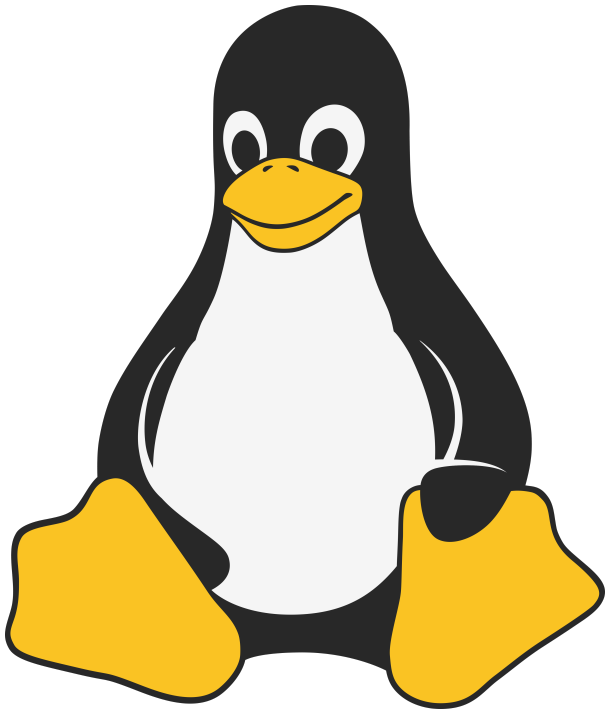
Systemd und System-Initialisierung



Inhaltsverzeichnis

-

Systemd - Historie

Systemd entstand als Reaktion auf die Komplexität und die langen Startzeiten des traditionellen `SysVinit`-Systems. Ein weiterer Grund war die mangelnde Parallelisierung des Startvorgangs.

Konzepte und Ideen von Systemd stammen aus *Launchd* (Socket-Aktivierung) von *macOS* und *SMF* (*Service Management Facility*) von *Solaris*. Lange Zeit war *Upstart* von *Ubuntu* ein Konkurrent von Systemd.

Zum Inhaltsverzeichnis ...

- Systemd wurde von Lennart Poettering und Kay Sievers entwickelt und erstmals 2010 in Fedora 15 eingeführt.
- In den folgenden Jahren wurde Systemd in vielen Linux-Distributionen als Standard-Init-System eingeführt (*Fedora*, *openSUSE*, *Arch Linux*, *Mandriva*, *Mageia*, RHEL (*Red Hat Enterprise Linux*), *SLES* (*SUSE Linux Enterprise Server*)).
- *Ubuntu* wechselte 2015 vom selbst entwickelten Upstart-Init-System zu Systemd. Auch *Debian* (*Jessie*) führt 2015 Systemd ein.
- Damit wurde Systemd zum Standard-Init-System in den meisten Linux-Distributionen.

Zum Inhaltsverzeichnis ...

Systemd - Aufgaben

Systemd ist ein System- und Service-Manager für Linux, der das System Initialisiert und danach im laufenden Betrieb verwaltet. Systemd hat viele Aufgaben. Hier eine unvollständige Liste:

- Steuerung und Protollierung des Systemstarts (nach dem Laden des Kernels) und des Shutdowns
- Bereitstellen der Hardware
- Ein- und Aushängen der Dateisysteme (auch Automounts)
- Starten und Beenden von Systemdiensten

Zum Inhaltsverzeichnis ...

- Bereitstellen von Sockets zur Kommunikation zwischen Systemprozessen
- Steuerung von von User-Sessions (vom Login bis zum Logout)
- Logging und Protokollierung
- Ressourcenkontrolle
- Zeitgesteuerte Aufgaben

Mehr dazu in *L07a-Systemd-Vorlaeufer-Konkurrenten-Ideengeber*

Zum Inhaltsverzeichnis ...

`systemctl` - Benutzerschnittstelle zu Systemd

Systemd ist ein dämon-Prozess. Er wird vom Kernel als erster Benuzterprozess gestartet. Systemd startet dann alle anderen Prozesse und Dienste. Systemd läuft dauerhaft im Hintergrund vom Boot bis zum Shutdown des Systems. Er benötigt grundsätzlich keine direkte Benutzerinteraktion.

Indirekte Benutzerinteraktion ist mit dem Kommando `systemctl` möglich. `systemctl` ist die Benutzerschnittstelle zu Systemd.
Mit `systemctl` kann Systemd abgefragt und überwacht werden. Auch die Steuerung und Verwaltung von Systemd ist möglich.

Zum Inhaltsverzeichnis ...

Systemd-Units

Systemd organisiert die zu überwachenden Systemdienste und -ressourcen in sog. Units. Eine Unit ist also eine Ressource und gleichzeitig eine Überwachungseinheit. Grundsätzlich gibt es zu jeder Unit eine Konfigurationsdatei - das sog. Unit-File - das die Eigenschaften und das Verhalten der Unit definiert/konfiguriert.

Units werden gegliedert in sog. Unit-Types (Ressourcen-Typen oder Aufgabengruppen, denn an jeder Ressource hängt eine bestimmte Verwaltungs- und Überwachungsaufgabe).

Zum Inhaltsverzeichnis ...

Systemd-Unit-Types (Ressource-Gruppen)

- `service` - Systemdienste
- `socket` - Sockets
- `device` - Hardwaregeräte, repräsentiert durch Dateien im `/dev` -Dateisystem
- `mount` - Eingehängte Dateisysteme
- `automount` - Automatisch eingehängte Dateisysteme
- `target` - Gruppierung anderer Units

- `snapshot` - gespeicherte Systemzustände
- `timer` - Zeitgesteuerte Units
- `path` - Dateisystem-Änderungen
- `slice` - Ressourcenkontrolle
- `scope` - Prozessgruppen
- `swap` - Swap-Dateien

```
hermann@debian:~$ # show available 'systemd' unit types
hermann@debian:~$ systemctl list-units --type=help | nl -ba
     1  Available unit types:
     2  service
     3  mount
     4  swap
     5  socket
     6  target
     7  device
     8  automount
     9  timer
    10  path
    11  slice
    12  scope
```

Jeder Unit-Typ kann mit dem Parameter `--type=<unit-type>` angegeben werden, um die Units dieses Typs zu listen.

Zum Inhaltsverzeichnis ...

Units vom Typ `service`

```
hermann@debian:~$ systemctl list-units --type=service \
> ssh.service cron.service smbd.service nmbd.service
  UNIT         LOAD   ACTIVE SUB     DESCRIPTION
  cron.service loaded active running Regular background program processing daemon
  nmbd.service loaded active running Samba NMB Daemon
  smbd.service loaded active running Samba SMB Daemon
  ssh.service  loaded active running OpenBSD Secure Shell server

LOAD   = Reflects whether the unit definition was properly loaded.
ACTIVE = The high-level unit activation state, i.e. generalization of SUB.
SUB    = The low-level unit activation state, values depend on unit type. ...
```

Hier wurden vier Units vom Typ `service` angegeben. Werden keine Services angegeben, werden alle gelistet.

Units vom Typ `mount`

```
hermann@debian:~$ systemctl list-units --type=mount -- -.mount \
> boot-efi.mount dev-hugepages.mount dev-mqueue.mount
  UNIT                   LOAD   ACTIVE SUB     DESCRIPTION
  -.mount                loaded active mounted Root Mount
  boot-efi.mount         loaded active mounted /boot/efi
  dev-hugepages.mount loaded active mounted Huge Pages File System
  dev-mqueue.mount    loaded active mounted POSIX Message Queue File System

LOAD   = Reflects whether the unit definition was properly loaded.
ACTIVE = The high-level unit activation state, i.e. generalization of SUB.
SUB    = The low-level unit activation state, values depend on unit type. ...
```

Hier wurden vier Units vom Typ `mount` angegeben. Werden keine Mounts angegeben, werden alle gelistet.

Erläuterung der Unit-Types

Die Erläuterung der Unit-Types gibt einen Überblick über die verschiedenen Aufgaben, für die Systemd zuständig ist.

Das Suffix eines Unit-Files (s.u) gibt den Unit-Type an. Z.B. `ssh.service` ist ein Unit-File vom Typ `service`, `boot-efi.mount` ist ein Unit-File vom Typ `mount`.

Zum Inhaltsverzeichnis ...

Unit-Type `service` - Systemdienste

- Systemdienste sind Programme, die im Hintergrund laufen und bestimmte Aufgaben erfüllen.
- Systemdienste können von Systemd gestartet, gestoppt, neugestartet und neu geladen werden.
- Systemdienste können automatisch beim Systemstart gestartet werden.

Zum Inhaltsverzeichnis ...

Unit-Type `socket` - Sockets

- Sockets (Unix Domain Sockets) sind Kommunikationsendpunkte, die von Prozessen auf dem lokalen System genutzt werden, um miteinander zu kommunizieren.
- Sockets können von Systemd erstellt und verwaltet werden.
- Sie sind Kommunikationsendpunkte für Systemdienste, die auch Daten puffern können. Z.B. `systemd-journald.socket`. Über diesen Socket können andere Dienste ihre Log-Daten an `systemd-journald` senden.

Unit-Type `device` - Hardwaregeräte

- Hardwaregeräte sind Geräte, die an das System angeschlossen oder eingebaut sind und von Systemd erkannnt, initialisiert und verwaltet werden.
- Hardwaregeräte werden durch Dateien im `/dev`-Dateisystem repräsentiert.
- Die Dateien im `/dev`-Dateisystem werden teils dynamisch erstellt. Z.B. wird ein USB-Stick beim Einstecken dynamisch erkannt und ein Device-File im `/dev`-Verzeichnis erstellt. Oder ein Pseudo-TTY wird beim Anmelden eines Benutzers mit `ssh` dynamisch erstellt und beim Abmelden wieder gelöscht.

Zum Inhaltsverzeichnis ...

Unit-Type `mount` - Eingehängte Dateisysteme

- Dateisysteme (physikalische oder virtuelle) werden von Systemd eingehängt und verwaltet.
- Beim Systemstart werden die Dateisysteme eingehängt und beim Shutdown wieder ausgehängt.
- In `/etc/fstab` werden die Dateisysteme und ihre Mount-Optionen definiert. Systemd liest diese Datei und hängt die Dateisysteme ein.
- Wird `/etc/fstab` (vom Administrator) geändert, muss Systemd neu geladen werden, damit die Änderungen wirksam werden (`systemctl daemon-reload`).

Unit-Type `automount` - Automatisch eingehängte Dateisysteme

- Automounts sind Dateisysteme, die automatisch eingehängt werden, wenn auf sie zugegriffen wird.
- Automounts sind nützlich, wenn auf ein Dateisystem nur selten zugegriffen wird. Das Dateisystem wird erst eingehängt, wenn darauf zugegriffen wird, und wird nach einer gewissen Zeit der Inaktivität automatisch wieder ausgehängt.
- Systemd erkennt den Zugriff auf das Automount-Dateisystem und hängt es ad hoc ein.

Zum Inhaltsverzeichnis ...

Unit-Type `target` - Gruppierung anderer Units

- Targets sind Gruppen von Units, die zusammen gestartet oder gestoppt werden.
- Targets entsprechen (ganz grob) den Runlevels des traditionellen `SysVinit`-Systems.

- Targets definieren bestimmte Systemzustände, z.B.
  - `multi-user.target` - System ist im Multi-User-Modus
  - `graphical.target` - System ist im Grafik-Modus
  - `bluetooth.target` - System ist im Bluetooth-Modus
  - `network.target` - System ist im Netzwerk-Modus
  - `single.target` - System ist im Single-User-Modus
  - `default.target` - Standard-Target, Ziel-Target beim Start des Systems
  - etc.

Unit-Type `snapshot` - Gespeicherte Systemzustände

- Snapshots sind gespeicherte Systemzustände, die von Systemd erstellt und verwaltet werden.
- Snapshots können erstellt werden, um den Systemzustand zu sichern, bevor Änderungen vorgenommen werden.
- Snapshots können verwendet werden, um einen früheren Systemzustand wiederherzustellen.

Zum Inhaltsverzeichnis ...

Unit-Type `timer` - Zeitgesteuerte Units

- Timer sind Units, die zu bestimmten Zeiten oder in bestimmten Intervallen ausgeführt werden.
- Timer können verwendet werden, um bestimmte Aufgaben zeitgesteuert auszuführen.

Zum Inhaltsverzeichnis ...

Unit-Type `path` - Dateisystem-Änderungen

- Path-Units überwachen Dateisystem-Änderungen und führen eine Aktion aus, wenn eine bestimmte Datei oder ein bestimmtes Verzeichnis geändert/gelöscht/hinzugefügt wird.

Zum Inhaltsverzeichnis ...

Unit-Type `slice` - Ressourcenkontrolle

- Slices sind Gruppen von Prozessen, die gemeinsam Ressourcen nutzen.
- Slices können verwendet werden, um die Ressourcennutzung von Prozessen zu kontrollieren oder zu begrenzen.

Zum Inhaltsverzeichnis ...

Unit-Type `scope` - Prozessgruppen

- Scopes sind Gruppen von Prozessen, die gemeinsam gestartet und gestoppt werden.

Unit-Type `swap` - Swap-Dateien

- Swap-Units sind Swap-Dateien oder Swap-Partitionen, die von Systemd verwaltet werden.

Systemd-Unit-Files

Jede Unit hat eine Konfigurationsdatei, das sog. Unit-File. Das Unit-File definiert die Eigenschaften und das Verhalten der Unit. Unit-Files sind im Verzeichnis `/etc/systemd/system/` oder `/lib/systemd/system/` gespeichert.

- Das Suffix eines Unit-Files gibt den Unit-Type an. Z.B. `ssh.service` ist ein Unit-File vom Typ `service`, `boot-efi.mount` ist ein Unit-File vom Typ `mount`.

- `/lib/systemd/system/` : hier liegen die Default-Unit-Files, die von der System-Installation oder beim Installieren von Paketen erstellt werden.
- `/etc/systemd/system/` : hier liegen die vom Administrator erstellten oder geänderten Unit-Files. Diese Unit-Files genießen Vorrang vor den Default-Unit-Files in `/lib/systemd/system/` .
- Normalerweise kopiert der Administrator ein Default-Unit-File aus `/lib/systemd/system/` nach `/etc/systemd/system/` und ändert es dort ab. So gehen die Änderungen nicht verloren, wenn das Paket aktualisiert wird.

- `/run/systemd/system/` : hier liegen die dynamisch erstellten Unit-Files, die von Systemd zur Laufzeit erstellt werden.
- `/usr/lib/systemd/system/` : hier liegen die Unit-Files von Systemd selbst, die nicht geändert werden sollten.

Beispiel: Unit-File `ssh.service`

```
hermann@debian:~$ nl -ba /lib/systemd/system/ssh.service
     1  [Unit]
     2  Description=OpenBSD Secure Shell server
     3  Documentation=man:sshd(8) man:sshd_config(5)
     4  After=network.target auditd.service
     5  ConditionPathExists=!/etc/ssh/sshd_not_to_be_run
     6
```

```
 7  [Service]
 8  EnvironmentFile=-/etc/default/ssh
 9  ExecStartPre=/usr/sbin/sshd -t
10  ExecStart=/usr/sbin/sshd -D $SSHD_OPTS
11  ExecReload=/usr/sbin/sshd -t
12  ExecReload=/bin/kill -HUP $MAINPID
13  KillMode=process
14  Restart=on-failure
15  RestartPreventExitStatus=255
16  Type=notify
17  RuntimeDirectory=sshd
18  RuntimeDirectoryMode=0755
19
20  [Install]
21  WantedBy=multi-user.target
22  Alias=sshd.service
```

Ein Unit-File besteht aus drei Abschnitten (Sections):

- `[Unit]`: Definiert die Eigenschaften der Unit, z.B. die Beschreibung, die Abhängigkeiten und die Bedingungen.
- `[<Unit-Type>]`: Definiert die Eigenschaften der Unit. Der Name der Section entspricht dem Unit-Type (z.B. `[Service]` für Units vom Typ `service`). Beispiel: `[Service]`
  - `[Service]`: Definiert die Eigenschaften des Dienstes, z.B. den Startbefehl, die Umgebungsvariablen und Neustartbedingungen.
- `[Install]`: Definiert, wie die Unit installiert wird, z.B. in welchem Target sie gestartet wird.

Zum Inhaltsverzeichnis ...

Links

- *Systemd* bei Wikipedia (de)
- *Systemd* bei Wikipedia (en)
- *Systemd*-Homepage
- *Systemd*-Socket-Aktivierung (Lennart Poettering)
- https://www.digitalocean.com/community/tutorials/understanding-systemd-units-and-unit-files
- https://wiki.archlinux.org/title/Systemd
- https://www.tecmint.com/systemd-replaces-init-in-linux/
- https://www.linux-community.de/ausgaben/linuxuser/2014/04/systemd-als-