

# Help choosing the next crypto-standard

ES&S MOL: ay 2023-2024

- ES&S
- Introduction
- Library exercise
- Experiment
- Your turn!

# Emerging technologies, Systems & Security

- Research group connected to Electronics/ICT, MNS & COSIC
- Headed by prof. Nele Mentens & prof. Kris Myny
- Current research team:
  - 8 PhD students
  - 1 post-doc
  - 1 research expert

- ES&S
- Introduction
- Library exercise
- Experiment
- Your turn!

# Introduction - preparation

Q1

What did you find out about this "formula" ?

cryptology = cryptography + cryptanalysis

# Introduction - preparation

Q2

What is the difference between Symmetric key and Public key cryptography ?

# Introduction - preparation

Q3

What is a cryptographic algorithm ?

# Introduction - preparation

Q3

What is a cryptographic algorithm ?

Symmetric-key cryptography

- 
- 
- 

Public-key cryptography

- 
-



# Introduction - preparation

Q4

Why would you optimise some code or a design towards binary size, anno 2023 ?

# Introduction - preparation

Q5

How do you compile a static library in C, and how do you link with it ?

- what is the difference between a static and a dynamic library ?

# Introduction - labsetup

Experiments will be done a server

- wifi: Eduroam
- server IP address: 10.4.14.1
- connection: SSH
- user accounts: mol\_user $n$  met  $n \in \{1, 2, 3, 4, 5\}$ 
  - Password: 20231109\_mol\_user

[This Photo](#) by Unknown author is licensed under [CC BY-SA](#).

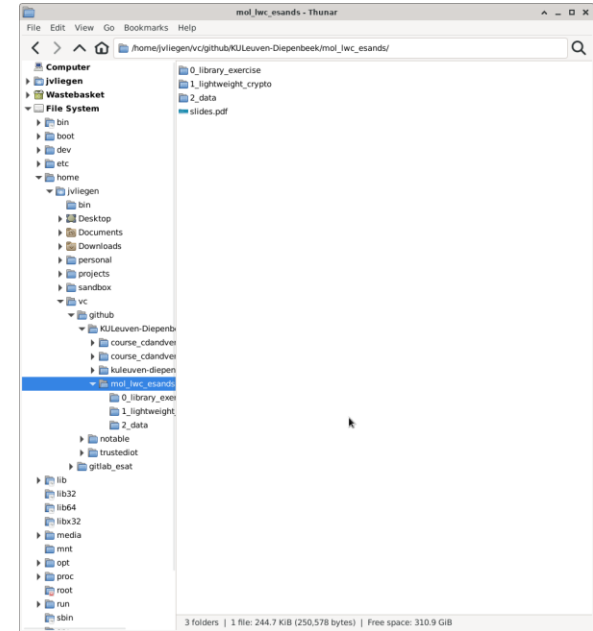


**PuTTY**

# Introduction - labsetup

## Linux – command line interface (CLI)

- File system is built from the root /
- Every user has its own home folder
  - e.g. /home/mol\_user4
- Directory listing: **ls**
- Changing directories with
  - `cd <folder_name>` and `cd ..`
  - e.g. `cd 0_library_exercise`



# Introduction - labsetup

## Linux – command line interface (CLI)

- Cheat sheet:

<https://files.fosswire.com/2007/08/fwunixref.pdf>

File Commands	System Info
<b>ls</b> - directory listing <b>ls -al</b> - formatted listing with hidden files <b>cd dir</b> - change directory to <i>dir</i> <b>cd</b> - change to home <b>pwd</b> - show current directory <b>mkdir dir</b> - create a directory <i>dir</i> <b>rm file</b> - delete <i>file</i> <b>rm -r dir</b> - delete directory <i>dir</i> <b>rm -f file</b> - force remove <i>file</i> <b>rm -rf dir</b> - force remove directory <i>dir</i> * <b>cp file1 file2</b> - copy <i>file1</i> to <i>file2</i> <b>cp -r dir1 dir2</b> - copy <i>dir1</i> to <i>dir2</i> ; create <i>dir2</i> if it doesn't exist <b>mv file1 file2</b> - rename or move <i>file1</i> to <i>file2</i> if <i>file2</i> is an existing directory, moves <i>file1</i> into directory <i>file2</i> <b>ln -s file link</b> - create symbolic link <i>link</i> to <i>file</i> <b>touch file</b> - create or update <i>file</i> <b>cat &gt; file</b> - places standard input into <i>file</i> <b>more file</b> - output the contents of <i>file</i> <b>head file</b> - output the first 10 lines of <i>file</i> <b>tail file</b> - output the last 10 lines of <i>file</i> <b>tail -f file</b> - output the contents of <i>file</i> as it grows, starting with the last 10 lines	<b>date</b> - show the current date and time <b>cal</b> - show this month's calendar <b>uptime</b> - show current uptime <b>w</b> - display who is online <b>whoami</b> - who you are logged in as <b>finger user</b> - display information about <i>user</i> <b>uname -a</b> - show kernel information <b>cat /proc/cpuinfo</b> - cpu information <b>cat /proc/meminfo</b> - memory information <b>man command</b> - show the manual for <i>command</i> <b>df</b> - show disk usage <b>du</b> - show directory space usage <b>free</b> - show memory and swap usage <b>whereis app</b> - show possible locations of <i>app</i> <b>which app</b> - show which <i>app</i> will be run by default
Process Management	Compression
<b>ps</b> - display your currently active processes <b>top</b> - display all running processes <b>kill pid</b> - kill process id <i>pid</i> <b>killall proc</b> - kill all processes named <i>proc</i> * <b>bg</b> - lists stopped or background jobs; resume a stopped job in the background <b>fg</b> - brings the most recent job to foreground <b>fg n</b> - brings job <i>n</i> to the foreground	<b>tar cf file.tar files</b> - create a tar named <i>file.tar</i> containing <i>files</i> <b>tar xf file.tar</b> - extract the files from <i>file.tar</i> <b>tar czf file.tar.gz files</b> - create a tar with Gzip compression <b>tar xzf file.tar.gz</b> - extract a tar using Gzip <b>tar cJf file.tar.bz2</b> - create a tar with Bzip2 compression <b>tar xJf file.tar.bz2</b> - extract a tar using Bzip2 <b>gzip file</b> - compresses <i>file</i> and renames it to <i>file.gz</i> <b>gzip -d file.gz</b> - decompresses <i>file.gz</i> back to <i>file</i>
File Permissions	Network
<b>chmod octal file</b> - change the permissions of <i>file</i> to <i>octal</i> , which can be found separately for user, group, and world by adding: <ul style="list-style-type: none"><li>• 4 - read (r)</li><li>• 2 - write (w)</li><li>• 1 - execute (x)</li></ul> Examples: <b>chmod 777</b> - read, write, execute for all <b>chmod 755</b> - rwx for owner, rx for group and world For more options, see <b>man chmod</b> .	<b>ping host</b> - ping <i>host</i> and output results <b>whois domain</b> - get whois information for <i>domain</i> <b>dig domain</b> - get DNS information for <i>domain</i> <b>dig -x host</b> - reverse lookup <i>host</i> <b>wget file</b> - download <i>file</i> <b>wget -c file</b> - continue a stopped download
SSH	Installation
<b>ssh user@host</b> - connect to <i>host</i> as <i>user</i> <b>ssh -p port user@host</b> - connect to <i>host</i> on port <i>port</i> as <i>user</i> <b>ssh-copy-id user@host</b> - add your key to <i>host</i> for <i>user</i> to enable a keyed or passwordless login	Install from source: <b>./configure</b> <b>make</b> <b>make install</b> <b>dpkg -i pkg.deb</b> - install a package (Debian) <b>rpm -Uvh pkg.rpm</b> - install a package (RPM)
Searching	Shortcuts
<b>grep pattern files</b> - search for <i>pattern</i> in <i>files</i> <b>grep -r pattern dir</b> - search recursively for <i>pattern</i> in <i>dir</i> <b>command   grep pattern</b> - search for <i>pattern</i> in the output of <i>command</i> <b>locate file</b> - find all instances of <i>file</i>	<b>Ctrl+C</b> - halts the current command <b>Ctrl+Z</b> - stops the current command, resume with <b>fg</b> in the foreground or <b>bg</b> in the background <b>Ctrl+D</b> - log out of current session, similar to <b>exit</b> <b>Ctrl+W</b> - erases one word in the current line <b>Ctrl+U</b> - erases the whole line <b>Ctrl+R</b> - type to bring up a recent command <b>!!</b> - repeats the last command <b>exit</b> - log out of current session
	* use with extreme caution.



- ES&S
- Introduction
- Library exercise
- Experiment
- Your turn!

All code in 1 file

```
demo_v1.c x
1  #include <stdio.h>
2
3  int sum(int x, int y) {
4      return (int)(x+y);
5  }
6
7  int main(void) {
8      int a, b, c;
9
10     a = 3;
11     b = 2;
12
13     c = sum(a, b);
14
15     printf("The sum of a + b = %d + %d = %d\n", a, b, c);
16
17     return 0;
18 }
```

All code in 1 file

```
demo_v1.c x
1  #include <stdio.h>
2
3  int sum(int x, int y) {
4      return (int)(x+y);
5  }
6
7  int main(void) {
8      int a, b, c;
9
10     a = 3;
11     b = 2;
12
13     c = sum(a, b);
14
15     printf("The sum of a + b = %d + %d = %d\n", a, b, c);
16
17     return 0;
18 }
```

	main( )	
--	---------	--

	sum( )	
--	--------	--



All code in 1 file

```
demo_v1.c x
1  #include <stdio.h>
2
3  int sum(int x, int y) {
4      return (int)(x+y);
5  }
6
7  int main(void) {
8      int a, b, c;
9
10     a = 3;
11     b = 2;
12
13     c = sum(a, b);
14
15     printf("The sum of a + b = %d + %d = %d\n", a, b, c);
16
17     return 0;
18 }
```

All code in a single file

`gcc -c demo_v1.c`

`gcc -o demo_v1 demo_v1.o`

C source  
object file  
binary  
static library

All code in 3 files

```
1 // demo_v2_lib.h
2
3 int sum(int x, int y);
4
5
6 int main() {
7     int a, b, c;
8     a = 3;
9     b = 2;
10
11     c = sum(a, b);
12
13     printf("The sum of a + b = %d + %d = %d\n", a, b, c);
14
15     return 0;
16 }
```

```
1 #include "demo_v2_lib.h"
2
3 int sum(int x, int y) {
4     return (int)(x+y);
5 }
6
```

```
1 #include <stdio.h>
2
3 int sum(int x, int y);
```

All code in 3 files

```
demo_v2_lib.h
1 int sum(int x, int y);
2
3 int main()
4 {
5     int a, b, c;
6     a = 3;
7     b = 2;
8
9     c = sum(a, b);
10
11     printf("The sum of a + b = %d + %d = %d\n", a, b, c);
12
13     return 0;
14 }
15
16 }
```

```
demo_v2_lib.c
1 #include "demo_v2_lib.h"
2
3 int sum(int x, int y) {
4     return (int)(x+y);
5 }
6
```

```
demo_v2_lib.h
1 #include <stdio.h>
2
3 int sum(int x, int y);
```

main( )

sum( )

All code in 3 files

```
demo_v2.c
1 int main()
2 {
3     int a, b, c;
4     a = 3;
5     b = 2;
6     c = sum(a, b);
7     printf("The sum of a + b = %d + %d = %d\n", a, b, c);
8     return 0;
9 }
```

```
demo_v2_lib.c
1 #include "demo_v2_lib.h"
2
3 int sum(int x, int y) {
4     return (int)(x+y);
5 }
6
```

```
demo_v2_lib.h
1 #include <stdio.h>
2
3 int sum(int x, int y);
```

All code in separate files

`gcc -c demo_v2_lib.c`

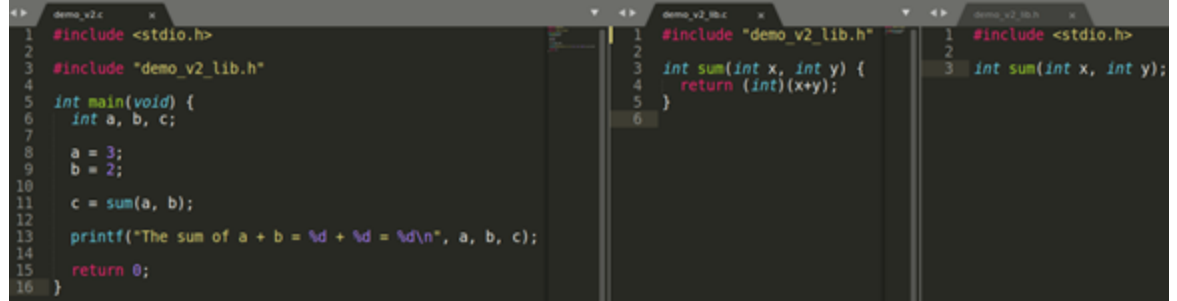
`gcc -c demo_v2.c`

`gcc -o demo_v2 demo_v2.o demo_v2_lib.o`

C source  
object file  
binary  
static library

Using a library

code is unaltered!!



```
demo_v2.c
1 #include <stdio.h>
2
3 #include "demo_v2_lib.h"
4
5 int main(void) {
6     int a, b, c;
7
8     a = 3;
9     b = 2;
10
11     c = sum(a, b);
12
13     printf("The sum of a + b = %d + %d = %d\n", a, b, c);
14
15     return 0;
16 }
```

```
demo_v2_lib.c
1 #include "demo_v2_lib.h"
2
3 int sum(int x, int y) {
4     return (int)(x+y);
5 }
6
```

```
demo_v2_lib.h
1 #include <stdio.h>
2
3 int sum(int x, int y);
```

Using a library

code is unaltered!!

```
demo_v2.c
1 #include <stdio.h>
2
3 #include "demo_v2_lib.h"
4
5 int main(void) {
6     int a, b, c;
7
8     a = 3;
9     b = 2;
10
11     c = sum(a, b);
12     printf("The sum of a + b = %d + %d = %d\n", a, b, c);
13
14     return 0;
15 }
```

```
demo_v2_lib.c
1 #include "demo_v2_lib.h"
2
3 int sum(int x, int y) {
4     return (int)(x+y);
5 }
```

```
demo_v2_lib.h
1 #include <stdio.h>
2
3 int sum(int x, int y);
```

main( )

The Great Library

sum( )

Using a library

code is unaltered!!

```
demo_v2.c
1 #include <stdio.h>
2
3 #include "demo_v2_lib.h"
4
5 int main(void) {
6     int a, b, c;
7
8     a = 3;
9     b = 2;
10
11     c = sum(a, b);
12     printf("The sum of a + b = %d + %d = %d\n", a, b, c);
13
14     return 0;
15 }

demo_v2_lib.c
1 #include "demo_v2_lib.h"
2
3 int sum(int x, int y) {
4     return (int)(x+y);
5 }
6

demo_v2_lib.h
1 #include <stdio.h>
2
3 int sum(int x, int y);
```

```
gcc -c demo_v2_lib.c
```

```
ar -rcs libdemo_v2.a demo_v2_lib.o
```

```
gcc -c demo_v3.c
```

```
gcc -o demo_v3 demo_v3.o -L. -ldemo_v2
```

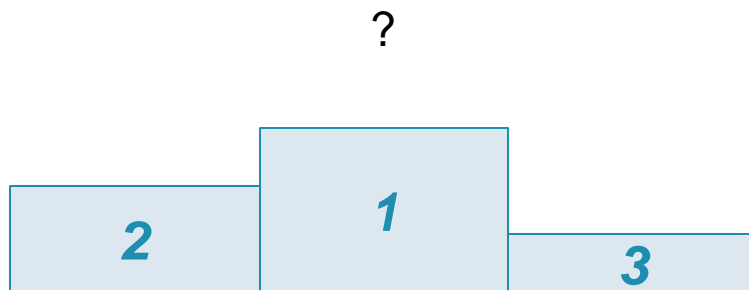
C source  
object file  
binary  
static library

- ES&S
- Introduction
- Library exercise
- Experiment
- Your turn!



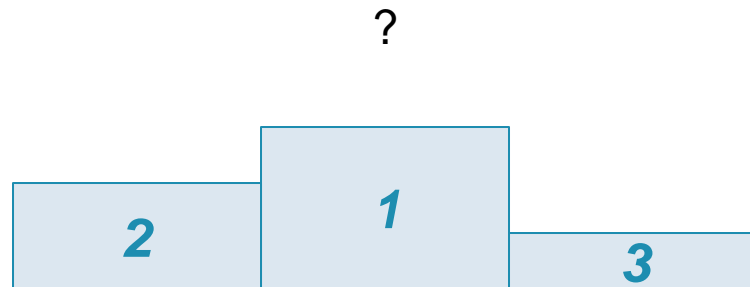
# Setting

- Ongoing competition:  
<https://csrc.nist.gov/projects/lightweight-cryptography/>
- 10 finalists



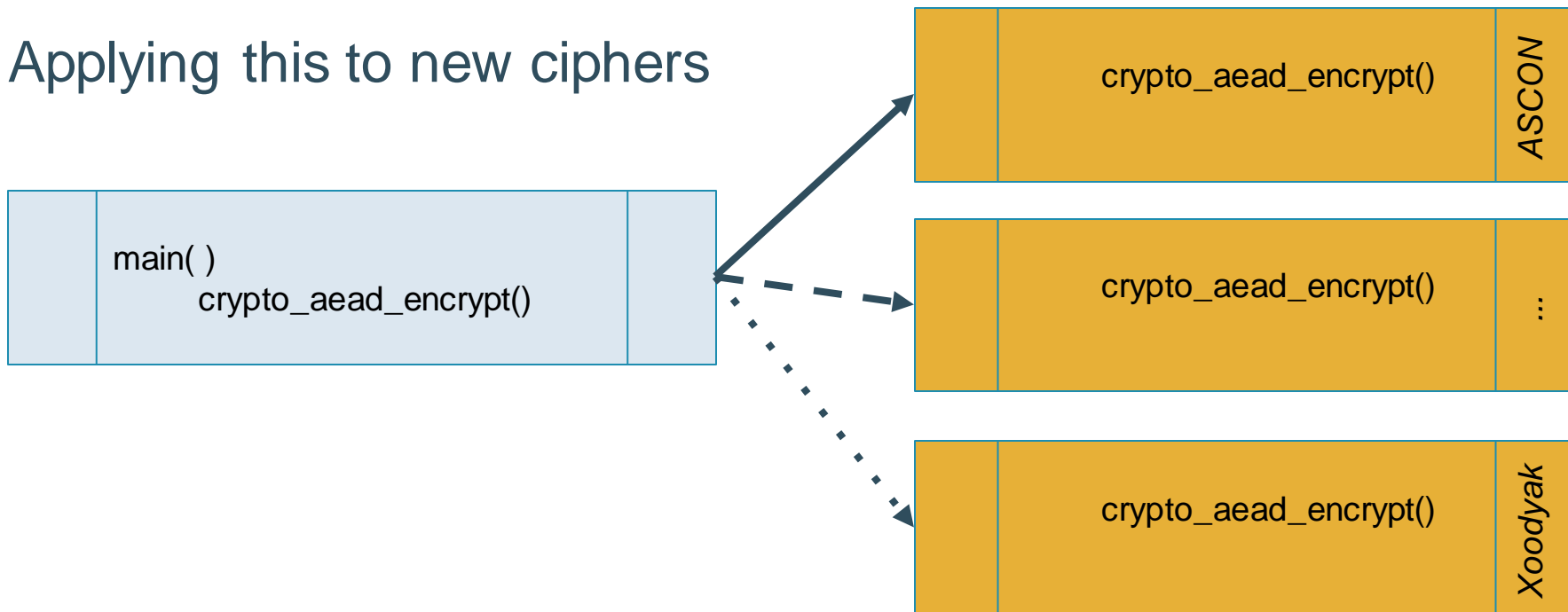
# Setting

- speed
- file size
- optimal input size
- ...



# A little help

Applying this to new ciphers



- ES&S
- Introduction
- Library exercise
- Experiment
- Your turn!