

Literature for Linux

- On the Internet
- As part of the installation: man, info
- O'Reilly Publishers:
 - Siever, Figgins, Love, Robbins:
LINUX in a Nutshell 6th Edition
- Addison-Wesley:
 - Helmut Herold: Linux/Unix Grundlagen, Kommandos
und Konzepte
- Much more books and online resources

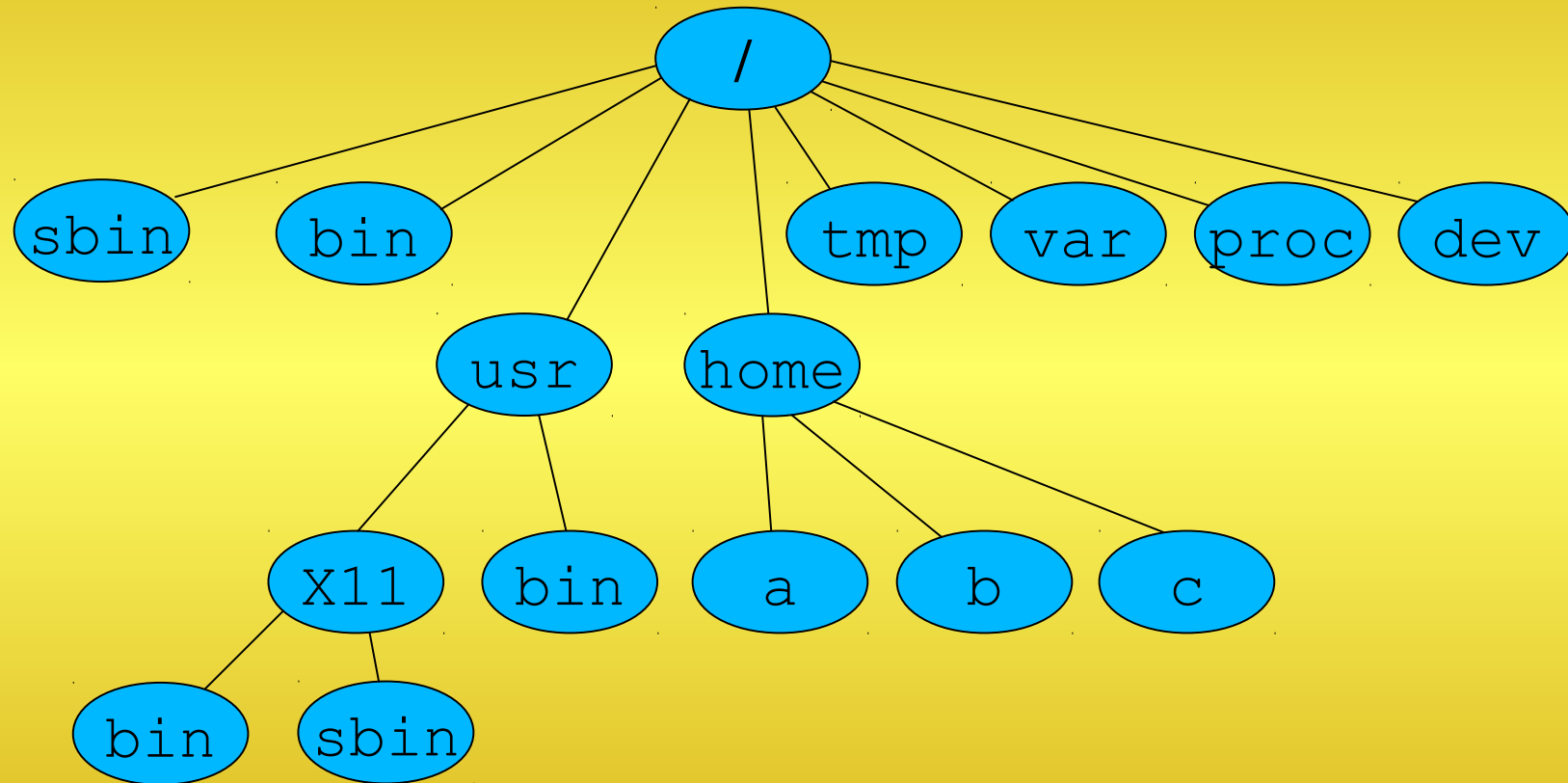
Structure of a Linux System

- Linux:
 - Actually name of the kernel
 - Unix-like operating system
 - Developed by Linus Torvalds
 - Subjected to GPL-License (open source)
- Multitasking / Multi-User System
- Available for many different types of CPUs
- specialized Versions for embedded systems, real-time systems, etc.
- Completed by open-source-tools (GNU).

Structure of a Linux System 2

- Hierarchical structured system, i.e.
- Each system component (HW or SW) is represented as a file
- Each component is uniquely identified by its path
- Absolute path: complete path starting with the root directory, always starts with „/“
- Relative Path: starts from the current directory
- . Current directory,
.. Parent directory (one level up)
- File types: regular file, directory, block – a. character-oriented device, sockets, named pipes

File System - Overview



<ftp://tsx-11.mit.edu/pub/linux/docs/linux-standards/fsstnd/>

Command Overview

- File System:
cd, ls, mkdir, rmdir, ln, touch, mv, pwd, rm, cp, chmod, chown, chgrp
- Files:
cat, less, head, tail, wc, strings, cut, paste, join, uniq, file
- Processes
ps, top, kill, killall
- Search
grep, find, apropos, locate

Command Overview 2

- Archiving
(g)zip, tar
- System status
df, du, env, date, who, stty, mount
- Documentation
man, info
- User administration
passwd, id

Log into the System

- Get yourself access to the system (Login):
 - Enter your Login name (user name issued by the systems administrator)
 - Enter your password (invisible)
- Change your password with the command **passwd**:
passwd *Username*
old password
new password (2x)

Informations about Users

- **Id** *login*
- gives informationen about user name, user id, primary and additional group id's and group names
- information about currently logged-in users:
command **who**

Online Documentation

- **man**: manual
 - original Unix help system
 - Manpages are organized in 9 sections
 - indexed in its own database
 - Restriction to certain sections or with keyword possible
- **apropos**: searches to index with a keyword, yields all occurrences in document titles

Online Documentation II

- **info**: GNU documentation tool
 - hyper text system
 - handling needs getting used to, but very powerful
 - indexed in own database
 - actual information source for the big software packages provided by GNU like GNU compiler, Latex, etc

File System 1

- get the current directory

pwd: print working directory

- change to another directory

cd: change directory

cd changes to your home directory

cd . . changes one level toward root

cd *Path_to_Dest* changes to directory *Dest*

File System 2

- List the content of a directory:
ls: list directory
- important options:
 - a all files (included hidden files)
 - l with comprehensive information
 - d do not list directories
 - t sort according to date of last modification
 - r reverse sorting
 - h „human readable“, size in Kilo-Mega byte
 - S sorts according to file size

File System 3

- Create a directory:
mkdir: make directory
mkdir *directoryName*
- Delete a directory:
rmdir: remove directory
rmdir *directoryName*
works only if the directory is empty

File System 4

- Copying files:
cp: copy files
cp *fromFile toFilei*
cp *fromFile toDirectory*
- options:
 - p keeps all informations, permissions, etc.
 - r -R recursive, if you want to cpy whole directories

File System 5

- moving resp. renaming of files:
mv: move files
mv *fromFile toFile*
mv *fromFile toDirectory*
mv *fromDirectory toDirectory*
- *Warning:*
 - existing files are overwritten
 - existing directories collect moved directories

File System 6

- Deleting Files:
rm: remove files
rm *file*
rm *-r directory*
- *Options*:
 - r recursive deletion of directories
 - f suppress request for confirmation when deleting write-protected files
 - i interactive, user confirms deletion
- **WARNING**: Deletion is final and irrevocable,
Use **rm -rf** only with extreme caution

File System 7

- Creation of links:
ln: link files (alias, nic name)
ln *(path/)file path/(LinkName)*
- Options:
 - s creates a symbolic link, i.e. an own link file
 - f suppress request for confirmation if existing files are overwritten
- symbolic links are safer in respect to undesired deletion of the target file

Files and Permissions

- Each user has an ID and is member of at least one group
- Permissions can be granted on owner (user), group or others.
- The three groups are represented by u (owner), g (group) and o (other) as well as a (all).

Files and Permissions 2

- the following permission can be granted:
r read read permission
w write write permission
x execute execute permission
s (set user/group ID), t(sticky bit) are not covered here, see man page
- grant a permission with “+”, revoke with “-”
grant exactly with “=”
- specify by letters or octal numbers

Files and Permissions 3

- examples:
ls -l lists these permissions:
-rw-r--r-- i.e. the owner has write and read permissions for this file, the remainder only read permissions
octal 644
- grant write permission to the group:
chmod g+w *filei* resp. **chmod** 664 *file*
- grant execution permission to all:
chmod a+x *file* bzw. **chmod** 775 *file*

Files and Permissions 4

- execution permission on directories allows to change to that directory
- read permission is not required to change into
- deleting a write protected file is possible if you have write permission to the directory!
- Options:
 - R recursive change of permission in the file system tree

Files and Permissions 5

- assignment (hand over) of files:
chown: change owner
chown *user.group fileName*
chgrp *group fileName*
- Options:
 - R recursive assignment in the file system tree

Files

- Displaying of text files:
cat: catenate
cat *fileName*
The file is displayed in one piece to standard out.
- Comfortably reading text file:
less *fileName*
Allows paging and searching for key words
- Determine the format (file type) of an unknown file:
file *fileName*

File Editing

- Extracting strings from binary files:
strings *fileName*
- Output the first n lines of a file:
head *-n file*
- Output the last n lines of a file:
tail *-n file*
- Option -f: displays always the last lines of a file, useful for monitoring log files

File Editing II

- Count lines, word, characters:
wc *Dateiname (word count)*
- Sorting words:
sort *fileName*
- Spilt a file into columns:
cut
- Joining file using a common key column: **join**
- columnwise merging of files:
paste

Editing Files III

- Remove multiply occurring lines:
uniq
- Create an empty file or update the time stamp:
touch fileName

System Status

- Get the seizure (use space) of a hard disk:
df: disk free
shows the available space on file systems and the used space
- space consumption of files and directories:
du: disk usage
shows you the space consumption of files, many options to improve the readability
- mounted file systems (drives):
mount

System Status II

- Display the environment variables:
env: environment
displays the name and the values of the environment variables of your shell
- show the system time:
date
- time consumption of a program:
time
- logged-in user and processes: **who**
- terminal settings: **stty**
 - if your terminal is misconfigured: **stty sane**

Processes

- process: Activity which gets computing time from the operating system
- characterized by: own ID, parent-ID, consumed cpu time, used (virtual) memory, status
- can run in front or as background
- can run without control terminal (demon process)

Processes 2

- **ps**: report process status
ps [optionen]
- Options:
 - u UID prozesses for user UID,
w/o UID for all users
 - a all processes having a terminal
 - x shows processes w/o control terminal
 - l verbose informations (long)
 - f display as tree

Prozesses 3

- **kill**: kill processes
kill -Signal PID
- Options:
Signal 15 (terminate, geordnet beenden),
 9 (abort, sofort unbedingt beenden)
- **killall ProgName**
finishes all processes of program ProgName,
assumed you have permission to do so
- **top**: tool for online display of active processes

Piping

- Many tools use as standard output device the screen and as standard input device the keyboard
- These standard channels can be redirected and combined into a series of connected processing steps

Simple In/Output Redirection

- Redirect screen output into a file:
`myProgram > myPutputFile`
- Redirection of keyboard input from a file:
`myProgram < mInputFile`
- Combination of both:
`mProgram > mOutputFile < mInputFile`

Piping

- Pipes is a mechanism provided by the operating system to automatically connect program in a series:

Prog1 | Prog2 | Prog3

i.e. putput of program 1 is directly used as input for program 2 and the output of program 2 is used by program 3

- Goal: many small but specialized tools can be connected to fulfill complex tasks

Filtering

- In a series of connected programs normally there is a processing at each step. I.e. certain data are modified or filter so this is also called filtering.

grep

- **grep**: get regular expression
- Search for patterns in input:
 - Pattern are regular expressions
 - regular expression similar to those in Perl
- Input: files or standard input
- Options:
 - c counts occurrences (count)
 - r search directories recursively
 - e specify a pattern
 - v all lines that do not match
 -

find

- **find** find files
- searches the file system starting at a certain point for (specified) files
- the desired files are specified by conditions
- Conditions / Options:
 - name, -type, -size, -ctime, -mtime, -depth.
 - follow, etc.

locate / which

- **locate**: finding files
 - use its own database
 - the database is regularly updated by a cron job
 - supports search pattern, too
 - faster than **find**, but depends from its database
- **which**: finds executable program in the search path, programs are found if they are reside in a directory listed in the `PATH` variable

Archiving

- **(g) zip**: compressing of files
(g) unzip: resp. for uncompressing
- **tar** : tape archive
tar -directive [options]
 - combine with g(un)zip to compress archives
 - most important directives: c,x,t
 - c create a new archive
 - x extract an existing archive
 - t list the contents of an existing archive
 - Ex.: **tar -czf *archiv.tgz* *directory/*** to create an compressed archive containing *directory* in the file *archiv.tgz*
tar -xzf *archiv.tgz* unpack and uncompress *archiv.tgz* to directory *directory*

Examples for Piping

- `find /usr/local/include -name "*.h" | grep gnu | wc -l`
counts the number of header files having "gnu" in the filename
"grep gnu *" would search the file contents for "gnu"
- `cd ; ls .* | wc -l`
counts hidden files in your home directory

Examples for Piping 2

- `cat /etc/passwd/ | grep system | less`
searches /etc/passwd for all line containing "system" and display them with the pager less
- `cd; find -type f | grep -c mp3`
counts all mp3 file in your home directory

Additional Tools

- **tee** *fileName*: read from standard input and writes to standard output and to the file *fileName* you can “tap” the pipeline at this point
- **sort**: input line are sorted numerically or alphanumerically, depending on the selected option

More Examples

- `cat /etc/passwd | cut -f1,5 -d:`
- `find . -name *.pl | wc`
- `cd; du -h --max-depth=3 | sort -rn
| less`
- `cd; grep -lvir gnu * | wc -l`
searches files in your home for the occurrence of "gnu" and counts the lines