systemd

“If you can’t beat them, join them”
“Like it or not, systemd is here to stay”

Inhoudsopgave

Naam en adoptatie

Naam komt van: (de **system d**aemon)

Maar ook van: “System D”: vermogen om snel te handelen, aan te passen en te improviseren.

Start: rond 2010, Lennart Poettering en Kay Sievers (RedHat software engineers)

Adoptatie: eerst in Fedora (15), daarna openSUSE 12.2, CentOS 7, RHEL7, SUSE12, Debian8 en Ubuntu 15.04.

Wat is systemd?

Een ***system*** en ***service*** manager wat een Linux systeem in *userspace* brengt.

Maar niet alleen een vervanging van init:

- event logging (systemd-journald, binair maar tooling (journalctl), ook *alle* bootmessages)
- system configuration (hostnamctl, timezonectl, localectl, ...)
- login management (systemd-logind)
- device management (systemd-udevd)
- network connection management
- scheduling
- tmpfiles
- connectie tussen user-space en kernel-space
- software development platform
- …

Wel PID **1** maar *niet* **1** programma maar een “software suite”.

systemd belangrijkste eigenschappen

- een default init system (system/service manager)
- ook vervanging andere services (eerder genoemd)
- bouwblok: units
- gebruik van cgroups (resource management)
- correct killen van services
- snelle boot (parallel, on-demand (socket/bus activation))
- altijd logging (ook begin van boot)
- declaratieve configuratie (“ini-stijl”) in ascii files

Waarom systemd

Doel was in eerste instantie om het init systeem te verbeteren:

- SystemV / BSD init was aan vervanging toe (zeker voor desktops)
- Om een **standaard** framework voor de diverse Linux services op de diverse Linux distributies te krijgen: meer consistentie in service configuratie en uitvoering.
- Betere afhankelijkheid (“dependency”) checks (bv een apache daemon pas opstarten als het netwerk beschikbaar is).
- De shell “overhead” verminderen

“Bonus”: Meer processen gelijktijdig (“concurrent”) ipv na elkaar opstarten (bv. een apache daemon al opstarten voordat het netwerk beschikbaar is.. ;-)) -> “socket activation”

Componenten systemd 1/2

- units - basisblok systemd - bepaald wat wanneer moet worden opgestart - systemd regelt dependencies tussen units
  - services - controle van daemons
  - sockets - controle van netwerk connecties
  - mounts, path, timer, slice, ...
- targets - logisch groepering van units/targets (vergelijk “runlevels”)
- cgroups - systemd maakt hevig gebruik van cgroups - groepering van services
- systemd daemons
  - systemd - system and service manager
  - systemd-logind- login service
  - systemd-journald - binaire logging service
  - systemd-udevd- usb device managing service
  - ...

systemd componenten 2/2

- commando’s
  - systemctl - controle van de system en service manager (systemd) - controle over units (stoppen, starten, enable, disable, status, …)
  - journalctl - bekijk de journal (logging)
  - hostnamectl
  - timedatectl
  - localectl
  - ...

Units

- **Basis** component van systemd (service units, target units, socket units, mount units, …)
- “Onderlinge *afhankelijkheid* waarmee systemd overweg kan en die het bootproces bepalen”
- Configuratie met unit-files (“ini” stijl). Naamgeving: ***name***.***type***
- (httpd.service, ssh.socket, logrotate.timer, …)
- Paden:
  - /usr/lib/systemd/system/ (maintainer)
  - /etc/systemd/system/ (beheerder)
  - /run/systemd/system/ (runtime aanpassingen (niet persistent)).

Targets

- Logische groepering van units
- Geeft bepaald “runlevel” weer:
  - rescue.target
  - multi-user.target
  - graphical.target
- Unit files: “naam”.target
- Default target
  - systemctl get-default
  - systemctl set-default “target”
- Overgang naar bepaalde target
  - systemctl isolate “target”
- Meerdere targets tegelijk actief (systemctl list-units -t target)

Services

Belangrijkste units (voor het starten van de diverse services/daemons).

Voorbeeld: /usr/lib/systemd/system/**nginx.service**

```
[Unit]
Description=A high performance web server and a reverse proxy server
Documentation=man:nginx(8)
After=network.target

[Service]
Type=forking
PIDFile=/run/nginx.pid
ExecStartPre=/usr/sbin/nginx -t -q -g 'daemon on; master_process on;'
ExecStart=/usr/sbin/nginx -g 'daemon on; master_process on;'
ExecReload=/usr/sbin/nginx -g 'daemon on; master_process on;' -s reload
ExecStop=-/sbin/start-stop-daemon --quiet --stop --retry QUIT/5 --pidfile /run/nginx.pid
TimeoutStopSec=5
KillMode=mixed

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

systemctl commando

Beheren van units doe je met het commando `systemctl`

(gebruik bash completion!)

- systemctl start nginx.service elasticsearch ... (*glob* units)
- systemctl stop nginx ...
- systemctl status nginx
- systemctl reload nginx
- systemctl restart nginx
- systemctl enable [--now] nginx
- systemctl disable nginx
- systemctl show [--all] nginx
- systemctl cat nginx
- systemctl daemon-reload

Maar systemctl kan nog veel meer van systemd prijsgeven (komt later)

Voorbeelden: een eigen service unit file maken

Source: http://kwalinux.nl/systemd/data.tgz

go_listen - go programma wat luister op poort 8000 en output geeft.

Unit file: /usr/lib/systemd/system/go_listen.service

mydaemon - real daemon - schrijft elke 5 seconden in de systeem log

Unit file: /lib/systemd/system/mydaemon.service

Enkele handige commando's 1/2

- `systemctl list-unit-files`
- `systemctl list-units --type [target|service]`
- `pstree -pu`
- `systemctl status`
- `systemd-cgls; systemd-cgtop`
- `systemctl get-default`
- `systemctl -H hostname restart elasticsearch`
- `systemctl list-dependencies graphical.target`
- `systemctl -t service (wat is` *`running`*`?)`
- `systemctl is-active apache2`
- `systemctl edit elasticsearch`
- `systemctl revert elasticsearch`
- `systemctl --failed`
- `systemctl daemon-reload`
- `systemctl mask "service" - volledige disable`
- `systemctl poweroff`
- `systemctl reboot`
- `man systemd.unit`
- `man systemd.service`

Enkele handige commando's 2/2

- `systemd-detect-virt`
- `journalctl -xe`
- `journalctl -k`
- `journalctl -u "unit"`
- `journalctl -p0..4 -xb`
- `journalctl -f`
- `journalctl --unit elasticsearch --since "2020-07-10 00:00:00"`
- `journalctl -k -S "2020-6-01 00:00:00" -U "2020-07-01 00:00:00"`
- `journalctl --since yesterday`
- `journalctl -S "-5 minutes"`
- `journalctl --list-boots`
- `journalctl -b "boot id"`
- `journalctl -o json-pretty -u ssh`
- `systemd-analyze blame`
- `systemd-analyze security "unit"`
- `hostnamectl`
- `timedatectl`
- `networkctl status "interface"`
- `loginctl list-sessions`
- `loginctl session-status "ID"`

Opmerkingen

- RedHat videos ietwat “lovend” (..)
  - https://www.youtube.com/watch?v=tY9GYsoxeLg (..)
- “My way or the highway” mentaliteit?
- (security) bugs aanpak wat “losjes” (volgens Linus Torvalds)
- Mission creep?
  - linux kernel: 27.800.000 lines - systemd: 1.300.000 lines (2020)
- (te) complex?
  - systemctl -t target --state=active
  - systemctl show go_listen.service | wc -l
  - systemd-analyze dot --order | dot -Tsvg > /vagrant/systemd.svg
- Vooral veel verbeteringen (ook meer standaard)

Referenties

- Homepage: https://freedesktop.org/wiki/Software/systemd/
- Hoe het begon: http://0pointer.net/blog/projects/systemd.html
  (hier staan ook veel vervolgstukken over systemd)
- manpages (systemd.unit,m systemd.service, systemd.exec, systemd….., systemctl, journalctl, …)

Vragen?

Bier?