

# Advanced Operating System with UNIX

Christian Kniep

Internation Center of Applied Technologies Bandung

27. July 2010

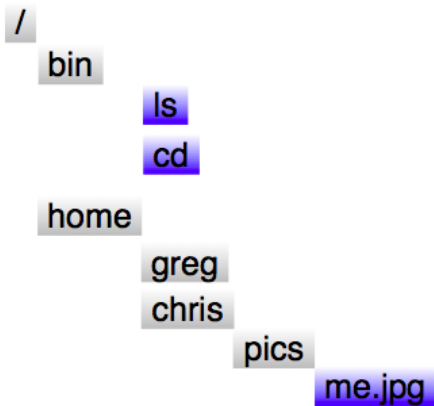
# Table of content

- 1 Unit 2
  - Introduction to UNIX-Filesystem
  - Filesystem
  - Introduction to UNIX-Filesystem
  - Mount and Unmount
  - Fck: File System Checking
  - The Boot Procedure
  - Kernel
  - System Processes
  - Startup Scripts
  - User-/Group-Handling
  - Backup & Restore

- There are only directories or files, that's it!
- everything is a file, whether it is
  - ▶ a command, textfile, archive, etc.
  - ▶ a resource, setting

# In General

- The filesystem looks like a tree



# In Detail

- The basic folder-hierarchie should be

/

bin	Essential command binarys
boot	Static files of the boot loader
dev	Device files
etc	Host-specific system configuration
lib	Essential share libraries and kernel
media	Mount point for removeable media
mnt	Mount point for mounting temporarily
opt	Add-on application software packages
sbin	Essential system binarys
srv	Data for serives provided by this system
tmp	Temporary files
var	Variable Data
usr	Secondary hierarchy
bin	
sbin	

# File System Types 1

- The Filesystem-Type defines how to use (speak to) the physical device
- There are several ones out there
  - ▶ historical types
    - ★ **s5** for the old SystemV OS
    - ★ **msdos,pcfs** for old versions of DOS / Windows
  - ▶ to use windows-partitions
    - ★ **fat16,fat32** the old Windows-Filesystem
    - ★ **ntfs-3g** to support WindowsXP,Vista and 7

# File System Types 2

- not that common
  - ▶ **ufs(2)** for FreeBSD
  - ▶ **bfs** boot file system for SystemV
- in broad use
  - ▶ **iso9660, hfs** for cdroms
  - ▶ **proc, procfs** pseudo-FS in the memory to handle processes
  - ▶ **ext2, ext3** RedHat, debian-derivates
  - ▶ **xf**s SUSE
- upcoming
  - ▶ **ext4** due to long history of ext2, ext3
  - ▶ **zfs** pushed by sun, manage a consistant FS
  - ▶ **btfs** speak BetterFS, like ZFS but OpenSource

# Swap? Whats that about?

- If the memory is fully loaded the OS could outsource some process data to the swap partition
- a drawer (swap) in your desk (RAM) instead of the cabinet (filesystem)
- when the OS uses the swap we say 'the machine is swapping'
- very bad, because its way slower then the normal RAM!



# Fdisk,mkfs

- with fdisk you are able to create,alter and delete the partition-table
- new partitions could be set up with a filesystem by using mkfs

# mount,umount

- if you want a device be part of your 'File-System'-tree?
- use `mount -t type device mountpoint` to do it
- if you want it to disappear use `umount device` or `umount mountpoint`

# data erode to your fingertips

- power blackout
- 'my dog bites on my pen-drive'
- you name it!

# it causes inconsistent states

- multiple inodes claim the same disk block
- a free block is not listed in the superblocks
- a used block is marked free
- ...

# soloution 'fsck'

- Phase 1 (simple stuff)
  - ▶ Validates the inodes for correctness (format, block numbers)
  - ▶ declares blocks BAD (number out of Range), DUP (claimed by another inode)
- Phase 2 (what files/directories are involved?)
  - ▶ Starting from root, searches for OUT OF RANGE inode numbers detected in P1
  - ▶ found one, than removes the 'dir' or 'file'

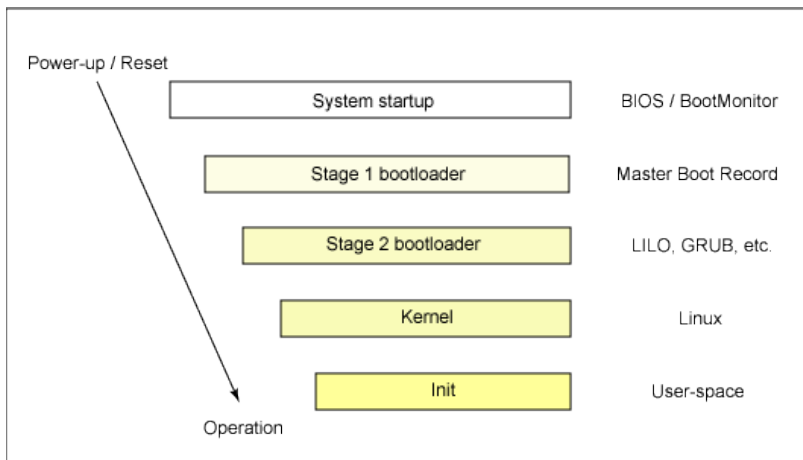
# soloution 'fsck'

- Phase 3 (lost+founds)
  - ▶ Looking for unreferenced directories and stores their files in ' $l + f$ '
  - ▶ the files are named as the inode number
- Phase 4 (check counter)
  - ▶ compares link count information from Phases 2 & 3, correcting discrepancies

# Get it on

- Hole process from pushing the button to have a login prompt
  - ▶ The memory-resident code
  - ▶ self-test
  - ▶ probes bus for boot device
  - ▶ Reading boot-sector from boot device
  - ▶ Boot-program reads kernel and initrd and passes control
  - ▶ Kernel identifies, initialises and configures the devices
  - ▶ Runs appropriate startup scripts (single- / multi-usermode)

# Get it on





# Startup-process

- The kernel is kind of a puppetmaster who pulls the strings
- If he hadn't initialise a device, it will not be available

# What comes up?

- To start the machine there are various processes that have to run
- the first process called 'swapper' it became PID 0 (but lives not long)
- the 2nd one is the init-Process (PID 1), which forks all the startup-scripts (init-scripts)
- on usual Unix-Systems the PID 2 is the first process created by the init-process and gets the PID 2

# process table

UID	PID	PPID	C	STIME	TTY	TIME	CMD
root	1	0	0	Jul11	?	00:00:01	init [2]
root	2	0	0	Jul11	?	00:00:00	[kthreadd]
root	3	2	0	Jul11	?	00:00:00	[migration/0]
root	4	2	0	Jul11	?	00:31:45	[ksoftirqd/0]
root	1071	2	0	Jul11	?	00:00:00	[ata/1]
root	1072	2	0	Jul11	?	00:00:00	[ata/2]
root	1500	1	0	Jul11	?	00:00:02	udevd --daemon
root	2297	2	0	Jul11	?	00:00:00	[kpsmoused]
root	2302	2	0	Jul11	?	00:00:00	[hd-audio0]
root	2552	2	0	Jul11	?	00:00:00	[kjournald]
daemon	2666	1	0	Jul11	?	00:00:00	/sbin/portmap

# As we are speaking of it...

- the init-process (PID 1) are able to start scripts in various ways
- inittab (like in the SMU-book)
- Entry looks like `1:2345:respawn:/sbin/getty 38400 tty1`
- The entrys are `id:runlevels:action:command`

`id` Identifier

`runlevel` in which runlevels the command should be started

`action` what action should be taken for this command, maybe:

`commands` specifies the shell command to be run

# add Users











- if you simply want to add a user you use `useradd USERNAME`  
The User will be created without any further Questions.  
Parameters like `-G, -g, -s` are possible to add more information
- if you want to be guided you should use `adduser USERNAME`  
You will be asked about the Username and some other Information
- To alter user information you use `usermod USERNAME`
- `userdel USERNAME` deletes the user (-r wipes the home too)

# Groups

- same as for users ( `groupadd` , `groupdel` , `groupmod` )
- the information are stored in `/etc/groups`

## /etc/passwd

- As we said earlier, all settings are stored in files
- So do user settings:

<code>root</code>	<code>:</code>	<code>S3ah8kaR</code>	<code>:</code>	<code>0</code>	<code>:</code>	<code>0</code>	<code>:</code>	<code>System Administrator</code>	<code>:</code>	<code>/var/root</code>	<code>:</code>	<code>/bin/sh</code>	
													
name		crypt. PW		uid		gid		description		home-dir		command/shell	
<code>kniep</code>	<code>:</code>	<code>...</code>	<code>:</code>	<code>1000</code>	<code>:</code>	<code>1000</code>	<code>:</code>	<code>Christian Kniep,,,</code>	<code>:</code>	<code>/home/kniep</code>	<code>:</code>	<code>/bin/bash</code>	
													
				uid > 999		gid > 999		GEC0-String					

- Some User-ID explanation :
  - From start there where 15bit UID ( $2^{15} = 32768$ )
  - modern systems may provide more (now up to 64)

`id=0` tied to the root-User

`id<100` reserved for system-user

`id<1000` reserved for users which are running services

`id>999` normal useraccounts

`id=32767` traditionally the user nobody (opposite to root)

# you should now

- since everything is a file you should backup the whole filesystem, so you will be well prepared for malfunction
- there are a couple of ways to do that:
  - dump** according to the book you could use dump, this will copy blockwise on e.g. a tape
  - tar** tar is the right choice to zip files/directories
  - dd** with (one name-legend said) DiskDump you could copy a data stream. This could be a harddisk, or a file, 'you name it!'. The beauty of it is the simplicity and the power.  
You could cut byte out, count it, whatever you want.



## additional stuff

**split** do you want to split a large file in many smaller ones?

- ▶ `split -a 1 -l 50 test.txt c`

This will split the test.txt in files ca,cb,cc,.. containing 50 lines

- ▶ `split -a 1 -b 50 m test.img d`

This will split the test.img in files da,db,dc,.. containing 50MB

**cat** to bring them back together you use:

```
cat c* > test.txt
```

```
cat d* > test.img
```