2025/11/04 18:38 1/5 Systemd

Systemd

Ein Service bearbeiten und personalisieren.

```
systemctl edit --full rc-local
```

Oder wenn es ein komplett neues Unitfile ist:

```
systemctl edit -f -l rc-local
```

Man könnte auch Dienste direkt in /etc/systemd/system/blabla-custom.service kopieren. Von dem wird abgeraten. Vor allem da viele Services erst von anderen Systemddiensten nur temporär angelegt werden. Das Kommando oben extrahiert die richtigen Files automatisch.

Beispiel Einbau von Sleep beim Start eines Services

```
[Unit]
Description=Puppet agent
Requires=network.target
[Service]
Type=forking
EnvironmentFile=-/etc/default/puppet
PIDFile=/run/puppet/agent.pid
ExecStartPre=/bin/sleep 15
ExecStart=/usr/bin/puppet agent $DAEMON_OPTS
[Install]
WantedBy=multi-user.target
```

Timeout beim Beenden eines Services

Sehr nützlich wenn durch z.B. nicht mehr erreichen von Services wie NB's - WLAN Dienste ihr maximales Timeout erreichen würden.

```
[Unit]
Description=Make remote CUPS printers available locally
After=cups.service avahi-daemon.service
Wants=cups.service avahi-daemon.service

[Service]
TimeoutStopSec=2
ExecStart=/usr/sbin/cups-browsed

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

Autologin systemd auf der Konsole ohne Displaymanager

Hierzu legt man sich folgendes File an:

nano /etc/systemd/system/getty@ttyl.service.d/override.conf

Mit folgenden Inhalt

```
[Service]
ExecStart=
ExecStart=-/sbin/agetty --autologin xbmc --noclear %I 38400 linux
```

Hier wird z.B. der xbmc Benutzer automatisch eingeloggt. Danach werden natürlich .zshrc .bashrc und auch die .xinitrc beachtet.

Systemdservices über Remote ausführen

Mit Systemd ist es sehr bequem möglich Dinge zu organisieren ohne das man direkt am Host ist. z.b.

```
systemctl -H root@myhost.supertux.bla status apache2
```

NFS-Client

```
systemctl enable nfs-client.target
systemctl enable rpc-statd.service
systemctl enable rpcbind.service
```

Mounten mit Systemd - FSTAB ruhe in Frieden

<u>Testsystem: Debian 10/Proxmox 6.3</u>

Die FSTAB ist mehr oder weniger überholt. Benötigt wird sie vom System wohl nur mehr für den Root Mount. Generell wird bereits bei jedem manuellen FSTAB-Eintrag ein Systemd-Unit-File generiert dass auf die FSTAB verweist. Daher wird auch empfohlen statt der FSTAB nur mehr Systemd zu verwenden, was sehr viele Vorteile mit sich bringt. Z.B. hat man damit die Möglichkeit auch Abhängigkeiten von anderen Programmen und States anzugeben. Ein praktischen nerviges Beispiel wäre wenn ein Netzwerkmount nicht immer automatisch in der FSTAB gemountet wird, weil das Netzwerk vielleicht nicht immer gleich schnell verfügbar ist. Und obwohl man in der FSTAB die Option gesetzt hat dass, das Netzwerk verfügbar sein muss, funktioniert es trotzdem doch immer nicht. Systemd schafft hier für dich Abhilfe. Auch Proxmox verwendet den Systemd-Mounter als Default.

Hier als Beispiel ein einfacher Mount einer lokalen HDD. Als erstes legst du ein sogenanntes Unit-File

2025/11/04 18:38 3/5 Systemd

an. Der Mountpoint wird dabei automatisch erstellt.

```
systemctl edit -f -l "/mnt/datastore/HDD-extern-OSIT"
```

Wie du siehst muss der Name der exakte Mountpoint sein. Nun befüllst du das File mit diesem Inhalt:

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

[Unit]
Description=Mount datatstore 'sicherung-osit-extern' under
'/mnt/datastore/HDD-extern-OSIT'

[Mount]
Options=defaults
```

Type=ext4

What=/dev/disk/by-uuid/d6b3aa86-aa6c-4b41-b6b2-16457820169629

Where=/mnt/datastore/HDD-extern-OSIT

Mit dem nächsten Befehl hast eine tolle Übersicht für alle Mountpoints die es gibt, und ob diese im Autostart sind oder nicht.

systemctl list-unit-files -t mount

```
UNIT FILE
                                           STATE
-.mount
                                           generated
boot-efi.mount
                                           generated
dev-hugepages.mount
                                           static
dev-mqueue.mount
                                           static
mnt-datastore-HDD\x2dextern\x2dOSIT.mount disabled
proc-fs-nfsd.mount
                                           static
proc-sys-fs-binfmt misc.mount
                                           static
run-rpc_pipefs.mount
                                           static
sys-fs-fuse-connections.mount
                                           static
sys-kernel-config.mount
                                           static
sys-kernel-debug.mount
                                            static
```

In den Autostart damit:

```
systemctl enable "mnt-datastore-HDD\x2dextern\x2dOSIT.mount"
```

Und mounten:

```
systemctl start mnt-datastore-HDD\\x2dextern\\x2dOSIT.mount
```

Bei der Mountübersicht sieht das ganze nun so aus:

```
UNIT FILE STATE
-.mount generated
boot-efi.mount generated
```

<pre>dev-hugepages.mount dev-mqueue.mount mnt-datastore-HDD\x2dextern\x2d0SIT.mount proc-fs-nfsd.mount proc-sys-fs-binfmt misc.mount</pre>	static static enabled static static
run-rpc_pipefs.mount sys-fs-fuse-connections.mount sys-kernel-config.mount sys-kernel-debug.mount	static static static static

Bestehende Unitfiles kann mit dem folgenden Befehlen editieren:

```
systemctl edit -l mnt-datastore-HDD\\x2dextern\\x2d0SIT.mount
```

oder auch:

```
systemctl edit -l "/mnt/datastore/HDD-extern-OSIT"
```

Für die Erweiterung deines Unitfiles empfehle ich diesen Artikel und auch diesen auf Ubuntuusers.

Debuging

Um z.B. Zeiten beim Systemstart ansehen zu können gibt es zwei nette Befehle:

```
systemd-analyze plot > bootchart.svg
systemd-analyze blame
```

Weitere nützliche Systemd-Befehle

```
systemctl reset-failed
systemctl --failed
```

Links

- Hersteller|Dokumentation Systemd
- https://wiki.ubuntuusers.de/systemd/systemctl/
- https://wiki.ubuntuusers.de/systemd/Mount_Units/

2025/11/04 18:38 5/5 Systemd

From:

https://wiki.deepdoc.at/dokuwiki/ - DEEPDOC.AT - enjoy your brain

Permanent link:

https://wiki.deepdoc.at/dokuwiki/doku.php?id=server_und_serverdienste:systemd&rev=1611871240

Last update: 2021/01/28 22:00

