Prozesse und Logs

Linux-Kurs der Unix-AG

Benjamin Eberle

25. Januar 2016

UNIX AG
TU Kaiserslautern

RH Regionales Hochschul-Rechenzentrum Kaiserslautern RK

Prozesse unter Linux

- gestartete Programme laufen unter Linux als Prozesse
- jeder Prozess hat eine eindeutige Prozess-ID (PID)
- jeder Prozess gehört einem Benutzer
- Prozesse werden immer von anderen Prozessen gestartet
- alter Prozess heißt Eltern-Prozess, neuer Prozess heißt Kind-Prozess
- beim Hochfahren wird vom Kernel der erste Prozess `init` gestartet (PID 1)
- alle anderen Prozesse sind Nachfahren von `init`

ps

- `ps` zeigt Informationen über laufende Prozesse an
- aus Kompatibilitätsgründen Optionen mit und ohne „-“
- wichtige Optionen:
  - `f` zeigt rudimentäre Baumstruktur
  - `a` zeigt auch Prozesse anderer Benutzer an
  - `u` zeigt zu jedem Prozess den Besitzer an
  - `x` zeigt auch Prozesse an, die nicht aus einem Terminal heraus gestartet wurden
- optionales Argument: eine Prozess-ID, dann nur Informationen zu diesem Prozess

Felder in der Ausgabe von ps aux

- `USER`: Besitzer des Prozesses
- `PID`: Prozess-ID
- `%CPU` / `%MEM`: Durch diesen Prozess verursachte Prozessor-/Speicherauslastung
- `VSZ`: Durch diesen Prozess belegter virtueller Speicher (RAM + Swap + shared libraries)
- `RSS`: Durch diesen Prozess belegter RAM
- `TTY`: Terminal, in dem der Prozess läuft oder „?“
- `STAT`: Zustand des Prozesses
- `TIME`: Zeit, die der Prozess die CPU benutzt hat
- `COMMAND`: Befehl, mit dem der Prozess gestartet wurde

Zustände eines Prozesses

- R (running): Prozess möchte auf der CPU rechnen
- S (sleeping): Prozess wartet auf etwas (z. B. Eingabe des Benutzers)
- D (uninterruptible sleep): Prozess „hängt“
- T (stopped): Prozess wurde angehalten
- Z (Zombie): Prozess ist beendet, aber der Eltern-Prozess hat den Exit-Status noch nicht abgefragt

kill

- `kill` sendet Signale an Prozesse
- manche Signale können vom Prozess abgefangen und dann speziell behandelt oder ignoriert werden
- Option: zu sendendes Signal (hier die wichtigsten)
  - `-9`, `-KILL`: Erzwingt Beenden des Prozesses (kann nicht abgefangen werden)
  - `-15`, `-TERM`: Fordert den Prozess zum Beenden auf (Standard-Signal)
- Argument: Prozess-ID oder -1 (alle Prozesse)
- `init` und Zombies können nicht beendet werden
- nur `root` darf Signale an Prozesse anderer Benutzer senden

killall

- `killall` sendet Signale an Prozesse
- Option: wie `kill`
- Argument: Name des Prozesses
- Beendet alle Prozesse des Namens!

free

- `free` zeigt an, wieviel RAM und Swap belegt bzw. frei ist
- nützliche Option: `-m` gibt Größen in Mebibyte aus
- Bedeutung der Ausgabe:
  - `total` zeigt die Gesamtgröße von RAM/Swap an
  - `used` zeigt an, wieviel davon verwendet wird
  - `free` zeigt an, wieviel davon frei ist
  - `buffers` / `cached` zeigt an, wieviel RAM für Festplattencache verwendet wird
  - die zweite Zeile enthält den verwendeten/freien RAM ohne Festplattencache

top

- `top` zeigt eine regelmäßig aktualisierte Liste aller Prozesse und allgemeiner Daten an
- erste Zeile:
  - Uhrzeit
  - Zeit seit dem Hochfahren (**uptime**)
  - angemeldete Benutzer (dazu zählen u. U. auch alle offenen Terminals)
  - Systemauslastung (**load average** der letzten Minute, der letzten fünf Minuten und der letzten fünfzehn Minuten)
  - die **load average** ist theoretisch nach oben offen; 0 bedeutet keine Auslastung; ein Wert, der der Anzahl der Prozessorkerne entspricht bedeutet Vollauslastung; bei höherer Last reagiert das System langsamer

top

- zweite Zeile: Anzahl der laufenden Prozesse, aufgeschlüsselt nach Zuständen
- dritte Zeile: Verwendung von CPU-Zeit, aufgeschlüsselt nach Prozess-Typen
  - `us` (User): Benutzer-Prozesse
  - `sy` (System): Kernel-Prozesse
  - `id` (Idle): Leerlauf
  - `wa` (Wait): Warten (vor allem auf Festplatten)
- vierte Zeile: Speicherauslastung, wie bei `free`

top

- restliche Ausgabe von `top` stellt sortierte Prozessliste dar
- Standardsortierung nach CPU-Belastung
- `M` schaltet auf Sortierung nach Speicherverbrauch um
- `P` schaltet wieder zurück
- `q` beendet `top`
- `VIRT` und `RES` entsprichen `VSZ` und `RSS` aus `ps`
- `S` entspricht `STAT` aus `ps`

htop

- `htop` ist eine moderne, übersichtlichere Alternative zu `top`

Protokolldateien

- Kernel und im Hintergrund laufende Dienste können Meldungen nicht direkt ausgeben
- solche Meldungen werden in Protokolldateien (log files) geschrieben
- nach FHS liegen Protokolldateien in `/var/log`
- können vertrauliche Informationen enthalten, daher in der Regel nicht für normale Benutzer lesbar

Syslog

- die meisten Dienste schreiben Meldungen nicht direkt in Protokolldateien, sondern senden sie an einen Syslog-Dienst (`syslogd`)
- schreibt die Meldungen dann in die passende Datei oder verschickt sie über das Netzwerk an einen zentralen Server
- unterstützt verschiedene Wichtigkeitsstufen für Meldungen (debug bis emergency)

Kernel-Meldungen

- Syslog speichert auch Kernel-Meldungen (bei manchen Distributionen über einen zusätzlichen `klogd`)
- da der Syslog-Dienst erst spät beim Hochfahren startet, können Kernel-Meldungen zusätzlich über den Befehl `dmesg` abgerufen werden

Spickzettel

Alle Befehle

Befehl	Optionen
`ps`	`f`, `a`, `u`, `x`
`kill`	`-9` / `-KILL`, `-15` / `-TERM`
`killall`	(wie `kill`)
`free`	`-m`
`top`	
`dmesg`	