Prozesse, Logs und Systemverwaltung

Linux-Kurs der Unix-AG

Zinching Dang

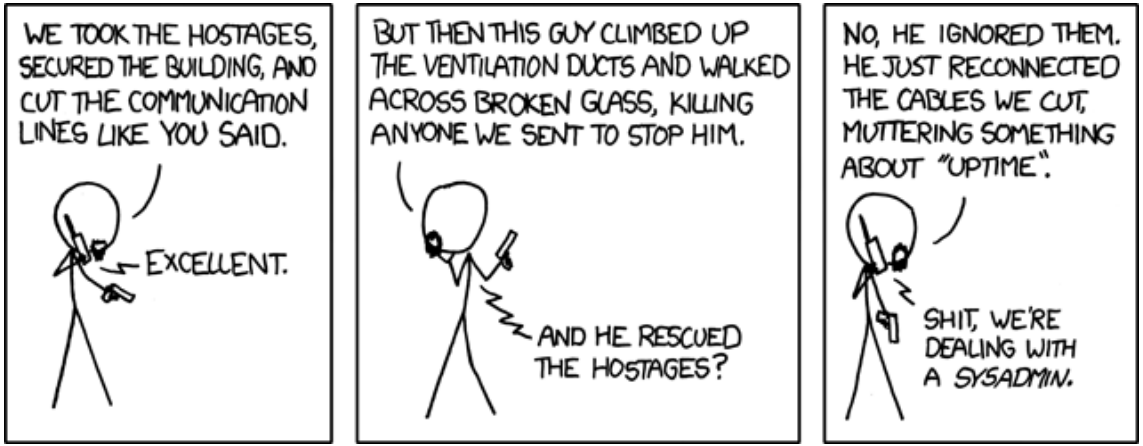
31. Januar 2018

UNIX AG
TU Kaiserslautern

RH Regionales Hochschul-Rechenzentrum Kaiserslautern RK

Übersicht

Übersicht



https://xkcd.com/705

Wiederholung & Vertiefung: Benutzer & Gruppen

UID, GID, Zugriffsrechte und Datei-Besitzer/Gruppen

- jeder Benutzer hat eine UID und GID
- Dateien sind einem Benutzer (**u**ser, Datei-Besitzer) und einer Gruppe (**g**roup, Datei-Gruppe) zugeordnet
- die Zugriffsrechte **r**ead, **w**rite und e**x**ecute sind definiert für:
  - **u**ser (Datei-Besitzer)
  - **g**roup (Datei-Gruppe)
  - **o**thers (alle anderen)

Zugriffsrechte

Zugriffsrechte für Benutzer & Gruppen

- beispielhafteAusgabe von `ls -ha` :

```
-rw-rw-r--  1 linuxkurs proj1 6K  Nov 6 Linux
```

user **o**thers Besitzer Gruppe

group

Wiederholung & Vertiefung: Benutzer & Gruppen

Symbolische und oktale Notation von Zugriffsrechten

```
-rw-rw-r-- 1 linuxkurs proj1 6K  Nov 6 Linux
```

- Symbolische Notation: drei Zugriffsrechte (**r**, **w**, **x**) je Zugriffs-Gruppe (**u**, **g**, **o**)
  - neun Zeichen um alle Zugriffsrechte zu beschreiben
- Oktale Notation: Summe aus den Zugriffsrechte jeder Zugriffs-Gruppe
  - **r**ead $\hat{=}$ 4, **w**rite $\hat{=}$ 2, e**x**ecute $\hat{=}$ 1, keine/**-** $\hat{=}$ 0
  - drei Ziffern von 0 – 7 um alle Zugriffsrechte zu beschreiben

Wiederholung & Vertiefung: Benutzer & Gruppen

Besitzer, Gruppen und Zugriffsrechte ändern

- `chown [besitzer][:[gruppe]] datei`
  - ändert den Besitzer und/oder die Gruppe
- `chmod <Zugriffsrechte> datei`
  - `<Zugriffsrechte>` in symbolischer oder oktaler Notation
  - bspw. `u=rwx,g=rx,o=` bzw. `750`
  - nur symbolische Notation: Ändern mit `+` , `-` oder `=`

Wiederholung & Vertiefung: Benutzer & Gruppen

Besondere Zugriffsrechte

- Set-UID-Bit: `s` zusätzlich bei den Besitzer-Zugriffsrechten
  - Programm wird mit der Rechten des Datei-Besitzers ausgeführt
  - keine Relevanz bei Verzeichnissen
- Set-GID-Bit: `s` zusätzlich bei den Gruppen-Zugriffsrechten
  - Programm wird mit der Rechten der Datei-Gruppe ausgeführt
  - neu erstellte Dateien und Verzeichnisse erben Verzeichnis-Gruppe
- Sticky-Bit: `t` zusätzlich bei den Zugriffsrechten aller anderen
  - keine Relevanz bei Dateien
  - in Verzeichnissen kann nur der Besitzer einer Datei diese löschen

Prozesse unter Linux

Ein Prozess

- entsteht durch einen Programmaufruf
- hat eine eindeutige PID (**P**rocess **ID**)
- ist einem Benutzer zugeordnet
- wird von einem Eltern-Prozess (Parent Process) gestartet
- wird Kind-Prozess (Child Process) genannt

Prozesse unter Linux

Der `init`-Prozess

- hat die PID 1
- wird durch den Kernel als erstes gestartet
- hat sämtliche Prozesse als Nachfahren

Prozess-Eigenschafen

- können von jedem Benutzer angeschaut werden
- dürfen nur vom Prozess-Besitzer (oder `root`) verändert werden

Prozesse unter Linux

Befehle

- `ps` , `pstree` – Prozesse auflisten
- `uptime` – Systemlaufzeit und -auslastung anzeigen
- `free` – Arbeitsspeicherbelegung
- `kill` , `killall` – Prozesse beenden
- `top` – Taskmanager für die Shell anzeigen

Prozesse auflisten – `ps` & `pstree`

Allgemeines

- zeigen die aktuell laufenden Prozesse an
- `ps` zeigt diese in einer Liste mit vielen Information an
- `pstree` stellt diese in einem Baumdiagramm dar

Prozesse auflisten – `ps`

Allgemeines

- detailierte Informationen zu Prozessen
  - optional auch Prozesse aller Benutzer
  - Informationen können anschließend weiterverarbeitet werden
- Argument: optional PID eines Prozesses

Wichtige Optionen – `ps`

- `-a` / `a` – Prozesse anderer Benutzer anzeigen
- `-u` / `u` – Besitzer zu jedem Prozess anzeigen
- `-x` / `x` – alle laufenden Prozesse anzeigen

Prozesse auflisten – `ps`

Beispiel-Ausgabe (gekürzt) von `ps aux`

USER	PID	%CPU	%MEM	VSZ	RSS	TTY	STAT	START	TIME	COMMAND
root	1	0.0	0.1	135200	6860	?	Ss	09:59	0:01	/sbin/init
lk	943	0.0	0.1	47928	7108	pts/6	Ss	11:42	0:00	/usr/bin/bash

- Prozess-Besitzer und PID
- aktuelle Prozessor- und Arbeitsspeicher-Auslastung
- virtuell und tatsächlich belegter Speicher
- Terminal in dem der Prozess läuft und aktueller Status
- Prozess-Start und beanspruchte Rechen-Zeit
- Startbefehl des Prozesses

Prozesse auflisten – `ps`

Prozess-Status

- **R**: running – Prozess rechnet aktuell
- **S**: sleeping – Prozess wartet auf etwas (Benutzereingaben)
- **D**: uninterrupible sleep – Prozess „hängt“ (Festplattenzugriff)
- **T**: stopped – Prozess wurde angehalten
- **Z**: zombie – Prozess wurde beendet, aber der Exit-Status wurde noch nicht vom Eltern-Prozess abgefragt

Prozesse im Baumdiagramm darstellen – `pstree`

Allgemeines

- stellt alle laufenden Prozesse als hierarchisch dar
- Ausgabe kann daher größer ausfallen
- Argument: optional Benutzername oder PID

Wichtige Optionen

- `-p` – PID anzeigen

Systemlaufzeit und -auslastung anzeigen – `uptime`

Allgemeines

- zeigt die aktuelle Uhrzeit, Systemlaufzeit an
- zusätzlich die Systemauslastung der letzten 1, 5 und 15 Minuten
- keine wichtigen Optionen oder Argumente

Arbeitsspeicherbelegung anzeigen – `free`

Allgemeines

- zeigt die Arbeitsspeicher- und Swapbelegung an
  - Swap entspricht der Auslagerungsdatei
- Arbeitsspeicher wird stark zum Cachen verwendet
  - Zwischenspeichern von Festplatteninhalten

Wichtige Optionen

- `-m` – Einheiten in MebiBytes anzeigen

Arbeitsspeicherbelegung anzeigen – `free`

Beispielausgabe von `free -m`

```
             total       used       free     shared    buffers     cached
Mem:          5985       5844        141        135        165       4133
-/+ buffers/cache:       1545       4440
Swap:           65         57       6450
```

- installierter, belegter und freier Arbeitsspeicher (mit Cache)
- Verteilung des Festplattencache
- tatsächlich belegter und freier Arbeitsspeicher
- installierter, belegter und freier Swapspeicher

Prozesse beenden – `kill` & `killall`

Allgemeines

- sendet Signale an Prozesse, insbesondere zum Beenden
- nur `root` darf Signale an Prozesse anderer Benutzer senden
- `init` und Zombies können nicht beendet werden

Wichtige Optionen

- `-9` / `-KILL` – Prozess zum Beenden erzwingen
- `-15` / `-TERM` – Prozess zum Beenden auffordern (Standard)

Prozesse beenden – `kill` & `killall`

Einzelnen Prozess beenden – `kill`

- einzelner Prozess wird anhand einer PID ausgewählt
- Argument: PID eines Prozesses

Mehrere Prozesse beenden – `killall`

- alle Prozesse mit dem gleichen Namen werden beendet
- Argument: Name des Prozesses

Taskmanager – `top`

Allgemeines

- stellt Informationen der folgenden Befehle kombiniert dar
  - `ps` , `uptime` , `free`
- stellt die Funktionalität von `kill` bereit
- Sortierung der Prozesse möglich
  - `P` : nach CPU-Auslastung
  - `M` : nach RAM-Verbrauch
- Beenden mit `q`

Log-Dateien

Allgemeines

- Hintergrundprozesse können keine Meldungen ausgeben
- Meldungen werden oft an `syslogd` geschickt und von diesem in Log-Dateien geschrieben
- werden nach FHS in `/var/log/` gesammelt
- können vertrauliche Informationen enthalten, daher für normale Benutzer nicht lesbar
- Kernelmeldungen können mit `dmesg` abgerufen werden

Wichtige Befehle & Optionen

Befehl	Optionen	Funktion
`ps`	`a` , `u` , `x`	Informationen zu Prozessen anzeigen
`pstree`	`-p`	Prozesse als Baumdiagramm
`uptime`		Systemauslastung und -laufzeit anzeigen
`free`	`-m`	Arbeitsspeicherauslastung anzeigen
`kill`	`-9` , `-15`	Prozesse beenden
`killall`	wie `kill`	Prozesse beenden
`top`		Taskmanager
`dmesg`		Kernelmeldungen anzeigen

Zusammenfassung & Ausblick

Zusammenfassung

- Informationen zu Prozessen auslesen
- Systemauslastung überprüfen und Prozesse beenden
- Log-Dateien und Systemmeldungen

Nächstes Mal

- Paketverwaltung
- Netzwerkgrundlagen