systemd
our next-generation init system

PABLO N. HESS
Instructor
Red Hat São Paulo
December 2011
General info & History

Authors:
Lennart Poettering (Red Hat)
Pulseaudio, Avahi
Kay Sievers (openSUSE)
Udev

Current default init for:
Fedora
openSUSE
Mandriva

Future default init for:
Gentoo
Arch
Mageia
Probably everyone else

Spelling:
It’s systemd
not system D
not System D
not SystemD
not system d
Major features

Massively parallel service initialization
Replaces Upstart and SysVinit

On-demand network service initialization
Replaces (x)inetd

On-demand fsck’ing & mounting
Replaces fstab and autofs

On-demand socket-based initialization
Better than Upstart
Motivation: What current init systems do not/can not provide

- Reliable supervisioning
- Reliable dependencies
- Parallel service initialization
- Socket-based initialization
- Better-than-shellscript speeds
- Code deduplication/sanitization
- Low first user PIDs
- Path-based initialization
SysV/Upstart

Daemon process

double fork()

Re-parented to init

daemon stopped

Child not stopped

Reliable supervisioning
Reliable dependencies
Parallel service initialization
Socket-based initialization
Better-than-shellscript speeds
Code deduplication/sanitization
Low first user PIDs
Path-based initialization
Daemon process
double fork()
Re-parented to init
daemon stopped
Whole cgroup stopped

Reliable supervisioning
Reliable dependencies
Parallel service initialization
Socket-based initialization
Better-than-shellscript speeds
Code deduplication/sanitization
Low first user PIDs
Path-based initialization
**SysV/Upstart**

- CUPS starts, needs D-bus
- D-bus started, open socket
- init starts D-bus
- CUPS may send D-bus messages

**Reliable supervisioning**

**Reliable dependencies**

**Parallel service initialization**

**Socket-based initialization**

**Better-than-shellscript speeds**

**Code deduplication/sanitization**

**Low first user PIDs**

**Path-based initialization**
systemd

systemd has already opened needed sockets

CUPS starts, needs D-bus

CUPS may send D-bus messages

D-bus started, retrieves messages

Reliable supervisioning
Reliable dependencies
Parallel service initialization
Socket-based initialization
Better-than-shellscript speeds
Code deduplication/sanitization
Low first user PIDs
Path-based initialization
start() {
  [ -x $exec ] || exit 5
  # Source config
  if [ -f /etc/sysconfig/rsyslog ]; then
    . /etc/sysconfig/rsyslog
  fi
  umask 077
  echo -n "Starting system logger: "
  daemon --pidfile="${PIDFILE}" \
    $exec $SYSLOGD_OPTIONS
  RETVAL=$?
  echo
  [ $RETVAL -eq 0 ] && touch $lockfile
  return $RETVAL
}

ExecStartPre=/bin/systemctl stop \systemd-kmsg-syslogd.service
ExecStart=/usr/sbin/rsyslogd -n -c5
Sockets=syslogd.socket
StandardOutput=null

SystemV

Reliable supervisioning
Reliable dependencies
Parallel service initialization
Socket-based initialization
Better-than-shellscript speeds
Code deduplication/sanitization
Low first user PIDs
Path-based initialization
Rsyslog as an example

sysvinit script
106 lines
75 lines of code

systemd “unit file”
11 lines
9 lines of code

Reliable supervisioning
Reliable dependencies
Parallel service initialization
Socket-based initialization
Better-than-shellscript speeds
Code deduplication/sanitization
Low first user PIDs
Path-based initialization
CUPS as an example

- CUPS started
- Processes job
- File created in /var/spool/cups

Reliable supervisioning
Reliable dependencies
Parallel service initialization
Socket-based initialization
Better-than-shellscript speeds
Code deduplication/sanitization
Low first user PIDs
Path-based initialization
Working with systemd

Everything is a unit:
- home.automount
- rsyslog.service
- sshd.socket
- cups.path

auto-mounted FS
regular service
socket definition
path definition

Targets “want” units
- multi-user.target.wants/
  - postfix.service
  - cron.target
  - sysinit.target

- sysinit.target.wants/
  - remount-rootfs.service
  - quotaon.service
## Invocation: systemd versus SystemV

<table>
<thead>
<tr>
<th>SystemV</th>
<th>systemd</th>
</tr>
</thead>
<tbody>
<tr>
<td><code># service sshd start</code></td>
<td><code># systemctl start sshd.service</code></td>
</tr>
<tr>
<td><code># chkconfig sshd on</code></td>
<td><code># systemctl enable sshd.service</code></td>
</tr>
<tr>
<td><code>add autofs map</code></td>
<td><code># systemctl enable home.automount</code></td>
</tr>
<tr>
<td><code>add fstab entry</code></td>
<td><code># systemctl enable home.mount</code></td>
</tr>
<tr>
<td><code># init 5</code></td>
<td><code># systemctl isolate graphical.target</code></td>
</tr>
</tbody>
</table>
Unit files example: automount & mount

**home.automount**

[Unit]
Description=Automount my /home

[Automount]
Where=/home

[Install]
WantedBy=sysinit.target

**home.mount**

[Unit]
Description=My home directory

[Mount]
#What=UUID=fd6e2ed9-d430-45b3-9...
What=/dev/sdb9
Where=/home
Type=ext4
Options=noatime,discard,nobARRIER

---

**man systemd.automount**

**man systemd.mount**

**triggers home.mount**
Unit files example: swap

**dev-sda5.swap**

```
[Unit]
Description=Swap on /dev/sda5

[Swap]
What=/dev/sda5
Priority=1
TimeoutSec=5

[Install]
WantedBy=swap.target
```

man systemd.swap
Unit files example: services

sshd.service

[Unit]
Description=OpenSSH server daemon.
After=syslog.target network.target auditd.service

[Service]
Type=simple
ExecStart=/usr/sbin/sshd -D
ExecReload=/bin/kill -HUP $MAINPID

[Install]
WantedBy=multi-user.target

or

man systemd.service
Unit files example: services

getty@.service

```
...[Service]
Environment=TERM=linux
ExecStart=-/sbin/agetty %I 38400
Restart=always
RestartSec=0
UtmpIdentifier=%I
TTYPath=/dev/%I
...
```

```
# systemctl --full --no-pager |grep getty
getty@tty2.service
getty@tty3.service
getty@tty4.service
getty@tty5.service
getty@tty6.service
```

man systemd.service
Unit files example: services

**sshd@.service**

```ini
[Unit]
Description=SSH Per-Connection Server
After=syslog.target

[Service]
ExecStart=/usr/sbin/sshd -i
StandardInput=socket
```

**sshd.socket**

```ini
[Unit]
Conflicts=sshd.service

[Socket]
ListenStream=22
ListenStream=2200
Accept=yes

[Install]
WantedBy=sockets.target
```

```
# systemctl --full --no-pager |grep sshd
sshd@192.168.123.241:22-192.168.123.100:50083.service
sshd@192.168.123.241:22-192.168.123.245:35623.service
sshd@192.168.123.241:22-192.168.123.245:35624.service
sshd@192.168.123.241:22-192.168.123.245:60016.service
```
Unit files example: services (oneshot)

iptables.service

[Unit]
Description=IPv4 firewall with iptables
After=syslog.target
ConditionPathExists=/etc/sysconfig/iptables

[Service]
Type=oneshot
RemainAfterExit=yes
ExecStart=/usr/libexec/iptables.init start
ExecStop=/usr/libexec/iptables.init stop
Environment=BOOTUP=serial
Environment=CONSOLETYPE=serial
StandardOutput=syslog
StandardError=syslog

[Install]
WantedBy=basic.target
Unit files example: services (forking)

dnsmasq.service

[Unit]
Description=DNS caching server.
After=syslog.target network.target

[Service]
Type=forking
PIDFile=/var/run/dnsmasq.pid
EnvironmentFile=-/etc/sysconfig/network
ExecStart=/usr/sbin/dnsmasq -s $HOSTNAME

[Install]
WantedBy=multi-user.target

the choice for legacy SysV init scripts
## One dir for the packager

```bash
$ ls /lib/systemd/system

<table>
<thead>
<tr>
<th>Service Name</th>
<th>Service Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>abrt-ccpp.service</td>
<td>poweroff.service</td>
</tr>
<tr>
<td>abrtd.service</td>
<td>poweroff.target</td>
</tr>
<tr>
<td>abrt-oops.service</td>
<td>poweroff.target.wants</td>
</tr>
<tr>
<td>abrt-vmcore.service</td>
<td>pppoe-server.service</td>
</tr>
<tr>
<td>accounts-daemon.service</td>
<td>prefdm.service</td>
</tr>
<tr>
<td>alsa-restore.service</td>
<td>printer.target</td>
</tr>
<tr>
<td>alsa-store.service</td>
<td>proc-sys-fs-binfmt_misc.automount</td>
</tr>
<tr>
<td><a href="mailto:anaconda-shell@.service">anaconda-shell@.service</a></td>
<td>proc-sys-fs-binfmt_misc.mount</td>
</tr>
<tr>
<td>anaconda.target</td>
<td>psacct.service</td>
</tr>
<tr>
<td>arp-ethers.service</td>
<td>quotacheck.service</td>
</tr>
<tr>
<td>atd.service</td>
<td>quotaon.service</td>
</tr>
<tr>
<td>auditd.service</td>
<td>rc-local.service</td>
</tr>
<tr>
<td><a href="mailto:autovt@.service">autovt@.service</a></td>
<td>rdisc.service</td>
</tr>
<tr>
<td>avahi-daemon.service</td>
<td>reboot.service</td>
</tr>
<tr>
<td>avahi-daemon.socket</td>
<td>reboot.target</td>
</tr>
<tr>
<td>basic.target</td>
<td>reboot.target.wants</td>
</tr>
<tr>
<td>basic.target.wants</td>
<td>remote-fs.target</td>
</tr>
<tr>
<td>bluetooth.service</td>
<td>remount-rootfs.service</td>
</tr>
<tr>
<td>bluetooth.target</td>
<td>rescue.service</td>
</tr>
<tr>
<td>canberra-system-bootup.service</td>
<td>rescue.target</td>
</tr>
<tr>
<td>canberra-system-shutdown-reboot.service</td>
<td>restorecond.service</td>
</tr>
<tr>
<td>canberra-system-shutdown.service</td>
<td>rpcbind.target</td>
</tr>
<tr>
<td>chronyd.service</td>
<td>rsyslog.service</td>
</tr>
<tr>
<td>chrony-wait.service</td>
<td>rtkit-daemon.service</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```
One dir for the packager

...and one for the sysadmin

$ ls /lib/systemd/system

$ ls /etc/systemd/system

home.automount
basic.target.wants
bluetooth.target.wants
my-own-target.target.wants
dbus-org.freedesktop.NetworkManager.service
default.target
default.target.wants
getty.target.wants

home.mount
graphical.target.wants
multi-user.target.wants
network.target.wants
printer.target.wants
sockets.target.wants
sysinit.target.wants
Troubleshooting

Select a target ("runlevel") at boot time:

kernel /vmlinuz-3.1 (...) systemd.target=emergency.target
loads the basic stuff

kernel /vmlinuz-3.1 (...) systemd.target=multi-user.target
equivalent to runlevel 3

kernel /vmlinuz-3.1 (...) systemd.log_level=debug
sets log level

kernel /vmlinuz-3.1 (...) systemd.log_target=kmsg
logs to dmesg
Benchmarking

# systemd-analyze time
Startup finished in 1812ms (kernel) + 3722ms (initramfs) + 3912ms (userspace) = 9446ms

# systemd-analyze blame
9682ms sshd-keygen.service
4483ms abrtd.service
4382ms plymouth-start.service
4365ms systemd-readahead-replay.service
2268ms sendmail.service
2182ms udev-settle.service
...
16ms rpcbind.service
13ms dnsmasq.service
```
Benchmarking

# systemd-analyze plot

```

```
<table>
<thead>
<tr>
<th>Userspace</th>
</tr>
</thead>
<tbody>
<tr>
<td>.mount</td>
</tr>
<tr>
<td>proc-sys-fs-binfmt_misc.automount</td>
</tr>
<tr>
<td>cryptsetup.target</td>
</tr>
<tr>
<td>systemd-ask-password-console.path</td>
</tr>
<tr>
<td>boot.automount</td>
</tr>
<tr>
<td>home.automount</td>
</tr>
<tr>
<td>swap.target</td>
</tr>
<tr>
<td>mnt-pequenos.automount</td>
</tr>
<tr>
<td>usr-portage.automount</td>
</tr>
<tr>
<td>usr-src.automount</td>
</tr>
<tr>
<td>systemd-stdout-syslog-bridge.socket</td>
</tr>
<tr>
<td>var-run.mount</td>
</tr>
<tr>
<td>media.mount</td>
</tr>
</tbody>
</table>

```

```
<table>
<thead>
<tr>
<th>Kernel</th>
</tr>
</thead>
<tbody>
<tr>
<td>systemd-remount-api-vfs.service</td>
</tr>
<tr>
<td>systemd-modules-load.service</td>
</tr>
<tr>
<td>systemd-vconsole-setup.service</td>
</tr>
<tr>
<td>dev-mqueue.mount</td>
</tr>
<tr>
<td>sys-kernel-debug.mount</td>
</tr>
<tr>
<td>dev-hugepages.mount</td>
</tr>
<tr>
<td>systemd-shutdown.socket</td>
</tr>
</tbody>
</table>

```

```
| Running on beren (3.1.1-tam #1 SMP PREEMPT Sat Nov 12 15:56:54 BRST 2011) x86_64 |
```

```
0s | 1s | 2s | 3s | 4s | 5s |
---|----|----|----|----|----|
kernel | | | | | |
PABLO N. HESS
Instructor
Red Hat São Paulo
December 2011