Filesystems Overview

- FAT: *File Allocation Table*
  - The original MS-DOS filesystem, tree structure
  - Limited depth of sub-directories, limited filename lenght
  - Follow-up systems: FAT8/16/32, VFAT, exFAT
    - VFAT and exFAT are nowadays used on external/flash USB disks
- NTFS: *NT File System*
  - First appeared on Windows NT
  - Current MS filesystem on Windows
- ISO9600: Read-only files system for CD/DVD
  - Limited subdirectory length, limited filenames
  - Enhanced systems: Joliet, ElTorito, Rock Ridge

MS filesystems in Linux

- Linux can work with all of them (R/W).
- MS formats are not open $\Longrightarrow$ reverse engineering
- Support for the newest exFAT is yet somewhat experimental

Journaling filesystems on Linux

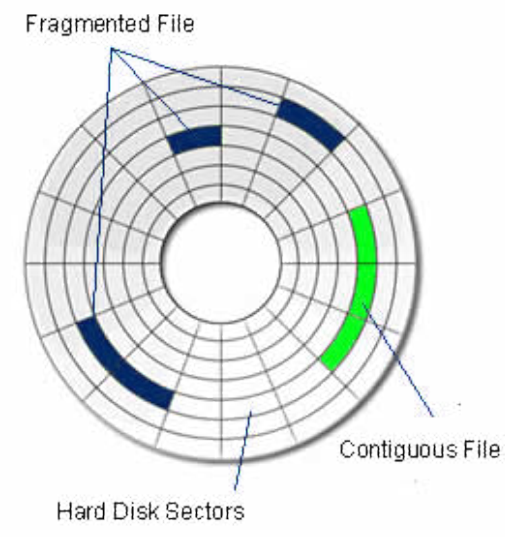
- Keeps track of disk operations not yet performed / finished
- Minimizing loss of data

Swap

- Enhancing RAM of the system
- When RAM gets full, swap is used to store temporarily unused data
- No special structure
- Windows equivalent: pagefile.sys

- `Ext2`, `Ext3`, `Ext4`: The most common on Linux
  - `Ext3` = `Ext2` + journal
  - `Ext4` allows for larger files and for more files per dir (32000 vs 64000)
- `XFS`: Silicon Graphics for the OS *IRIX*
  - Good performance for large data handling
- `ReiserFS`, `Reiser4`:
  - Good performance for large number of small files
  - Connecting advantages of filesystems and databases
  - Developed by Hans Raiser (in 2008 convicted murderer)a
- `JFS`: IBM for the OS *AIX* and later for *OS/2* and Linux
  - First journaling system
- `AFS`: *Andrew File System*
  - Distributed file system for large networks of servers and client workstations

Filesystém	Max. velikost filesystému	Velikost bloků	Max. velikost souboru
Ext2	4 TB	1KB-4KB	2 GB
Ext3	4 TB	1KB-4KB	2 GB
ReiserFS	16 TB	až 64KB	2^10 PB *1
XFS	18000 PB *1	512B - 64KB	9000 PB *1
JFS	512 B / 4 PB *2	512B, 1024B, 2048B, 4096B	512B / 512Tb *2



No defragmentation on Linux

- MS filesystems need occasional defragmentation:
  - Files are placed to the next free sector on the physical disk $\Longrightarrow$ when a file is enlarged, the next sector of physical disk is usually already occupied by another file $\Longrightarrow$ file fragmentation
- Linux filesystems defragment on the fly:
  - Files are placed on distant sections on the physical disk $\Longrightarrow$ space around existing file is usually free
  - If there is a danger file would become fragmented, the system tries to find a continuous space on the physical disk and moves the file in there
  - Fragmentation thus appears only when disk is close to full

Linux Files

- *Directory* (`d`): container of files
- *Plain file* (`-`): ordinary file (collection of bytes)
- *Symbolic link* (`l`): analogy of hypertext link or shortcut in Windows
- *Hard link*: similar to ordinary file, just another name for one file
- *Block and character device* (`b`,`c`): representation of HW devices
- *Name pipe* (`p`): named pipes for communication between applications instead through RAM
- *Socket* (`s`): duplex communication between processes

```
meop35 exam # ls -lai
total 21
1433763 drwxr-xr-x  5 petr petr     288 Oct  9 11:59 .
   4350 drwxr-xr-x 11 petr users    976 Oct  9 11:57 ..
1434761 -rw-r--r--  1 petr petr   13643 Oct  8 21:37 FILESYSTE.win
1434763 lrwxrwxrwx  1 petr petr      11 Oct  9 11:59 core -> /proc/kcore
1434735 drwxr-xr-x  2 petr petr      48 Oct  9 11:57 doc
1434764 srwxrwxrwx  1 petr petr       0 Oct  9 11:59 gpmctl
1434757 drwxr-xr-x  2 petr petr      72 Oct  9 11:58 hudba
1434758 -rw-------  1 petr petr      31 Aug 21 13:52 hymna.mp3
1434756 drwxr-xr-x  2 petr petr      80 Oct  9 11:58 prednasky
1416878 crw-rw----  1 root tty   2, 190 Oct  9 11:59 ptyae
1434762 lrwxrwxrwx  1 petr petr       9 Oct  9 11:58 sym_link -> hymna.mp3
meop35 exam #
```

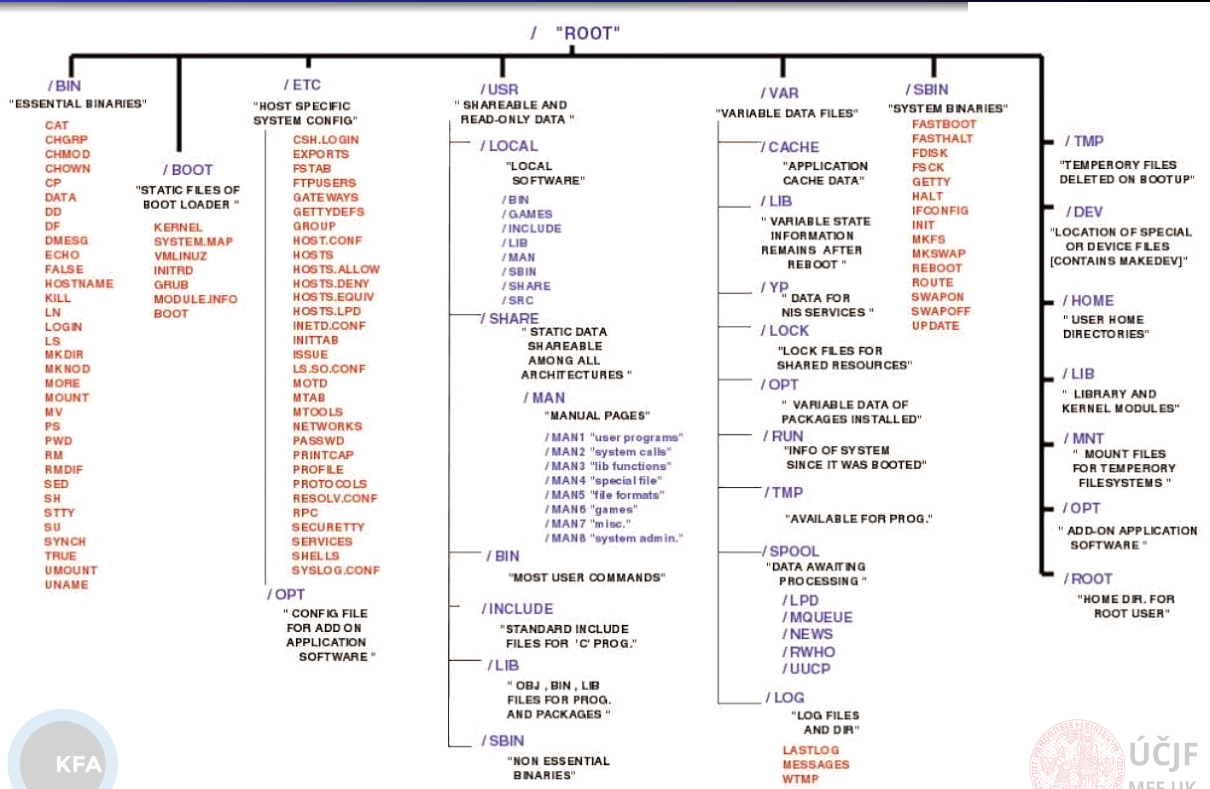
List files using command `ls -lah`

Structure of Directories in Linux

Directories and files in Linux

- Physical disks not visible in the structure (no `C:` and `D:`), disks are *mounted* to directories (admin can define what is mounted where)
  - e.g. typically "disk D:" is mounted to `/home` directory with users data
- Using '`/`' (slash) instead of Windows '`\`' (backslash)
- Linux filesystems (like `ext4`) features:
  - Directory and file names are CaSe SeNsItIvE
  - Native support of access control (user/group/all, read/write/execute)
  - Links (a bit similar to shortcuts in Windows, but for applications behave as real files)
- Hidden dirs and files starting with dot (`.*`, e.g. `.config`), usually keeping configuration of applications
- Special files representing e.g. HW devices and their configurations
- Special directory names:
  - Root directory '`/`' (top directory of the Linux system)
  - Home directory of a user '`~/`' (representing typically `/home/username`)
  - Current directory '`./`'
  - Directory one level up '`../`'
- **Absolute paths** starting from root of the system `/`
- **Relative paths** from current directory (may or may not start with '`./`')
- Text files has different standard for end-of-line (EOF) code from Windows/DOS/Old MACs
  - Conversion via commands `dos2unix` and `unix2dos`

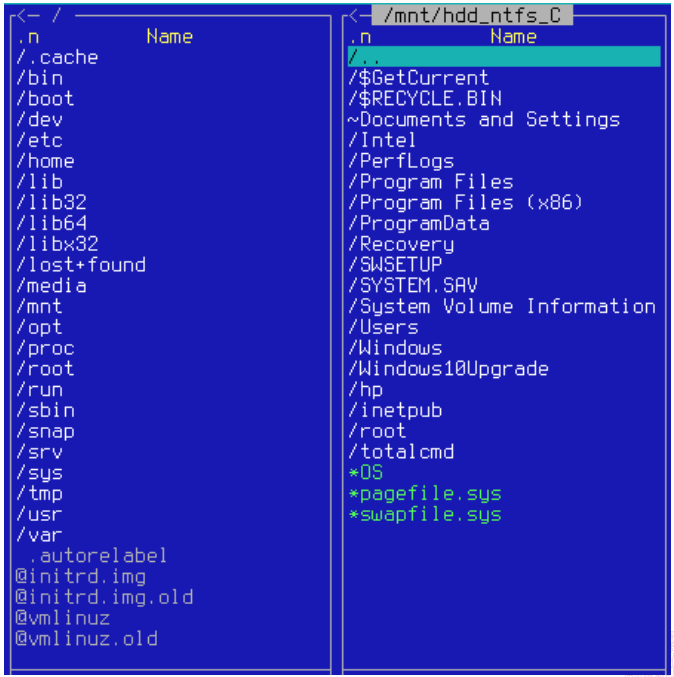
Diagram of the Directories



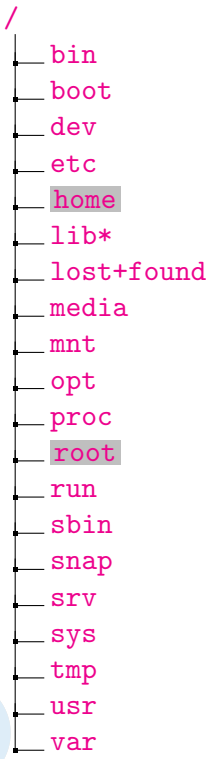
Linux Root Directory

- /
  - bin
  - boot
  - dev
  - etc
  - home
  - lib*
  - lost+found
  - media
  - mnt
  - opt
  - proc
  - root
  - run
  - sbin
  - snap
  - srv
  - sys
  - tmp
  - usr
  - var

Snapshot of top directories in Linux and Windows 10:



User Home Directory



/home

- Home directory of users in `/home/$(USER)`
- Place used by all ordinary users to store data, configurations, possibly also local applications
- Other users can usually read files of other users, unless manually disabled
- `/home/$(USER)/Desktop`: Any file/directory placed here will appear on GUI desktop
- `/home/$(USER)/.*`: User configuration files of various applications
  - Especially in `.config` subdirectory
  - Files associations, user shortcuts etc. in `.local/share/` (`applications`) subdirectory
- Analogy of `C:\Users`, resp. `C:\Documents and Settings` in Windows

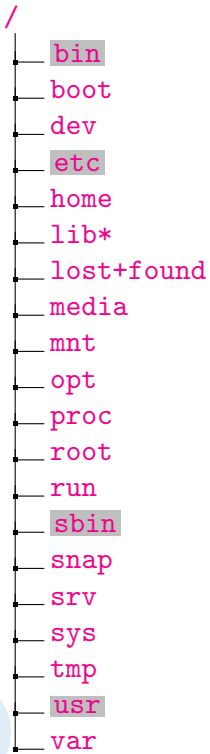
/root

- Home directory of administrator (*superuser*/*root* user)
- Same structure as for users in `home/$(USER)` directory

KFA MFF UK

ÚČJF MFF UK

Application Installation



Ordinary application files

- Executable files in `/usr/bin`
- Data files in `/usr/share`
- Documentation, licence, examples etc. in `/usr/share/doc`
- Manual pages (help) for executables in `/usr/share/man`
- Libraries in `/usr/lib*`
- Header files of libraries in `/usr/include`

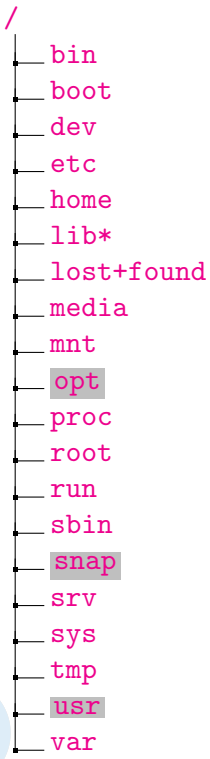
System application files

- Executable files in `/bin`
- *Superuser* executable files in `/sbin` and/or `/usr/sbin`
- `/etc`: Configuration of the system (services) and applications, common for all users
  - `etc/default`: Most common default settings
  - Analogy of Windows registry

KFA MFF UK

ÚČJF MFF UK

Application Installation (Special, Local)



Local installations

- Usually not going through the ordinary package system, but directly copied to the system
- These programs and libraries placed in `/usr/local`
  - Contains similar sub-structure as the root `/` and/or `/usr` directories
  - I.e.: `etc`, `bin`, `lib`, `share`, ...

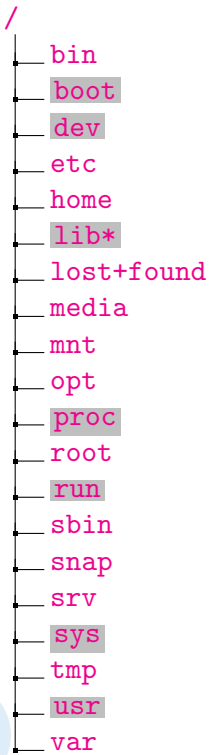
3rd party / commercial applications

- Whole application stored in `/opt`, with links to `/usr/bin` etc.
- `/snap` stores distribution-agnostic applications
  - All dependencies (libraries etc.) are included

KFA MFF UK

ÚČJF MFF UK

Boot, Linux kernel, devices



Boot configuration in /boot

- Linux kernels to boot
- Configuration of the boot manager (*GRUB*)

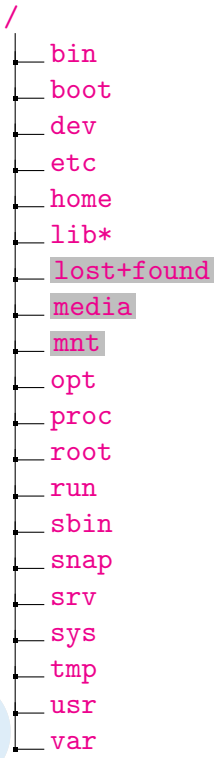
Linux kernel modules and source

- Linux kernels sit in /boot
- Linux kernel modules in /lib/modules
- Linux kernel source code in /usr/src

Hardware devices and processes

- Connected HW devices represented by special files in /dev
  - Storage devices to mount
  - Dustbin /dev/null
  - Console, random generator device, ...
- Configuration of the HW devices in /sys, *superuser* can use the files to modify the configuration of the HW drivers
- /proc directory keeps information about running processes, HW configuration (memory, cpu)
  - Interface to the kernel internal data structures
- Runtime information about processes in /run (e.g. connected WiFi)

Storage devices



Lost files

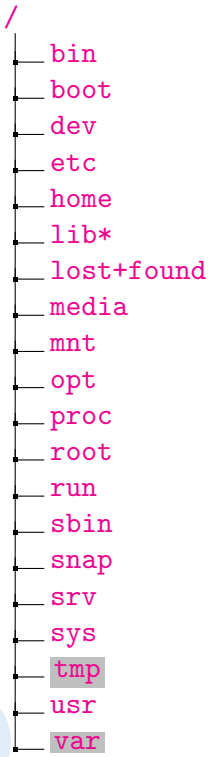
- Lost and recovered files after OS crash

Storage devices connected to the system

- Removable devices usually appear in `/media`, resp. `/media/$(USER)`
  - Modern distributions connected them (*mount*) automatically
- Windows partitions, network storages etc. are usually mounted to `/mnt`
  - Follow rules defined in `/etc/fstab` file
  - And/or rules for auto-mounting in `/etc/auto*` files

KFA MFF UK

ÚČJF MFF UK



Temporary application settings in /var

- /var/cache: Temporary files of applications, e.g.:
  - Downloaded installation packages of apt system in /var/cache/apt/archives
- /var/lib: Variable system information, e.g.:
  - Network connections
  - Installed packages info in /var/lib/dpkg/info
  - Content of package repositories in /var/lib/apt/lists
- /var/spool: Data awaiting further processing, e.g.:
  - E-mails
  - Print jobs
- /var/log: System log files:
  - Crucial for administrators
  - Boot, kernel messages, logins, services (e.g. WWW servers) etc.
- /var/www: WWW server data
- /var/lock: Application and system lockers

Temporary files in /tmp

- Temporary files of applications, e.g.:
  - Opened / downloaded files from web-browsers
  - Archives unzipped in file-browsers

Users and Groups

Users and Groups:

- /etc/passwd: users configutation: home, shell, title, uid
- /etc/shadow: users password hash
- /etc/groups: groups

Superusers:

- su / sudo (-i)
- su needs further config to run graphical applications
- visudo
- separate home dir in /root

Work with the Command Line in Terminals

Command Line

- Very powerfull in Linux, allows to master the OS
- Commands are taken by a *shell* and given to the OS
- Allows complicated scripts including loops, macros, conditions, etc.
- Several *shells* exists:
  - `bash`: most commont
  - `dash`: minimalistic, for system scripts
  - `zsh`: programmers focused
  - `ksh`:
  - `csh`: different syntax from bash-like shells above
  - `tcsh`: enhances csh
  - `fish`: friendly interactive shell
- `chsh` command to change user shell

Shortcut Keys

- **Left/Right/Home/End**: navigate cursor through the command line
- **Up/Down-Arrows**: browsing through history of commands
- **Tabulator**: complements commands or file names
  - Search for commands in the standard executable paths
  - Shows all possibilities in case the completion is ambiguous
  - Possibility to enhance completion for specific commands (ssh and remote host names etc.)
- **Ctrl-r**: search in history of commands backward
- **Ctrl-s**: search in history of commands forward
- **Ctrl-g**: end of search (not-only search) mode
- **Ctrl-l**: clear terminal window
- **Ctrl-q**: unblocks blocked terminal
  - Some terminal emulators get blocked with the Ctrl-s command
- **Ctrl+c**: interrupt running process
- **Ctrl+d**: interrupt writing into file (e.g. in `cat > filename`)

Shell Settings

- Configuration / startup files:
  - `/.bash_history`: history of commands
  - `/.bashrc`: startup script for non-login interactive shells
  - `/.bash_profile`: startup script for `login shells` (login in text console or from remote host)
    - `/.profile` is read by bash too, to be backward-compatible with old `sh`
  - `/etc/profile`: system-wide
  - `/etc/motd`: message on login shells
- Other shells have similarly-named files
- Typical configurations in `/.bashrc`:
  - Format of command prompt: PS1 (also PS2-PS4 for 2nd etc. level of prompts)
  - PATH to executables
  - PATH to 3rd party or local libraries (LD_LIBRARY_PATH)
  - Enhanced completion and other plugins (e.g. list of commands in not-installed packages)
  - Aliases (alias/unalias commands)
  - Default limits (ulimit command)
  - Environmental variables / program setups (`export | less`)
  - History length

Commands in Linux

Programs available in shell or in the executable PATHs.

- General structure: [path/]command [options] [arguments]
- List of options and arguments (e.g. for `systemctl` command):
  - `command --help`, resp. `command -h`: basic syntax and options of the command
  - `man command`: manual pages with more detailed info
  - `info command`: interactive manual pages (jump to references etc.)
  - `command -v`. resp. `command --verbose`: be verbose about what is being done
- Run in background, while continuing work in shell: `command ... &`
- `fg`: Put command already running in background back to foreground of the shell
- **Ctrl+z**: Suspend process / command (stops processing)
- `bg`: Put command into background (typically used to let suspended command running again in the background)
- When command / program ends, it usually returns a *return code*
  - 0 ... usually means "no error"
  - non-zero is usually indication of premature exit of the command because of some problem
- Find full path to a command: `which command` (or search for it in aliases: command `alias` lists existing aliases)