Brandenburg
University of Technology
Cottbus - Senftenberg

Introduction into Cyber Security,
Cottbus, Germany, October 17th, 2018

Introduction into Linux

A. Panchenko, T. Ziemann
Research Group BTU Cottbus-Senftenberg,
Chair of IT Security

1. The Way from Unix to GNU/Linux

2. Understanding GNU/Linux

3. Important Unix Commands and Tools

UNIX

History

1969, Bell Laboratories: development of Unix to support software developer

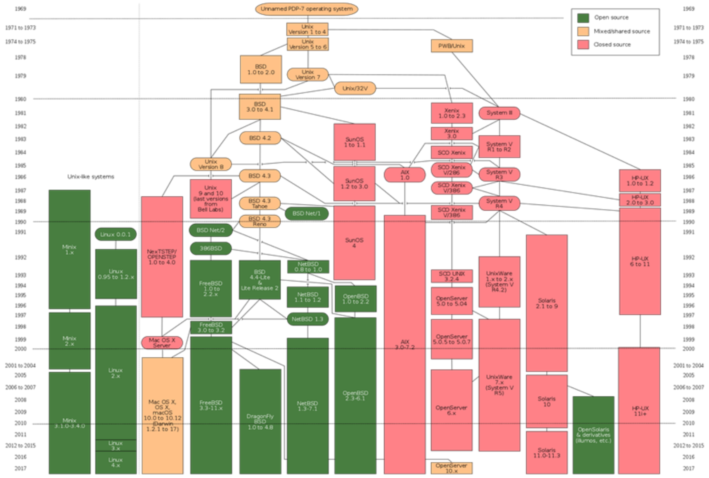
What is Unix?

Today, Unix are a term to denote any operating system which either is an descendant of UNIX or implements it concepts.

Properties

- Multi-user system
- Multi-tasking capabilities
- Multi-threating capable
- Memory protection / virtual memory

Ancestral Chart of UNIX



From UNIX to Linux

GNU-Project (GNU: GNU's Not Unix)

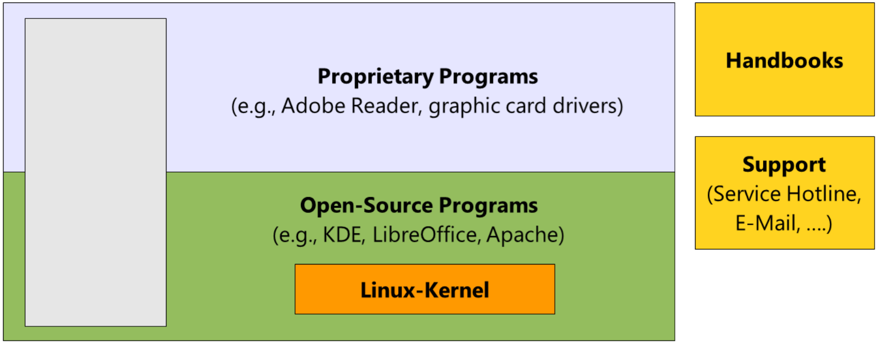
1983, Richard Stallman started the GNU-Project to develop a free equivalent of the UNIX operation system

Development of Linux

- 1991, Linus Torvalds: development of the OS-Kernel (Open-Source)
- 1992, Kernel was licensed by GNU GPL
- Linux: similar OS to UNIX, based on Linux-Kernel/GNU-Software
- Today, Linux is the most widely used open source version of UNIX
- Discussion: Linux or GNU/Linux

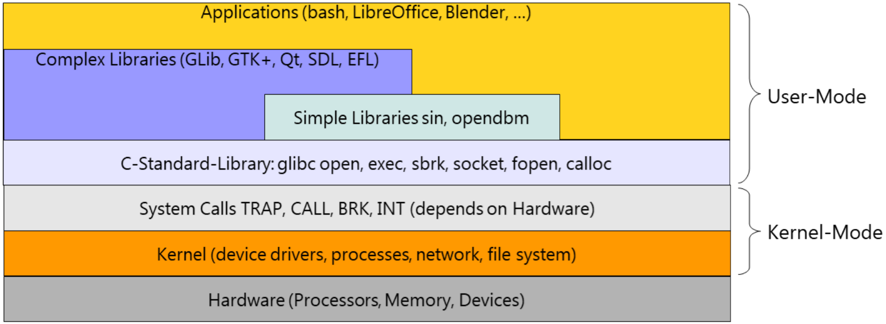
Linux Distributions

Contents of a Distribution:


Proprietary Programs
(e.g., Adobe Reader, graphic card drivers)
Open-Source Programs
(e.g., KDE, LibreOffice, Apache)
Linux-Kernel
Handbooks
Support
(Service Hotline, E-Mail, ...)

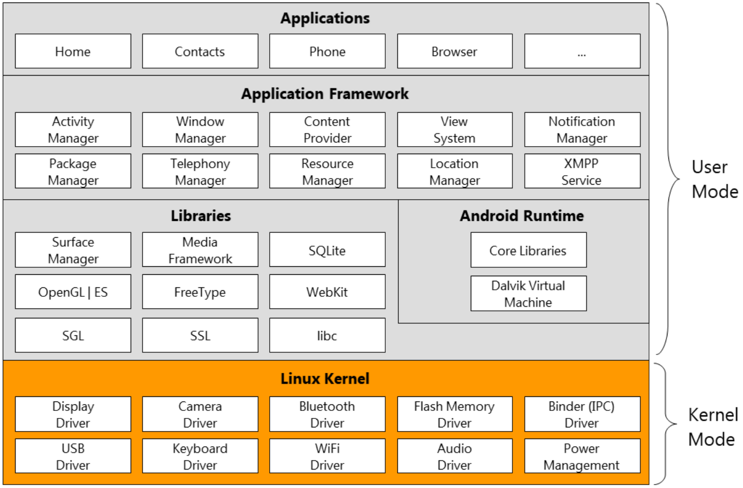
Examples:
Debian, Fedora, Red Hat (REL), Gentoo, Android, Firefox OS, . . .

Linux Architecture

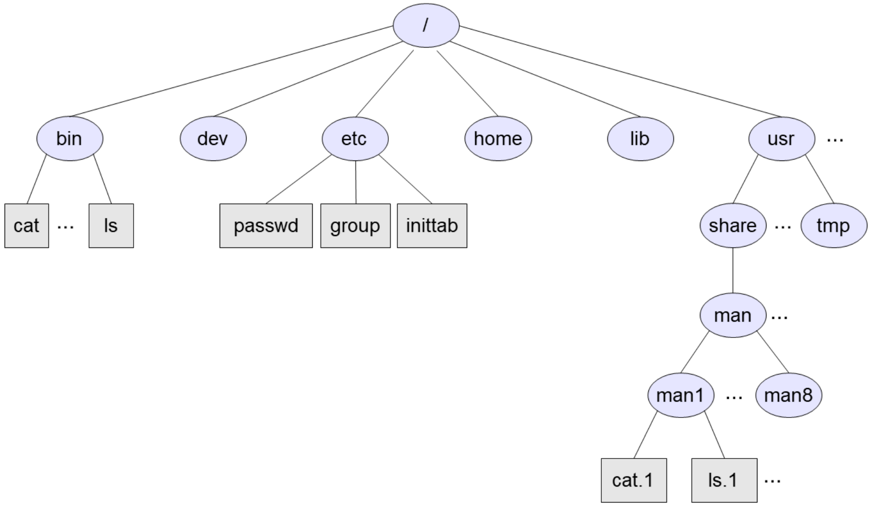

Applications (bash, LibreOffice, Blender, ...)
Complex Libraries (GLib, GTK+, Qt, SDL, EFL)
Simple Libraries sin, opendbm
C-Standard-Library: glibc open, exec, sbrk, socket, fopen, calloc
System Calls TRAP, CALL, BRK, INT (depends on Hardware)
Kernel (device drivers, processes, network, file system)
Hardware (Processors, Memory, Devices)
User-Mode
Kernel-Mode

- System calls as interface between user mode and kernel mode

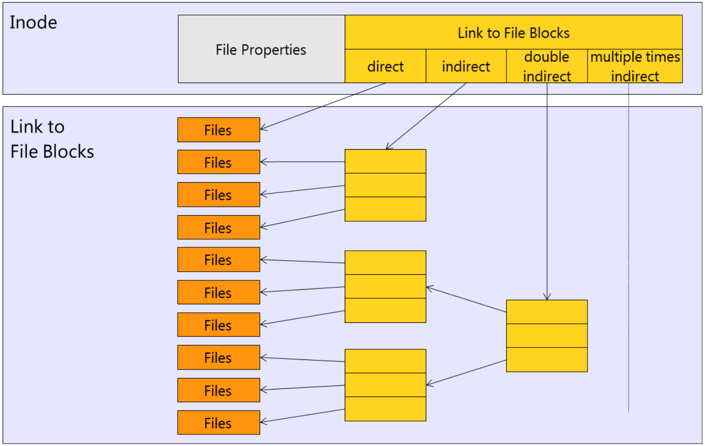
Android – An Example



Structure of File System



File System: Inode



GNU/Linux File Types

Regular File

text and binary files like programs, scripts, configuration files, . . .

Directory

contains Inode-numbers of any files in the directory

Device File

interface to hardware;
distinction between block based (buffered) and character devices (non buffered)

File System: Access Control

User Domains

- **User:** creator of the file
- **Group:** all users within the same group as the creator of the file
- **Others:** all remaining users

File Operations

- read (r)
- write (w)
- execute (e)

⇒ some practical examples

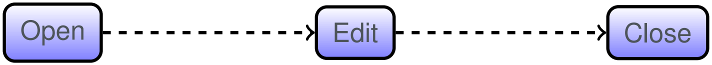
File System: Access Control

File access permissions in the case of directories:

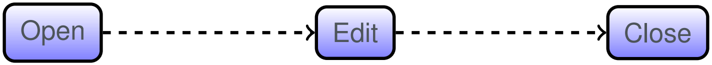
dir permissions	Octal	del rename create files	dir list	read file contents	write file contents	cd dir	cd subdir	subdir list	access subdir files
---	0								
-W-	2								
R--	4		only file names (*)						
RW-	6		only file names (*)						
--X	1			X	X	X	X	X	X
-WX	3	X		X	X	X	X	X	X
R-X	5		X	X	X	X	X	X	X
RWX	7	X	X	X	X	X	X	X	X

`https://unix.stackexchange.com/questions/21251/execute-vs-read-bit-how-do-directory-permissions-in-linux-work`

Working with Files



Working with Files



Open

- Open File by absolute or relative path
- Check file access on execution
- return file descriptor on success

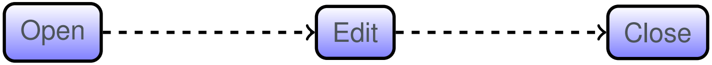
Working with Files

$$\begin{tikzcd}\boxed{\text{Open}} \arrow[r, dashed] & \boxed{\text{Edit}} \arrow[r, dashed] & \boxed{\text{Close}}\end{tikzcd}$$

Edit

- Reference file by its file descriptor
- Read or Write File

Working with Files



Close

- release file descriptor

The Shell

Command Line Interpreter

- Started by login service after successful authentication of user
- Interprets and executes user commands with the access rights of the callee
- Provides:
  - script language for automation
  - wildcards (e.g., $*$)
  - environment variables (e.g., $HOME)
  - input/output pipelining
  - command history

Important GNU/Linux Shell Commands

File Commands

ls – directory listing
ls -al – formatted listing with hidden files
cd *dir* - change directory to *dir*
cd – change to home
pwd – show current directory
mkdir *dir* – create a directory *dir*
rm *file* – delete *file*
rm -r *dir* – delete directory *dir*
rm -f *file* – force remove *file*
rm -rf *dir* – force remove directory *dir* *
cp *file1 file2* – copy *file1* to *file2*
cp -r *dir1 dir2* – copy *dir1* to *dir2*; create *dir2* if it doesn't exist
mv *file1 file2* – rename or move *file1* to *file2* if *file2* is an existing directory, moves *file1* into directory *file2*
ln -s *file link* – create symbolic link *link* to *file*
touch *file* – create or update *file*
cat > *file* – places standard input into *file*
more *file* – output the contents of *file*
head *file* – output the first 10 lines of *file*
tail *file* – output the last 10 lines of *file*
tail -f *file* – output the contents of *file* as it grows, starting with the last 10 lines

File Permissions

chmod *octal file* – change the permissions of *file* to *octal*, which can be found separately for user, group, and world by adding:

- 4 – read (r)
- 2 – write (w)
- 1 – execute (x)

Examples:
chmod 777 – read, write, execute for all
chmod 755 – rwx for owner, rx for group and world
For more options, see **man chmod**.

SSH

ssh *user@host* – connect to *host* as *user*
ssh -p *port user@host* – connect to *host* on port *port* as *user*
ssh-copy-id *user@host* – add your key to *host* for *user* to enable a keyed or passwordless login

Searching

grep *pattern files* – search for *pattern* in *files*
grep -r *pattern dir* – search recursively for *pattern* in *dir*
command* | grep *pattern – search for *pattern* in the output of *command*
locate *file* – find all instances of *file*

Additional References

- Linux Command line Reference
  `https://ss64.com/bash/`
- Linux Shell Scripting Tutorial: A Beginners Handbook
  `http://www.freeos.com/guides/lsst/`
- Linux Services: A list of UNIX and GNU/Linux services
  `http://www.linux-services.org/shell/`
- Galileo Computing: Shell Programming
  `http:`
  `//openbook.galileocomputing.de/shell_programmierung/`

Thank you for your attention!