

# Systemadministration

## Teil 9

Prof. Dr.-Ing. Jörn Schneider

# Wiederholung

# Runlevel

- S: Start (Single User)
- 0: Halt
- 1: Single User („sauber“)
- 2: Default Multi User, d.h. Netzwerkdienste gestartet
- 3-5: Weitere Multi User Runlevel
- 6: Reboot

# cron Daemon

- Aufgabe - regelmäßige Ausführung von Aktivitäten (cron Jobs) in festen Zeitintervallen
  - z.B.
    - Jede Minute prüfen, ob Mails eingegangen sind
    - tägliches Backup durchführen (inkrementell)
    - wöchentliches Backup durchführen (vollständig)

# System cron Jobs

/etc/crontab

- Tabelle mit System cron Jobs

Format:

- <Zeit> <Kommando>
- Zeit=  
<Min> <Stunden> <Tag des Monats> <Monat> <Tag der Woche>
- Kommando=  
<ausführbare Datei> [<Parameter>] [%<Text für stdin>]

Beispiele

- 0,30 \* \* \* \* write notroot %,,Wieder eine 1/2 Stunde rum,,
- 30 10 \* \* 1 write notroot %,,Manic Monday,,

# User cron Jobs

- Kommando crontab
  - Anzeigen
    - `crontab -l`
  - Einrichten/Ändern/Löschen
    - `VISUAL=`which vi`; export VISUAL`
    - `crontab -e`
- Files sind abgelegt in z.B. `/var/spool/cron/crontabs/`

# Ende Wiederholung

# Fingerübung Cron Jobs

- `0,30 * * * * write notroot %,,Wieder eine 1/2 Stunde rum,,`
- `30 10 * * 1 write notroot %,,Manic Monday,,`

Ändere Beispiel so ab, dass sie an den Vorlesungstermin für Systemadministration, erinnern und die täglichen Vorlesungspausen



# AT DAEMON

# at Daemon

- Aufgabe - Einmalige Ausführung von Aktivitäten (at Jobs) zu festen Zeitpunkten
  - z.B.
    - In Mittagspause rechenintensiven Job starten
    - Um 19:00 Uhr Nachricht an User „bitte abmelden“
    - Um 20:00 Uhr System zur Wartung runterfahren
    - Nächsten Montag Erinnerung an SysAdmin Vorlesung

# Verwaltung von at Jobs

Einrichten

- at

Anzeigen

- atq

Löschen

- atrm

- Files sind abgelegt in z.B. /var/spool/cron/atjobs/

# Start des at Daemons

- In allen Multiuser Runlevel
- z.B. `/etc/rc2.d/S89atd`
  - bewirkt Ausführung von `"/etc/rc2.d/S89atd start"` bei Wechsel in Runlevel 2
    - `S89atd` ist symbolischer link auf `/etc/init.d/atd`

# SYSLOG DAEMON

# syslogd

- Aufgabe - Protokollieren von Systemmeldungen
  - z.B.
    - Ergebnis von File System Überprüfungen in Log Datei schreiben
    - Datum und Uhrzeit von reboot vermerken
    - Fehlgeschlagene Login-Versuche dokumentieren
    - Kritische Fehlermeldungen von Dämonen auf root Konsole schreiben

# Konfiguration des syslogd

/etc/syslog.conf

- Format

- <Quelle>.<Priorität> <Ziel>
- Beispiel:
  - mail.alert /var/log/mail.log
  - auth.crit /var/log/auth.log

# Quellen

- Kernel
- User
- Mail
- Daemon
- Authorization
- Line Printer
- ...



# Prioritäten

- 0 Emergency
- 1 Alert
- 2 Critical
- 3 Error
- 4 Warning
- 5 Notice
- 6 Informational
- 7 Debug

# Ziele

- Lokale Logdateien
- Terminals von Benutzern (z.B. root)
- Logserver (über das Netzwerk)

# Senden von Meldungen an syslogd

Kommando logger

- sendet Meldungen an den syslogd
- Beispiel:
  - `echo "Var x an Stelle s ist 10" | logger -p user.debug`

# Inhalt

- Was ist ein Rechnersystem?
- Was ist ein Betriebssystem?
- Aufgaben eines Systemadministrators
- Rechneraufbau
- Betriebssystemkonzepte
- Benutzer
- Prozesse und Threads
- Dienste und Bootvorgang (Teil 3) – Syslogd
- **Benutzerverwaltung unter UNIX**

# BENUTZUMGEBUNG

# Wiederholung – Ablauf Benutzeranmeldung

# Boot unter UNIX(2)

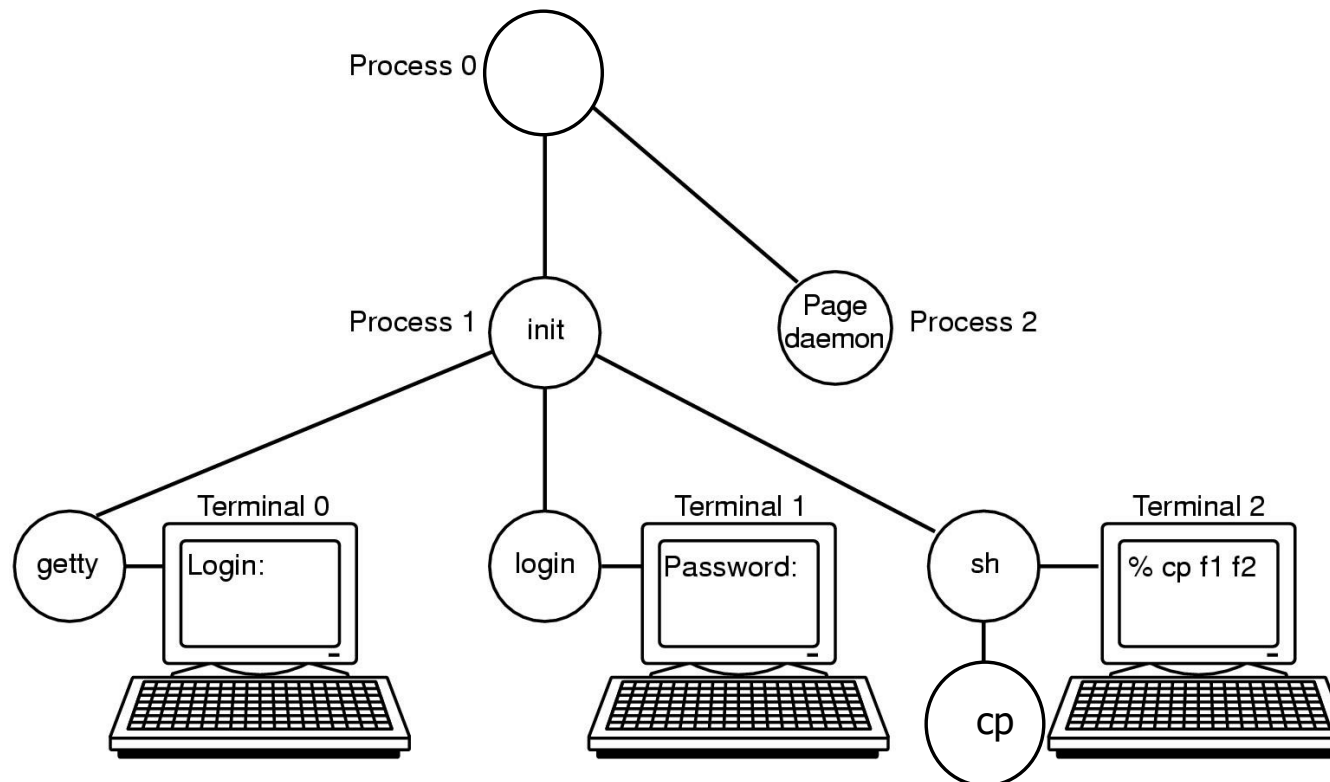
- Der Init Prozess ist Stammvater aller weiteren Prozesse
- Weitere Initialisierungen (siehe Initphase)
- Normalbetrieb: Start von getty für jedes Terminal
  - TTY kommt von Teletype Writer (Fernschreiber)
  - getty konfiguriert Terminal, schreibt "login:" und wartet auf Eingabe
- Bei Eingabe Benutzername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

# Beispiel UNIX (I)

- Beim anmelden, suche nach User (z.B. notroot) in
  - /etc/passwd
- Username gefunden ➔
  - Verschlüsseln eingegebenes Passwort
  - Vergleich mit abgelegtem Passwort
- Vergleich OK ➔
  - Starte Shell



# Booting UNIX



Bildquelle: Zusatzmaterial "Modern Operating Systems", 2<sup>nd</sup> ed. Andrew S. Tanenbaum, <http://www.cs.vu.nl/~ast/books/>

## Prozessabfolge bei einigen UNIX Systemen

## Beispiel UNIX (II)

- `/etc/passwd`
- Jede Zeile ein User, mit Einträgen:
  - Benutzername
  - Verschlüsseltes Passwort (oder ,x‘)
  - UID (User ID)
  - GID (ID der primären Gruppe des Users)
  - Kommentarfeld (Name des Benutzers)
  - Home-Verzeichnis
  - Shell die der User verwendet
- Bsp.:  
`hugo:x:1047:1000:Hugo Müller:/home/hugo:/bin/bash`

## Beispiel UNIX (III)

- `/etc/shadow`
- Enthält verschlüsselte Passwörter anstelle von `/etc/passwd`
- Steuert Passwort Aging

# Beispiel UNIX (IV)

- Bei erfolgreicher Anmeldung:
  - Eintrag in utmp file (Ubuntu Linux: /var/log/utmp)
    - Anzeige über `who`
  - Setzen der Umgebungsvariablen
  - Wechsel in Home-Verzeichnis
  - Ausführung der Login Skripte in aktueller Prozessumgebung, z.B.:
    - `.profile`
    - `.bashrc`

# Ende der Wiederholung

# Benutzerspezifische Umgebung

- Shell (z.B.: /bin/sh, /bin/csh, /bin/ksh, /bin/tcsh, /bin/bash, ...)
- Home Verzeichnis
- Umgebungsvariablen
- Aliase

# Umgebungsvariablen

## Environment

- Menge von Shellvariablen, die samt ihren Werten an Kindprozesse vererbt werden
- Achtung:
  - Nicht jede Variable ist eine Umgebungsvariable
  - Sonstige Variablen werden nicht an Kindprozesse vererbt
- Das System definiert gewisse Umgebungsvariablen vor
  - SHELL
  - HOME
  - ...
- Der Benutzer kann Umgebungsvariablen definieren

# Wichtige Umgebungsvariablen

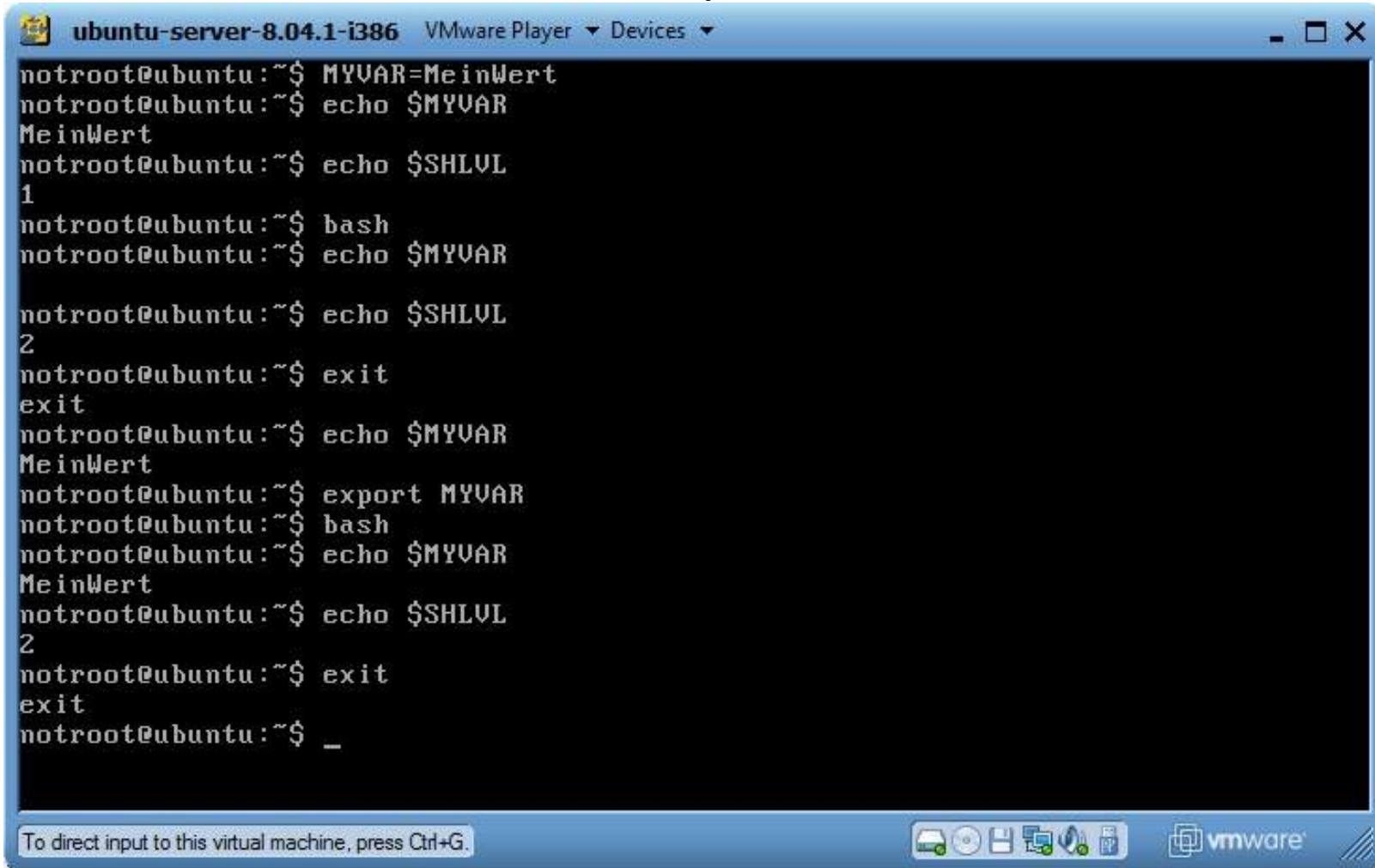
- SHELL
  - Pfad der verwendeten Shell
- HOME
  - Heimatverzeichnis
- LOGNAME
  - Benutzername
- PATH
  - Liste der Pfade in der die Shell nach Kommandos sucht
- TERM
  - Typ des verwendeten Terminals
- VISUAL
  - Zu verwendender Editor bei Aufruf über andere Programme, z.B. `crontab -e`



# Umgebungsvariablen definieren

- `MYVAR=xyz`
  - Weist der Variable MYVAR den Wert „xyz“ zu
- `export MYVAR`
  - Nimmt die Variable MYVAR in die Liste der Umgebungsvariablen auf
- Beispiel siehe nächste Folie

# Beispiel



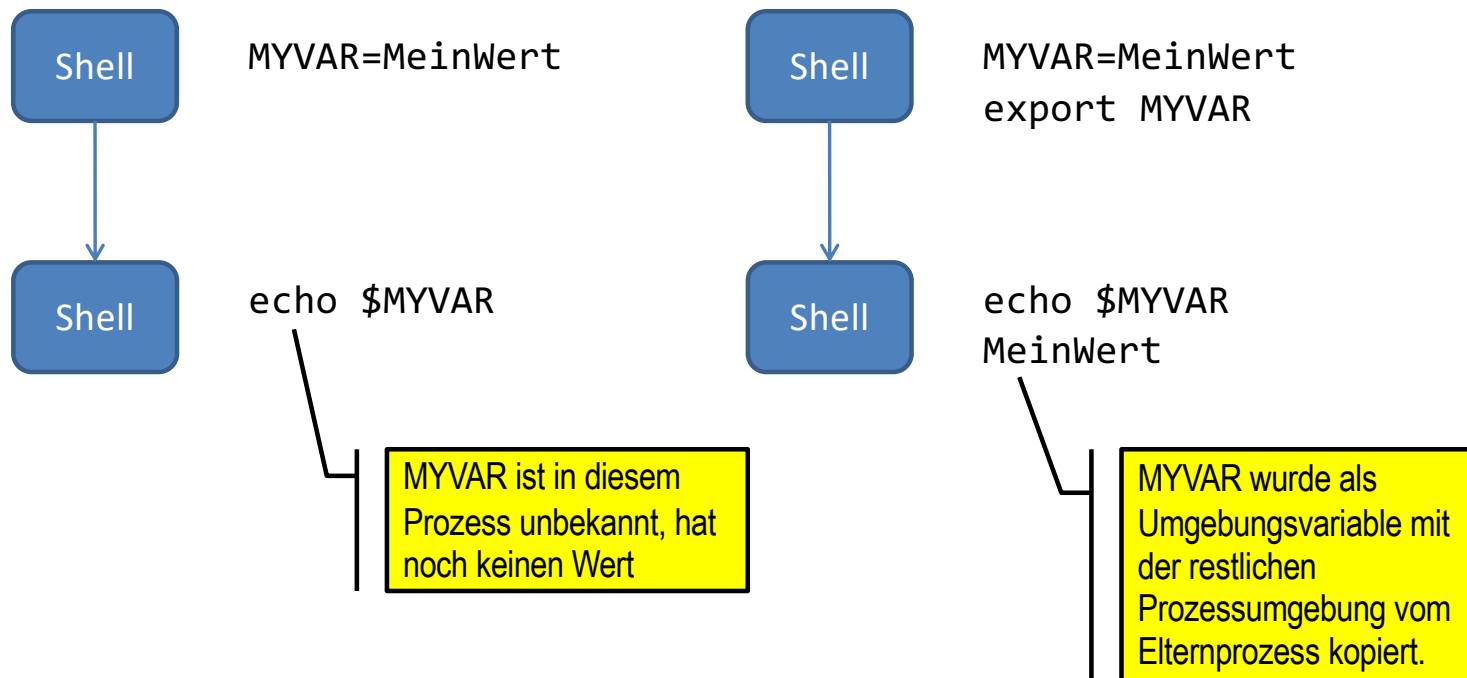
The screenshot shows a VMware Player window titled "ubuntu-server-8.04.1-i386". Inside the window is a terminal window with the following commands and output:

```
notroot@ubuntu:~$ MYVAR=MeinWert
notroot@ubuntu:~$ echo $MYVAR
MeinWert
notroot@ubuntu:~$ echo $SHLVL
1
notroot@ubuntu:~$ bash
notroot@ubuntu:~$ echo $MYVAR

notroot@ubuntu:~$ echo $SHLVL
2
notroot@ubuntu:~$ exit
exit
notroot@ubuntu:~$ echo $MYVAR
MeinWert
notroot@ubuntu:~$ export MYVAR
notroot@ubuntu:~$ bash
notroot@ubuntu:~$ echo $MYVAR
MeinWert
notroot@ubuntu:~$ echo $SHLVL
2
notroot@ubuntu:~$ exit
exit
notroot@ubuntu:~$ _
```

At the bottom of the terminal window, there is a status bar that says "To direct input to this virtual machine, press Ctrl+G." and a VMware logo.

## Beispiel Forts.



# Wiederholung

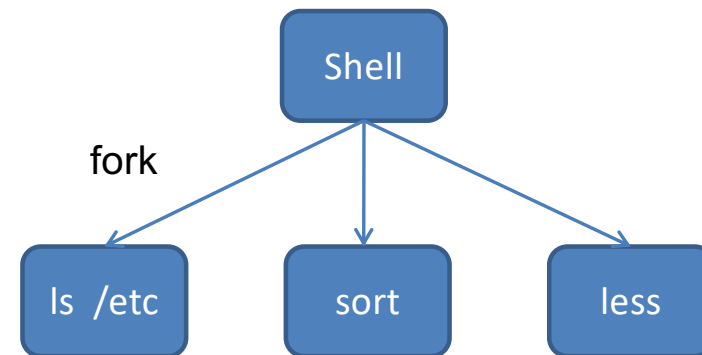
# UNIX: fork & exec

## fork

- Erzeugt Kindprozess (Child) mit gleichem Umfeld:
  - Programmcode
  - Speicherimage (Kopie)
  - **Umgebungsvariablen (Kopie)**
  - Offene Dateien

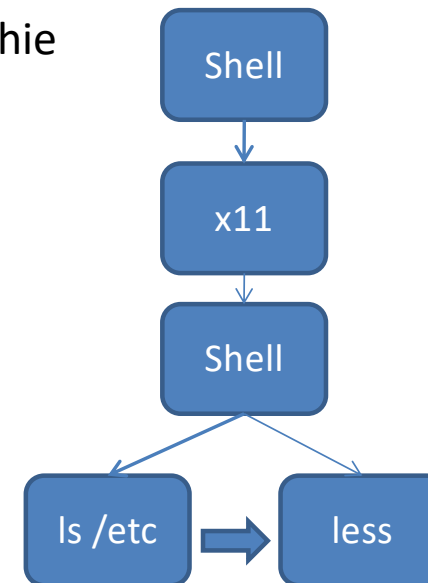
## exec

- Führt neues Programm anstelle des bisherigen aus



# Prozesshierarchie

- Elternprozess erzeugt Kindprozess, Kindprozess kann eigene Kindprozesse erzeugen, etc.
- Es ergibt sich eine Hierarchie von Prozessen
  - Unter UNIX redet man hier von Prozessgruppen (process group)
- Windows hat kein Konzept zur Prozesshierarchie
  - Alle Prozesse werden gleich erzeugt



# Ende der Wiederholung

# Wie Umgebungsvariable in Shellskript setzen?

- Bei Ausführung eines Shellskripts wird eigener Kindprozess erzeugt
- Funktioniert das Setzen einer Umgebungsvariablen auch so, dass der Elternprozess den neuen Wert übernimmt?



# Wie Umgebungsvariable in Shellskript setzen?

Trick, Ausführung des Shellskripts in derselben Umgebung mit:

- `source myscript.sh`

oder

- `. myscript.sh`

# Alias-Mechanismus

# Alias-Mechanismus

## Textuelle Ersetzungen bei Shellkommandos

- Erstes Wort des Kommandos wird überprüft, bei Übereinstimmung mit Aliasnamen wird es ersetzt
- Beispiel:  
`alias ll='ls -l'`

# Benutzerspezifische Startupskripte

- Beim Start der Shell werden die von der jeweiligen Shell unterstützten Startupskripte im Heimatverzeichnis des Benutzers gestartet
- Beispiele:
  - .profile
  - .bashrc
  - .alias