Informatik
Hauptcampus

H O C H
S C H U L E
T R I E R

Systemadministration
Teil 7

Prof. Dr.-Ing. Jörn Schneider

Wiederholung

Klassifizierung Online Schedulingverfahren (1)

- Prioritätsgesteuert
  - Schedulingobjekten werden Prioritäten zugeordnet
  - z.B.:
    - Ticket System des r/ft
    - Persönlicher Zeitplaner

- Nicht prioritätsgesteuert
  - Es werden keine expliziten Prioritäten vergeben
  - z.B.:
    - Warteschlange beim Bäcker
    - Druckerwarteschlange

Klassifizierung Online Schedulingverfahren (2)

- Unterbrechbar/Preemptable
  - Bereits gestartete Vorgänge/Programme können unterbrochen werden

- Nicht Unterbrechbar/Non preemptable
  - Einmal gestartete Vorgänge/Programme können nicht unterbrochen werden

Klassifizierung Prioritätsgesteuerte Schedulingverfahren

- Dynamische Prioritäten/Dynamic Priority Scheduling
  - Prioritäten werden zur Laufzeit vergeben

- Feste Prioritäten/Fixed Priority Scheduling
  - Prioritäten werden zur Entwicklungszeit vergeben

First Come First Serve (FCFS) Scheduling

- Scheduling nach dem FIFO-Prinzip

Round-Robin Scheduling

- Jeder Task der zur Ausführung gelangt erhält ein Quantum an zugestandener Rechenzeit
- Hat er das Quantum aufgebraucht, wird er unterbrochen und ans Ende der Ready-Liste gehängt, dann wird der nächste Task in der Liste ausgeführt
- Beendet sich der Task vor Aufbrauchen des Quantums wird der Scheduler aufgerufen, der den nächsten Task auswählt

Prozessarten: Interaktiv vs. Batch-Prozess

- Interaktive Prozesse erlauben bzw. erwarten einen steuernden Eingriff durch den Benutzer
  - Shell
  - Editor in IDE
  - …
  - Textverarbeitung
  - Datenbankfrontend
- Batch Prozesse können vom Benutzer nach erfolgtem Start nicht mehr direkt gesteuert werden
  - sort
  - Druckerdaemon
  - Datenbankbackend
  - mkfs
  - Compiler
  - …

Foreground vs. background

- Hintergrundprozesse arbeiten ohne Verbindung zu Eingabemedien des Benutzers
  - Beispiel: Daemonen, Rechenintensive Programme

- Vordergrundprozesse haben eine Verbindung zu Eingabemedien des Benutzers
  - Beispiel: cat, vi, cfdisk, more
  - Beispiel Shell: Zu einem gegebenen Zeitpunkt kann je Shell immer nur ein Vordergrundprozess laufen

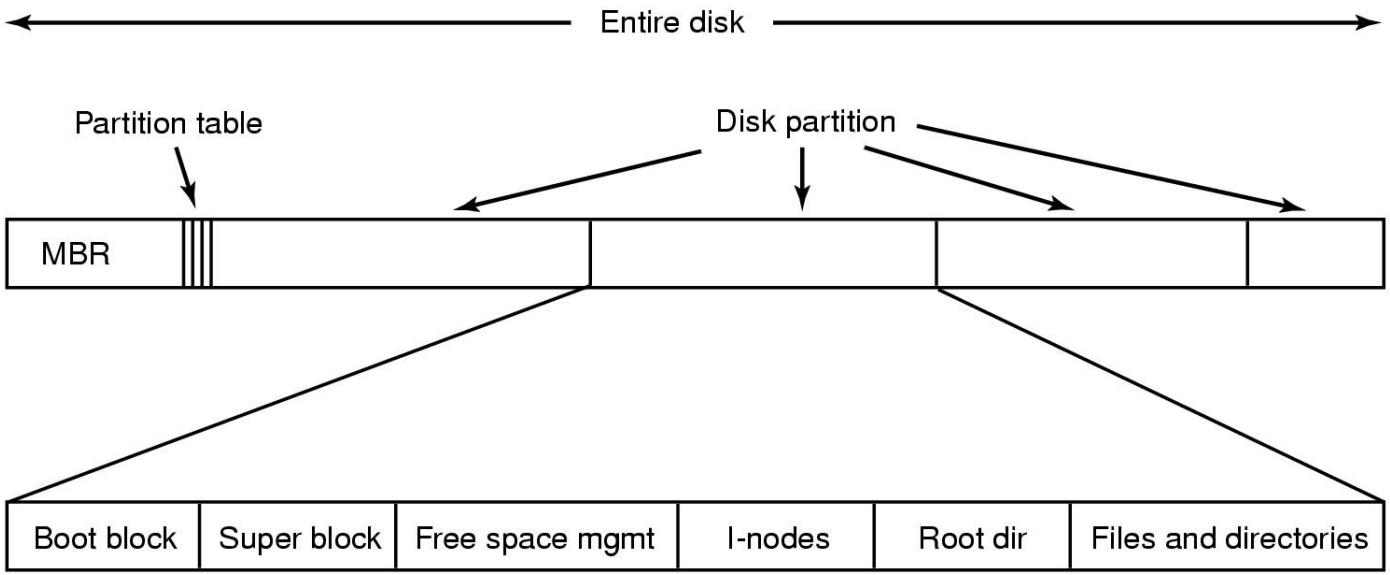
Wiederholung - Ende

Teil 7

- Was ist ein Rechnersystem?
- Was ist ein Betriebssystem?
- Aufgaben eines Systemadministrators
- Rechneraufbau
- Betriebssystemkonzepte
- Benutzer
- Prozesse und Threads (2)
- Bootvorgang
  - Booten
  - Initphase

Boot Vorgang in HW

File System Implementation



A possible file system layout

Boot-Vorgang

- Beim Einschalten: HW sorgt für Reset am Prozessor (hard Reset)
- Fetch von Startadresse (Typischerweise Adresse 0x0)
- Hier ist NV-RAM (früher ROM, heute Flash) eingeblendet von dem aus die Firmware ausgeführt wird
  - NV-RAM = Non-Volatile Random Access Memory
  - Bei PCs bezeichnet man die Firmware als BIOS
- Die Firmware lokalisiert den Datenträger von dem das zu startende Bootprogramm geladen werden soll
  - Weitere wichtige Schritte der Firmware
    - Durchführen HW-Test, POST (Power-on-self-test)
    - Konfiguration der HW

Boot-Vorgang (2)

- Die Firmware lädt das Bootprogramm aus dem MBR (Master Boot Record) in den Speicher und führt es aus
- Das Bootprogramm (aus dem MBR) identifiziert die aktive Partition und lädt von dort den Betriebssystemkernel oder die nächste Bootloader-Stufe
  - Die nächste Bootloader-Stufe kann insbesondere auch ein Bootmanager sein, bzw. vor dem Betriebssystemkernel einen solchen laden

Boot unter Linux

1. Evtl. Ausführung Bootmanager, z.B.
   - LILO
   - GRUB
   - Das U-Boot
2. Der Kernel wird geladen und ausgeführt
3. Kernel initialisiert die Hardware und lädt die notwendigen Treiber
   - autoconfigure, d.h. ansprechen aller möglicherweise vorhandenen Hardware und Registrierung der antwortenden Geräte
4. Prozess 0 wird gestartet
5. Prozess 0 mounted das root File System und erzeugt die Prozesse 1 (init) und 2 (page daemon)

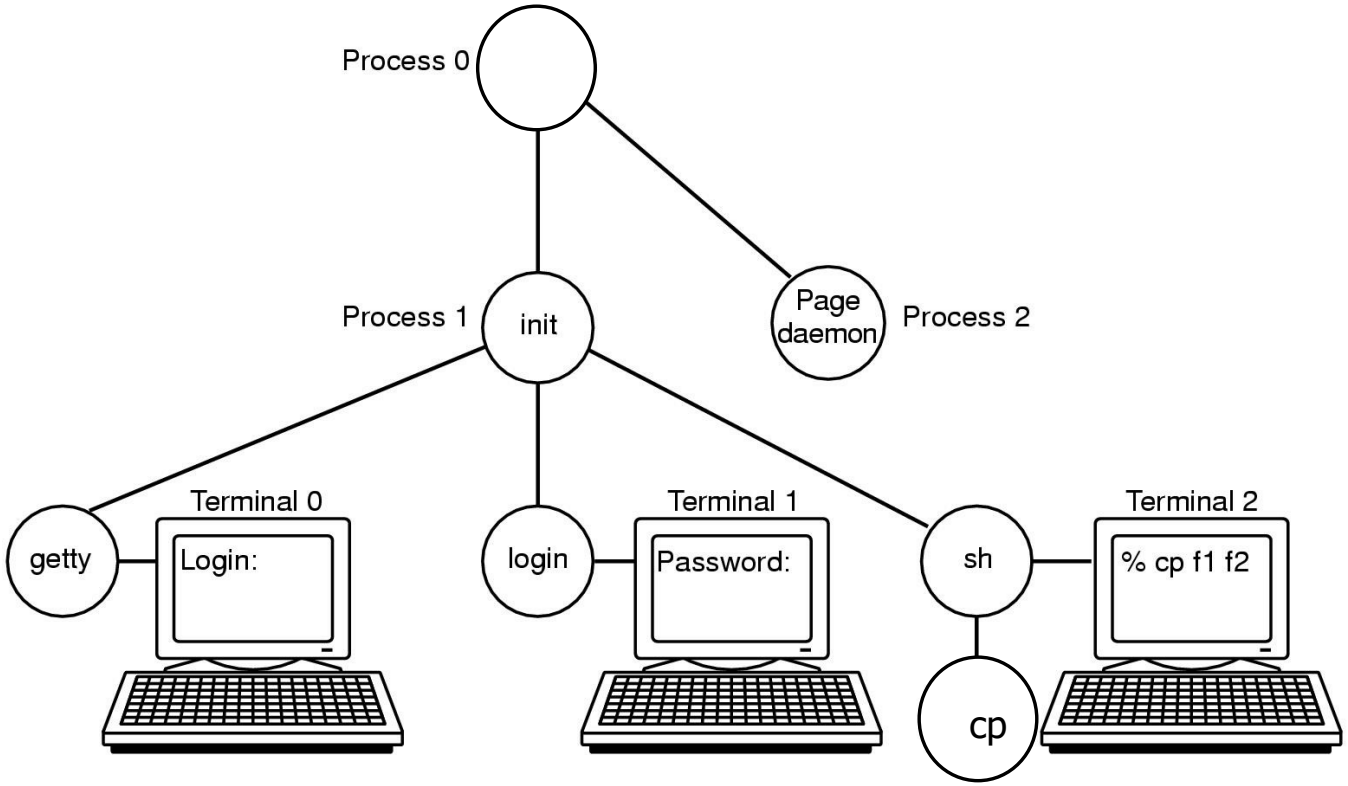
Boot unter Linux (2)

6. Der *Init* Prozess ist Stammelter aller weiteren Prozesse
7. Weitere Initialisierungen (siehe Initphase)
8. Normalbetrieb: Start von *getty* für jedes Terminal
   - TTY kommt von Teletype Writer (Fernschreiber)
   - *getty* konfiguriert Terminal, schreibt "login:" und wartet auf Eingabe

---

9. Bei Eingabe Benutzername terminiert getty durch Start von login
10. login fragt nach Passwort, verschlüsselt dieses und vergleicht mit Eintrag in /etc/shadow
11. Nach erfolgreichem Anmelden terminiert login durch Ausführung der Shell des Benutzers

Booting UNIX



The sequences of processes used to boot some systems

Teil 7

- Was ist ein Rechnersystem?
- Was ist ein Betriebssystem?
- Aufgaben eines Systemadministrators
- Rechneraufbau
- Betriebssystemkonzepte
- Benutzer
- Prozesse und Threads (2)
- Bootvorgang
  - Booten
  - Initphase

Boot unter Linux (2)

6. Der *Init* Prozess ist Stammelter aller weiteren Prozesse
7. **Weitere Initialisierungen (Initphase)**
8. Normalbetrieb: Start von getty für jedes Terminal
   - TTY kommt von Teletype Writer (Fernschreiber)
   - getty konfiguriert Terminal, schreibt "login:" und wartet auf Eingabe
9. Bei Eingabe Benutzername terminiert getty durch Start von login
10. login fragt nach Passwort, verschlüsselt dieses und vergleicht mit Eintrag in /etc/shadow
11. Nach erfolgreichem Anmelden terminiert login durch Ausführung der Shell des Benutzers

Was passiert in Initphase?

- Überprüfen des Systems
  - Filesystem Check
- Einrichten der benötigten Umgebung
  - Mount der benötigten Filesysteme
  - Netzwerkanbindung
- Starten der jeweils benötigten Dienste (Dämonen) in der korrekten Reihenfolge
  - Druckerdienst (Printer Dämon)
  - Mailserver

Wie wird die korrekte Abfolge eingehalten?

- Es werden verschiedene sogenannte Runlevel durchlaufen
- Für den Normalfall ist die Abfolge der Runlevel beim Bootvorgang per Konfiguration festgelegt.
- Im Betrieb kann der Systemadministrator in andere Runlevel wechseln
- Auch das Herunterfahren des Systems wird über spezielle zu durchlaufende Runlevel realisiert

Initphase

- Unterscheidung zwischen Single User und Normalbetrieb
- Single User: Shell starten für root
- Multi User: Ausführung Initalisierungs Shell Skripte gemäß runlevel
  - File system checks
  - Mounten Dateisysteme
  - Starten von Daemon Prozessen

Single User Modus

- Single User Modus wird verwendet für reine Wartung, ohne dass Netzwerkdienste gestartet sind oder Benutzer sich anmelden können, z.B.:
  - Filesystem auf andere Partition umziehen
  - Kernel-Update
  - …

Was passiert bei Wechsel von Runlevel

- 1. Dienste (Daemon) stoppen, die in diesem Runlevel nicht laufen sollen
- 2. Dienste starten, die in diesem Runlevel laufen sollen

Runlevel Realisierung (System V UNIX)

- init führt zentrales Startup Skript aus (z.B. /etc/rc)
- /etc/inittab wird gelesen, um default runlevel zu ermitteln
- Ausführung der Shellskripte in /etc/rcS.d (“S“ steht hier für Startup), und dann in /etc/rc2.d (unter der Annahme, dass 2 der default Runlevel ist)
  - 1. Ausführung der Knn* Skripte (beenden der nicht benötigten Dienste)
    - n = Ziffer zwischen 0 und 9
  - 2. Ausführung der Snn* Skripte (starten der benötigten Dienste)
  - Reihenfolge der jeweiligen Skripte: Skript mit kleinstem nn zuerst

Realisierung Runlevel Skripte

- rc-Skripte (in den Verzeichnissen /etc/rc?.d) sind in Wirklichkeit symbolische Links auf Skripte im Verzeichnis /etc/init.d
- Dort gibt es für jeden Dienst **ein** Konfigurationsskript
- Link beginnt mit K → Aufruf mit Parameter „ stop“
- Link beginnt mit S → Aufruf mit Parameter „ start“

Fingerübung: Runlevel

- Im Verzeichnis /etc/rcS.d existieren folgende Einträge:
  S01Bdienst, S14Adienst
- Im Verzeichnis /etc/rc1.d stehen folgende Einträge:
  S01Cdienst, K99Ddienst
- Im Verzeichnis /etc/rc2.d stehen folgende Einträge:
  S01Ddienst

- Das System läuft im default runlevel (2).
- Welche Skripte werden in welcher Reihenfolge gestartet, wenn der Systemadministrator mit dem Kommando "init 1" in den Single User Modus schaltet und welche Parameter werden dabei jeweils übergeben?