systemd: the new init system for Linux

Glenn Holmer
Milwaukee Linux Users Group
June 14, 2014
History
init, the father of processes

✔ BIOS loads bootloader from hard drive
✔ bootloader loads GRUB
✔ GRUB loads kernel
✔ kernel mounts filesystems and loads drivers
✔ kernel starts first process (init)
sysvinit startup

✔ init looks for default runlevel and runs its scripts to start the appropriate services

✔ each runlevel has a directory `/etc/rcN.d/` with start and stop symlinks to scripts in `/etc/init.d`
init systems

traditional: sysvinit (Linux)
alternate: OpenRC (Gentoo), upstart (Ubuntu)
other OS: BSD, SMF (Solaris), launchd (Mac)
systemd development
started by Lennart Poettering, Kay Sievers
What's wrong with sysvinit?

synchronous
everything started at boot time
can't control double-forked child processes
What's better about systemd?

asynchronous

services started only when needed (via socket)

services run in cgroups
Why is systemd controversial?

“do one thing and do it well” (feature creep)
depends on dbus/kdbus
Linux-specific (because of cgroups)
binary log file (the journal)

http://0pointer.de/blog/projects/the-biggest-myths.html
Why is systemd controversial?

“do one thing and do it well” (feature creep)
depends on dbus/kdbus
Linux-specific (because of cgroups)
binary log file (the journal)

http://0pointer.de/blog/projects/the-biggest-myths.html

personality conflicts...
I hate Lennart Poettering and his systemd, journ-

JUST RTFM, WILL YOU?
systemd adoption

Ubuntu: upstart 2006 (native init 2009)
Fedora: upstart 2008, systemd 2011
RHEL: RHEL 7 (just released) uses systemd
SUSE: upstart option 2010, systemd 2011
SLES: SLES 12 (Q3) will use systemd
Debian: systemd 2014, Ubuntu will follow
Mechanics
Why is sysvinit synchronous?

services wait for their dependencies (which open a socket when ready)
Why not open the socket on startup?

services with dependencies can start at once
socket buffer holds messages until ready
Why is sysvinit slow?

shell scripts! (shell loaded over and over)
multiple invocations of grep, awk, sed...
Why not use configuration files?

in `/usr/lib/systemd/system` locals, overrides in `/etc/systemd/system`
[Unit]
Description=PostgreSQL database server
After=network.target

[Service]
Type=forking
User=postgres
Group=postgres
Environment=PGPORT=5432
Environment=PGDATA=/var/lib/pgsql/data
OOMScoreAdjust=-1000
ExecStartPre=/usr/bin/postgresql-check-db-dir ${PGDATA}
ExecStart=/usr/bin/pg_ctl start -D ${PGDATA} -s -o "-p ${PGPORT}" -w -t 300
ExecStop=/usr/bin/pg_ctl stop -D ${PGDATA} -s -m fast
ExecReload=/usr/bin/pg_ctl reload -D ${PGDATA} -s
TimeoutSec=300

[Install]
WantedBy=multi-user.target
some systemd unit types:

**service:** traditional daemon (ssh, http, kdm...)

**socket:** listener socket for service activation

**target:** like a runlevel, but not exclusive
no /etc/inittab:

/etc/systemd/system/default.target

is a symlink to e.g.

/lib/systemd/system/graphical.target
some distros use symlinks:

runlevel0.target -> poweroff.target
runlevel1.target -> rescue.target
runlevel2.target -> multi-user.target
runlevel3.target -> multi-user.target
runlevel4.target -> multi-user.target
runlevel5.target -> graphical.target
runlevel6.target -> reboot.target
more unit types:

**slice:** resource control via cgroups (can control CPU share, memory usage, IO bandwidth, device access)

**snapshot:** saves current runtime state, can return to it with `isolate` (lost on reboot)
more unit types:

device: when device is ready (udev)
mount: (/etc/fstab still preferred)
automount: (requires matching mount unit)
swap: swap space
path: unit activation on path change
timer: event based on boot time, time since a unit's activation, calendar time
sysvinit compatibility

“virtual” service units created in-memory
look for **LSB**: or **SYSV**: in the output of
  systemctl list-units
service, chkconfig, telinit... still work

for more information:
freedesktop.org/wiki/Software/systemd/Incompatibilities/
Usage
GUI front-end for systemd:

systemadm

On Fedora 20 and OpenSUSE 13.1, install systemd-ui package.

Good for exploring and learning systemd.
<table>
<thead>
<tr>
<th>Load State</th>
<th>Active State</th>
<th>Unit State</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>loaded</td>
<td>active</td>
<td>running</td>
<td>cups.path</td>
</tr>
<tr>
<td>loaded</td>
<td>active</td>
<td>running</td>
<td>cups.service</td>
</tr>
<tr>
<td>loaded</td>
<td>active</td>
<td>running</td>
<td>cups.socket</td>
</tr>
</tbody>
</table>

**Id:** cups.service(running)

**Description:** CUPS Printing Service

**Dependencies:**
- requires: basic.target(active)
- wants: system.slice(active)
- conflicts: shutdown.target(dead)
- after: basic.target(active), cups.path(running), cups.socket(running), system.slice(active), systemd-journald.socket(running)
- before: shutdown.target(dead)

**Fragment Path:** /usr/lib/systemd/system/cups.service

**Control Group:** n/a

**Load State:** loaded

**Active State:** active

**Unit State:** running

**Activated:** Mon, 26 May 2014 06:25:20

**Can Start/Stop:** Yes

**Deactivated:** n/a

**Can Reload:** No
Most frequently-used commands:

systemctl
journalctl

Completion prompting and color coding!
systemctl [list-units]
systemctl list-unit-files
systemctl -t service
systemctl --state failed
systemctl enable <servicename>
systemctl start <servicename>
systemctl status <servicename>
systemctl daemon-reload
systemctl halt
journalctl -f  (follow, like tail -f)
journalctl -x  (show extra)
journalctl -n99  (last 99 entries)
journalctl -b  (since boot)
journalctl -b -1  (since previous boot)
journalctl --since  (since date/time)
journalctl -p err  (by priority)
journalctl -u  (by unit)
journalctl /usr/...  (by executable)
journalctl /dev/...  (by device)
switch “runlevel”:
  systemctl isolate <target-name>

emergency boot: start kernel with
  systemd.unit=<target-name>
systemd diagnostics:

systemd-cgls ("control group ls")
systemd-cgtop ("control group top")

systemd-analyze
systemd-analyze blame
systemd-analyze plot > filename.svg
systemd-analyze dot | \
  dot -Tsvg > systemd.svg

[root@orac ~]# systemctl enable postgresql.service
ln -s '/usr/lib/systemd/system/postgresql.service' '/etc/systemd/system/multi-user.target.wants/postgresql.service'
[root@orac ~]#
# systemctl enable postgresql.service

ln -s '/usr/lib/systemd/system/postgresql.service' '/etc/systemd/system/multi-user.target.wants/postgresql.service'

# systemctl start postgresql.service

Job for postgresql.service failed. See 'systemctl status postgresql.service' and 'journalctl -xn' for details.

#
systemctl enable postgresql.service
ln -s '/usr/lib/systemd/system/postgresql.service' '/etc/systemd/system/multi-user.target.wants/postgresql.service'

Job for postgresql.service failed. See 'systemctl status postgresql.service' and 'journalctl -xn' for details.

postgresql.service - PostgreSQL database server
Loaded: loaded (/usr/lib/systemd/system/postgresql.service; enabled)
Active: failed (Result: exit-code) since Sat 2014-04-19 08:27:49 CDT; 10s ago
   Process: 13117 ExecStartPre=/usr/bin/postgresql-check-db-dir $PGDATA (code=exited, status=1/FAILURE)

Apr 19 08:27:49 orac.lyonlabs.org postgresql-check-db-dir[13117]: "/var/lib/pgsql/data" is missing or empty.
Apr 19 08:27:49 orac.lyonlabs.org postgresql-check-db-dir[13117]: Use "postgresql-setup initdb" to initialize the database cluster.
Apr 19 08:27:49 orac.lyonlabs.org systemd[1]: postgresql.service: control process exited, code=exited status=1
Apr 19 08:27:49 orac.lyonlabs.org systemd[1]: Failed to start PostgreSQL database server.
Apr 19 08:27:49 orac.lyonlabs.org systemd[1]: Unit postgresql.service entered failed state.

[root@orac ~]#
[root@rac ~]# systemctl enable postgresql.service
ln -s '/usr/lib/systemd/system/postgresql.service' '/etc/systemd/system/multi-user.target.wants/postgresql.service'

[root@rac ~]# systemctl start postgresql.service
Job for postgresql.service failed. See 'systemctl status postgresql.service' and 'journalctl -xn' for details.
[root@rac ~]# systemctl status postgresql.service
postgresql.service - PostgreSQL database server
   Loaded: loaded (/usr/lib/systemd/system/postgresql.service; enabled)
   Active: failed (Result: exit-code) since Sat 2014-04-19 08:27:49 CDT; 10s ago
   Process: 13117 ExecStartPre=/usr/bin/postgresql-check-db-dir $PGDATA (code=exited, status=1/FAILURE)

Apr 19 08:27:49 orac.lyonlabs.org postgresql-check-db-dir[13117]: "/var/lib/pgsql/data" is missing or empty.
Apr 19 08:27:49 orac.lyonlabs.org postgresql-check-db-dir[13117]: Use "postgres-setup initdb" to initialize the database cluster.
Apr 19 08:27:49 orac.lyonlabs.org systemd[1]: postgresql.service: control process exited, code=exited status=1
Apr 19 08:27:49 orac.lyonlabs.org systemd[1]: Failed to start PostgreSQL database server.
Apr 19 08:27:49 orac.lyonlabs.org systemd[1]: Unit postgresql.service entered failed state.

Apr 19 08:27:49 orac.lyonlabs.org systemd[1]: journalctl -xn
Apr 19 08:25:38 orac.lyonlabs.org su[13017]: pam_unix(su::session): session opened for user root by czenbe(uid=1002)
Apr 19 08:26:07 orac.lyonlabs.org fprintf[13018]: ** Message: No devices in use, exit
Apr 19 08:27:43 orac.lyonlabs.org systemd[1]: Reloading.
Apr 19 08:27:49 orac.lyonlabs.org systemd[1]: Starting PostgreSQL database server...
   -- Subject: Unit postgresql.service has begun with start-up
   -- Defined-By: systemd
   -- Unit postgresql.service has begun starting up.
Apr 19 08:27:49 orac.lyonlabs.org postgresql-check-db-dir[13117]: "/var/lib/pgsql/data" is missing or empty.
Apr 19 08:27:49 orac.lyonlabs.org postgresql-check-db-dir[13117]: Use "postgres-setup initdb" to initialize the database cluster.
Apr 19 08:27:49 orac.lyonlabs.org systemd[1]: postgresql.service: control process exited, code=exited status=1
Apr 19 08:27:49 orac.lyonlabs.org systemd[1]: Failed to start PostgreSQL database server.
   -- Subject: Unit postgresql.service has failed
   -- Defined-By: systemd
   -- Unit postgresql.service has failed.
   -- The result is failed.
Apr 19 08:27:49 orac.lyonlabs.org systemd[1]: Unit postgresql.service entered failed state.

[root@rac ~]#
[root@crac ~]# postgresql-setup initdb
Initializing database ... OK

[root@orac ~]#
[root@orac ~]# postgresql-setup initdb
Initializing database ... OK

[root@orac ~]# systemctl start postgresql.service
[root@orac ~]#
[root@orac ~]# postgresql-setup initdb
Initializing database ... OK

[root@orac ~]# systemctl start postgresql.service
[root@orac ~]# systemctl status postgresql.service
postgresql.service - PostgreSQL database server
   Loaded: loaded (/usr/lib/systemd/system/postgresql.service; enabled)
   Active: active (running) since Sat 2014-04-19 08:43:36 CDT; 7s ago
   Process: 13748 ExecStart=/usr/bin/pg_ctl start -D ${PGDATA} -s -o -p ${PGPORT} -w -t 300 (code=exited, status=0/SUCCESS)
   Process: 13741 ExecStartPre=/usr/bin/postgresql-check-db-dir ${PGDATA} (code=exited, status=0/SUCCESS)
   Main PID: 13751 (postgresql)
   CGroup: /system.slice/postgresql.service
            └─13751 /usr/bin/postgres -D /var/lib/pgsql/data -p 5432
   └─13752 postgres: logger process
        └─13754 postgres: checkpoint process
    └─13755 postgres: writer process
       └─13756 postgres: wal writer process
    └─13757 postgres: autovacuum launcher process
        └─13758 postgres: stats collector process

Apr 19 08:43:35 orac.lyonlabs.org systemd[1]: Starting PostgreSQL database server...
Apr 19 08:43:35 orac.lyonlabs.org pg_ctl[13748]: LOG:  redirecting log output to logging collector process
Apr 19 08:43:35 orac.lyonlabs.org pg_ctl[13748]: HINT:  Future log output will appear in directory "pg_log".
Apr 19 08:43:36 orac.lyonlabs.org systemd[1]: Started PostgreSQL database server.
[root@orac ~]#
resources:

overview:
http://www.freedesktop.org/wiki/Software/systemd/

“Demystifying systemd” slides:
http://bit.ly/1jm87CJ

“Getting Ready for Systemd” video:
https://access.redhat.com/site/videos/403833