DEMYSTIFYING SYSTEMD

Strengthening service concepts and management in Red Hat Enterprise Linux 8

Ben Breard, RHCA
Principal product manager

Herr Lennart Poettering
Sr. consulting engineer
AGENDA

- Concepts and unit files
- Security and sandboxing
- Resource management
- Unprivileged units
- Miscellaneous awesome stuff
<table>
<thead>
<tr>
<th>PROJECT STATS</th>
</tr>
</thead>
<tbody>
<tr>
<td>39,776</td>
</tr>
<tr>
<td>1,129</td>
</tr>
<tr>
<td>181</td>
</tr>
<tr>
<td>20</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>----------------</td>
</tr>
<tr>
<td>Commits</td>
</tr>
<tr>
<td>Contributors</td>
</tr>
<tr>
<td>systemd releases</td>
</tr>
<tr>
<td>Releases since RHEL 7</td>
</tr>
</tbody>
</table>
PROJECT STATS

<table>
<thead>
<tr>
<th>39,776</th>
<th>Commits</th>
</tr>
</thead>
<tbody>
<tr>
<td>1,129</td>
<td>Contributors</td>
</tr>
<tr>
<td>181</td>
<td>systemd releases</td>
</tr>
<tr>
<td>20</td>
<td>Releases since RHEL 7</td>
</tr>
</tbody>
</table>

10 Years of systemd

(Nov 18th 2009)
A BIG YEAR FOR RED HAT
A BIG YEAR FOR SYSTEMD
A BIG YEAR FOR SYSTEMD
A BIG YEAR FOR SYSTEMD
A BIG YEAR FOR SYSTEMD

Here’s what an AI/ML Logo Service
POLL: VOTE FOR YOUR FAVORITE

2nd

systemd

1st

systemd

3rd

systemd

#redhat #rhsummit
SYSTEMD HIGHLIGHTS
Red Hat Enterprise Linux 8

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

Usability
- Many improvements to systemctl, journalctl, etc.
- Additional service & unit files settings
- Resource management using cgroups v2 (tech preview)
- Better journal compression and performance

New technology Previews
- IP accounting and filtering
- Portable system services
CONCEPTS AND 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:

/etc/systemd/system

Administrator:

/usr/lib/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`

**journalctl** - view and filter the system journal
- `journalctl -fu chronyd`

**systemd-run** - use transient unit files

**systemd-analyze** - analyze and debugging systemd

**systemd-cgls** - view cgroup hierarchy

**systemd-cgtop** - view cgroup accounting
BASIC USAGE

Limit the CPU usage of a task to 15% of 1 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 w/ read-only fs access
```
systemd-run -p DynamicUser=1 -t /bin/bash
```
SECURITY SANDBOXING AND CAPABILITIES
SECURING UNITS

Reduce system attack surface per unit
- Namespace isolation
- Syscall filters
- Linux capabilities (breakdown of root perms)

Provides container-style isolation for traditional services

Simple to apply as another layer of security for systems
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

JoinsNamespaceOf=

- Enables multiple units to share
  PrivateTmp= & PrivateNetwork=

SELinuxContext=

- Specify an SELinux security context for the process/service

https://www.freedesktop.org/software/systemd/man/systemd.exec.html
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` will appear empty
- Alternatively can set to “read-only”
- **New: tmpfs** - masks w/ tmpfs mount

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

- Configure file system namespaces
- Ensure a process & children cannot elevate privileges

- NoNewPrivileges=
  - Configure file system namespaces

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

- RestrictAddressFamilies=
  - AF_INET AF_INET6 AF_UNIX
  - ~AF_PACKET
SECURING UNITS
New in Red Hat Enterprise Linux 8

ProtectKernelTuneables=
- Disable modification to /proc & /sys

ProtectKernelModules=
- Prohibit load/unload of modules.
- Masks /usr/lib/modules

ProtectControlGroups=
- Disable write access to /sys/fs/cgroup

RestrictNamespaces=
- Boolean to restrict all or a subset of namespaces
  - cgroup ipc net mnt pid user uts

ConditionSecurity=
- uefi-secureboot selinux
SECURING UNITS
New in Red Hat Enterprise Linux 8

**MemoryDenyWriteExecute=**
- Disable memory mapping that is simultaneously writable & executable

**DynamicUsers=**
- (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, & message queues
## SECURING UNITS

New in Red Hat Enterprise Linux 8

<table>
<thead>
<tr>
<th>@io</th>
<th>@file-system</th>
<th>@mount</th>
<th>@reboot</th>
<th>@system-service</th>
</tr>
</thead>
<tbody>
<tr>
<td>@basic-iod</td>
<td>@io-event</td>
<td>@network-iod</td>
<td>@resources</td>
<td>@timer</td>
</tr>
<tr>
<td>@chown</td>
<td>@ipc</td>
<td>@obsolete</td>
<td>@setuid</td>
<td></td>
</tr>
<tr>
<td>@clock</td>
<td>@keyring</td>
<td>@privileged</td>
<td>@signal</td>
<td></td>
</tr>
<tr>
<td>@cpu-emulation</td>
<td>@memlock</td>
<td>@process</td>
<td>@swap</td>
<td></td>
</tr>
<tr>
<td>@debug</td>
<td>@module</td>
<td>@raw-io</td>
<td>@sync</td>
<td></td>
</tr>
</tbody>
</table>

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

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

#redhat #rhsummit
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]
SYSTEMD-ANALYSE SECURITY (8.1)

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=-~CLONE_NEWUSER Service has network configuration privileges | 0.3
✓ CapabilityBoundingSet=-~CLONE_NEWUSER Service may allocate exotic sockets | 0.2
✓ CapabilityBoundingSet=-~CAP_NET_ADMIN Service has no raw I/O access | 0.2
✓ CapabilityBoundingSet=-~CAP_RAWIO Service may load kernel modules | 0.2
✓ CapabilityBoundingSet=-~CAP_SYS_MODULE Service processes may change the system clock | 0.2
✓ CapabilityBoundingSet=-~CAP_SYS_TIME Service has a minimal device ACL | 0.2
✓ DeviceAllow= Service does not define an IP address whitelist | 0.2
✓ IPAddressDeny= Service doesn't share key material with other services | 0.2
✓ NoNewPrivileges= Service processes may acquire new privileges | 0.2
✓ NotifyAccess= Service child processes cannot alter service state | 0.2
✓ PrivateDevices= Service has no access to hardware devices | 0.2
✓ PrivateMounts= Service cannot install system mounts | 0.2
✓ PrivateTmp= Service has no access to other software's temporary files | 0.2
✓ PrivateUsers= Service has access to other users | 0.2
✓ ProtectDirGroups= Service may modify to the control group file system | 0.2
✓ 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 | 0.2
✓ ProtectSystem= Service has very limited write access to the OS file hierarchy | 0.1
✓ SystemCallFilter=~@clock System call whitelist defined for service, and @clock is not included | 0.2
✓ SystemCallFilter=~@debug System call whitelist defined for service, and @debug is not included | 0.2
✓ SystemCallFilter=~@module System call whitelist defined for service, and @module is not included | 0.2
✓ SystemCallFilter=~@mount System call whitelist defined for service, and @mount is not included | 0.2
✓ SystemCallFilter=~@raw-io System call whitelist defined for service, and @raw-io is not included | 0.2
✓ SystemCallFilter=~@reboot System call whitelist defined for service, and @reboot is not included | 0.2
✓ SystemCallFilter=~@swap System call whitelist defined for service, and @swap is not included | 0.2
✓ SystemCallFilter=~@privileged System call whitelist defined for service, and @privileged is included | 0.2

Overall exposure level for httpd.service: 6.7 MEDIUM
RESOURCE MANAGEMENT
SLICE, SCOPES, AND SERVICES

**Slice**
Unit type for creating the cgroup hierarchy for resource management

**Scope**
Organizational unit that groups a service's worker processes

**Service**
Process or group of processes controlled by systemd
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
```
CONTROL GROUPS
Red Hat Enterprise Linux 8

cgroups 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: (*=deprecated)
  - CPUAccounting=,*CPUShares=,*CPUQuota=
  - MemoryAccounting=,*MemoryLimit=
  - *BlockIOAccounting=,*BlockIOWeight=,*BlockIODEviceWeight=
  - TasksAccounting=,TasksMax=

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

#redhat #rhsummit
cgroups v2—tech preview

- Unified hierarchy with **vastly improved controllers**
  - Delivers more coherent and holistic resource management
- Perfectly integrated with systemd
  - Ecosystem in-progress (virt & containers work remains)
  - Fedora 31 may default to v2
  - Support planned for 8.1 or 8.2

- Append `systemd.unified_cgroup_hierarchy` to kernel
- Best effort translation for relevant controllers:
  - `CPUWeight=` replaces `CPUShares=`
  - `MemoryMax=` replaces `MemoryLimit=`
  - `IO*=` replaces `BlockIO*=`

## V2 CHEAT SHEET

<table>
<thead>
<tr>
<th>v1</th>
<th>Min...Default...Max</th>
<th>v2</th>
<th>Min...Default...Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPUShares=</td>
<td>2...1024...262144</td>
<td>CPUWeight=</td>
<td>10...100...10000</td>
</tr>
<tr>
<td>StartupCPUShares=</td>
<td>2...1024...262144</td>
<td>StartupCPUWeight=</td>
<td>10...100...10000</td>
</tr>
<tr>
<td>MemoryLimit=</td>
<td>N/A</td>
<td>MemoryMax=</td>
<td>N/A</td>
</tr>
<tr>
<td>BlockIOWeight=</td>
<td>10...500...1000</td>
<td>IOWeight=</td>
<td>10...100...10000</td>
</tr>
</tbody>
</table>
CGROUP v2 CONTROLS

Red Hat Enterprise Linux 8

CPUWeight= CPUStartupWeight=
CPUQuota=

MemoryMin=
MemoryLow=
MemoryHigh=
MemoryMax=
MemorySwapMax=

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

https://www.freedesktop.org/software/systemd/man/systemd.resource-control.html
UNPRIVILEGED UNITS
$ 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 & .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
### IP ACCOUNTING AND FILTERING

**Technology Preview**

<table>
<thead>
<tr>
<th>IPAccounting=</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Ingress and egress IP traffic is counted for associated processes</td>
</tr>
<tr>
<td>• Applies to services, sockets, and slices</td>
</tr>
<tr>
<td>• Requires cgroup v2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IPAddressAllow=</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Filtering via cgroups eBPF hooks independent from iptables/nft</td>
</tr>
<tr>
<td>• IP/netmask for allowed traffic</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IPAddressDeny=</th>
</tr>
</thead>
<tbody>
<tr>
<td>• IP/netmask deny list</td>
</tr>
</tbody>
</table>

```
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

Journal
- Better compression & performance
- Familiar filtering options
  - journalctl --grep=
- Additional color coding for log levels
  - Debug entries are light grey

Mount Options in /etc/fstab
- x-systemd.growfs
- x-systemd.makefs

systemctl
- Restart counter for units (Restart=)
  - systemctl show --value
- Create a new unit file:
  - systemctl edit --force foo.service
- Reboot into UEFI Firmware setup:
  - systemctl reboot --firmware-setup
MISCELLANEOUS AWESOME STUFF

systemd-run
  ● --pipe use STDIN/STDOUT/STERR w/ 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 occurrence
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 w/ drop-ins
systemd-tmpfiles --cat-config

systemd-analyze cat-config
/etc/systemd/journald.conf
HELPFUL RESOURCES

- RHEL documentation: https://access.redhat.com/site/documentation/Red_Hat_Enterprise_Linux/
- Demystifying systemd 2018: https://www.youtube.com/watch?v=tY9GYsoxeLq
- 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://redhat.com/training/
- systemd FAQ
- Tips & Tricks
- Cgroups v2: https://www.kernel.org/doc/Documentation/cgroup-v2.txt
- Cgroups v2 @ facebook: https://facebookmicrosites.github.io/cgroup2/docs/overview
THANK YOU

linkedin.com/company/Red-Hat
facebook.com/RedHatInc
youtube.com/user/RedHatVideos
twitter.com/RedHat
PORTABLE SYSTEM SERVICES

Technology Preview

- Similar concepts as containers, except the focus is on the integration of units/apps as part of the system.
- When images or chroots are attached, the relevant units are copied to the system:

  ```
  /usr/lib/systemd/portablectl attach example.raw
  ```
- An alternative to system containers (atomic install in RHEL 7)
- Can use disk images (QCOW, RAW, etc) or chroots

- Services are managed as any other local unit file:

  ```
  systemctl enable --now example.service
  ```

- Detaching images cleans up the units and restores the default state of the system

  ```
  /usr/lib/systemd/portablectl detach example.raw
  ```

http://0pointer.net/blog/walkthrough-for-portable-services.html