Vorbereitungskurs Informatik - Tag 1

FSI Informatik

Uni Erlangen-Nürnberg

10. Oktober 2006

Wer sind wir?

- Fachschaftsinitiative (kurz FSI) Informatik
- Was machen wir?
  - Vertretung der studentischen Interessen in Gremien
  - Erstsemestereinführung
  - Bereitstellen von Prüfungsfragen und weiteren Infos
  - Genereller Ansprechpartner für Studenten
  - Sommerfest
  - Was noch so anfällt...
- Wie erreicht man uns?
  - online: `http://fsi.informatik.uni-erlangen.de`
  - per E-Mail `fsi@informatik.uni-erlangen.de`
  - oder einfach im 2. Stock im Blauen Hochhaus (direkt neben dem CIP) vorbeischauen, Zimmer 02.150

Wie schauts im CIP aus?

- CIP-Pools im 1. und 2. Stock des Blauen Hochhauses
- Linux-Arbeitsrechner
- Sunrays
- Drucker
- im RRZE:
  - Scanner
  - Farbdrucker
  - Posterdrucker

Linux - Was ist das?

- eigentlich nur der Kern eines Betriebssystems
- meistens meint man damit eine Zusammenstellung von:
  - Betriebssystem
  - Arbeitsprogrammen
  - Spielen
  - etc.
- diese „Distributionen“ haben eigene Namen und Versionsnummern, z.B.:
  - SuSE
  - Fedora
  - Ubuntu
  - Debian (hier im CIP installiert)

Arbeitsumgebung

- Window-Manager
- Shell
- weitere Software:
  - Editoren
  - Browser
  - etc.
- diese Komponenten kann man nach eigenen Bedürfnissen selbst zusammenstellen

Window-Manager

- bestimmt Aussehen und Verhalten der graphischen Oberfläche
- es existiert ein breites Spektrum:
  - geringer Ressourcenverbrauch
  - minimalistisch
  - nur per Tastatur steuerbar
  - sehr individuell konfigurierbar
  - verspielt
  - Rundum-Paket
  - Grenzen oft fließend

Konfiguration im CIP

Im CIP-Pool kannst du mit dem Befehl `envcfg` deinen Window-Manager auswählen.

KDE

- gut geeignet für den Einstieg
- viele Menüpunkte
- Konqueror
- Kontrollzentrum zur Konfiguration
- virtuelle Desktops
- bei vielen Distributionen der Standard

Viewer

Es gibt eine Vielzahl von Programmen für Linux um die unterschiedlichsten Dateiformate zu betrachten, z.B.:

- `gedit, scite, kate, gvim, ...` für Textdateien
- `gv, ggv` für Postscript-Dateien
- `acroread, xpdf, kpdf` für PDF-Dateien
- `gqview` für alle möglichen Bilddatein
- `mplayer, xine, vlc` für Videos

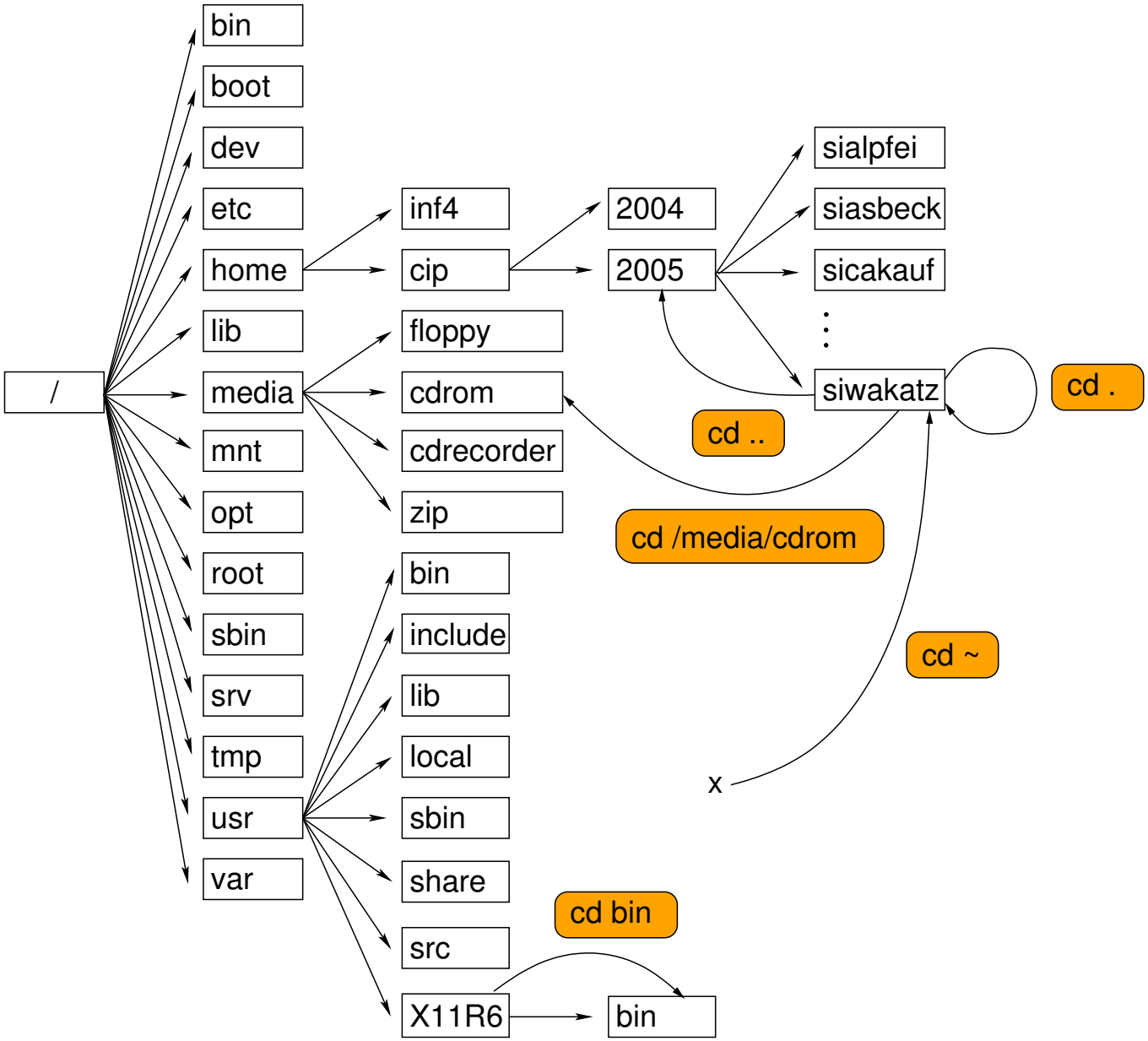
Tipp

Hängst du an das Kommando ein `&` hinten an, so wird das Programm im Hintergrund gestartet und du kannst auf der Shell weiterarbeiten.

Shell

- auf einer Shell kann man Befehle per Tastatur eingeben und ausführen
- unterschiedliche Shells (wie unterschiedliche Window Manager), die gängisten sind
  - tcsh (Standard im CIP-Pool)
  - **bash** (wird von uns empfohlen)
  - zsh
- Grundfunktionalität gleich
- jede Shell bietet gewisse Komfortfunktionen, z.B.:
  - Tab-Completion
  - Befehlshistory
  - Befehlsaliase
  - Pipes
  - Scripting

Verzeichnisbaum



Home und ciptmp

- jeder User besitzt ein Home-Verzeichnis (/home/cip/2006/userlogin):
  - es steht nur begrenzter Speicherplatz zur Verfügung
  - dort liegen Konfigurationen und Nutzdaten
  - der Inhalt wird täglich gesichert
- mehr Speicherplatz unter /proj/ciptmp/userlogin verfügbar:
  - insgesamt 2 GB
  - wird nicht gesichert!

Standard Unix-Kommandos

- grundlegend für die Arbeit mit dem Dateisystem sind:
  - ls: zeigt Dateien an
  - cd: Verzeichniswechsel
  - mkdir: legt Verzeichnis an
  - cp: kopiert Dateien und Verzeichnisse
  - mv: verschiebt Dateien und Verzeichnisse
  - rm: löscht Dateien und Verzeichnisse
- kleine nützliche Tools
  - grep: sucht im Inhalt von Dateien
  - cat: gibt Textdateien aus
  - less: Anzeige von Text-Dateien mit Scroll-Funktion

ls, pwd – Verzeichnisinhalte und Arbeitsverzeichnis

ls

- `ls` listet die Dateien im aktuellen Verzeichnis auf
- `ls verzeichnisname` zeigt den Inhalt des angegebenen Verzeichnisses an
- `ls -l` ausführliches Verzeichnislisting mit Dateigrößen, Rechten, Zeitstempel etc.
- `ls -a` listet auch versteckte Dateien (`.file`) auf

Optionen wie `-l` und `-a` können auch kombiniert werden: `ls -la`

pwd

`pwd` gibt den Pfad des aktuellen Verzeichnisses aus

cd – Verzeichniswechsel

cd

Mit `cd` wechselt man zwischen Verzeichnissen.

Beispiele:

`cd bin`	wechselt in das Unterverzeichnis 'bin' im aktuellen Verzeichnis (*relativer Pfadwechsel*)
`cd /bin`	geht in das Verzeichnis 'bin' unterhalb vom Root-Verzeichnis (/) (*absoluter Pfadwechsel*)
`cd ..`	wechselt eine Verzeichnisebene nach oben
`cd`	geht von allen Verzeichnis in das Home-Verzeichnis
`cd ../sicakauf`	wechselt eine Verzeichnisebene nach oben **und** darin in das Verzeichnis 'sicakauf'

mkdir, rmdir – Verzeichnisse erstellen und entfernen

mkdir

`mkdir foo`	`mkdir foo` legt ein Verzeichnis 'foo' im aktuellen Verzeichnis an

rmdir

`rmdir foo`	löscht das leere Verzeichnis 'foo' im aktuellen Verzeichnis

cp – Kopieren

cp

`cp` kopiert Dateien

Aufbau:	*cp Quelle Ziel*	
Beispiel:	`cp bsp bspkopie`	kopiert die Datei 'bsp' nach 'bspkopie' (im aktuellen Verzeichnis)
	`cp bsp test/`	kopiert die Datei 'bsp' in das Verzeichnis 'test'
	`cp -r test/ test2`	erstellt eine Kopie des Verzeichnisses 'test' mit dem Namen 'test2'

mv – Verschieben

mv

`mv` verschiebt Dateien oder benennt sie um

Aufbau:	*mv Quelle Ziel*	
Beispiel:	`mv alt neu`	benennt die Datei 'alt' in 'neu' um (geht auch für Verzeichnisse)
	`mv foo dinge/`	verschiebt die Datei 'foo' aus dem aktuellen Verzeichnis in das Verzeichnis 'dinge'

rm – Löschen

rm

`rm` löscht Dateien und Verzeichnisse

Beispiel:	`rm foo.pdf`	löscht die Datei 'foo.pdf' im aktuellen Verzeichnis
	`rm -r Mails/`	löscht das Verzeichnis 'Mails' und alle darin enthaltenen Dateien und Unterverzeichnisse

rm – Löschen

rm

`rm` löscht Dateien und Verzeichnisse

Beispiel:	`rm foo.pdf`	löscht die Datei 'foo.pdf' im aktuellen Verzeichnis
	`rm -r Mails/`	löscht das Verzeichnis 'Mails' und alle darin enthaltenen Dateien und Unterverzeichnisse

rm – Löschen

rm

`rm` löscht Dateien und Verzeichnisse

Beispiel:	`rm foo.pdf`	löscht die Datei 'foo.pdf' im aktuellen Verzeichnis
	`rm -r Mails/`	löscht das Verzeichnis 'Mails' und alle darin enthaltenen Dateien und Unterverzeichnisse

Achtung!

rm löscht ohne Nachfrage

Tab-Completion

- moderne Shells nehmen einem viel Tipparbeit ab, indem sie Namen von Befehlen, Dateien und Verzeichnissen ergänzen
- hierzu tippt man den Anfang des Namens und dann <TAB>
- wenn nicht eindeutig, Liste von Alternativen mit <TAB> <TAB>

Beispiel

```
faui00a:~$ ls
Desktop    Mail    mathe1_klausur_ws0506.pdf
faui00a:~$ xpdf ma<TAB>
faui00a:~$ xpdf mathe1_klausur_ws0506.pdf
```

History

- die Shell merkt sich die zuletzt eingegebenen Befehle
- aufrufen der letzten Befehle mit den Cursortasten hoch/runter
- Ausgabe der letzten Befehle mit `history`

Kombinationen von Kommandos

- viele Shell-Befehle arbeiten mit Standard-Ein- und Ausgabe
- oft ist die Ausgabe eines Programmes gleichzeitig wieder Eingabe für ein weiteres
- gebräuchlich sind:
  - $<$, um die Eingabe aus einer Datei lesen zu lassen
  - | (sprich: Pipe), um die Ausgabe des einen Programms als Eingabe des anderen zu verwenden
  - $>$, um die Ausgabe in eine Datei zu schreiben
- Beispiel: Suche alle Zeilen, in denen „backup“ vorkommt in der Datei /proj/ciptmp/README
  - cat /proj/ciptmp/README | grep backup
  - grep backup $<$ /proj/ciptmp/README