Packages (Ubuntu/Debian based distibitions)

APT system to download and install packages from repositories

- `apt-get`, `aptitude`: commands to run the tasks (install, remove, update package list)
- `/etc/apt`: configuration of the packaging system
- Temporary files (package list, downloaded installation files) in `/var/lib/apt/lists` and `/var/cache/apt` files
- Dependencies are flagged as auto-installed (`/var/lib/apt/extended_states`
- Simulate action (complicated upgrades): `apt-get -s`

DPKG is used to handle already installed packages:

- `-i`: install local package
- `-r`/`-P`: remove/purge installed package (purge = remove also config files)
- Allows to install even conflicting packages (`--force-depends`, `--force-conflicts`): use with care !
- Packages can ask for configuration options during installation process, reinvoke these questions by `dpkg-reconfigure` command
- Packages can provide "alternatives" for similar tasks: `update-alternatives` command to handle these selections

Linux on net is also target of attacks, especially through the ssh connection

- `sshguard`: service that stops ssh login after several failed attempts
- `rkhunter`: complex check of the most important system utilities
- `chkrootkit`: simple check of most known "viruses" in the system utilities
- `logcheck`: scans log files and reports suspicious activities

Many of these commands are set to run regularly and send reports by e-mail to the system admin (see `/etc/aliases` file)

Exercise 1

Write list of commands ala simple script, which will:

- List content of the /etc directory
- List file info of /etc/passwd (size, access rights etc.)
- Copy the file /etc/passwd to your home (hint: using cp /etc/passwd ~/passwd.copy)
- Add new user named "ukolvT11"
- Check by ls command that the file /etc/passwd has changed size etc.
- Use diff command to show difference between the /etc/passwd and ~/passwd.copy
- Change "shell" of the new user from /bin/bash to /usr/bin/zsh (hint: chsh)
- Change access rights of the ~/passwd.copy file so that no-one can write into the file and only the owner can read it (nor group, nor others)
- Change the ownership of the file to the superuser root and group root
- Check by ls command that you can't read this file now
- Remove the recently added user "ukolvT11"
- Check by diff command that the /etc/passwd and ~/passwd.copy are same again

Make sure the script has a header that to be run in shell /bin/bash. Use sudo on appropriate lines (not all operations above will be allowed to you as an ordinary user).

File and Directory Manipulation

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

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 ;-)

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
```

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
```

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 localtion 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
```

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 exists,
                                 # it will create parent dirs (-p)
```

Wildcards

Standard wildcards (also known as globbing patterns) are used by various command-line utilities to work with multiple files, especially usefull 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
- [!]– [!0] = any other character than 0
- 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
```

File Manipulation/Wildcards - Excercises

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

File and directory (de)compression

File and directory (de)compression

Prupose: 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
```