C2110 *UNIX and programming*

3rd lesson

File system

Petr Kulhánek

kulhanek@chemi.muni.cz

National Centre for Biomolecular Research, Faculty of Science,
Masaryk University, Kamenice 5, CZ-62500 Brno

Contents

- **Revision**
  - commands, documentation, ssh, homework
- **File system**
  - structure, comparison with the MS Windows, cluster WOLF, paths, wild characters, commands, search for files
- **Permissions**
  - POSIX permissions, user identity, commands
- **Homework**
  - practicing commands

Revision

- commands, documentation
- ssh
- homework

Manual, Description of commands

Manual pages (what to do if you do not know):

```
$ man [section_number] topic
```

Command description:

```
$ command [options] [--] [arguments]
```

expansion/change of command behaviour
options can be given in any order

terminate option specification, useful only in very specific cases.

arguments
main data or information given to a command
arguments must be given in a specific order

[] **optional** arguments or options
<> **mandatory** arguments or options

brackets are not typed

Remote login

Several possibilities exist for a remote login (rsh, XDMCP, etc.) but the most often used and **safest** is the Secure shell **(ssh).**

Syntax:

```
$ ssh [user@]hostname [command]
```

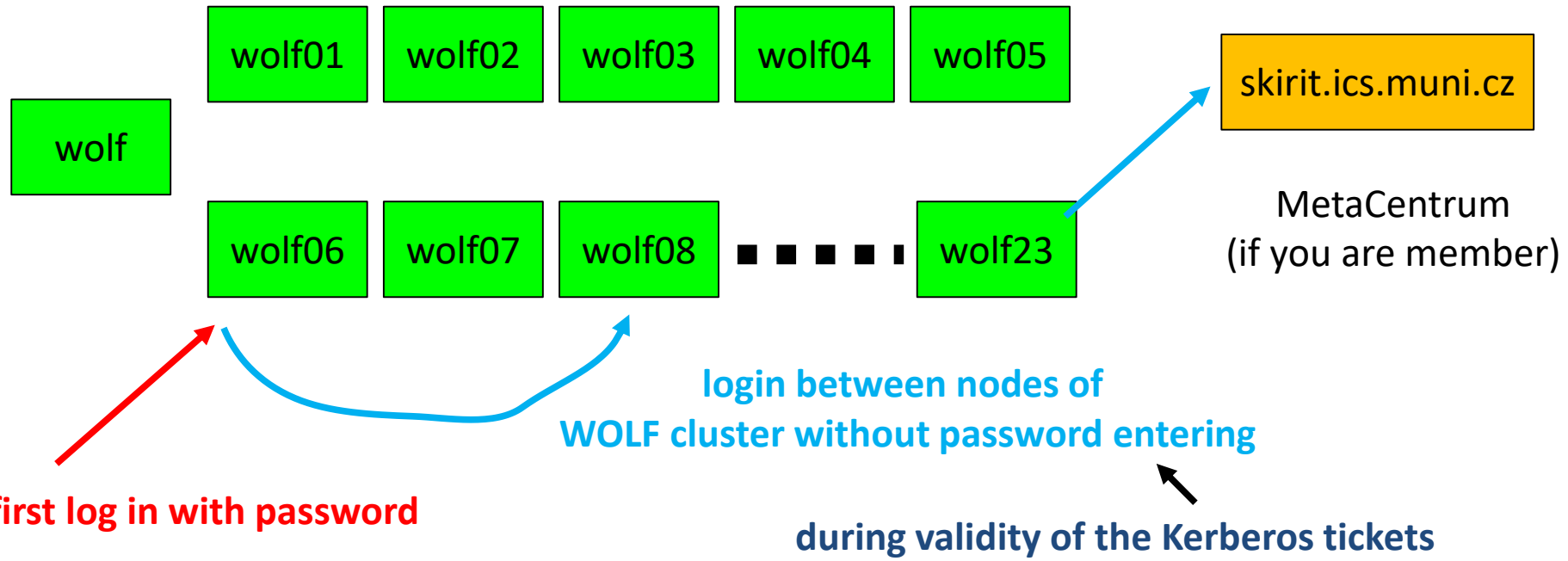
an user name;
if not provided, name of the logged user is used

machine name

a command to execute; if not provided, interactive connection is established

Logout:
Remote interactive sessions are terminated by the **exit** command.

Kerberos – passwordless login



Authentication Kerberos:

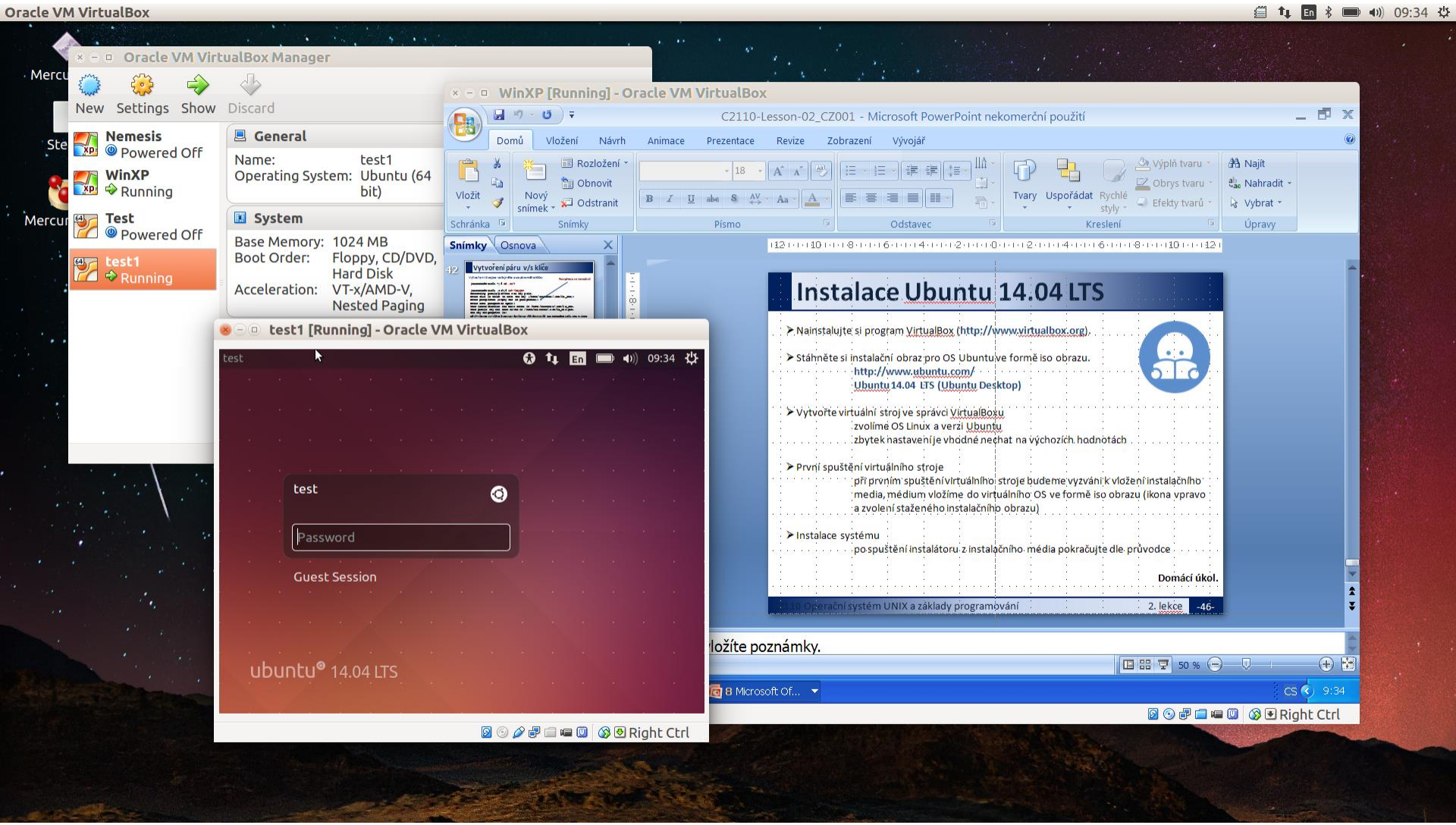
Advantages:

- do not have to continually enter your password
- safer usage of ssh and scp commands in scripts
- faster work

Disadvantages:

- when one computer is compromised, all computers are compromised at least for the duration of Kerberos ticket (if the attacker stole password)

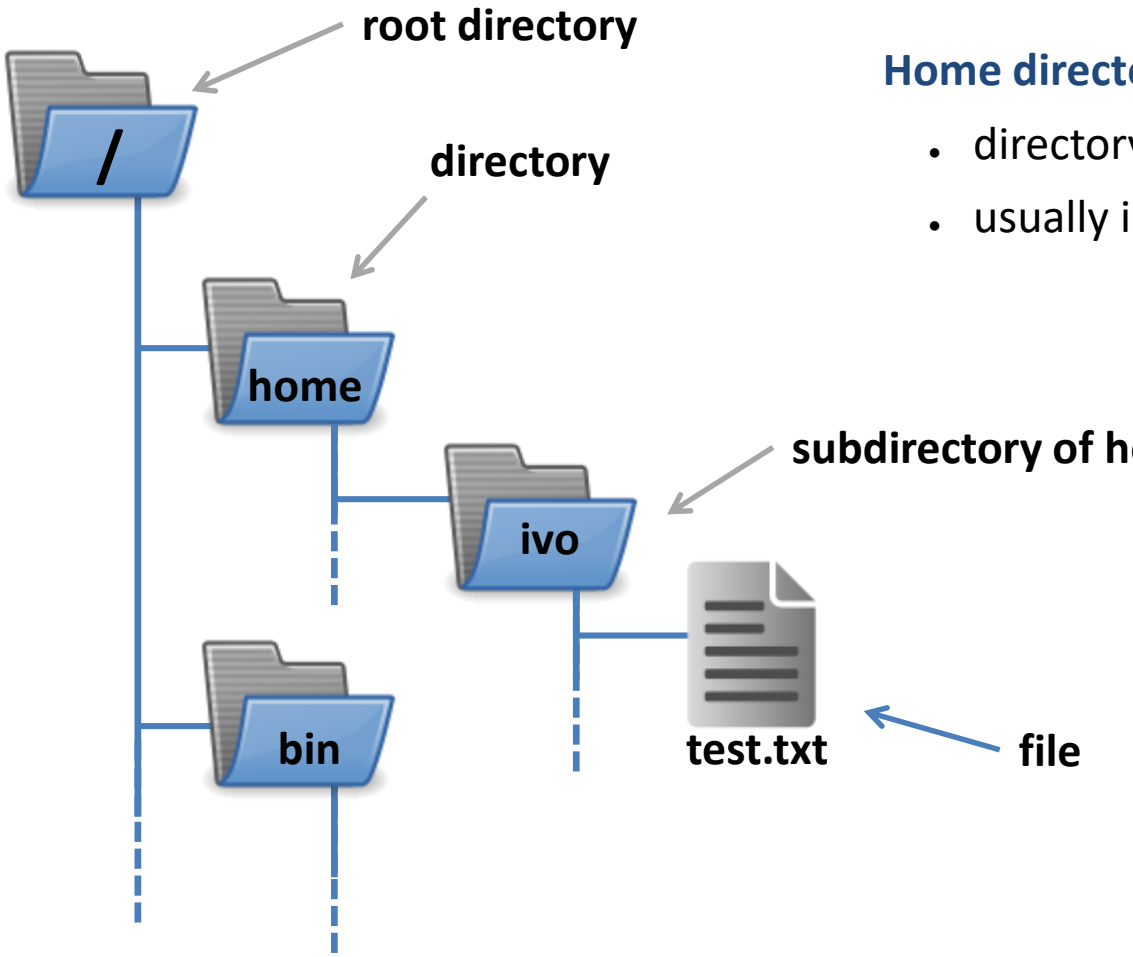
Ubuntu OS


Oracle VM VirtualBox
09:34
Oracle VM VirtualBox Manager
New
Settings
Show
Discard
Nemesis
Powered Off
WinXP
Running
Test
Powered Off
test1
Running
General
Name: test1
Operating System: Ubuntu (64 bit)
System
Base Memory: 1024 MB
Boot Order: Floppy, CD/DVD, Hard Disk
Acceleration: VT-x/AMD-V, Nested Paging
WinXP [Running] - Oracle VM VirtualBox
C2110-Lesson-02_CZ001 - Microsoft PowerPoint nekomerční použití
Instalace Ubuntu 14.04 LTS
Nainstalujte si program VirtualBox (http://www.virtualbox.org).
Stáhněte si instalační obraz pro OS Ubuntu ve formě iso obrazu.
http://www.ubuntu.com/
Ubuntu 14.04 LTS (Ubuntu Desktop)
Vytvořte virtuální stroj ve správci VirtualBoxu
zvolíme OS Linux a verzi Ubuntu
zbytek nastavení je vhodné nechat na výchozích hodnotách
První spuštění virtuálního stroje
při prvním spuštění virtuálního stroje budeme vyzváni k vložení instalačního media, médium vložíme do virtuálního OS ve formě iso obrazu (ikona vpravo a zvolení staženého instalačního obrazu)
Instalace systému
po spuštění instalátoru z instalačního média pokračujte dle průvodce
Domácí úkol.
2. lekce
-46-
test1 [Running] - Oracle VM VirtualBox
test
Password
Guest Session
ubuntu® 14.04 LTS
Right Ctrl

File system

File system structure

UNIX uses a **hierarchical file system** composed from directories (folders) and files. All directories and files are located in a **single root directory (/)**.



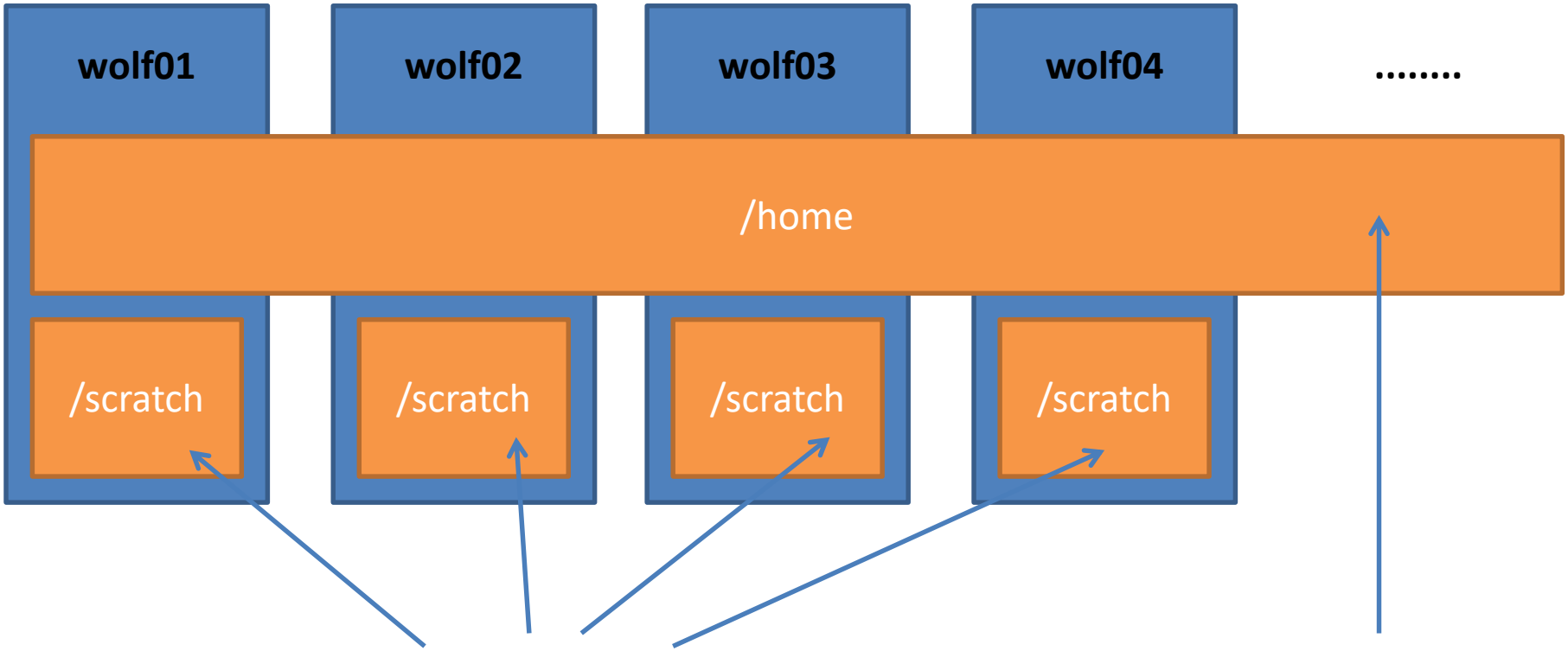
Home directory:

- directory used for **user data and settings**
- usually in the path **/home/username**

Comparison with MS Windows

Properties	Linux (ext2/ext3/ext4)	MS Windows (FAT32,NTFS)
Partitions	No Partitions are attached as directories.	C:, D :, etc. possible to attach as directory (NTFS).
Names	Names distinguishes between uppercase and lowercase letters (Case sensitive)	Does not distinguish between uppercase and lowercase letters (case insensitive).
Separators	Slash	Backslash
Permissions	Yes POSIX	Yes (only NTFS) ACL
Devices (hardware)	As special files	No

File system - cluster WOLF

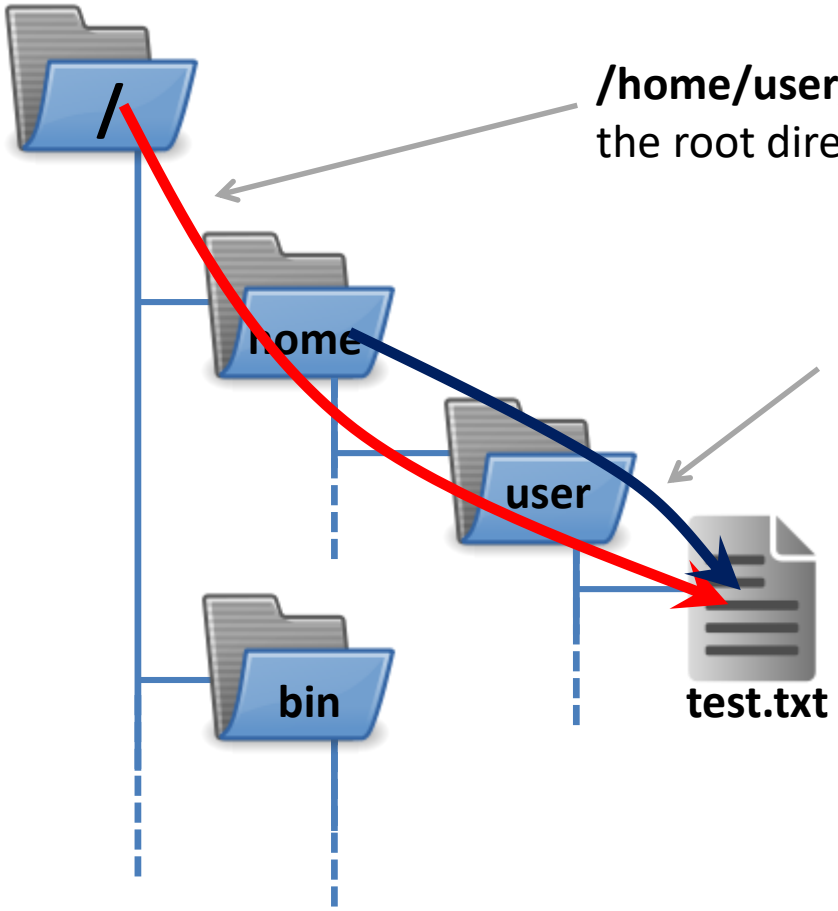


Different content on each node.
Data in scratch **can be deleted at any time** without previous notice (**No backup**).
Data capacity is **not limited** by any quota.

Shared content on each node of the cluster.
Data backup is accessible in form of snapshots from directory:
/backup/<date>/WOLF/wolf.ncbr.muni.cz/home
Data capacity is limited by **1.5 GB** quota per user.

Identification of directories and files

The path to the directory or file can be specified as **absolute** or **relative**. Names of directories and files are separated by **slash /**.



/home/user/test.txt - absolute path to file test.txt, starts in the root directory

user/test.txt - relative path to file test.txt, starts in the current directory (in this case it is a directory /home)

Path types

An absolute path always start in the root or home directory.
It must begin with either a slash **/** or tilde **~**.

/home/kulhanek/Documents/domaci_ukol.txt

Usage of tilde:

~	home directory of logged user
~username	home directory of user **username**

A relative path starts in the current/working directory. (Absolute path to the working directory can be obtained by the **pwd** command.)

../alois/Documents

Names of special directories:

. (dot)	current directory
.. (two dots)	parent directory

Special characters in filenames

Special characters (wildcards, wild characters) in filenames or directory:

***** - anything in the name (but no hidden files)

? - one character in the name

[] - range (one character) in the name, eg. [ajk], [a,j,k], [a-j]

Expansion of special character is executed by **shell** (command line environment) **before the execution** of a command. Expansion can be prevented by giving the file name in quotation marks or using a backslash before a special character. In this case, the expansion can be performed in the second round by an executed commands (e.g., the find command).

Examples:

```
$ cp *.pdf Documents/
```

copies all pdf file in the current directory to subdirectory Documents

```
$ rm *
```

deletes all files in the current directory (except directories)

```
$ mv A? Tmp/
```

moves files, which a name begin with the letter "A" and contain to characters to the "tmp" directory

Special characters in filenames

Expansion of special characters will be executed only if there is at least one file or directory that fits the template.

Exercise:

echo command prints given arguments.

```
$ cd
$ echo D*
Desktop Documents Downloads
$ echo A*
A*
$ echo "D*" D\* "D\*"
D* D* D\*
$ echo "D\"" "D\\"
D" D\
```

escape sequence (\) man bash: quoting

It is possible to use wild characters in directory and/or file names at **the same time**:

Examples:

$ cp ~/task[1,4,5]/*.pdf Documents/

copies all pdf documents from subdirectories task1, task4 and task5 from home directory to subdirectory Documents

Basic commands

File system:

cd	**changes the current working directory**
pwd	**displays the absolute path to the current working directory**
ls	**lists of directory content**
mkdir	creates directory
rmdir	deletes a directory (must be empty)
cp	copies file or directory
mv	moves a file or directory
rm	removes file or directory
find	searches for files or directories

"**Golden triple**" I have to know how to get there (**cd**), where I am (**pwd**), and what it contains (**ls**)

Helpful Hints:

$ cd sets the home directory as the working directory

It is useful **to open at least two terminals** (in source and destination directories) and control directory content before and after the operation.

Directory creation

- **Create directory**

```
$ mkdir directory
```

- **Create nested directories**

```
$ mkdir -p dir_name1/dir_name2/dir_name3
```

-p (parents) option makes parent directories if they do not exists

Copy

- **to copy file or dictionary use command cp**

```
$ cp file1 file2
```

creates copy of the file "file1" called "file2“

```
$ cp file1 file2 file3 directory1/
```

copies the files "file1", "file2", "file3" into the directory “directory1"

```
$ cp -r directory1 directory2
```

creates copy of the directory “directory1" as “directory2", if the directory “directory2" already exists, it creates a copy of the directory “directory1" as subdirectory of “directory2"

```
$ cp -r file1 directory2/ file3 directory1/
```

copies files “file1", “file3" and directory “directory2" into directory “directory1"

-r option (recursive) must be used when copying data containing subdirectories

Move

- **To move or rename file or directory use command mv**

```
$ mv file1 file2
```

renames the file "file1" to "file2“

```
$ mv file1 file2 file3 directory1/
```

moves the files "file1", "file2", "file3" into the directory “directory1"

```
$ mv directory1 directory2
```

renames the directory “directory1" to “directory2", if the directory “directory2" already exists, it moves the directory “directory1" to “directory2"

```
$ mv file1 directory2/ file3 directory1/
```

moves files “file1", “file3" a directory “directory2" to directory “directory1"

Remove

- **to remove dictionaries or files use command rm**

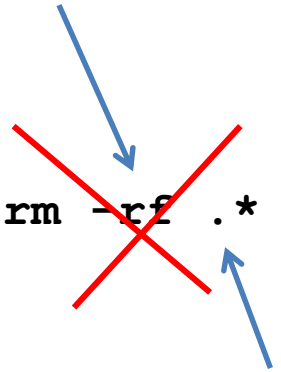
```
$ rm file1
```

removes file “file1"

```
$ rm -r directory1
```

removes directory “directory1"

recursion (r) and without asking (f - force).



`.* -> ..` (removes also parent directories)

Hidden files and directories

Names of **hidden** files or directories **start with a dot**. They are normally invisible for an user. But it is possible to list them using the command **ls -a**. Special characters *, ? and [] in the typical usage do not include hidden files.

Hidden files contain a system configuration, do not delete or change them and if you are not sure what you are doing.

Format of command line prompt

Appearance of command line prompt can be changed. For this purpose is used system variable PS1 (man bash). If the current format does not suits you (displays the name of the last directory from the path), you can change the appearance as follows:

Default settings:

```
$ PS1="[\u@\h \W]$ "
```

Modified settings:

```
$ PS1="[\u@\h \w]$ "
```

lowercase and uppercase w

Changed setting will show **the full path** to the working directory.

Changes affects only the terminal, where they is made. The changes are possible to make permanent (the command have to be inserted at the end **of the hidden file ~/.bashrc** on a separate line.) To change content of the file, use a text editor **gedit** or **kwrite**. Changing the settings is shown in the newly opened terminals.

Exercise I

1. Download from IS the study materials to the **~/Downloads** directory.
2. Create a directory **test** in the scratch directory **/scratch/username**
3. Create a **studmat** directory in your home directory.
4. Copy study materials from the **~/Downloads** directory to the **studmat** directory**.**
5. Open the presentation (**Lesson 02**) in the program **okular**
6. Copy the presentation to the **/scratch/username/test** directory.
7. In the directory **/scratch/username/test**, rename the presentation to **lesson2.pdf**
8. Open the **lesson2.pdf** presentation in okular
9. Log in to the wolf01 workstation. Why is not there the **lesson2.pdf** file in the **/scratch/username/test**?
10. Remove the presentations from the **~/Downloads** directory.
11. Remove the directories **test** and **studmat**

username – your username

Exercise to use:

- autocomplete (TAB)
- simplified copying
  - select with the left mouse button
  - insertion with the middle mouse button (wheel) of mouse
- command line history

File search

For file search, use command **find**

If not specified, the search is executed in a current directory

```
$ find [where] what
```

Search is recursive (default state)

Query (**what**) is composed from sub-queries, that can be chained using logic operators.

Basic queries:

-name *pattern*	finds all files that match the pattern pattern can contain special characters: *,?, [] **(for usage of special characters, pattern have to be in quotation marks)**
-type *c*	finds all files of type c (file, directory, etc. for list of types see. man find)

Logical operators:

-and	pattern have to match both left and right query in same time
-or	pattern have to match at least one of the queries

File search, examples

```
$ find /home/ -name '*.txt'
```
Finds all files with suffix .txt in directory /home/

```
$ find ~kulhanek -name '*.txt' -or -name '*.hpp'
```
Finds all files with suffix .txt or .hpp in directory /home/kulhanek

```
$ find -name 'D*' -and -type d
```
Finds all subdirectories, which start by D letter, in current directory

Exercise for homework

Permissions

Permissions

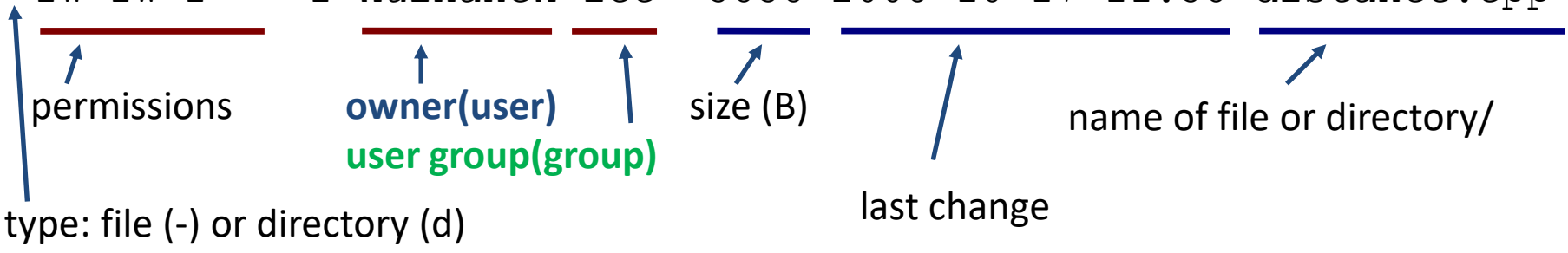
Permissions determine what operations a user can perform with file or directory in the file system.

Permissions:

r	to read file	to list of dictionary contents
w	to modify file	to modify the directory contents (create or delete a file or directory)
x	to execute file	to enter to the directory

For each file or directory, the owner and user group are set. The three separate permission types exist: **file owner (u)**, **user group (g)** a **other user (o)**.

```
$ ls -l
  u   g   o
drwxrwxr-x  3 kulhanek lcc  4096 2008-10-13 09:57 bin/
drwx------  2 kulhanek lcc  4096 2008-10-13 09:58 Desktop/
-rw-rw-r--  1 kulhanek lcc  5858 2008-10-17 11:58 distance.cpp
```



Permission evaluation procedure

```
$ ls -l
  u   g   o
drwxrwxr-x   3 kulhanek ncbr   4096 2008-10-13 09:57 bin/
drwx------   2 kulhanek ncbr   4096 2008-10-13 09:58 Desktop/
-rw-rw-r--   1 kulhanek ncbr   5858 2008-10-17 11:58 distance.cpp
```

permissions — owner (user) — user group (group)

Access of the user to the file or the directory:

1) username is consistent with the owner of the file, access is handled by permissions for the owner
2) user is a member of the group, access is handled by permissions for group
3) user is among the other users, access is handled by permissions for others

Order of permission evaluation for a given entity (directory or file)

When user accesses a file or directory specified by path, the rules above are applied gradually from root directory:

/home/user/test.txt

Order of permission evaluation

Default setting and its change

Created files or directories:

- are owned by the user who creates them
- user group is set as primary group of the owner at the time the file or directory was created or access group of parent directory, if the active flag set-group-ID
- default permission are given by the mask, which is set by command **umask**

Some commands or applications can have different default policies (eg. ssh-keygen and permissions for the private key).

Change:

- of the owner of the file is done only by root (**chown**)
- of the user group is done by file owner to his groups or by superuser to any group (**chgrp**)
- of permissions are done by the file owner or the superuser (**chmod**)
- of the mask by **umask** command is done by user, for a lasting effect is necessary to paste the command to your **~/.bashrc**

User and group indentity

Identity of the user and his **groups** is obtained by command **id:**

```
[kulhanek@wolf01 ~]$ id
uid=18773(kulhanek) gid=10000(ncbr) groups=10000(ncbr),10001(students),...
```

Username (login) and its numeric representation

primary user group to which the user is assigned and its numeric representation

rest of user groups to which the user is assigned and their numeric representation

Assigning user into the primary and other groups can be **changed only by the superuser**.

Users included in the group is displayed by **getent**:

```
[kulhanek@wolf ~]$ getent group students
students:x:10001:ambrozkl,balcon,claud145,david1992,djaron,dubak, .... (kráceno)
```

groupname

numerical representation

list of users (logins) in group separated comma

Command getent can also be used for other queries, eg. to list all system users (getent passwd).

Permission modification

Permissions of files and directories can be changed by the file's owner or by superuser with **chmod** command

```
$ chmod permissions file1 [file2 ...]
```

Permissions:

r	to read file	to list of dictionary contents
w	to modify file	to modify the directory contents (create or delete a file or directory)
x	to execute file	to enter to the directory
X	sets permission to execute a file that already has that right in another group of rules and always for directory (useful for recursive change of permissions)	

Groups of permissions:

u	owner (user)
g	user group (group)
o	other users (other)
a	all users (all), permission is applied to u,g,o

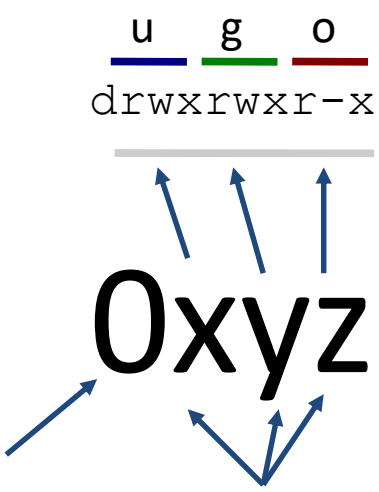
Příklad:

```
$ chmod u+x,g-w soubor
```

Add (+) permission to execute for the owner and remove (-) permission to write for group

Change of permissions

Permissions in octal notation:



zero (prefix of octal entry)

Sum of octal values for each group of permissions

Permission	Octal value
r	4
w	2
x	1

Exercises:

```
rwxrwxr-x      0775
r---w---x      0421
rwxr-x---      0750
```

Group change

User group of files and directories is changed by the owner or the superuser by command **chgrp**. The owner can only use the groups to which he belong (can list wiht **id**).

```
$ chgrp group_name file1 [file2 ...]
```

```
[kulhanek@wolf ~]$ id
uid=18773(kulhanek) gid=10000(ncbr) groups=10000(ncbr),10001(students),...
```

```
[kulhanek@wolf ~]$ ls -ld Documents/
drwx------ 3 kulhanek ncbr 4096 Sep 19 11:43 Documents/

[kulhanek@wolf ~]$ chgrp students Documents/

[kulhanek@wolf ~]$ ls -ld Documents/
drwx------ 3 kulhanek students 4096 Sep 19 11:43 Documents/
```

group change

Mask setting

Default permissions are set up using mask, that is set by command **umask**. The current setting of mask is possible to display by command umask without any argument. (Documentation: man bash, SHELL BUILTIN COMMANDS)

Default permissions for:

files are 0666
directories are 0777

Mask specifies permissions that **are removed from the default permissions** before permissions setting for created file or directory

For example: the default **mask for the cluster WOLF, which is 0077**, leads to the following permissions:

for file 0600
for directory 0700

Change of masks can be done by insertion of **umask** command at the end of the file **~/.bashr**c or settings made by **ams-config** command (Infinity environment installed on cluster WOLF).

List of commands

File system (permissions):

id	prints lists of groups, where is user included and user primary group
getent	prints information about users, user groups and other information
umask	default permissions for newly created files or directories
chmod	changes permissions for directory or file
chgrp	changes access group of users for directory or file
chown	changes owner of file or directory

Posix ACL (access control list):

setfacl	sets ACL
getfacl	prints actual ACL

NFSv4 ACL:

nfs4_setfacl	sets ACL
nfs4_getfacl	prints actual ACL
nfs4_editfacl	edits ACL

Detailed information in C2115

Exercise II

1. Create a subdirectory **Data i**n the directory **/scratch/username**.
2. What are permissions for the **Data** directory?
3. Change the access group for the **Data** directory to '**students**'.
4. Insert two files into the **Data** directory (e.g., two presentations for the C2110 course). What are permissions for these files?
5. Can your colleague open files by the **okular** program? Explain.
6. Grant permissions to your colleague to open the files.
7. Deny reading for one file to your colleague. Can he/she open the file? Can he/she access the second file?
8. Grant access to modify contents of the **Data** directory. Can your colleague delete files in that directory?
9. Delete the **Data** directory.
10. Restores default permissions for the **/scratch/username** directory.

Work in pairs

Conclusion

- Linux uses **a hierarchical file system**. Names of file and directories are **case sensitive**. **Slash** is used to separate directories and files.
- Access to files and directories is managed by **access permission** at a relatively coarse level, but it is sufficient for routine work.

Homework

- **practicing commands**

Homework

1. 1. In your home directory, create a directory Data
2. 2. Copy content of directory /home/kulhanek/Documents/C2110/Lesson03 to Data directory including subdirectories.
3. 3. Find all files with suffix .cpp, which are located in directory Data (print filenames on the screen)
4. In the directory / cratch/username, create a directory Headers
5. Copy all files with suffix .h to Headers from directory /home/kulhanek/Documents/C2110/Lesson03/dev/src
6. Move all files with suffix .cpp to Headers from directory /home/kulhanek/Documents/C2110/Lesson03/dev/src

   What really happens and why?
7. What is size (in B and kB)of file /home/kulhanek/Documents/C2110/Lesson03/dev/src/GraphicsSetup.cpp
8. In directory Headers, delete all files starting with word Graphic and with suffix .h
9. Remove directory Headers