EBERHARD KARLS
UNIVERSITÄT
TÜBINGEN

FACULTY OF
SCIENCE
Interfakultäres Institut für Biomedizinische Informatik

Bioinformatics preparatory course
Basic Unix II

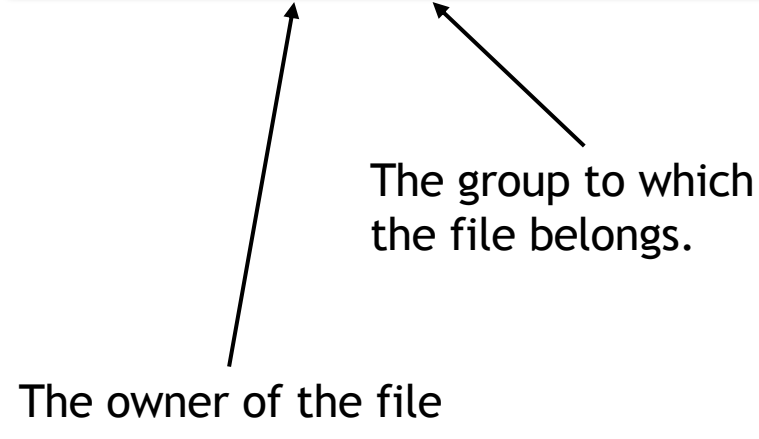
Léon Kuchenbecker and Sam Wein

Users, groups and permissions

- UNIX file systems allow to set permissions or access rights to users and groups of users
- A user may correspond to a real-world person, but also users representing *programs* exist e.g., http
- UNIX defines each user by its user identification number (UID) and user name
- Groups may contain one or more users
- Every file in UNIX belongs to one user and one group
- Permissions are based on this ownership

Ownership

```
vorkurs@vorkurs-vm:~$ ls -l Exercise/data/ ⏎
total 13960
-rw-r--r-- 1 vorkurs vorkurs 14293917 Mar 27 13:42 clinvar_20180225.vcf.gz
```

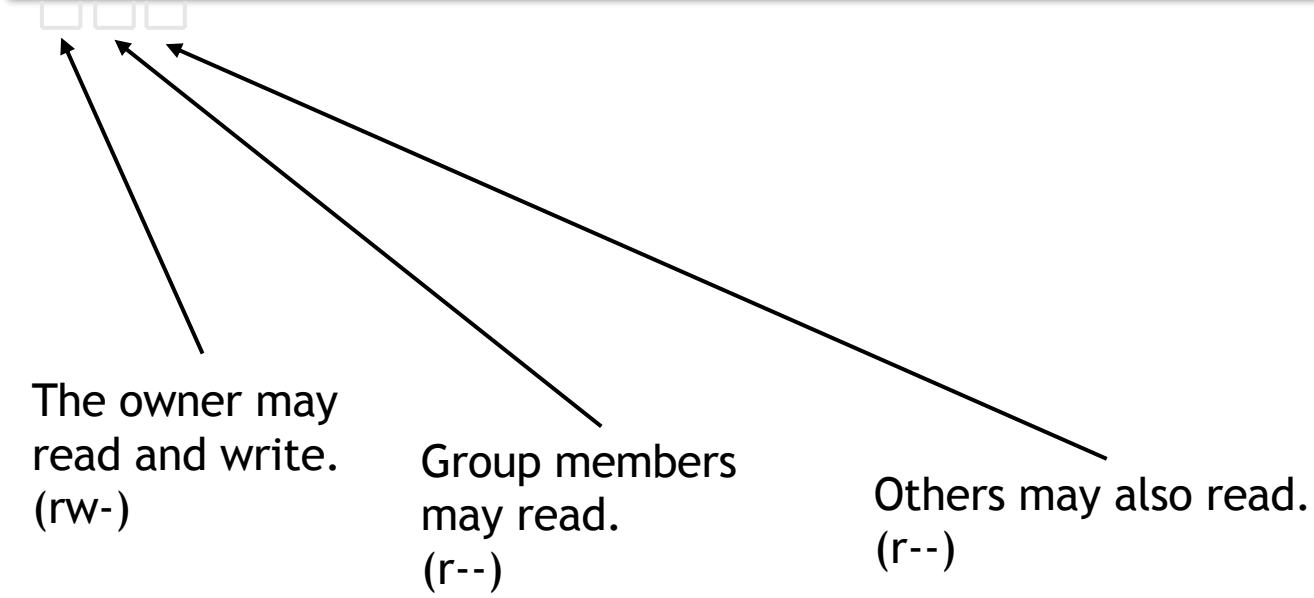


Permissions

- A set of permissions is associated with each file
- These permissions determine what can be done with a file and what not:
  - **Read**
  - **Write**
  - **Execute**
- This triplet of permissions (rwx) is defined for three types of users: *owner, group member, others*

Permissions

```
vorkurs@vorkurs-vm:~$ ls -l Exercise/data/ ⏎
total 13960
-rw-r--r-- 1 vorkurs vorkurs 14293917 Mar 27 13:42 clinvar_20180225.vcf.gz
```



Permissions

- The permissions for UNIX directories follow basically the same logic as the one for files. Anyhow some details differ.
- **Read** determines if the content of a directory is visible
- **Write** determines if new files can be created or deleted
- **Execute** determines if a user can change (`cd`) into the directory

chown

- Allows to change the ownership of a file or directory
- Allows to change the group
- Only the superuser (root) can change the owner

Syntax

```
chown␣[OPTION]␣[OWNER][:[GROUP]] FILE
```

- **`-R, --recursive`**
  operate on files and directories recursively
- **`-v, --verbose`**
  output a diagnostic for every file processed

chmod

- Allows to set and modify permissions for files and directories
- Requires ownership or super user status
- **chmod** needs to know for whom the permissions should be set:
  - **u** the owner user
  - **g** the owner group
  - **o** others (neither u, nor g)
  - **a** all users
- **chmod** uses **+** to set a bit and **-** to clear it

Example

- `chmod g+w FILE`
- Would allow the group to modify or delete the file

```
vorkurs@vorkurs-vm:~$ ls -l Exercise/data/ ⏎
total 13960
-rw-r--r-- 1 vorkurs vorkurs 14293917 Mar 27 13:42 clinvar_20180225.vcf.gz
vorkurs@vorkurs-vm:~$ chmod a+w Exercise/data/clinvar_20180225.vcf.gz ⏎
vorkurs@vorkurs-vm:~$ ls -l Exercise/data/ ⏎
total 13960
-rw-rw-rw- 1 vorkurs vorkurs 14293917 Mar 27 13:42 clinvar_20180225.vcf.gz
```


- Note: you can also change the permissions using numerical values for user, group, and all
  - 4: read (r)
  - 2: write (w)
  - 1: execute (x)

- Now you have to sum up the numbers:
  - **`chmod 750 FILE`**
    - 7=4+2+1: user has all privileges (rwx)
    - 5=4+1: group as read and execute privileges (rx)
    - 0: all others have no privileges

Downloading data from the Internet

wget

- Allows to download files from remote sources
- Very useful when working in a remote shell without graphical web browsers
- Not installed by default on macOS, but `curl` is

Syntax

```
wget␣[OPTION]␣URL
```

- **`-b, --background`**
  Go to background immediately after startup.
- **`-c, --continue`**
  Continue getting a partially-downloaded file.

- `wget` example

```
vorkurs@vorkurs-vm:~$ wget https://compbio.de/vorkurs/material.zip ⏎
--2018-04-03 14:10:03--  https://compbio.de/vorkurs/material.zip
Resolving compbio.de (compbio.de)... 188.40.166.122
Connecting to compbio.de (compbio.de)|188.40.166.122|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 10586 (10K) [application/zip]
Saving to: ‘material.zip’

material.zip     100%[======================>]  10.34K  --.-KB/s    in 0s

2018-04-03 14:10:03 (45.7 MB/s) - ‘material.zip’ saved [10586/10586]
```

- `curl` example (use `curl -LO` as wget replacement)

```
vorkurs@vorkurs-vm:~$ curl -LO https://compbio.de/vorkurs/material.zip ⏎
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 10586  100 10586    0     0   144k      0 --:--:-- --:--:-- --:--:--  143k
```


File compression and file archives

- The most common archive file formats are .tar.gz and .zip
- tar is an archive file format to store multiple files in one file, gz (GZIP) is a compressed file format
- zip is an “all in one” solution: archive and compression file format

ZIP file handling

- List the contents of a ZIP file

```
vorkurs@vorkurs-vm:~$ unzip -l material.zip⏎
Archive:  material.zip
  Length      Date    Time    Name
---------  ---------- -----   ----
        0  2017-03-29 09:36   material/
       51  2017-03-22 10:59   material/test_file_1.txt
       69  2017-03-22 10:59   material/test_file_2.txt
    34753  2017-03-22 12:36   material/large.fasta
       67  2017-03-22 12:39   material/small.fasta
      595  2017-03-28 08:58   material/duplicated_file.txt
       35  2017-03-29 09:29   material/my_diff_2.txt
       46  2017-03-29 09:29   material/my_diff_1.txt
       35  2017-03-29 09:34   material/my_sort_1.txt
      557  2017-03-29 09:36   material/tmp.txt
      557  2017-03-29 09:36   material/my_sort_2.txt
---------                     -------
    36765                     11 files
```


ZIP file handling

- Unpack the contents of a ZIP file

```
vorkurs@vorkurs-vm:~$ unzip material.zip ⏎
Archive:  material.zip
   creating: material/
  inflating: material/test_file_1.txt
  inflating: material/test_file_2.txt
  inflating: material/large.fasta
  inflating: material/small.fasta
  inflating: material/duplicated_file.txt
 extracting: material/my_diff_2.txt
  inflating: material/my_diff_1.txt
  inflating: material/my_sort_1.txt
  inflating: material/tmp.txt
  inflating: material/my_sort_2.txt
vorkurs@vorkurs-vm:~$ ls -F ⏎
Desktop/  Documents/  Exercise/  material/  material.zip
vorkurs@vorkurs-vm:~$ ls material ⏎
duplicated_file.txt  my_diff_1.txt  my_sort_1.txt  small.fasta
test_file_2.txt
large.fasta          my_diff_2.txt  my_sort_2.txt  test_file_1.txt  tmp.txt
```


ZIP file handling

- Create a new ZIP file from a directory

```
vorkurs@vorkurs-vm:~$ zip -r new_material.zip material/ ⏎
  adding: material/ (stored 0%)
  adding: material/my_diff_2.txt (stored 0%)
  adding: material/test_file_2.txt (deflated 41%)
  adding: material/duplicated_file.txt (deflated 92%)
  adding: material/test_file_1.txt (deflated 63%)
  adding: material/my_sort_1.txt (deflated 3%)
  adding: material/small.fasta (deflated 45%)
  adding: material/tmp.txt (deflated 52%)
  adding: material/my_sort_2.txt (deflated 51%)
  adding: material/my_diff_1.txt (deflated 4%)
  adding: material/large.fasta (deflated 77%)
```

- The `-r` (recursive) switch tells zip to not only add the folder itself to the archive, but recursively add all its contents, too

GZIPed TAR file handling

- List the contents of a GZIPed TAR file

```
vorkurs@vorkurs-vm:~$ tar tzf material.tar.gz ⏎
material/
material/test_file_2.txt
material/tmp.txt
material/my_sort_1.txt
material/test_file_1.txt
material/small.fasta
material/my_diff_2.txt
material/my_sort_2.txt
material/my_diff_1.txt
material/duplicated_file.txt
material/large.fasta
```

- Note that for `tar` the switches are commonly specified en bloc without a leading '-'. This is a historical convention and an exception.

GZIPed TAR file handling

- Unpack the contents of a GZIPed TAR file

```
vorkurs@vorkurs-vm:~$ tar xvzf material.tar.gz ⏎
material/
material/test_file_2.txt
material/tmp.txt
material/my_sort_1.txt
material/test_file_1.txt
material/small.fasta
material/my_diff_2.txt
material/my_sort_2.txt
material/my_diff_1.txt
material/duplicated_file.txt
material/large.fasta
```


GZIPed TAR file handling

- Create a new GZIPed TAR file from a directory

```
vorkurs@vorkurs-vm:~$ tar cvzf new_material.tar.gz material/ ⏎
material/
material/my_diff_2.txt
material/test_file_2.txt
material/duplicated_file.txt
material/test_file_1.txt
material/my_sort_1.txt
material/small.fasta
material/tmp.txt
material/my_sort_2.txt
material/my_diff_1.txt
material/large.fasta
```

- If you add the `v` (verbose) switch, `tar` outputs the list of files while creating the archive

GZIPed TAR file handling

- TAR files are sometimes compressed with compression tools other than GZIP
- For `.tar.bz2` use the ‘`j`’ switch instead of ‘`z`’
- For `.tar.xz` use the ‘`J`’ switch instead of ‘`z`’
- In fact, recent `tar` versions can automatically detect the compression and you can skip the compression flag altogether:

```
vorkurs@vorkurs-vm:~$ tar xf material.tar.gz ⏎
```


GZIP is also used independently of TAR

`gzip`

- Compresses a file, adds ‘.gz’ extension

`gunzip`

- Decompresses a file, removes ‘.gz’ extension

Syntax

```
g[un]zip␣[OPTION]␣FILE
```

`-k --keep`	Keeps the original file
`-v --verbose`	Verbose output

GZIP is also used independently of TAR

- It is commonly used to compress all types of files, e.g. FASTQ

```
vorkurs@vorkurs-vm:~$ wget http://compbio.de/vorkurs/SRR6800317.fastq.gz ⏎
URL transformed to HTTPS due to an HSTS policy
--2018-04-03 15:56:35--  https://compbio.de/vorkurs/SRR6800317.fastq.gz
Resolving compbio.de (compbio.de)... 188.40.166.122
Connecting to compbio.de (compbio.de)|188.40.166.122|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 14192892 (14M) [application/x-gzip]
Saving to: ‘SRR6800317.fastq.gz’

SRR6800317.fastq.gz    100%[============================>]  13.54M  10.9MB/s    in 1.2s

2018-04-03 15:56:36 (10.9 MB/s) - ‘SRR6800317.fastq.gz’ saved [14192892/14192892]

vorkurs@vorkurs-vm:~$ ls -lh SRR6800317.fastq.gz ⏎
-rw-r--r-- 1 vorkurs vorkurs 14M Apr  3  2018 SRR6800317.fastq.gz
vorkurs@vorkurs-vm:~$ gunzip SRR6800317.fastq.gz ⏎
vorkurs@vorkurs-vm:~$ ls -lh SRR6800317.fastq ⏎
-rw-r--r-- 1 vorkurs vorkurs 89M Apr  3  2018 SRR6800317.fastq
```

File sizes

Check sizes of files and directories

- **`du`** stands for ‘disk usage’, counting bits and bytes recursively
- Prints out the disk space used by a file, directory, …
- In particular useful when checking whole branches of the file system tree

- **`-a, --all`**
  write counts for all files, not just directories
- **`-c, --total`**
  produce a grand total
- **`-h, --human-readable`**
  print sizes in human readable format (e.g., 1K 234M 2G)

Disk Usage vs. File Size

- Note that there is a difference between the outputs of `ls` and `du`:

```
vorkurs@vorkurs-vm:~$ du -h material.zip ⏎
12K   material.zip
vorkurs@vorkurs-vm:~$ ls -lh material.zip ⏎
-rw-r--r-- 1 vorkurs vorkurs 11K Apr  3 22:05 material.zip
```


du Examples

```
vorkurs@vorkurs-vm:~$ du SRR6800317.fastq.gz ⏎
13864 SRR6800317.fastq.gz
```

```
vorkurs@vorkurs-vm:~$ du -h SRR6800317.fastq.gz ⏎
14M   SRR6800317.fastq.gz
```

```
vorkurs@vorkurs-vm:~$ du -h material ⏎
76K   material
```

```
vorkurs@vorkurs-vm:~$ du -ah material ⏎
4.0K  material/my_diff_2.txt
4.0K  material/test_file_2.txt
4.0K  material/duplicated_file.txt
4.0K  material/test_file_1.txt
4.0K  material/my_sort_1.txt
4.0K  material/small.fasta
4.0K  material/tmp.txt
4.0K  material/my_sort_2.txt
4.0K  material/my_diff_1.txt
36K   material/large.fasta
76K   material
```


File Comparison

Check Files for Differences

`diff` can be used to compare plain text files

- `-i  --ignore-case`
  Ignore case differences in file contents.
- `-w  --ignore-all-space`
  Ignore all white space.
- `-B  --ignore-blank-lines`
  Ignore changes of whitespace
- `-y  --side-by-side`
  Output in two columns.

Check Files for Differences

- The output is not quite human readable by default

```
vorkurs@vorkurs-vm:~$ cd material/ ⏎
vorkurs@vorkurs-vm:~/material$ ls -l test_file_* ⏎
-rw-r--r-- 1 vorkurs vorkurs 51 Mar 22  2017 test_file_1.txt
-rw-r--r-- 1 vorkurs vorkurs 69 Mar 22  2017 test_file_2.txt
vorkurs@vorkurs-vm:~/material$ diff test_file_* ⏎
5c5
< bla
---
> important information
```

- If the two files are identical, the output is empty

```
vorkurs@vorkurs-vm:~/material$ cp test_file_1.txt test_file_1.txt.copy ⏎
vorkurs@vorkurs-vm:~/material$ diff test_file_1.txt* ⏎
vorkurs@vorkurs-vm:~/material$
```


Find differences in the files “my_diff_1.txt” and “my_diff_2.txt”

Sorting and Counting

Sorting Text

`sort` can sort text

- **-b, --ignore-leading-blanks**
  ignore leading blanks
- **-n, --numeric-sort**
  compare according to string numerical value
- **-r, --reverse**
  reverse the result of comparisons
- **-u, --unique**
  Output redundant lines only once


```
vorkurs@vorkurs-vm:~$ curl -LO https://compbio.de/vorkurs/sortme.txt ⏎
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   189  100   189    0     0   1393      0 --:--:-- --:--:-- --:--:--  1400
vorkurs@vorkurs-vm:~$ cat sortme.txt ⏎
This is an example file with content
that may be sorted. If you sort this file
its contents might no longer make any sense
because the line order is critical. None
of this makes any sense!
vorkurs@vorkurs-vm:~$ sort sortme.txt ⏎
because the line order is critical. None
its contents might no longer make any sense
of this makes any sense!
that may be sorted. If you sort this file
This is an example file with content
```


1. Sort the file “my_sort_1.txt”

2. Sort the file „my_sort_2.txt“ in reverse numerical order

`uniq` collapses consecutive repetitive lines into one line

- `-c, --count`
  Count the number of occurrences
- `-i, --ignore-case`
  Ignore differences in case when comparing


```
vorkurs@vorkurs-vm:~/material$ uniq -c test_file_1.txt ⏎
      2 bla
      1 blabla
      1 blablabla
      1 bla
      2 blabla
      1 bla
      1 end
```


1. Which lines in *duplicated_file.txt* are duplicated and how often do they occur?

2. Are all globally redundant lines collapsed by `uniq`?

`wc` (***w**ord **c**ount*) can count characters, words and lines

- `wc␣-c␣<filename>`
  Print the number of characters in a file
- `wc␣-l␣<filename>`
  Print the number of lines in a file
- `wc␣-w␣<filename>`
  Print the number of words in a file
- `wc␣<filename>`
  Prints all three numbers


```
vorkurs@vorkurs-vm:~/material$ wc -l *.txt ⏎
 100 duplicated_file.txt
   5 my_diff_1.txt
   4 my_diff_2.txt
   5 my_sort_1.txt
 167 my_sort_2.txt
   9 test_file_1.txt
   9 test_file_2.txt
 167 tmp.txt
 466 total
```


Searching Text for Expressions

`grep` is a powerful tool to search for *regular expressions*

- **`grep␣<pattern>␣<filename>`**
  Print all lines of the file containing the text
- **`grep␣-v␣<pattern>␣<filename>`**
  Print all lines NOT containing the text
- **`grep␣-l␣<pattern>␣<filename>`**
  Print only the names of all files containing the text
- **`grep␣-n␣<pattern>␣<filename>`**
  Print the line numbers along with the matches


```
vorkurs@vorkurs-vm:~$ grep --color -n is sortme.txt ⏎
1:This is an example file with content
2:that may be sorted. If you sort this file
4:because the line order is critical. None
5:of this makes any sense!
```


Standard Streams II – The `stdin` stream

- The | (pipe character) can be used to redirect the **standard output** of one process to the **standard input** of another process

- Remember the common structure of the interfaces for programs we have seen today:

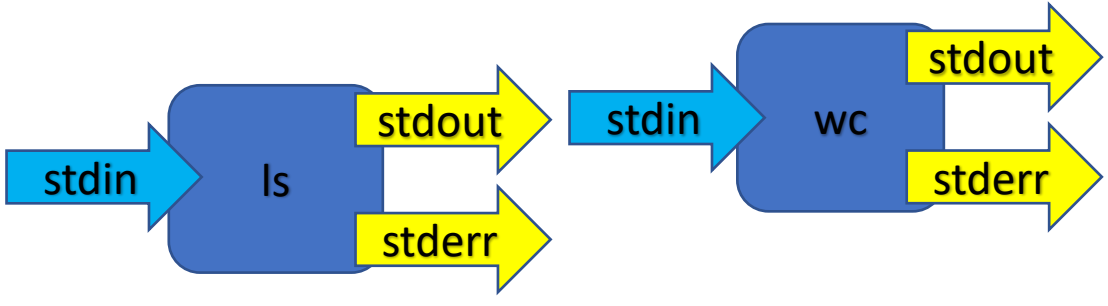
```
COMMAND [OPTIONAL SWICHES] [INPUT FILENAME]
```

- Most of the programs we have seen today process the **standard input** if not input filename is given

Standard Streams II

```
vorkurs@vorkurs-vm:~/material$ ls -l 1> temp.txt
vorkurs@vorkurs-vm:~/material$ wc -l temp.txt
15 temp.txt
```

```
vorkurs@vorkurs-vm:~/material$ ls -l | wc -l
15
```



Standard Streams II

```
vorkurs@vorkurs-vm:~/material$ uniq -c duplicated_file.txt
     22 bla
     28 blabla
      1 this line is not duplicatedbla
     21 bla
     28 blabla
```

```
vorkurs@vorkurs-vm:~/material$ sort duplicated_file.txt | uniq -c
     43 bla
     56 blabla
      1 this line is not duplicatedbla
```


Standard Streams II

- What does this do? And why?

```
vorkurs@vorkurs-vm:~/material$ gunzip -c material.tar.gz | tar t
```

