Faculty of Mathematics and Physics
CHARLES UNIVERSITY

NOFY077

Introduction to the Linux OS

Peter Huszár
KFA: Department of Atmospheric Physics

Pavel Řezníček
ÚČJF: Institute of particle and nuclear physics

September 30, 2025

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)

Scripts: Basics

Sequence of commands can be put into script files.

- `#` are used for comments
- Special header "comment": `#!/usr/bin/zsh` instructs the script to be run by the `zsh` shell. Not only for shells, but also for interpreters like `python`
- `exit` [`number`] to quit script [and possibly return a *return code*]
  - Not needed at the very end of a script, it will end by itself
- `set -x` command inside a script instruct to show the commands being run by the script
- Two ways how to run a script:
  - *./script.sh*: starts a new shell and runs the script in it
  - *source ./script.sh* (or also `. ./script.sh`: runs the commands from the script one by one in the current shell → i.e. as if one would write them manually in the current terminal
  - PS: The *'./'* makes sure it runs `script.sh` in the current directory, and not somewhere from the `$PATH` paths

```
#!/bin/sh
echo $SHELL
sleep 0.5
```

Commands: User Information

- `login`: Login user to the session, normally not directly invoked
- `logout`: Logout user from the session
- `exit`: Exit from the terminal/shell/script
- `id` [user]: User and group id information about myself
- `groups` [user]: List groups user is in
- `whoami`: Return user-name
- `who`: Show who is logged in
- `w`: Show who is logged in and what is he/she doing
- `last`: Show list of last logged-in users
- `su/sudo`: Become superuser
  - `su -` to make sure superuser paths are set
  - `sudo -i` to start superuser shell

Commands: File / Directories

- `ls` [path/file]: List files/directories
  - `ls -1`: List one file/dir per line
  - `ls -l`: List full information:

```
-rw-r--r--  1 reznicek reznicek 15360 Oct 14 19:40 osnova_Uvod_do_Linuxu.doc
drwxr-xr-x 18 reznicek reznicek  4096 Jan 31 2019  skola
```

    - File type (file='-', dir='d', ...), access rights
    - Number of links (copies)
    - User name
    - Group name
    - Size in bytes
    - Date and time of last modification
    - File name
  - `ls -h`: Human readable file size
  - `ls -a`: List also hidden files and directories (`.*`)
  - `ls -d`: Show information about directory instead of listing its content
  - `ls -R`: List recursivelly also content of sub-directories
  - `ls -t`: Sort by modification time
  - `ls -r`: Reverse sort order (name, time)
- `stat file`: Disk-storage info about `file`
  - Physical place on disk
  - Creation, modification and access times

Commands: File / Directories

- `cd` [path]: Change directory
  - `cd ../`: Go one directory up
  - `cd ./`: Go to (stay in) current directory
  - `cd /`: Go to root (top filesystem) directory
  - `cd ~/`, `cd`: Go to home directory
  - `cd -`: Go to previous directory (back from last cd command)
  - Change directory *relatively* to the current dir or *absolutelly* specified path from the root dir
    - *relative* path example: `cd work/papers` or `cd ./work/papers`
    - *absolute* path example: `cd /home/username/work/papers`
- `pwd`: Print current directory
- `readlink -f file`: Print full path to the given `file`
- `cat file`: Dump content of `file` on the screen
- `less file`: Browse `file` content on the screen
  - Enhanced `more` command (also on Windows)
  - Search for content ('/' key)
- `touch file`: Update *modification time*
- `diff file1 file2`: Show difference between two files, GUI versions exist too (`meld`, `diffuse`)
- File and directory manipulations (see later lectures in details):
  - `cp`, `mv`, `rm`: copy, move/rename, remove files
  - `mkdir`, `rmdir`: create directory, remove (empty!) directory
- Terminal-based file browser: `mc`

Work on Remote Machine via SSH

Secure Shell Connection, assuming the host PC is running `sshd` server (just install the ssh-server package...)

Client can:

- `ssh -Y username@hostname`
- Gains access to login shell on remote machine
- Can send graphical windows (e.g. xterm)
  - `-Y` allows to transfer graphics
- With sudo/su commands can manage the computer (including reboots, poweroff etc.)
- Configure automatic options for ssh via `$HOME/.ssh/config` file:

```
Host lxplus
  user reznicek
  Hostname lxplus.cern.ch
Host ipnp-stick
  User mlive
  Hostname 10.77.0.11
```

At MFF there is a faculty computing Linux cluster `chimera` for all faculty members:
https://www.mff.cuni.cz/en/hpc-cluster/general-information

Manipulation with users and groups

Users:

- Users stored in `/etc/passwd`

```
student:x:1001:1001:Student linuxu,T11,1234,5678,Poznamka:/home/student:/bin/bash
```

- `adduser`: add new user, copy `/etc/skel` content to the newly created home directory. Default options in `/etc/adduser.conf`
- `deluser`: remove user (home directory is kept by default)
- `usermod`: modifications to user settings (home dir, id, password-expiry, groups, shell, lock-password)
- `chfn`: changer user information (full name etc.)
- `passwd`: change user password
- `chsh`: change user shell

Groups:

- Groups stored in `/etc/group`
- `groupadd`: add new group
- `groupdel`: delete group
- `groupmod`: name, id
- `gpasswd`: groups can have passwords, administrators

Change ownership of a file/dir

Users:

- `chown username:groupname file`: change user and group ownership of a file/dir
  - `chown -R`: recursively for all sub-directories
  - `chown --dereference`: for symbolic links "jump" to the real file the link is pointing to
- `chgrp groupname file`: changer group ownership of a file/dir
- Username change mostly only for superuser, user cannot "give" file to another user. But owner can modify group =¿ give file for r/w to other users via groups

Change read/write/execute/access rights to files/dir

```
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 #
```

Command `chmod`

- `chown ugoa+/-rwxst file`
  - `u`: user rights
  - `g`: group rights
  - `o`: rights of others
  - `a`: rights of all (user, group, others), not specifying who defaults to "all"
  - `+`: add rights
  - `-`: remove rights
  - `r`: read rights
  - `w`: write/modify rights, for dirs only works when 'x' is set too
  - `x`: executable (file), directory enter and listing allowed (directories)
  - `s`: setuid bit, allows to run command with file-owner privilegies (`/etc/passwd`)
  - `t`: sticky bit for directory (o+t): everyone can create files, but files can be deleted only by the owners (Despite the "xw" rights for all in the directory - e.g. `/tmp`)

Change read/write/execute/access rights to files/dir

```
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 #
```

Command `chmod`

- `chown 0644 file`
- Rights can be also defined by octal numbers:
  - 777 = (binary) 111 111 111 = rwx rwx rwx
- Default permissions driven by `umask` (022 = files ar 644, dirs are 755 =¿ i.e. it is subscracted from 666 and 777)
- Some filesystems support more advanced file-atributes (e.g. immutable, nodeletable, no copy, no write, ...) via commands `chattr` (and listing these attributes via `lstattr`)