bw | HPC – C5

# Linux: Basic Concepts and Command Line Usage
# Unit 1

Funding: Baden-Württemberg
MINISTERIUM FÜR WISSENSCHAFT, FORSCHUNG UND KUNST

# Outline

- Interactive use
- File System Paths
- Environment variables
- Search Patterns
- Redirection, pipes

Overall goal: give you a starting point on most common tasks

# Interactive Use

- Copy and Paste (not like in Windows):
- Copy:

  mark the text with mouse (auto-copy) or CTRL+Insert
- Paste to cursor:

  middle button or Shift+Insert, Putty: right button

- Command line history
  - Up/Down arrows
  - history | less
  - Ctrl+r: reverse incremental search in command history

# Documentation (see Cheat Sheet)

- Man(ual) pages:

  `man "command"` (e.g. `man cp`)

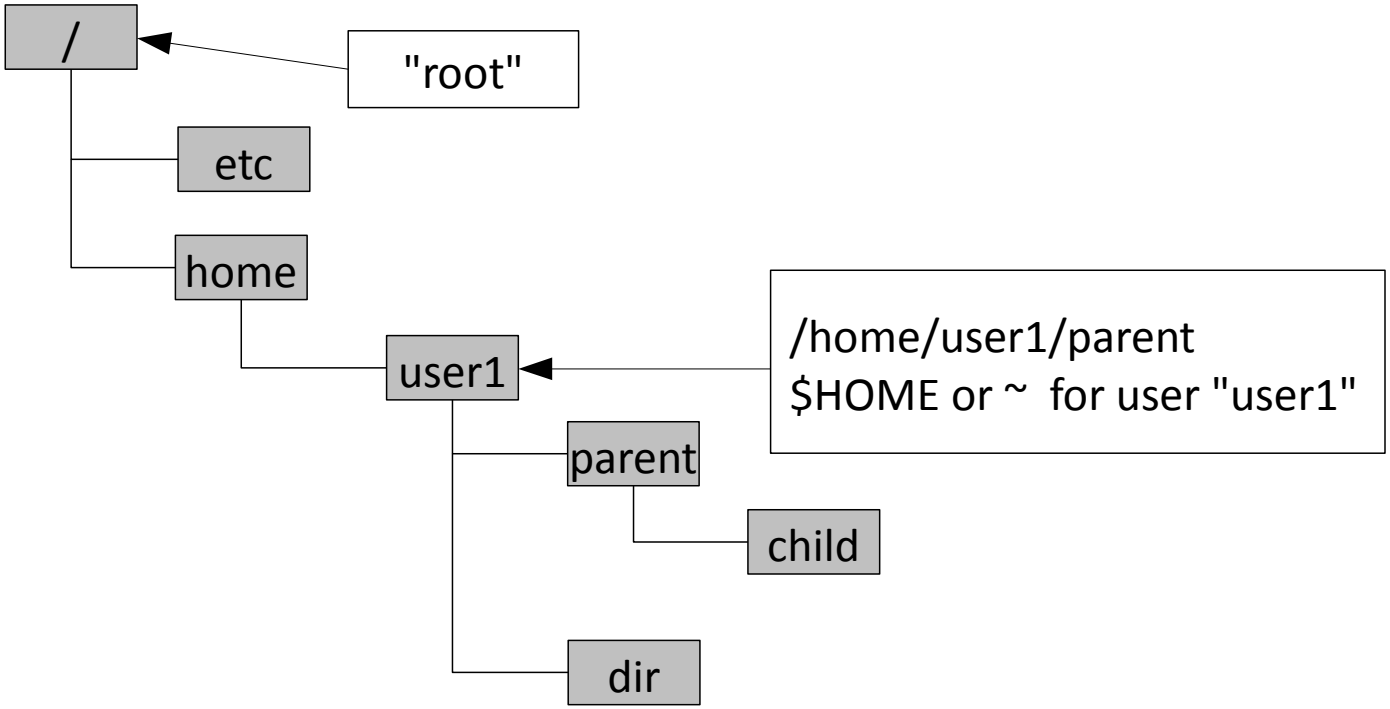
  You usually **search** in man pages
- info pages

  `info command`
- Help from the command:

  `"command" --help` (e.g. `cp –help`)
- bash internal commands

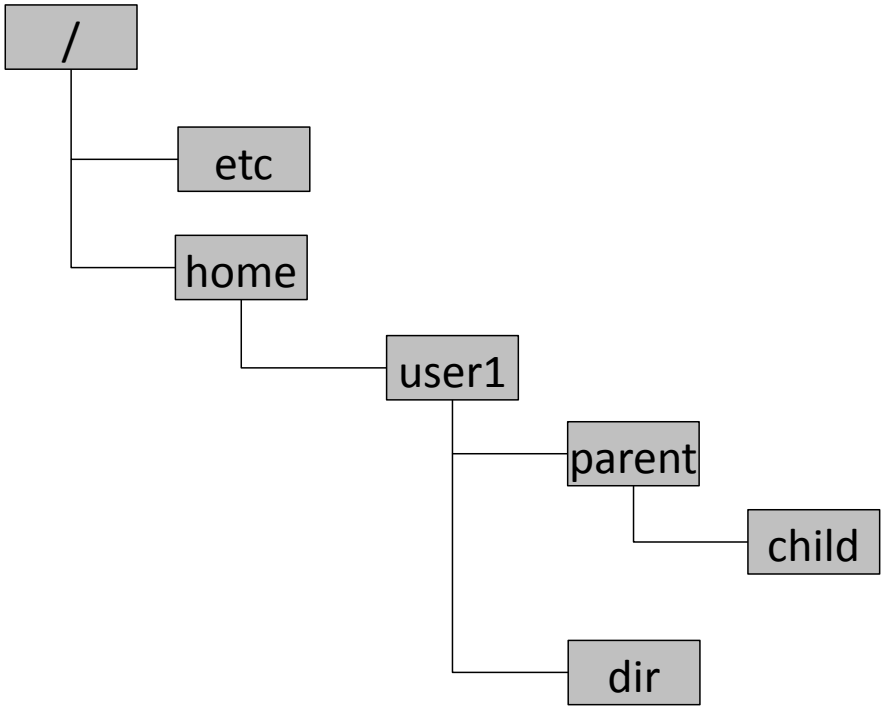
  `help command (e.g. help type)`

# Directory Tree



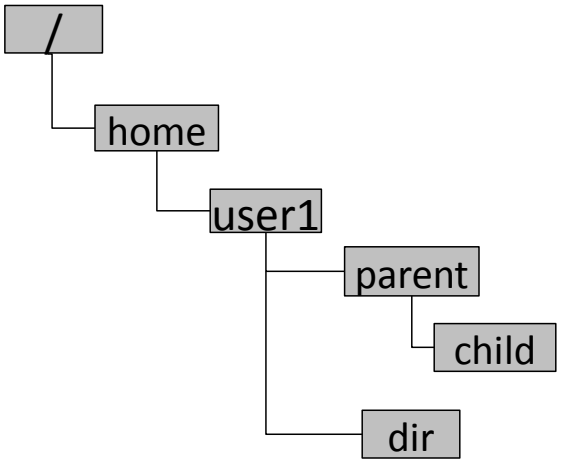

- Special Directories:
  - . = current directory
  - .. = parent directory
- forward slash "/" between directory names in path.
- current directory: You are here

# Directory Tree




- Commands:
  - `pwd` Show current directory ("**p**resent **w**orking **d**irectory")
  - `cd` Change directory
  - `cd child` Change current directory to "child"
  - `cd ..` Change current directory ("child") to the parent directory ("parent")

# Working with file system




- Absolute path (starts from / )
  - `less /home/user1/dir/file2.txt`
- Relative path (starts from pwd)
  - `pwd` → `/home/user1`
  - `less file1.txt`
  - `less dir/file2.txt`
  - `cd dir`
  - `less file2.txt`
    - Equal to: `less ./file2.txt`
  - `less ../file1.txt`

# Commands - See Cheat Sheet

- `ls -l` list a directory ; -l uses long format
- `du -hs dir` "disk usage" - space used in dir and subdirs
  (-h: human readable, -s: summary)
- `cp` copy a file
- `mv` move or rename a file
- `rm file1 [file2...]` remove files
  -r: also remove all subdirectories -f: force (**careful!**)
- `less filename` Progressively dump a file to the screen:
  SPACEBAR = page down
  Page-Up
  q=quit
- `nano filename` Edit a file using the "nano" `editor`
- `grep "string" file_name` prints all the lines in a file that contain the string

# Archives (tar)

- *.tar
  - `tar cvf archive.tar file1 [file2...]`

    c: Create a tar archive as a file “archive.tar” containing file1, file2...

    t: List the contents of “archive.tar“

    x: Extract  from the archive file
- *.tar.gz or .tgz
  - add “z” to list of options
- `.rar`
  - `unrar (rar is not free software)`
- `.zip`
  - `use unzip (and zip)`

# Environment variables

- `export DIRROOT=/usr/local/dir`

  Defines and exports to child process the variable DIRROOT with the value /usr/local/dir

- `echo $DIRROOT` Prints out the value of DIRROOT, or /usr/local/dir

- `printenv (or env)` Prints all available environment variables

# Environment variables

- environment variables used by the shell
  - $PATH All directories with executables
  - $USER User name
  - $HOME User's home folder (e.g. /home/user_name)
  - $TMPDIR Special variable for temporary folder (/scratch/user_name)

# Environment variables

- `ls`, `du`, `scp` etc. are executables.
- How does the shell find these programs?
- The shell looks in a list of directories.

- This list is saved in an environment variable: `PATH`

- Example:
- `echo $PATH`
  `/usr/local/bin:/usr/bin:/bin`

# Bash Globbing vs. Regex

Globbing (bash and find)

- strings: match itself
- * any number of any character
- [] character list
- [a-z] can use ranges
- [^a-z] negated character list

Regex (grep, sed, perl, …)

- strings: match itself
- . any character
- * any number of previous match
- [] character list
- [a-z] can use ranges
- [^a-z] negated character list
- many more rules for matches, depending on type of regex

# Quoting

- Use More Quotes!

- The shell expands arguments before passing them to a program
- Quotes prevent unwanted expansion
  - ' prevents variable substitution
  - '” does not (but prevents filename expansion)
  - \ only escapes one character

# Choosing Filenames

- Linux allows you to use **any** characters in a filename
- To avoid pain: Don't.  Only use  alphanumeric

  For example, avoid
  - $
  - ?
  - *
  - space
  - …

- Ok are: A-Za-z0-9._

# Working with files

- Output of ls -l, file attributes

```
$ ls -l
drwxr-xr-x    4 cliff    user        1024 Jun 18 09:40 WAITRON_EARNINGS
-rw-r--r--    1 cliff    user      767392 Jun  6 14:28 scanlib.tar.gz
^ ^  ^  ^     ^   ^       ^           ^      ^    ^      ^
| |  |  |     |   |       |           |      |    |      |
| |  |  |     | owner   group       size   date  time    name
| |  |  |     number of links to file or directory contents
| |  |  permissions for world
| |  permissions for members of group
| permissions for owner of file: r = read, w = write, x = execute -=no permission
type of file: - = normal file, d=directory, l = symbolic link, and others...
```

- Shared file access - change permissions for different groups and users
  - `chmod u=rw,g=rw,o=r file.txt`

# Thank you!