

Systemd - die Zukunft von Linux?

Bernd Seidler

March 25, 2014

Inhalt

- 1 Motivation
- 2 Über systemd
- 3 Beispiele
- 4 Praxis
- 5 Kritik
- 6 Fazit

Motivation

Warum überhaupt systemd anschauen?

- 1 Mein Debian-System wollte verschlüsselte LVM-Volumes beim Start nicht korrekt einbinden.
- 2 Neugierde!

systemd - kurz und klein

URL: <http://www.freedesktop.org/wiki/Software/systemd/>

Author: Lennard Poettering

systemd is a system and service manager for Linux, compatible with SysV and LSB init scripts.

systemd provides aggressive parallelization capabilities, uses socket and D-Bus activation for starting services, offers on-demand starting of daemons, keeps track of processes using Linux control groups, supports snapshotting and restoring of the system state, maintains mount and automount points and implements an elaborate transactional dependency-based service control logic.

Holy flipping penguins, Batman!

Möglichkeiten von systemd

- 1 Prozesse parallelisiert starten
- 2 Prozesse kontrollieren, bei Bedarf neu starten.
- 3 Intensive Nutzung von control groups (cgroups)
- 4 Ersetzt Init-Skripte durch Config-Files mit “clean syntax”
- 5 Dienste on-demand starten (inetd, anyone?)
- 6 Ersatz für cron, atd
- 7 Integration von verschlüsselten Devices,
- 8 Unterstützung von Polkit
- 9 ... und so ziemlich alles, was Lennard noch so einfällt.

Architektur-Überblick

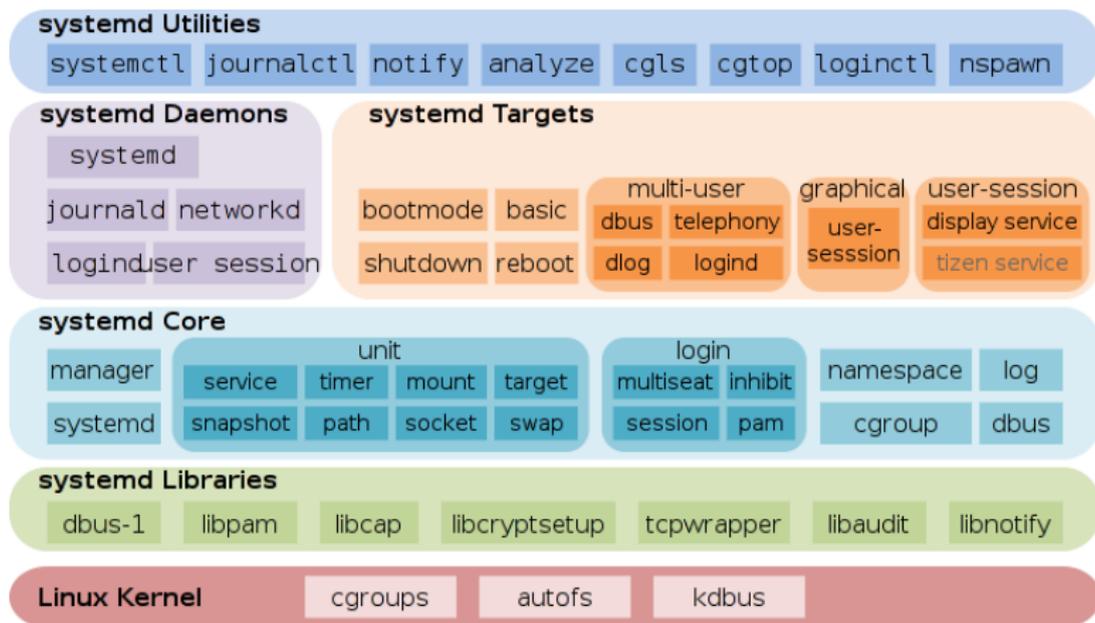


Figure : http://en.wikipedia.org/wiki/File:Systemd_components.svg

Kompatibilität

Systemd ist ja grundsätzlich kompatibel zu klassischen init-Skripten, aber: Um richtig Sinn zu machen, sollen die Dienste sich nicht wegforken. Das Konstrukt mit ".service", ".target", ".socket"-Files ist nicht immer einfach zu durchschauen.

Wenn man dann noch "teilweise" SystemV und "teilweise" systemd macht, kommt man ziemlich schnell zu einem hochgradig komplexen System.

Verschlüsseltes Dateisystem mounten

Datei: `/etc/systemd/system/local-fs.target.wants/var.mount`

[Unit]

Description=Mount /var

DefaultDependencies=yes

After=cryptsetup@cryptdb.service

[Mount]

What=/dev/mapper/cryptdb

Where=/var

Type=btrfs

Options=compress=lzo,nodiratime,noatime

Das entschlüsselt das LVM-Volumen cryptdb und hängt es nach /var.

Syslog starten

Datei /etc/systemd/system/syslog.service

[Unit]

Description=System Logging Service

Requires=syslog.socket

[Service]

Type=notify

ExecStart=/usr/sbin/rsyslogd -n

StandardOutput=null

[Install]

WantedBy=multi-user.target

Alias=syslog.service

Syslog-Socket binden

```
[Unit]
```

```
Description=Syslog Socket
```

```
DefaultDependencies=no
```

```
Before=sockets.target syslog.target shutdown.target
```

```
# Don't allow logging until the very end
```

```
Conflicts=shutdown.target
```

```
# Pull in syslog.target to tell people[...]
```

```
Wants=syslog.target
```

```
[Socket]
```

```
ListenDatagram=/run/systemd/journal/syslog
```

```
SocketMode=0666
```

```
PassCredentials=yes
```

```
PassSecurity=yes
```

```
ReceiveBuffer=8M
```

Mein Setup

Mein Rechner hat schon knapp sechs Jahre auf dem Buckel.

- Intel(R) Core(TM)2 Duo CPU E6550 @ 2.33GHz
- 8 GB RAM
- 2x500 GB HDD, mit LVM verwaltet
- OS: Debian unstable, 32 Bit - vom Vorgängersystem (Baujahr 1999) mitgewandert
- Desktop: XFCE - mit KDE4 und Gnome3 bin ich nie warm geworden
- Datenpartitionen /home, /var, /space auf LVM, mit LUKS verschlüsselt

Und eben diese Unterstützung machte im Setup mit sysvinit Schwierigkeiten.

Erfahrungen

- Nach einigen Versuchen bekam ich meine Daten-Partitionen mit systemd ohne weiteres zum Laufen.
- Allerdings, ich wollte es selbst kaum glauben, brauchte cron satte **2min 8.975s** zum Start. Lösung: Eigene `cron.service` mit der Option `Type=forking`
- Mein Displaymanager wollte mal starten, mal nicht. Das nervt!
- Lennard-Poettering-Gedenkminute: Es dauerte doch gut und gerne ein bis zwei Minuten zwischen Login und Anmeldung nach dem Neustart.
- Trotz einschlägiger Debugging-Tools kam ich partout nicht darauf, was denn eigentlich für das seltsame Verhalten verantwortlich war.

Und zurück zu sysvinit

- Das Switchen zwischen systemd und sysvinit geschieht einfach per Kommandozeilenoption im Bootmanager: `init=/bin/systemd` bzw `init=/sbin/init`
- Nach Einbau einer SSD für `/` und `/usr` startete sysvinit schneller als systemd. So viel zum Thema...
- Cryptdisks und ein passendes Skript habe ich mittlerweile in den Baum für die initrd reingepriemelt bekommen - das ursprüngliche Problem war also gelöst.
- Und der Displaymanager wollte wieder.

Tja, das war's dann mit systemd.

- Systemd ist im Vergleich zu Sysvinit sehr mächtig
- Diese Mächtigkeit macht es schwer zu pflegen und schwer zu ersetzen. Aus Security-Perspektive sehr bedenklich.
- Es beschreitet einen komplett neuen Ansatz - und damit muss man nochmal die Schulbank drücken
- Löst Probleme, die eigentlich woanders entstanden sind: Stabile Benennung von Netzwerkinterfaces wäre eigentlich Kernel-Aufgabe.
- Durch Parallelisierung und lazy loading erzeugt systemd race conditions

Alternativen

OpenBSD verhält sich deutlich anders als Linux:

- init ruft das Skript `etc/rc` auf
- Das wiederum `sourcet: etc/rc.conf` und `etc/rc.local`
- Außerdem werden Skripte aus `etc/rc.d` eingebunden. Diese Option ist für Dienste aus dem Portstree da.
- Der Neustart von Diensten geschieht manuell

Auf die Tour ist der Init-Prozess unter OpenBSD recht schlank und ziemlich zackig.

Fazit

- Systemd hat sicher das Potential, einiges am Bootprozess zu verbessern: Geschwindigkeit, Hotplugging, Suspend..
- Allerdings fehlt es dem Projekt an einem klaren Fokus
- Man sollte sich bei produktiven Systemen allerdings Zeit lassen, bis Debian Stable mit systemd als Default daherkommt. Es ist noch mit Optimierungen seitens der Packet Maintainer und diversen Bugfixes bei sysvinit zu rechnen.
- Bei Servern ist fraglich, ob die Vorteile von systemd die schlechtere Vorhersagbarkeit und das komplexe Debugging wert sind.
- Bestandssysteme sollten weiterhin mit sysvinit laufen.

Fragen

Fragen?

Ende

Vielen Dank für Eure Aufmerksamkeit!