Admin's Workshop:
Eine Reise durch das system(d)

Secure Linux
Administration
Conference

Berlin, 17.06.2016

Bjørn Bürger
bbu@pengutronix.de

motp @ {IRCnet, freenode, hackint, wunder-nett}

```
0183 3553 7110 3405 10B5  1CF6 4F4B B9AE 2074 ED8E
5B10 E8EB 86B7 2F61 65E6  21EA 41D4 4696 088D AB19
```

bbu:~$ ▮

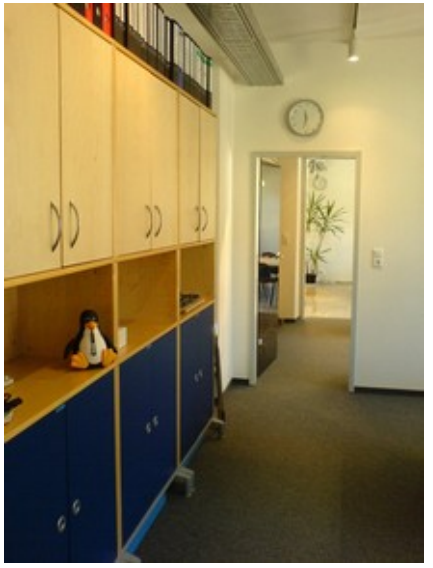
Bjørn Bürger

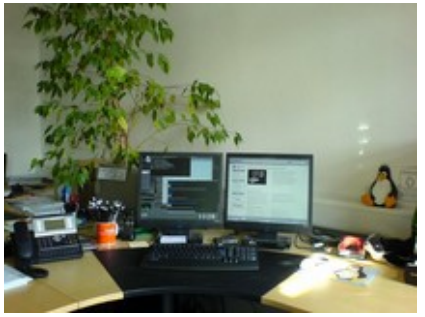
- 80er Jahre
  - Erster Computer-Kontakt
  - UniFlex (S-Plus South West Computer / MC68B09 / 2 MHz)
  - AMSDOS (Schneider CPC 6128 / Z80 / 4 MHz)
- 90er Jahre
  - Atari ST / PAK68 — Linux m68k
  - PC — SuSE 4.2 [...]
- TU Braunschweig
  - Linux User Group BS
  - Braunschweiger Linuxtage
  - Studenten-Netz
- Seit 2003
  - Embedded Linux / Entwicklung
  - Infrastruktur / Admin

Pengutronix

- Dienstleister
  - Embedded Linux Projekte
  - Free/Libre Open Source Software
  - > 3600 Beiträge im offiziellen Linux Kernel
- Seit 2001 in Hildesheim
- 22 Festangestellte

- Infrastruktur: Debian GNU/Linux
  Zentrale Server-Infrastruktur
  Remote-Labore
  Desktop-Clients: PC





User / Desktop

- Schneller Start
- Plug n' Play
- Multiseat Management
- Sicherheit
- Privatsphäre

- Voraussagbares Verhalten
- Gute Dokumentation
- Leicht auffindbare Logfiles

- Session Management
  - für das System
  - incl. Powermanagement Features, Netzwerk, etc.
  - für die User Session

- Desktop Virtualisierung

Admins

- Robuster Betrieb
  - Uptimes von mehr als 300 Tagen keine Seltenheit
  - Einheitliche Konfiguration verschiedener Distributionen
  - Einbindung in Configuration Management
- Container Management
  - Hohe Service-Dichte
  - Heterogene VM-Landschaft
- Gute Konzepte aus der Unix-Welt nutzen
  - kleine, spezialisierte Tools
  - Scripting als „Kleber“
  - "Everything is a file"
  - In-System Dokumentation
  - POSIX
- Schlechte Beispiele aus der Unix-Welt vermeiden
  - Kommerzielle Spezial-Tools
  - fehlende Standards

(Embedded-) Entwickler

- Service Manager
  - Uptimes im Bereich von Jahren keine Seltenheit
  - Restart eines "toten" Service
  - Recovery
  - softwaregestützte Ausfallerkennung
  - hardwaregestützte Ausfallerkennung

- Management der Systemressourcen
  - Services nur bei Bedarf starten
  - Services abhängig von Hardware-Aspekt starten
    - sshd nur während der Wartung
    - Debug Console, wenn USB-Seriell-Adapter steckt
    - Upgrade bei Anschluß eines bestimmten USB-Sticks

- Schneller Start
- Trusted Boot / IMA
- R/O Dateisystem
- Factory Reset
- Remote Update

Community

- Erwartungen der Nutzer:
  - Eine lebendige Community
  - Gute Dokumentation
  - Die Möglichkeit zum Mitgestalten
- Erwartungen der Entwickler
  - Wenig Aufwand für die Upstream Projekte
  - Verläßliche Roadmaps
  - Keine Unangenehmen Überraschungen

https://www.freedesktop.org/wiki/Software/systemd/InterfacePortabilityAndStabilityChart/

Community

- Erwartungen der Nutzer:
  - Eine lebendige Community
  - Gute Dokumentation
  - Die Möglichkeit -
- Erwartunge

eam Projekte

ps

genehmen Überraschungen

h freedesktop.org/wiki/Software/systemd/InterfacePortabilityAndStabilityChart/

systemd - Stolperfallen

- Mehr Magie im System
  - Instanziierte Services
  - Beispiel: ttys
  - Impliziter Automounter für eine Menge Filesysteme
  - sysctl.conf.d

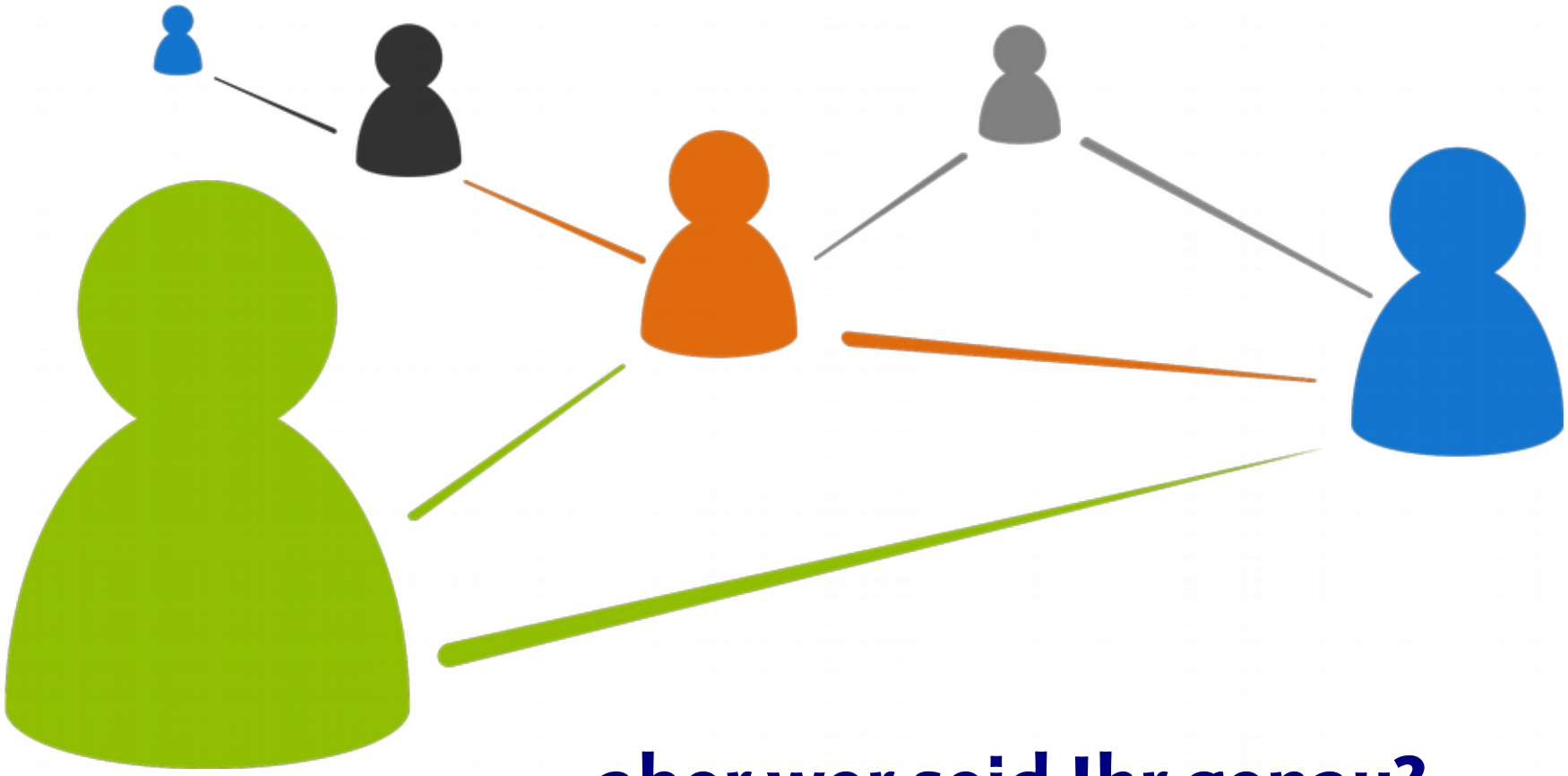
- shutdown
  - -H ist jetzt wirklich wieder "halt"
  - -P ist jetzt wirklich "poweroff"

Folie von 2015
„Admin's Diary“

- Es wird hinter den Prozessen aufgeräumt
  - cgroups (!)
  - Bug oder Feature:
    Alle zu einer ssh session gehörenden
    Prozesse werden beim logout gekillt...

- Realtime Budget für bestimmt Prozesse?

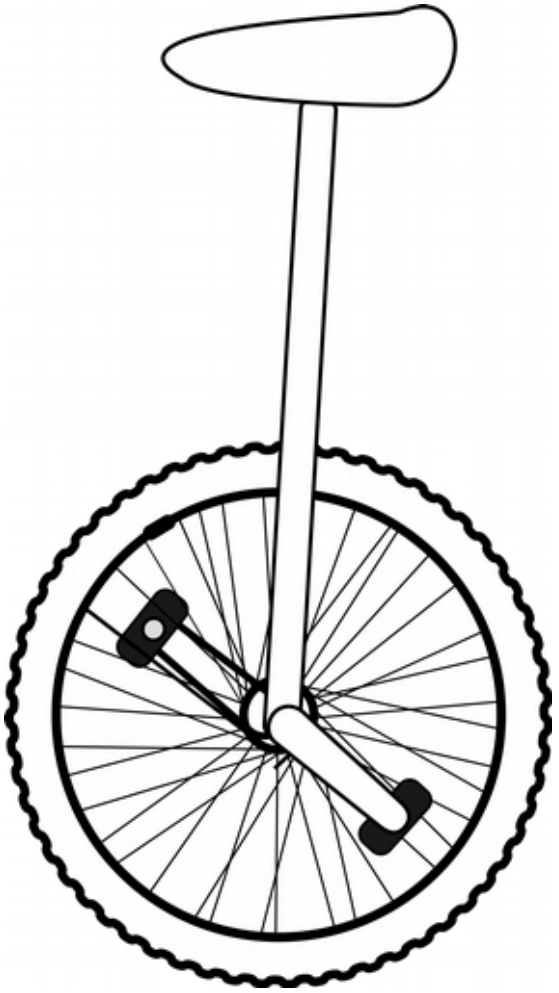
Ihr seid die Community...



... aber wer seid Ihr genau?

https://www.penguin.de/public/talks/2016/slac/

systemd



~~system D~~
System D
~~SystemD~~

https://www.freedesktop.org/wiki/Software/systemd/#spelling

Init

Embedded Init

Individuelle Lösungen

systemd

Kompatibilität

systemd

lsb / sysVinit

systemd

- Ein System-, Service- und Session Manager für Linux
- Kompatibel mit SysV und LSB Init Scripts
- Wird inzwischen von allen neueren Linux Distros benutzt
- LFS: http://linuxfromscratch.org/lfs/view/stable-systemd/

- systemd nutzt die aktuelle Linux Infrastruktur
  - udev
  - inotify
  - dbus
  - cgroups
  - pam
  - Unix Ressource Limits
  - Capabilities,
  - SELINUX, AppArmor, [...]

- Tracking von **Services und Sessions**, nicht nur von Prozessen
- Überwachung des Service während der gesamten Laufzeit
- Watchdog Kette von der Hardware bis zur Applikation

systemd - Für wen ist es gemacht?

- Server
  - Container Support
  - Lange Laufzeiten
  - Zentrales Management
- Embedded
  - „Headless“ Betrieb
  - „Unattended Upgrades“
  - Ressourcen!
- Desktop
  - Speed!
  - Desktops Session Management
  - Multiseat Support
- Deine Anwendung ist nicht dabei?
  - Send Patches!

systemd – Ziele

- Gesteigerte Anforderungen an Linux Systeme
  - Modularisierung
  - asynchrone Job Bearbeitung
  - deklarative Konfiguration
  - Tooling (Analyse, Tests) integraler Bestandteil
- Distributionsübergreifende Vereinheitlichung
  Upstream ist beste Quelle für systemd units
- Einfachere Nutzung aktueller Kernel APIs
  durch simple Konfigurationsdateien
- Schaffung sicherer Default Konfigurationen
- Flexibilität durch vielfältige Language-Bindings
- Dinge grundlegend RICHTIG machen,
  statt ewig mit Kompromissen zu leben

systemd – Ziele

- Gesteigerte Anforderungen an Linux Systeme
  - Modularisierung
  - asynchrone Job Bearbeitung
  - deklarative Konfiguration
  - Tooling (Analyse, Tests) integraler Bestandteil
- Distributionsübergreifende Vereinheitlichung
  Upstream ist beste Quelle für systemd units
- Einfachere Nutzung aktueller Kernel APIs
  durch simple Konfigurationsdateien
- Schaffung sicherer Default Konfigurationen
- Flexibilität durch vielfältige Language-Bindings
- Dinge grundlegend RICHTIG machen,
  statt ewig mit Kompromissen zu leben

Wir verlassen hier die Komfortzone :-)

systemd – user sessions

- Wozu braucht man das?
- Service Management für User Services
- Autostart unabhängig von ...
  - Desktop-Environment
  - Terminal-Art (X, Wayland, TTY)
  - login des users
- z.B. screen / tmux Session schon beim Systemstart

```
~/.config/systemd/user/mypersonaldaemon.service

[Unit]
Description=My personal daemon
Documentation=man:mypersonaldaemon(1)

[Service]
ExecStart=/usr/local/bin/mypersonald
ExecStop=/usr/local/bin/mypersonald --shutdown

[Install]
WantedBy=default.target
```

Stateless Systems

- /usr -> Betriebssystem
- /etc -> Konfiguration (opt)
- /var -> State (opt)

- Ziel: Read-Only System im /usr (stateless)
- /etc ist nur noch minimale, lokale Konfiguration

```
.include /usr/lib/systemd/system/some.service

[Service]
IOSchedulingClass=idle
```

- Alternativ: Droplets
  /{run,etc,lib}/systemd/system/some.service.d/*.config

- Container-Instanzen können vom Basis-System "erben"
- Factory Reset made simple

systemd – Ökosystem

- Stabile Grundlage für andere Projekte
  - Desktop Environments
  - Netzwerk Setup
  - VM / Container Management
- Zentrale Ressourcenkontrolle
  - Reduktion redundanter Implementierungen
  - Verbessertes Power Management
    - Stichwort Cron, Lidswitch, etc
  - Abstraktion komplexer Kernel-APIs
  - Verbesserte Sicherheit

systemd - Ökosystem

- systemd entwickelt sich nicht im luftleeren Raum
- Immer mit hinreichend aktuellen Kerneln einsetzen
- Andere Projekte beginnen, systemd Funktionalität vorauszusetzen
- Es gibt im Umfeld von systemd immer mehr spannende Projekte
  - Tools
    - systemd-cron
    - journal-brief
    - gnome-logs
    - libnss-resolve
  - Frameworks
    - CoreOS
    - fleet / etcd

Guter Einstieg ins Thema:
The systemd for Administrators Blog Series
http://www.freedesktop.org/wiki/Software/systemd/