# systemd

Alex Barton @ CETiK 2017

„***systemd*** is a suite of basic building blocks for a Linux system. It provides a system and service manager that runs as PID 1 and starts the rest of the system.“

*–https://www.freedesktop.org/wiki/Software/systemd/*

„***systemd*** is a suite of basic building blocks for a Linux system. It provides a system and **service manager** that runs as PID 1 and starts the rest of the system. ***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**, maintains **mount and automount points**, and implements an elaborate **transactional dependency-based service control logic**. systemd supports **SysV and LSB init scripts** and works as a replacement for sysvinit. Other parts include a **logging daemon**, utilities to control basic system configuration like the **hostname**, **date**, **locale**, maintain a list of **logged-in users** and **running containers** and **virtual machines**, system **accounts**, **runtime directories** and settings, and daemons to manage simple **network configuration**, network **time synchronization**, **log forwarding**, and **name resolution**.“

# systemd als Service Manager

- **Default** bei allen großen Distributionen (RHEL, SLES, CentOS, Debian, Ubuntu, Fedora, …)
- Schon deshalb sollte man sich ein wenig damit befassen – unabhängig der persönlicher Präferenz 😎

# systemd als Service Manager

- **`systemd`**(1) ersetzt **`init`**(8) vollständig als PID 1
- Keine Skripte wie bei SysV-init, sondern **Konfigurationsdateien** („unit files“)
- Verschiedene Arten von „Units“: **Service**, **Mount Point**, **Socket**, **Timer**, **Target**, …
- Keine „Runlevel“ sondern „Targets“
- API zur Verwaltung: **`systemctl`**(1), D-BUS

```
debian9:~ # systemctl status
● debian9
    State: running
     Jobs: 0 queued
   Failed: 0 units
    Since: Sat 2017-10-07 18:20:56 CEST; 3 weeks 5 days ago
   CGroup: /
           ├─user.slice
           │ └─user-1000.slice
           │   ├─session-c878.scope
           │   │ ├─20206 sshd: alex [priv]
           │   │ ├─20213 sshd: alex@pts/0
           │   │ └─20315 -zsh
           │   ├─user@1000.service
           │   │ ├─gpg-agent.service
           │   │ │ └─6928 /usr/bin/gpg-agent --supervised
           │   │ ├─init.scope
           │   │ │ ├─592 /lib/systemd/systemd --user
           │   │ │ └─685 (sd-pam)
           │ [……]
           ├─init.scope
           │ └─1 /sbin/init
           └─system.slice
             ├─ttyd.service
             │ └─10851 /usr/bin/ttyd -i lo -c arthur:G3tC0ns0le! login
             ├─nfs-mountd.service
             │ └─3072 /usr/sbin/rpc.mountd --manage-gids
             ├─isc-dhcp-server.service
             │ └─1662 /usr/sbin/dhcpd -4 -cf /etc/dhcp/dhcpd.conf lan0
           […]
```

**Overall System Status**

**„User Slices“**

**„System Slices“**

**„Service“**

```
debian9:~ # systemctl status postfix@-.service
● postfix@-.service - Postfix Mail Transport Agent (instance -)
   Loaded: loaded (/lib/systemd/system/postfix@.service; enabled-runtime; vendor preset: enabled)
   Active: active (running) since Sat 2017-10-07 18:24:29 CEST; 3 weeks 5 days ago
     Docs: man:postfix(1)
    Tasks: 8 (limit: 4915)
   CGroup: /system.slice/system-postfix.slice/postfix@-.service
           ├─ 2099 pickup -l -t fifo -u -c
           ├─ 2213 /usr/lib/postfix/sbin/master -w
           ├─ 7354 qmgr -l -t fifo -u
           ├─ 7516 tlsmgr -l -t unix -u -c
           ├─ 7644 smtpd -n smtp -t inet -u -c -o stress= -s 2
           ├─29197 anvil -l -t unix -u -c
           ├─30366 smtpd -n smtp -t inet -u -c -o stress= -s 2
           └─30432 showq -t unix -u -c

Nov 03 10:58:04 arthur postfix/smtpd[30366]: warning: hostname newsoft1011.example.com does not
resolve to address 91.200.12.117: Name or service not known
Nov 03 10:58:04 arthur postfix/smtpd[30366]: connect from unknown[91.200.12.117]
Nov 03 10:58:06 arthur postfix/smtpd[30366]: warning: unknown[91.200.12.117]: SASL LOGIN
authentication failed: authentication failure
Nov 03 10:58:06 arthur postfix/smtpd[30366]: lost connection after AUTH from
unknown[91.200.12.117]
Nov 03 10:58:06 arthur postfix/smtpd[30366]: disconnect from unknown[91.200.12.117] ehlo=1
auth=0/1 commands=1/2
```

```
debian9:~ # systemctl cat sshd.service
# /lib/systemd/system/ssh.service
[Unit]
Description=OpenBSD Secure Shell server
After=network.target auditd.service
ConditionPathExists=!/etc/ssh/sshd_not_to_be_run

[Service]
EnvironmentFile=-/etc/default/ssh
ExecStart=/usr/sbin/sshd -D $SSHD_OPTS
ExecReload=/bin/kill -HUP $MAINPID
KillMode=process
Restart=on-failure
RestartPreventExitStatus=255
Type=notify

[Install]
WantedBy=multi-user.target
Alias=sshd.service

debian9:~ # wc -l /etc/init.d/ssh
174 /etc/init.d/ssh
```

17!

```
debian9:~ # systemctl cat netdata.service
# /lib/systemd/system/netdata.service
[Unit]
Description=netdata real-time system monitoring
After=network.target

[Service]
Type=simple
EnvironmentFile=-/etc/default/netdata
ExecStart=/usr/sbin/netdata -D $EXTRA_OPTS
TimeoutStopSec=30
Restart=always
RestartSec=5

[Install]
WantedBy=multi-user.target

# /etc/systemd/system/netdata.service.d/override.conf
[Service]
TimeoutStopSec=2m
```

- **systemctl** {**start**|**stop**|**restart**|**reload**|**try-restart**|**reload-or-restart**|**try-reload-or-restart**|…} <service> [<service> […]]
- **systemctl** {**enable**|**disable**|**mask**} [--now] <service> [<service> […]]
- **systemctl status** <pattern|PID> [<pattern|PID> […]]
- **systemctl list-unit-files** [<pattern>]
- **systemctl list-**{**units**|**timers**|**sockets**} [<pattern>]
- **systemctl** {**is-enabled**|**is-active**|**is-failed**} <pattern>
- …

# systemd als Service Manager

- Für bestehende Init-Scripte werden dynamisch „Units“ generiert, ebenso für Dateisysteme in `/etc/fstab` etc.
  - **systemd.generator**(7)
  - **systemd-sysv-generator**(8)
  - **systemd-fstab-generator**(8)
  - **systemd-cryptsetup-generator**(8)

# systemd als Service Manager

- Gute **Parallelisierung** beim Systemstart
- Gute Verwaltung der **Abhängigkeiten**
- **Socket Activation**: Dienste erst bei Bedarf starten
- **Control Groups** (cgroups): „Isolation“ der gestarteten Services (Prozessgruppen)
- Reagieren auf **Ereignisse**: z. B. automatisches Neustarten von Diensten im Fehlerfall
- CLI-Tools, API (D-BUS)
- Für System- und Benutzersitzungen geeignet

# Unit Files

- „INI-Dateien“ mit Verschiedene Sektionen, wie [**Unit**], [**Service**], [**Install**] – abhängig vom Unit-Typ
  Manual pages: `systemd.unit(5)`, `systemd.service(5)`, …
- Gute Trennung zwischen System („Vendor“) Konfiguration, lokaler Konfiguration und dynamischer Konfiguration:
  - `/lib/systemd/…`
  - `/etc/systemd/…`
  - `/run/systemd/…`
- Override-Dateien als „drop-in“

```
# /etc/systemd/system/sleep.service
[Unit]
Description=Sleep is good!

[Service]
ExecStart=/usr/sbin/sleep 30

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

```
# /lib/systemd/system/ssh.service
[Unit]
Description=OpenBSD Secure Shell server
After=network.target auditd.service
ConditionPathExists=!/etc/ssh/sshd_not_to_be_run

[Service]
EnvironmentFile=-/etc/default/ssh
ExecStart=/usr/sbin/sshd -D $SSHD_OPTS
ExecReload=/bin/kill -HUP $MAINPID
KillMode=process
Restart=on-failure
RestartPreventExitStatus=255
Type=notify

[Install]
WantedBy=multi-user.target
Alias=sshd.service
```

# „Override Files“

- In `/etc/systemd/system/<unit>.service.d/`
- Am Besten mit folgendem Befehl bearbeiten: **`systemctl edit`** `<service>`
- Kann gut kopiert & automatisiert werden
- Wir nicht bei Updates durch den Paketmanager überschrieben, d. h. Änderungen sind dauerhaft
- Achtung: **`systemd`**`(1)` nach manueller Änderung Unit-Dateien neu lesen lassen: **`systemctl daemon-reload`**

# Neue Unit anlegen

1. Unit File in **`/etc/systemd/system/<name>.service`** mit beliebigem Editor anlegen / kopieren / …

2. systemd(1) informieren: **`systemctl daemon-reload`**

3. Unit starten und automatisch bei Systemstart aktivieren: **`systemctl enable --now`** `<service>`

   Letzteres entspricht:
   **`systemctl enable`** `<service>`
   **`systemctl start`** `<service>`

4. Ergebnis prüfen: **`systemctl status`** `<service>`

„***systemd*** is a suite of basic building blocks for a Linux system. It provides a system and **service manager** that runs as PID 1 and starts the rest of the system. ***systemd*** provides **aggressive parallelization** capabilities, uses **socket and D-Bus ac** rting services, offers **on-demand sta** of daem keeps track of processes using Li control gr s, maintains **mount and automount points**, and im ents an elaborate **transactional dependency-ba ervice control logic**. systemd supports **SysV an B init scripts** and works as a replacement for sy t. Other parts include a **logging daemon**, utilit o control basic system configuration like the **hostname**, **date**, **locale**, maintain a list of **logged-in users unning containers** and **virtual machines**, system unts, **runtime directories** and settings, and daemons to manage simple **network configuration**, network **time synchronization**, **log forwarding**, and **name resolution**.“

- **systemctl**(1) – Dienste verwalten [systemd(1)]
- **journalctl**(1) – „System-Journal“ abfragen [systemd-journald(1)]
- **hostnamectl**(1) – Hostname setzen & verwalten
- **localectl**(1) – System „locale“ und Tastatur verwalten
- **loginctl**(1) – „Login manager“ verwalten [systemd-logind(8)]
- **machinectl**(1) – VMs und Container verwalten [systemd-machined(8)]
- **networkctl**(1) – Netzwerkstatus abfragen [systemd-networkd(8)]
- **coredumpctl**(1) – core dumps auswerten [systemd-coredump(8)]

- **systemd-cgls**(1) – Control-Group-Hierarchie anzeigen
- **systemd-cgtop**(1) – Control Groups nach Nutzung anzeigen
- **systemd-nspawn**(1) – „Namespace Container“ starten
- **systemd-run**(1) – Programm als „unit“ starten
- **systemd-analyze**(1) – Systemstart analysieren
- **systemd-cat**(1) – Daten ins Journal schreiben
- …

# systemd Journal

„*systemd-journald* is a system service that **collects and stores logging data**. It creates and maintains **structured**, **indexed** journals based on logging information that is received from a **variety of sources**“

*–systemd-journald.service(8)*

# systemd Journal

- Bei Debian per Default installiert, allerdings nicht persistent: das Journal wird nur im RAM gehalten und überdauert keine Systemstarts.

- Achtung: Logs nicht doppelt speichern, d. h. entweder mit **`syslogd`**`(8)` oder mit **`systemd-journald`**`(8)` – nicht mit beiden gleichzeitig (IO-Last …)!

- Persistentes Journal aktivieren (Debian):
  **`mkdir -p /var/log/journal`**
  **`systemd-tmpfiles --create --prefix /var/log/journal`**
  (vgl. `/usr/share/doc/systemd/README.Debian.gz`)

# systemd Journal

- Journal ist eine **binäre Datenbank**, kein reiner Text
- Einträge sind **strukturiert** und werden mit zusätzlichen **Attributen** versehen (wie z. B. Zeitstempel, Loglevel, Prozess, Benutzer,  Service Unit, Fehlernummer, Boot-ID, Maschinen-ID, Hostname, SELinux Context, …)
- Zusätzliche Attribute können zur Suche genutzt werden
- Position im Journal („cursor“) kann abgefragt und angesteuert werden – nützlich für Skripte

# Journal abfragen

```
alex@Arthur:~ $ journalctl -u ngircd.service -p warning --since yesterday
-- Logs begin at Mon 2017-09-25 01:41:30 CEST, end at Sat 2017-11-04 02:21:54 CET. --
Nov 03 05:01:52 arthur ngircd[1528]: Can't resolve address "117.50.7.159":
                                     Name or service not known [getnameinfo].
Nov 03 15:29:39 arthur ngircd[28345]: Can't resolve address „66.213.67.10":
                                      Name or service not known [getnameinfo].
Nov 03 17:40:19 arthur ngircd[646]: ID "chat.cetik.netz" already registered
                                    (on connection 31)!
Nov 03 17:41:32 arthur ngircd[646]: ID "chat.cetik.netz" already registered
                                    (on connection 31)!
Nov 03 17:42:32 arthur ngircd[646]: ID "chat.cetik.netz" already registered
                                    (on connection 31)!
Nov 03 23:56:51 arthur ngircd[646]: SSL error, client disconnected
                                    [in ConnSSL_Read()]!
```

# Journal abfragen

- Alle Fehlermeldungen im Journal seit dem letzten Systemstart anzeigen:
  `journalctl -b -p err`

- Alle Warn- und Fehlermeldungen seit Mitternacht („heute“) anzeigen:
  `journalctl -p warning --since today`

- Meldungen von einem bestimmten Binary fortlaufend anzeigen:
  `journalctl -f _EXE=/usr/sbin/sshd`

- Alle Kernel-Meldungen mit Level „Warnung“ und höher anzeigen:
  `journalctl -b _TRANSPORT=kernel -p warning`

# Nützliche Befehle

- Systemstatus anzeigen:

```
systemctl status
```

- Alle Units mit Fehlern auflisten:

```
systemctl --all --failed
0 loaded units listed.
To show all installed unit files use
'systemctl list-unit-files’.
```

# Nützliche Befehle

- Eigene Daten ins Journal schreiben mit eigenem Identifier:
  ```
  echo "hello world" | systemd-cat -t my_identifier
  journalctl -t my_identifier
  ```

- Alle aktivierten Unis auflisten:
  ```
  systemctl list-unit-files -state=enabled
  ```

- Unit bearbeiten (ohne `--full`: Override bearbeiten):

  ```
  export SYSTEMD_EDITOR=vim
  sudo -E systemctl edit --full <unit_name>
  ```

# Nützliche Befehle

- Abhängigkeiten von einer Uni anzeigen:
  `systemctl list-dependencies <service>`

- „Control Group Tree“ anzeigen:
  `systemd-cgls`
  `systemd-cgtop`

- `ps(1)` mit Angabe der „Control Group“:
  `ps xawf -eo pid,user,cgroup,args`

# Fragen?

# Demo?