>
<tr>
<td>multi-user.target</td>
<td>loaded</td>
<td>active</td>
<td>active</td>
<td>Multi-User System</td>
</tr>
<tr>
<td>network.target</td>
<td>loaded</td>
<td>active</td>
<td>active</td>
<td>Network</td>
</tr>
<tr>
<td>paths.target</td>
<td>loaded</td>
<td>active</td>
<td>active</td>
<td>Paths</td>
</tr>
<tr>
<td>remote-fs.target</td>
<td>loaded</td>
<td>active</td>
<td>active</td>
<td>Remote File Systems</td>
</tr>
<tr>
<td>slices.target</td>
<td>loaded</td>
<td>active</td>
<td>active</td>
<td>Slices</td>
</tr>
<tr>
<td>sockets.target</td>
<td>loaded</td>
<td>active</td>
<td>active</td>
<td>Sockets</td>
</tr>
<tr>
<td>sound.target</td>
<td>loaded</td>
<td>active</td>
<td>active</td>
<td>Sound Card</td>
</tr>
<tr>
<td>swap.target</td>
<td>loaded</td>
<td>active</td>
<td>active</td>
<td>Swap</td>
</tr>
</tbody>
</table>

lines 1-14/23 61%
# Shutting Down, Suspending, Etc.

<table>
<thead>
<tr>
<th>Old Command</th>
<th>New Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>halt</td>
<td>systemctl halt</td>
<td>Halts the system</td>
</tr>
<tr>
<td>poweroff</td>
<td>systemctl poweroff</td>
<td>Powers off the system</td>
</tr>
<tr>
<td>reboot</td>
<td>systemctl reboot</td>
<td>Restarts the system</td>
</tr>
<tr>
<td>pm-suspend</td>
<td>systemctl suspend</td>
<td>Suspends the system</td>
</tr>
<tr>
<td>pm-hibernate</td>
<td>systemctl hibernate</td>
<td>Hibernates the system</td>
</tr>
<tr>
<td>pm-suspend-hybrid</td>
<td>systemctl hybrid-sleep</td>
<td>Hibernates and suspends the system</td>
</tr>
</tbody>
</table>

```
[root@rhel7 ~]# ls -al /usr/sbin/shutdown
lrwxrwxrwx. 1 root root 16 Feb 13 17:00 /usr/sbin/shutdown -> ../bin/systemctl
[root@rhel7 ~]#
```
systemd-cgtop

Show top control groups by their resource usage:

```
[root@rhel7 ~]# systemd-cgtop
```

<table>
<thead>
<tr>
<th>Path</th>
<th>Tasks</th>
<th>%CPU</th>
<th>Memory</th>
<th>Input/s</th>
<th>Output/s</th>
</tr>
</thead>
<tbody>
<tr>
<td>/</td>
<td>453</td>
<td>20.9</td>
<td>19.3G</td>
<td>0B</td>
<td>11.8K</td>
</tr>
<tr>
<td>/machine.slice</td>
<td>-</td>
<td>2.7</td>
<td>132.1M</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>/machine.slice...tance\x2d00000017.scope</td>
<td>2</td>
<td>2.7</td>
<td>132.1M</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>/machine.slice...00000017.scope/emulator</td>
<td>2</td>
<td>2.7</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>/machine.slice...x2d00000017.scope/vcpu0</td>
<td>1</td>
<td>0.0</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>/system.slice/auditd.service</td>
<td>1</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>/system.slice/avahi-daemon.service</td>
<td>2</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

- May need to enable accounting – perfect

```
[root@rhel7 ~]# vi /etc/systemd/system/mariadb.service.d/accounting.conf
```

```
[Service]
CPUAccounting=1
MemoryAccounting=1
BlockAccounting=1
```
systemd-cgls

Recursively show control group contents:

```bash
[root@rhel7 ~]# systemd-cgls
```

```
-1 /usr/lib/systemd/systemd --switched-root --system --deserialize 23
- user.slice
  - user-1000.slice
    - session-2.scope
      - 311 -bash
    - 2830 sshd: mruzicka [priv
    - 2866 sshd: mruzicka@pts/1
    - 2867 -bash

- system.slice
  - systemd-localed.service
    - 1810 /usr/lib/systemd/systemd-localed
  - colord.service
    - 1644 /usr/libexec/colord
  - upower.service
    - 1145 /usr/libexec/upowerd
  - polkit.service
    - 680 /usr/lib/polkit-1/polkitd --no-debug
```
systemd Logging: journalctl
Improved Logging

- Don’t need to wait for syslog to start
- No More Losing STDERR and STDOUT
- More detail than classic syslog alone
- Logging with metadata
- Improved debugging and profiling
journalctl

- Does not replace rsyslog in RHEL 7
  - rsyslog is enabled by default
- The journal is not persistent by default.
  - Enable persistence: `mkdir /var/log/journal`
- Stored in key-value pairs
  - `journalctl [tab] [tab]`
  - Man 7 systemd.journal-fields
- Collects event metadata along with the message
- Simple to filter
  - Interleave units, binaries, etc.
Using the Journal

- Tail the journal: `journalctl -f`
- Show X number of lines: `journalctl -n 50`
- View from boot: `journalctl -b`
- Filter by priority: `journalctl -p [level]`

<table>
<thead>
<tr>
<th>Level</th>
<th>Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>emerg</td>
</tr>
<tr>
<td>1</td>
<td>alert</td>
</tr>
<tr>
<td>2</td>
<td>crit</td>
</tr>
<tr>
<td>3</td>
<td>err</td>
</tr>
<tr>
<td>4</td>
<td>warning</td>
</tr>
<tr>
<td>5</td>
<td>notice</td>
</tr>
<tr>
<td>6</td>
<td>debug</td>
</tr>
</tbody>
</table>
journalctl

View basic logs:

```
[root@rhel7 ~]# journalctl
-- Logs begin at Tue 2015-02-17 17:56:24 EST, end at Tue 2015-02-17 22:01:01 EST
Feb 17 17:56:24 rhel7.mruzicka systemd-journald[90]: Runtime journal is using 6.2
Feb 17 17:56:24 rhel7.mruzicka systemd-journald[90]: Runtime journal is using 6.2
Feb 17 17:56:24 rhel7.mruzicka kernel: Initializing cgroup subsys cpuset
Feb 17 17:56:24 rhel7.mruzicka kernel: Initializing cgroup subsys cpu
Feb 17 17:56:24 rhel7.mruzicka kernel: Initializing cgroup subsys cpuacct
Feb 17 17:56:24 rhel7.mruzicka kernel: Linux version 3.10.0-229.el7.x86_64 (mock
Feb 17 17:56:24 rhel7.mruzicka kernel: Command line: BOOT_IMAGE=/vmlinuz-3.10.0-
```

- Time stamps converted to system local time zone
- All logged data is shown, including rotated logs
- Non-persistent by default, can be preserved
journalctl

View most recent logs: (-f to follow)

```
[root@rhel7 ~]# journalctl -n 10
-- Logs begin at Tue 2015-02-17 17:56:24 EST, end at Tue 2015-02-17 22:05:37 EST
Feb 17 22:00:21 rhel7.mruzicka dbus[623]: [system] Successfully activated service
Feb 17 22:01:01 rhel7.mruzicka systemd[1]: Created slice user-0.slice.
Feb 17 22:01:01 rhel7.mruzicka systemd[1]: Starting Session 37 of user root.
Feb 17 22:01:01 rhel7.mruzicka systemd[1]: Started Session 37 of user root.
Feb 17 22:01:01 rhel7.mruzicka CROND[24501]: (root) CMD (run-parts /etc/cron.hourly)
Feb 17 22:01:01 rhel7.mruzicka run-parts(/etc/cron.hourly)[24507]: starting Oana
Feb 17 22:01:01 rhel7.mruzicka run-parts(/etc/cron.hourly)[24513]: finished Oana
Feb 17 22:01:01 rhel7.mruzicka run-parts(/etc/cron.hourly)[24515]: starting Yum
Feb 17 22:01:01 rhel7.mruzicka run-parts(/etc/cron.hourly)[24519]: finished Yum
Feb 17 22:05:37 rhel7.mruzicka [24590]: blah blah blah
```

- Can force stdout/stderr to write to journal with systemd-cat if wanted

```
[root@rhel7 ~]# systemctl-cat echo 'blah blah blah'
```
journalctl

Filter by priority:

Filter by time and priority:

- Advanced filtering by field, UID, unit, etc..
Using journalctl

• Other useful filters:
  - -r reverse order
  - -u [unit]
  - binary e.g. /usr/sbin/dnsmasq [additional binaries]
  - --since=yesterday or YYYY-MM-DD (HH:MM:SS)
  - --until=YYYY-MM-DD

• View entire journal
  - journalctl -o verbose (useful for grep)
Systemd Journal

How to enable persistent logging for the systemd journal

- https://access.redhat.com/solutions/696893

System Administrator's Guide


Lennart Poettering - The systemd Journal

- https://www.youtube.com/watch?v=i4CACB7paLc
systemd - Review
Review: systemd

- Replaces init and does much more
- It is here and it’s powerful
- New boot and root password reset process
- New commands and functionality
- Plenty of great information and resources available
Start using the new commands

Bash Completion is your friend!
- `# yum install bash-completion`

systemd Cheat Sheet for Red Hat Enterprise Linux 7
- [https://access.redhat.com/articles/systemd-cheat-sheet](https://access.redhat.com/articles/systemd-cheat-sheet)

Common Administrative Commands in RHEL 5, 6, & 7
- [https://access.redhat.com/articles/1189123](https://access.redhat.com/articles/1189123)
Compatibility

- Systemd maintains 99% backwards compatibility with LSB compatible initscripts and the exceptions are well documented.

- While we do encourage everyone to convert legacy scripts to service unit files, it's not a requirement.

- Incompatibilities are listed here:
  http://www.freedesktop.org/wiki/Software/systemd/Incompatibilities/

- Converting SysV Init Scripts:
  http://0pointer.de/blog/projects/systemd-for-admins-3.html
Systemd Resources

- RHEL 7 documentation: https://access.redhat.com/site/documentation/Red_Hat_Enterprise_Linux/
- Systemd project page: http://www.freedesktop.org/wiki/Software/systemd/
- Lennart Poettering's systemd blog entries: (read them all) http://0pointer.de/blog/projects/systemd-for-admins-1.html
- Red Hat System Administration II & III (RH134/RH254) http://redhat.com/training/
- Systemd FAQ
- Tips & Tricks