Red Hat Summit

Demystifying systemd

Strengthening service concepts and management in Red Hat Enterprise Linux 8

Ben Breard
Principal Product Manager

Herr Lennart Poettering
Sr. Consulting Engineer

Red Hat

Agenda

- Concepts and unit files
- Security and sandboxing
- Resource management
- Unprivileged units
- Miscellaneous awesome stuff

systemd highlights

New Project Logo!

https://brand.systemd.io/

Red Hat Enterprise Linux 8

Security

- Improved sandboxing and isolation options for services
- Unprivileged unit files
- Additional hardening of systemd services
- Dynamic users

Usability

- Many improvements to systemctl, journalctl, etc.
- Additional service and unit files settings
- Resource management using cgroup v2
- Better journal compression and performance

New in 8.2

- Fine-grain NUMA scheduling and policy support
- CPUSet with cgroup v2
- Additional security controls

Concepts and unit files

Unit files

```
[Unit]
Description=The Apache HTTP Server
Wants=httpd-init.service
After=network.target remote-fs.target nss-lookup.target httpd-init.service
Documentation=man:httpd.service(8)

[Service]
Type=notify
Environment=LANG=C

ExecStart=/usr/sbin/httpd $OPTIONS -DFOREGROUND
ExecReload=/usr/sbin/httpd $OPTIONS -k graceful
KillSignal=SIGWINCH
KillMode=mixed
PrivateTmp=true

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

Unit types

- foo**.service**
- bar**.socket**
- baz**.device**
- qux**.mount**
- waldo**.automount**
- thud**.swap**
- grunt**.target**
- snork**.timer**
- grault**.path**
- pizza**.slice**
- tele**.scope**

Unit file locations

Maintainer

```
/usr/lib/systemd/system
```

Administrator

```
/etc/systemd/system
```

Non-persistent, runtime

```
/run/systemd/system
```

Identify and compare overriding unit files

```
systemd-delta
```

Note

Unit files in `/etc` take precedence over `/run`, and `/run` over `/usr`

Basic usage

`systemctl` - Primary command for interacting with systemd

- e.g. start, stop, reload, restart, enable, disable, status
- `systemctl enable --now httpd`
- `systemctl set-property --runtime CPUShares=2048 httpd`

List loaded services:

`systemctl -t service`

List installed services (similar to chkconfig):

`systemctl list-unit-files -t service`

Execute on remote system via SSH

`systemctl -H [hostname] [command]`

`journalctl` - View and filter the system journal

- `journalctl -fu chronyd`
- `journalctl --grep`

Targets replace runlevels

- multi-user.target ≅ runlevel3
- graphical.target ≅ runlevel5

Change target at runtime

`systemctl isolate [target]`

Configure default target

`systemctl set-default [target]`

Security and sandboxing

Securing units

Reduce system attack surface per unit

- Namespace isolation
- Syscall filters
- Linux capabilities

Provides container-style isolation for traditional services

Simple to apply as another layer of security for systems

https://www.redhat.com/sysadmin/mastering-systemd

Security made simple

1 `systemctl edit [unit.service]`

2 Use `$EDITOR` to insert the following:

```
[Service]
ProtectSystem=strict
ProtectHome=1
PrivateDevices=1
ProtectKernelTunables=1
ProtectKernelModules=1
ProtectControlGroups=1
SystemCallFilter=@system-service
SystemCallErrorNumber=EPERM
NoNewPrivileges=1
PrivateTmp=1
```

3 `:wq`

4 `systemctl restart [unit]`

https://www.freedesktop.org/software/systemd/man/systemd.exec.html

Systemd analyze security

```
  Name                                  Description                                                     Exposure

✗ PrivateNetwork=                       Service has access to the host's network                          0.5
✗ User=/DynamicUser=                    Service runs as root user                                         0.4
✗ RestrictNamespaces=~CLONE_NEWUSER     Service may create user namespaces                                0.3
✗ RestrictAddressFamilies=~…            Service has network configuration privileges                      0.3
✗ CapabilityBoundingSet=~CAP_NET_ADMIN  Service may allocate exotic sockets                               0.2
✓ CapabilityBoundingSet=~CAP_RAWIO      Service has no raw I/O access
✗ CapabilityBoundingSet=~CAP_SYS_MODULE Service may load kernel modules                                   0.2
✗ CapabilityBoundingSet=~CAP_SYS_TIME   Service processes may change the system clock                     0.2
✓ DeviceAllow=                          Service has a minimal device ACL
✗ IPAddressDeny=                        Service does not define an IP address whitelist                   0.2
✓ KeyringMode=                          Service doesn't share key material with other services
✗ NoNewPrivileges=                      Service processes may acquire new privileges                      0.2
✓ NotifyAccess=                         Service child processes cannot alter service state
✓ PrivateDevices=                       Service has no access to hardware devices
✓ PrivateMounts=                        Service cannot install system mounts
✓ PrivateTmp=                           Service has no access to other software's temporary files         0.2
✗ PrivateUsers=                         Service has access to other users                                 0.2
✗ ProtectControlGroups=                 Service may modify to the control group file system
✓ ProtectHome=                          Service has no access to home directories                         0.2
✗ ProtectKernelModules=                 Service may load or read kernel modules                           0.2
✗ ProtectKernelTunables=                Service may alter kernel tunables                                 0.1
✗ ProtectSystem=                        Service has very limited write access to the OS file hierarchy
✓ SystemCallFilter=~@clock              System call whitelist defined for service, and @clock is not included
✓ SystemCallFilter=~@debug              System call whitelist defined for service, and @debug is not included
✓ SystemCallFilter=~@module             System call whitelist defined for service, and @module is not included
✓ SystemCallFilter=~@mount              System call whitelist defined for service, and @mount is not included
✓ SystemCallFilter=~@raw-io             System call whitelist defined for service, and @raw-io is not included
✓ SystemCallFilter=~@reboot             System call whitelist defined for service, and @reboot is not included
✓ SystemCallFilter=~@swap               System call whitelist defined for service, and @swap is not included
✗ SystemCallFilter=~@privileged         System call whitelist defined for service, and @privileged is included      0.2
  ----truncated----
  → Overall exposure level for httpd.service: 6.7
  MEDIUM
```

Securing units

New in Red Hat Enterprise Linux 8

`ProtectKernelTuneables=`

- Disable modification to `/proc` and `/sys`

`ProtectKernelModules=`

- Prohibit load/unload of modules
- Masks `/usr/lib/modules`

`ProtectControlGroups=`

- Disable write access to `/sys/fs/cgroup`

`RestrictSUIDSGID=` (new w/ 8.2)

- Prohibit the creation of SUID/SGID files

`RestrictNamespaces=`

- Boolean to restrict all or a subset of namespace

```
cgroup ipc net mnt pid user uts
```

`AssertSecurity=`

```
uefi-secureboot selinux
```

Red Hat

Securing units

New in Red Hat Enterprise Linux 8

`MemoryDenyWriteExecute=`

- Disable memory mapping that is simultaneously writable and executable

`DynamicUser=`

- (Restrictions apply to stateful data)
- Dynamically allocated UID/GID (61184 - 65519)
- `/etc/[passwd,group]` are not altered and users are removed when the service stops

`PrivateMounts=`

- Service is run in a private mount namespace

`RestrictRealtime=`

- Prohibit real-time scheduling

`RemoveIPC=`

- Remove semaphores, shared memory, and message queues

Securing units

New in Red Hat Enterprise Linux 8

`SystemCallFilter=`

- seccomp filtering to whitelist/blacklist individual or groups of syscalls

`@aio`	`@file-system`	`@mount`	`@reboot`	`@system-service`
`@basic-io`	`@io-event`	`@network-io`	`@resources`	`@timer`
`@chown`	`@ipc`	`@obsolete`	`@setuid`	
`@clock`	`@keyring`	`@privileged`	`@signal`	
`@cpu-emulation`	`@memlock`	`@process`	`@swap`	
`@debug`	`@module`	`@raw-io`	`@sync`	

https://www.freedesktop.org/software/systemd/man/systemd.exec.html

Resource management

Slice, scopes, and services

Slice

- Unit type for creating the cgroup hierarchy for resource management

Scope

- Organizational unit that groups a services' worker processes

Service

- Process or group of processes controlled by systemd

Control groups

Red Hat Enterprise Linux 8

cgroup v1—the default

Well supported in the Linux ecosystem for over a decade

Same basic behavior as Red Hat Enterprise Linux 7

- systemd uses cgroups labels by default
- Accounting is opt-in for CPU & BlockIO

Memory and tasks accounting is now enabled by default

Same unit file options available

- `CPUAccounting=` `*CPUShares=` `CPUQuota=`
- `MemoryAccounting=` `*MemoryLimit=`
- `*BlockIOAccounting=` `*BlockIOWeight=` `*BlockIODeviceWeight=`
- `TasksAccounting=, TasksMax=`

*=deprecated

https://www.kernel.org/doc/Documentation/cgroup-v1/cgroups.txt

Control groups

Red Hat Enterprise Linux 8

cgroup v2– Full Support in 8.2

Unified hierarchy with vastly improved controllers

- Delivers more coherent and holistic resource management

Perfectly integrated with systemd

- Ecosystem in-progress
- Fedora 31 switched to v2 by default
- Likely to become the default in the next major release of RHEL

Append

- `systemd.unified_cgroup_hierarchy` to kernel

Best effort translation for relevant controllers

- `CPUWeight=` replaces `CPUShares=`
- `MemoryMax=` replaces `MemoryLimit=`
- `IO*=` replaces `BlockIO*=`

https://www.kernel.org/doc/Documentation/cgroup-v2.txt

Resource management

Configuration

Configure cgroup attributes

```
systemctl set-property --runtime httpd CPUShares=2048
```

Drop “--runtime” to persist (will create a drop-in):

```
systemctl set-property httpd CPUShares=2048
```

Or place in the unit file:

```
[Service]
CPUShares=2048
```

http://0pointer.de/blog/projects/resources.html

cgroup v2 controls

Red Hat Enterprise Linux 8

```
CPUWeight= CPUStartupWeight=
CPUQuota=
```

```
AllowedCPUs=
AllowedMemoryNodes=
```

New in 8.2!

```
MemoryMin=
MemoryLow=
MemoryHigh=
MemoryMax=
MemorySwapMax=
```

```
IODeviceLatencyTargetSec=
IOWeight=
IODeviceWeight=
IOReadBandwidthMax= IOWriteBandwidthMax=
IOReadIOPSMax= IOWriteIOPSMax=
```

Resource management

NUMA Scheduling - new in 8.2!

NUMAMask=

- List of NUMA nodes e.g. NUMAMask=0,1

NUMAPolicy=

- Memory policy for executed process: default, preferred, bind, interleave and local

Align with CPUAffinity= for performance & efficiency

```
CPUAffinity=12-23 NUMAMask=1 NUMAPolicy=bind
```

Or place in the unit file:

```
[Service]
CPUAffinity=0-11,12-23
NUMAMask=0-1
NUMAPolicy=interleave
```

https://www.freedesktop.org/software/systemd/man/systemd.exec.html#NUMAMask=

Unprivileged units

systemd --user

```
$ systemctl --user status
● localhost.localdomain
    State: running
     Jobs: 0 queued
   Failed: 0 units
    Since: Sat 2019-03-09 15:29:52 CST; 31min ago
   CGroup:
/user.slice/user-1000.slice/user@1000.service
           └─init.scope
             ├─1420 /usr/lib/systemd/systemd--user
             └─1427 (sd-pam)
```

systemd --user

User units

```
~/.config/systemd/user
```

Maintainer user units

```
/usr/lib/systemd/user &
~/.local/share/systemd/user
```

Global user units (all users)

```
/etc/systemd/user
```

Note:

`.bashrc` and `.bash_profile` are not sourced by systemd

```
~/.config/environment.d
```

```
systemctl --user import-environment
```

```
systemctl --user show-environment
```

systemd --user

Interact with the systemd user instance

- `systemctl --user`
- `e.g. start, stop, restart, enable, disable, status`
- `systemctl --user enable --now foo.service`

Filter the journal by user unit(s)

- `journalctl --user-unit=foo.service`

Enable/disable systemd user outside of sessions (start on boot)

- `loginctl enable-linger $USER`
- `loginctl disable-linger $USER`

“Shame back” view of user’s disgusting use of system resources

- `loginctl user-status`

Miscellaneous awesome stuff

Power usage

Transient Units

systemd-run [options] command [args]

- Leverage the security & resource management capabilities of systemd for more than typical services, e.g. commands, scripts, etc

```
SEC-HIGH="-p ProtectSystem=strict -p ProtectHome=1 -p PrivateDevices=1 -p ProtectKernelTunables=1 -p
ProtectKernelModules=1 -p ProtectControlGroups=1 -p SystemCallFilter=@system-service -p NoNewPrivileges=1 -p
PrivateTmp=1"

RES-HIGH="-p CPUWeight=500 -p MemoryLow=1G -p MemorySwapMax=0"

RES-LOW="-p CPUWeight=50 -p CPUQuota=20% -p MemoryHigh=1G"

NUMA0="-p CPUAffinity=0-11,24-35 -p NUMAMask=0 -p NUMAPolicy=bind"

NUMA1="-p CPUAffinity=12-23,36-47 -p NUMAMask=1 -p NUMAPolicy=bind"
```

- `systemd-run -P $SEC-HIGH $RES-LOW $NUMA0 /bin/foobar`

https://www.freedesktop.org/software/systemd/man/systemd-run

Power usage

Limit the CPU usage of a task to 15% of one core

- `systemd-run -p CPUQuota=15% /usr/bin/cpuhog`

Wait for the task to complete and provide stats and exit code

- `systemd-run -p CPUAccounting=1 --wait /usr/bin/long-job`

```
Running as unit: run-u1573.service
Finished with result: success
Main processes terminated with:
code=exited/status=0
Service runtime: 30.004s
CPU time consumed: 2ms
```

Schedule a timer

- `systemd-run --on-calendar=18:55 /usr/bin/dinner-is-ready`

Start a shell under an automatically picked, unused UID with read-only fs access

- `systemd-run -p DynamicUser=1 -t /bin/bash`

Miscellaneous awesome stuff

Journal

- Better compression and performance
- Familiar filtering options

```
journalctl --grep=
```

- Additional color coding for log levels

Mount options in `/etc/fstab`

```
x-systemd.growfs
x-systemd.makefs
```

systemctl

- Restart counter for units (Restart=)

```
systemctl show -p NRestarts --value
```

- Create a new unit file

```
systemctl edit --force foo.service
```

- Reboot into UEFI Firmware setup

```
systemctl reboot --firmware-setup
```

Red Hat

Helpful resources

- Red Hat Enterprise Linux documentation
  https://access.redhat.com/site/documentation/Red_Hat_Enterprise_Linux/
- Demystifying systemd 2018
  https://www.youtube.com/watch?v=tY9GYsoxeLg
- Systemd project page
  http://www.freedesktop.org/wiki/Software/systemd/
- Lennart Poettering's systemd blog series: (read them all)
  http://0pointer.de/blog/projects/systemd-for-admins-1.html
- Red Hat System Administration II & III (RH134/RH254
  http://0pointer.de/blog/projects/systemd-for-admins-1.html
- Cgroup v2
  https://www.kernel.org/doc/Documentation/cgroup-v2.txt
- Cgroup v2 @facebook
  https://facebookmicrosites.github.io/cgroup2/docs/overview

systemd FAQ

Tips & Tricks

Red Hat Summit

Thank you

linkedin.com/company/Red-Hat

facebook.com/RedHatinc

youtube.com/user/RedHatVideos

twitter.com/RedHat

Red Hat

Project stats

43,969	Commits
1,564	Contributors
107	systemd releases
31	Releases since Red Hat Enterprise Linux 7

11 years of systemd!

Securing units

`PrivateTmp=`

- File system namespace

```
/tmp & /var/tmp
```

- Files under

```
/tmp/systemd-private-*-[unit]-*/tmp
```

`PrivateNetwork=`

- Creates a network namespace with a single loopback device, private 127.0.0.1

https://www.freedesktop.org/software/systemd/man/systemd.exec.html

`JoinsNamespaceOf=`

- Enables multiple units to share

```
PrivateTmp= & PrivateNetwork=
```

`SELinuxContext=`

- Specify an SELinux security context for the process or service

Securing units

`ProtectSystem=`

- If enabled, `/usr` and `/boot` directories are mounted read-only
- If “full”, `/etc` is also read-only
- **New: strict** - whole system tree is read-only except `/dev`, `/proc`, `/sys`

`ProtectHome=`

- If enabled, `/home`, `/root`, `/run/user` will appear empty
- Alternatively can set to “read-only”
- **New: tmpfs** - masks w/ `tmpfs` mount

https://www.freedesktop.org/software/systemd/man/systemd.exec.html

`PrivateDevices=`

- If enabled, creates a private `/dev` namespace.
- Includes pseudo devices like `/dev/null`, `/dev/zero`, etc
- Disables `CAP_MKNOD`

Securing units

`ReadWriteDirectories=, ReadOnlyDirectories=, InaccessibleDirectories=`

- Configure file system namespaces

`NoNewPrivileges=`

Ensure a process and children cannot elevate privileges

`CapabilityBoundingSet=`

- CAP_SYS_ADMIN
- ~CAP_NET_ADMIN
- man:capabilities(7) for details

`RestrictAddressFamilies=`

- AF_INET AF_INET6 AF_UNIX
- ~AF_PACKET

https://www.freedesktop.org/software/systemd/man/systemd.exec.html

Red Hat

v1 → v2 cheat sheet

v1	Min	Default	Max
`CPUShares=`	2	**1024**	262144
`StartupCPUShares=`	2	**1024**	262144
`MemoryLimit=`	N/A	N/A	N/A
`BlockIOWeight=`	10	**500**	1000

v2	Min	Default	Max
`CPUWeight=`	10	**100**	10000
`StartupCPUWeight=`	10	**100**	10000
`MemoryMax=`	N/A	N/A	N/A
`IOWeight=`	10	**100**	10000

IP accounting and filtering

Technology preview

`IPAccounting=`

- Ingress and egress IP traffic is counted for associated processes
- Applies to services, sockets, and slices
- Requires cgroup v2

`IPAddressAllow=`

- Filtering via cgroups eBPF hooks independent from iptables/nft
- IP/netmask for allowed traffic

`IPAddressDeny=`

- IP/netmask deny list

```
systemd-run -p
IPAddressAllow=10.0.0.5 -p
IPAddressDeny=any -t mysqladm …
```

```
System-wide Example:
systemctl set-property system.slice
IPAddressDeny=any
IPAddressAllow=localhost
```

http://0pointer.net/blog/ip-accounting-and-access-lists-with-systemd.html

Miscellaneous awesome stuff

systemd-run

- `--pipe` use STDIN/STDOUT/STERR with transient units
- `--wait` for it to exit code

Unit files

- `ExecStart` accepts a relative path
- Improved drop-in prefixes
- Clickable links with `--no-pager`

Parse, normalize, and calculate next occurance

```
systemd-analyze calendar:'2019-05-8 11:45:00'
Original form:2019-05-8 11:45:00
Normalized form:2019-05-08 11:45:00
Next elapse:Wed 2019-05-08 11:45:00 EDT
(in UTC): Wed 2019-05-08 15:45:00 UTC
From now: 4 days left
```

Concatenate config files with drop-ins

```
systemd-tmpfiles --cat-config

systemd-analyze cat-config
/etc/systemd/journald.conf
```

Power usage

`systemd-delta` - View overridden config files

`systemd-cgls` - View cgroup hierarchy

`systemd-cgtop` - View cgroup accounting

`systemd-analyze` - Analyze and debugging systemd