

# Introduction to the Linux OS

Peter Huszár

KFA: DEPARTMENT OF ATMOSPHERIC PHYSICS

Pavel Řezníček

ÚČJF: INSTITUTE OF PARTICLE AND NUCLEAR PHYSICS

November 16, 2023

# Overview and Organization

**Introduction to the Operation system Linux, focus on the command line, scripting, basic services and tools used in (not only) physics: tasks automation in data processing and modeling**

## Organization

- Graded Assessment (KZ): attendance to the lectures, worked out homeworks

## Literature

- C. Herborth: Unix a Linux - Názorný průvodce, Computer Press, Praha, 2006
- D. J. Barrett: Linux - Kapesní přehled, Computer Press, Praha, 2006
- M. Sobell: Mistrovství v RedHat a Fedora Linux, Computer Press, Praha, 2006
- M. Sobell: Linux - praktický průvodce, Computer Press, Praha, 2002
- E. Siever: Linux v kostce, Computer Press, Praha, 1999
- **Number of online sources...**

## Study materials and homeworks

- <http://kfa.mff.cuni.cz/linux>



- 1 UNIX systems, history, installation, basic applications
- 2 Structure of the Linux OS, file systems, hierarchy of the file system
- 3 Command line, shells, remote access (ssh, ftp)
- 4 Processes and their administration, basic system commands, packages, printing
- 5 Users, file and directory permissions
- 6 Work with files and directories, file compression, links, partition
- 7 Text-file processing commands, redirection, pipeline
- 8 Regular expressions
- 9 Command line based text editors
- 10 User and system variables, output processing
- 11 Scripts: basic construction, conditionals, loops, functions, automation
- 12 Networking, server-client services: http, (s)ftp, scp, ssh, sshfs, nfs
- 13 Programming in Linux (examples of Fortran, C/C++, Python), version control systems, documents in Latex

# File and Directory Manipulation

In theory, files and directories can be:

- **Deleted** - remove file(s) and/or directory(-ies) from the filesystem
- **Copied** - copy file(s) and/or directory(-ies) to different part of the filesystem
- **Moved** - move file(s) and/or directory(-ies) to different part of the filesystem = copy and delete the original
- **Linked** - link file(s) and/or directory(-ies) to different part of the filesystem
- **Created** - create a new file(s) and/or directory(-ies)

i.e. operations on whole files, not on part of the data in the files (next lesson)

Wildcards – useful constructions to perform actions on more than one file/directory

# File and directory manipulation

## Deletion

Removes files and directories from the FS.

- For files

```
rm [OPTION]... [FILE]...  
rm /my/file # one or many files  
rm -i files/in/my/dir/file1 files/in/my/dir/file2 # interactive  
rm -f /my/files # force removal
```

- For directories

```
rm -r [OPTION]... [DIRECTORY]...  
rm -r /my/directory # one or many directories  
rm -ri dirs/in/my/dir/directory1 dirs/in/my/dir/directory1 # interactive  
rm -rf /my/dir # force removal  
# combination of file(s) and directory(-ies)  
rm -r files/in/my/dir/file1 dirs/in/my/dir/directory1  
rmdir dir # for empty directories only
```

**There is no undelete!!!**

Once you delete with `rm`, your data are gone! `rm -rf /` when logged as 'root' will delete everything ;-)

# File and directory manipulation

## Copy

Copy file(s) and/or directory(-ies) to different part of the filesystem or to different name

- For files

```
cp [OPTION]... [-T] SOURCE DEST
cp /my/file /another/file # DEST is a file
cp /my/file /another/directory # DEST is a directory
cp /my/file1 /my/file2 /another/directory # copy many file to destination directory
cp -i file1 file2 # interactive, if file2 already exists, it asks if to overwrite
```

- For directories

```
cp -r /my/dir1 /another/dir
# combination
cp -r /my/file /my/dir1 /dest/dir
# if the DEST dir is the actual working directory ('pwd') then
cp -r /my/file /my/dir1 .
cp -r /my/file /my/dir1 /my/dir2 # if dir2 does not exists, it is created and it will have
# the same content as /my/dir1+/my/file
```



# File and directory manipulation

## Move

Moves file(s) and/or directory(-ies) to different part of the filesystem or to different name

- For files

```
mv [OPTION]... SOURCE... DEST
mv /my/file /another/file # DEST is a file
mv /my/file /another/directory # DEST is a directory
mv /my/file1 /my/file2 /another/directory # move many file to destination directory
mv -i file1 file2 # interactive, if file2 already exists, it asks if to overwrite
```

- For directories

```
mv /my/dir1 /another/dir
# combination
mv /my/file /my/dir1 /dest/dir
# if the DEST dir is the actual working directory ('pwd') then
mv /my/file /my/dir1 .
mv /my/file /my/dir1 /my/dir2 # if dir2 does not exists, it is created and it will have
# the same content as /my/dir1+/my/file
```



### Hard links

- hard links reference a physical(!!!) file location. Each file is actually a hardlink.
- `ls -l` 2nd column shows the number of links (at least 1!!!).
- Links have actual file contents
- Removing any link, just reduces the link count, but doesn't affect other links.
- We cannot create a hard link for a directory to avoid recursive loops.
- If original file is removed then the link will still show the content of the file.
- Command to create a hard link is:

```
ln [original filename] [link name]
ln /my/file1 /other/file2
```

# File and directory manipulation

## Links - Soft links

### Soft links

- A soft link or symlink is similar to the file shortcut feature which is used in Windows Operating systems. Each soft points to the original file but not to the physical location of the data. As similar to hard links, any changes to the data in either file is reflected in the other. Soft links can be linked across different file systems, although if the original file is deleted or moved, the soft linked file will not work correctly (called hanging link).
- `ls -l` command shows all links with first column value 'l' and the link points to original file.
- Soft Link contains the path for original file and not the contents.
- Removing soft link doesn't affect anything but removing original file, the link becomes 'dangling' link which points to nonexistent file.
- A soft link can link to a directory.
- If you want to link files across the filesystems, you can only use soft links.

```
ln -s [original filename] [link name]
ln -s /my/file1 /other/file2 # link file to different filename
ln -s /my/file /my/dir # link file to different directory
ln -s /my/dir /my/other/dir # linking directories
```

# File and directory manipulation

## Creating files and directories

### Files

- A new file is created by many ways, some of them were seen on previous slides, others will be shown during next lessons
- Commands cp, mv, ln create new files (a symbolic link is a file too)
- Using touch - a new empty file is created:

```
touch myfile
```

- Redirecting the standard output (stdout) and standard error output (stderr) to a file (see next lesson)

```
command > /my/file  
ls -l /etc > etc.content # the long listing of the  
                        # /etc directory is saved in file etc.content
```

- Created by running programs (Text Editors, Photoeditors, Games etc.)

### Directories

```
mkdir /my/dir # the dir /my MUST exist  
mkdir -p /my/dir1/dir2/dir3/dir4 # if dir1 (and so on) does not exist,  
                                # it will create parent dirs (-p)
```

# Wildcards

# Wildcards

## Standard Wildcards (globbing patterns)

Standard wildcards (also known as globbing patterns) are used by various command-line utilities to work with multiple files, especially useful with file/directory manipulation.

- ? (question mark) – this can represent any single character (i.e. only one character). If you specified something at the command line like "hd?" GNU/Linux would look for hda, hdb, hdc and every other letter/number between a-z, 0-9 etc.
- \* (asterisk) – this can represent any (including zero) number of characters. If you specified a "cd\*" it would use "cda", "cdrom", "cdrecord" and anything that starts with "cd" also including "cd" itself.
- [] (square brackets) – specifies a range a[a-c]d = aad,abd, acd, same as a[abc]d, [0-9] all numbers between 0 and 9. [A-Z]. etc.
- {} (curly brackets) – terms are separated by commas and each term must be the name of something or a wildcard, {abc, def, ghi} mean one of the strings inside
- [!]- [!0km] = any other character than 0, k or m
- Wildcards can be combined

```
ls -l /etc/??? # list all files/dirs in /etc with three character long names
cp -r ~/* /tmp # copy all files/dirs from the home directory to /tmp
# (~ is a shortcut for the home directory)
rm /etc/[nm]* # removes all config files from /etc starting with n or m
cp /tmp/[ab]{*.txt,*.doc,*.pdf} ~ # copy or files starting with a or b with
# txt, doc and pdf extension from
# the /tmp directory to the home directory
```

- Exercise 1: create directories `~/cviceni/{dnesni datum}` and copy all files starting with b or c from `/usr`
- Exercise 2: make symlink of all file ending with "t" in `~/cviceni/{dnesni datum}` in a new directory `~/cviceni/{dnesni datum}/t`
- Exercise 3: move all files from `~/cviceni/{dnesni datum}` to `~/tmp` and check the content of dir `~/cviceni/{dnesni datum}/t`
- Exercise 4: remove dir `~/cviceni/{dnesni datum}` and everything from `~/tmp`

# File and directory (de)compression

# File and directory (de)compression

Purpose: creating one file to replace many files and to reduce the size of files/directories  
Commands to be learnt: zip, tar, gzip, bzip2, 7z ...

- ZIP - similar to WinZIP

```
# Compression
zip [options] zipfile files_list
zip myfile.zip my_files
zip -r student_home.zip /home/student
zip -P password zipfile files_list # zipping with "password"
# Decompression
unzip myfiles.zip
unzip -l myfiles.zip # shows the zip content without unzipping
```

- TAR - create a tar archive (originally tape archive = tar). Merges whole directories into one file called 'tarball'. No compression!!!

```
# "Compression" - no compression in fact
tar -cvf myfilesdirs.tar /my/dirs /and/files # make a tarball from dirs and files
# -c means compress

# Uncompress - untar
tar -xvf myfilesdirs.tar # -x means eXtract
# does not compresses
tar -cvf downloads.tar Downloads/
ls -l downloads.tar
976005120
du -cb Downloads
975353493
```



# File and directory (de)compression

TAR - with (de)compression

Tar can be combined with compression to reduce the size of the tarball file. Using compression algorithms from GZIP and BZIP2 utilities

- TAR with GZIP

```
# Compression
tar -zcvf myfiles.tar.gz /my/files /and/my/dirs # the extension is .tar.gz or .tgz
# Decompression
tar -zxvf myfiles.tar.gz
tar -tvf myfiles.tar.gz # view contents
```

- TAR with BZIP2

```
# Compression
tar -jcvf myfiles.tar.bz2 /my/files /and/my/dirs # the extension is .tar.bz2 or .tbz2
# Decompression
tar -jxvf myfiles.tar.bz2
tar -tvf myfiles.tar.bz2 # view contents
```

- gzip and bzip2 can be used directly for files

```
# Compression
gzip /my/file # creates file.gz
bzip2 /my/file # creates file.bz2
# Decompression
gunzip file.gz
bunzip2 file.bz2
```

# File and directory (de)compression

7z, RAR, xz

- 7z - From man pages "7-Zip is a file archiver with the highest compression ratio. Supports LZMA, LZMA2, XZ, ZIP, Zip64, CAB, RAR (if the non-free p7zip-rar package is installed), ARJ, GZIP, BZIP2, TAR, CPIO, RPM, ISO, most filesystem images and DEB formats. Compression ratio in the new 7z format is 30-50% better than ratio in ZIP format."

```
# Compression
7z a dir.7z /my/dir
# Decompression
7z e dir.7z
```

- RAR and UNRAR

```
# Compression
rar a myfiles.rar /my/files /my/dirs
# Decompression
unrar x myfiles.rar
```

- XZ - general-purpose data compression tool with command line syntax similar to gzip and bzip2, contained by 7z

```
# Compression
xz /my/files
# Decompression
xz -dk files.xz
```

- Exercise 1: Archive all files and directories in your home folder starting with . (dot) with tar+gzip
- Exercise 2: make a tarball from /etc directory - will it work for all the files
- Exercise 3: Compare the sizes of gzip, bzip2, zip, 7z, rar and xz for this file [http://meop3.troja.mff.cuni.cz:8010/linux/netcdf/PHA03\\_STS.2007-2011DJF.nc](http://meop3.troja.mff.cuni.cz:8010/linux/netcdf/PHA03_STS.2007-2011DJF.nc)
- Exercise 4: How to use 7z with password? Is it safe?

# Disk/partition/filesystem handling utilities

# Basic disk/partition handling utilities

Files/directories are structures physically or virtually written into harddrives/flashdrives, cdrom/dvdroms, tapes, virtual memory etc. These are usually divided into partitions which enables to separate logically distinct parts of the filesystem hierarchy structure. For example /home will be placed on different partition than /.

- Each partition has its own filesystem (FS)
- Partitions are 'mounted' to directories
- `df` – report file system disk space usage. It shows which partition is 'mounted' to which directory.
  - `/dev/sda`, `/dev/sdb`, `/dev/sdc` are 'device' files corresponding to individual devices
  - `/dev/sda1`, `sda2` mean the partition number on the disk (i.e. each partition has a separate device file in `/dev`)
  - `tmpfs` is a temporal filesystem created in the RAM, so it behaves as a normal `hrddisk`, but when the computer is turned off, it is gone.
  - `df -Th` shows the type of the filesystems mounted (`ext2`, `ext3`, `ext4`, `reiserfs`, `vfat`, `ntfs` etc).
- `lsblk` – very useful utility to show the disk/partitions structure/filesystems and IDs of partitions
- `hdparm` – get/set SATA/IDE device parameters. Tool to use when it comes to tuning your hard disk or DVD drive, but it can also measure read speed, deliver valuable information about the device, change important drive settings, and even erase SSDs securely.

```
lsblk -f # get complete information about the disks/partitions/filesystems and disk UUIDs (see /etc/fstab)
hdparm -I /dev/sda # For all kind of information about the SATA/IDE disk
```

# Basic disk/partition handling utilities

fdisk (partitioning) and filesystem creation

- **FDISK** - manipulate disk partition table. A very powerful utility to create, modify and delete partitions on a disk. **USE WITH CAUTION AS YOU CAN EASILY DAMAGE YOUR EXISTING FILES/DIRECTORIES!!!** More info on: [https://www.tldp.org/HOWTO/Partition/fdisk\\_partitioning.html](https://www.tldp.org/HOWTO/Partition/fdisk_partitioning.html) or `man fdisk`.

```
fdisk -l # shows all disks connected to the computer (needs root privileges)
fdisk /dev/sdb # prompt to manipulate with disk /dev/sdb, press m to get all the commands
```

- Once partitions are created on a disk, we can create different filesystems (Linux File System list <https://static.javatpoint.com/linux/images/linux-file-system2.png>, [https://en.wikipedia.org/wiki/List\\_of\\_file\\_systems](https://en.wikipedia.org/wiki/List_of_file_systems) `mkfs.filesystem`

```
mkfs.ext4 /dev/sdb1 # create Ext4 filesystem on disk /dev/sdb and partition 1.
```

- **fsck** - check and repair a Linux filesystem

```
fsck.ext4 /dev/sdb1 # check and repairs the ext4 filesystem on disk /dev/sdb and partition 1.
```

# Basic disk/partition handling utilities

## Mounting filesystems

In order to access the files/directories on the partitions, this partition has to be 'mounted'.

- mount - mount a filesystem

```
mount -t type device dir # The standard form of the mount command
# type is the filesystem type, device is the device file in /dev and dir is the directory where to mount
mount /dev/sdb1 /data # -t type may be omitted as the system will find it out automatically
mount -o ro /dev/sdb1 /data # "-o" = adding mount options, ro = read-only
```

- mounting ISO filesystem (CD/DVD \*.iso image)

```
mount /path/to/image.iso /media/iso -o loop
```

- Mounting NFS - network filesystem

```
mount -t nfs 10.76.120.40:/volume1/d01 /home/nas01
```

- tmpfs - temporal/virtual filesystem represented in the RAM

```
mount -F tmpfs -o size=1G swap /mount/tmp # this takes 1G from the RAM and allocate it to /mnt/tmp
```

- unmounting filesystems

```
umount /mount/point
```

# Basic disk/partition handling utilities

The `/etc/fstab` file

The `/etc/fstab` file is a system configuration file that contains all available disks, disk partitions, virtual disks and their options. Each file system is described on a separate line. Each line contains six fields separated by one or more spaces or tabs.

The `mount` command reads each line at system boot and mounts the disks according to the options given on the lines.

```
# <file system><mount point><type><options><dump><pass>
# filesystem root disk
UUID=f3b12014-faea-49c8-9e9a-f397b002454d / ext4 errors=remount-ro 0 1
# boot partitions
UUID=515B-6053 /boot/efi vfat umask=0077 0 1
UUID=141ad68b-b7f4-4074-a89d-43801222d5fb none swap sw 0 0
# 2T data disk
UUID=6f3ffadf-418f-4e15-baaf-93ca06ffaf54 /data ext4 errors=remount-ro 0 0
# NAS1 and NAS2
10.76.120.40:/volume1/d01 /home/nas01 nfs defaults 0 0
10.76.120.40:/volume2/d02 /home/nas02 nfs rw 0 0
# cesnet high capacity storage center Jihlava
sshfs#huszarpet@ssh.du4.cesnet.cz:/tape_tape/archive/V0_cuni_mff_meteo/home/huszarpet \
/mnt/cesnet fuse defaults,imap=user,IdentityFile=/home/huszi/.ssh/id_rsa,allow_other 0 0
```

- Disks can be specified either by the device file e.g. `/dev/sda2`, however, this is not uniq and can change if disk are connected to the computer in a different order.
- A much safer option is to specify the disks by their UUID number (`lsblk -f /dev/sda2`)
- Further options: type = FS type; options = mount options (like ro for read only); dump = backup operation with 'dump' (0/1); pass = FS check with fsck (root always 1, other 2, nocheck = 0)
- to mount the "lines" of fstab (when changes made) use `mount -a`.





# Excercise - disk/partition handling utilities

A simple but frequent example

Suppose we got an old (or new) disk and want to format it as a single partition to the `ext4` filesystem, then mount it to `/mnt/mydisk`.

- `fdisk` to repartition (delete old partitions)
- `mkfs.ext4` to create a FS
- mount it (for one time use) and permanently via `fstab`