KU LEUVEN

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

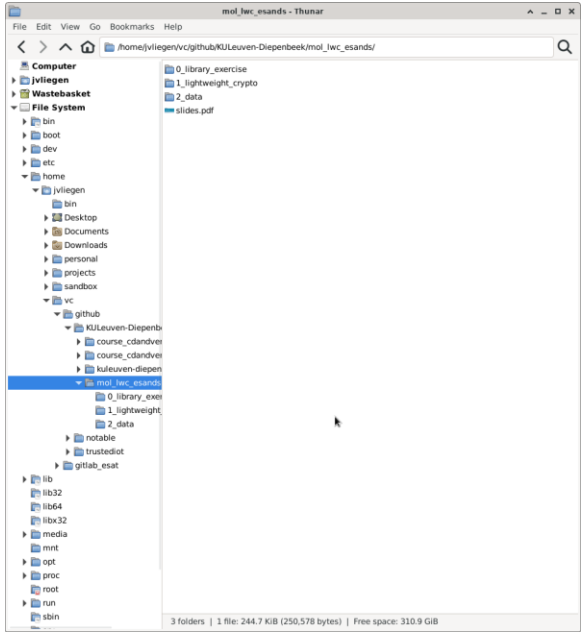
- wifi: Eduroam
- server IP address: 10.4.14.1
- connection: SSH
- user accounts: mol_user***n*** met **n** {1, 2, 3, 4}
  - Password: 20231130_mol_user

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

Automatic Zoom

Unix/Linux Command Reference

FOSSwire.com

File Commands

- **ls** – directory listing
- **ls -al** – formatted listing with hidden files
- **cd *dir*** – change directory to *dir*
- **cd** – change to home
- **pwd** – show current directory
- **mkdir *dir*** – create a directory *dir*
- **rm *file*** – delete *file*
- **rm -r *dir*** – delete directory *dir*
- **rm -f *file*** – force remove *file*
- **rm -rf *dir*** – force remove directory *dir* *
- **cp *file1 file2*** – copy *file1* to *file2*
- **cp -r *dir1 dir2*** – copy *dir1* to *dir2*; create *dir2* if it doesn't exist
- **mv *file1 file2*** – rename or move *file1* to *file2* if *file2* is an existing directory, moves *file1* into directory *file2*
- **ln -s *file link*** – create symbolic link *link* to *file*
- **touch *file*** – create or update *file*
- **cat > *file*** – places standard input into *file*
- **more *file*** – output the contents of *file*
- **head *file*** – output the first 10 lines of *file*
- **tail *file*** – output the last 10 lines of *file*
- **tail -f *file*** – output the contents of *file* as it grows, starting with the last 10 lines

Process Management

- **ps** – display your currently active processes
- **top** – display all running processes
- **kill *pid*** – kill process id *pid*
- **killall *proc*** – kill all processes named *proc* *
- **bg** – lists stopped or background jobs; resume a stopped job in the background
- **fg** – brings the most recent job to foreground
- **fg *n*** – brings job *n* to the foreground

File Permissions

chmod *octal file* – change the permissions of *file* to *octal*, which can be found separately for user, group, and world by adding:

- 4 – read (r)
- 2 – write (w)
- 1 – execute (x)

Examples:
chmod 777 – read, write, execute for all
chmod 755 – rwx for owner, rx for group and world
For more options, see **man chmod**.

SSH

- **ssh *user@host*** – connect to *host* as *user*
- **ssh -p *port user@host*** – connect to *host* on port *port* as *user*
- **ssh-copy-id *user@host*** – add your key to *host* for *user* to enable a keyed or passwordless login

Searching

- **grep *pattern files*** – search for *pattern* in *files*
- **grep -r *pattern dir*** – search recursively for *pattern* in *dir*
- ***command* | grep *pattern*** – search for *pattern* in the output of *command*
- **locate *file*** – find all instances of *file*

System Info

- **date** – show the current date and time
- **cal** – show this month's calendar
- **uptime** – show current uptime
- **w** – display who is online
- **whoami** – who you are logged in as
- **finger *user*** – display information about *user*
- **uname -a** – show kernel information
- **cat /proc/cpuinfo** – cpu information
- **cat /proc/meminfo** – memory information
- **man *command*** – show the manual for *command*
- **df** – show disk usage
- **du** – show directory space usage
- **free** – show memory and swap usage
- **whereis *app*** – show possible locations of *app*
- **which *app*** – show which *app* will be run by default

Compression

- **tar cf *file.tar files*** – create a tar named *file.tar* containing *files*
- **tar xf *file.tar*** – extract the files from *file.tar*
- **tar czf *file.tar.gz files*** – create a tar with Gzip compression
- **tar xzf *file.tar.gz*** – extract a tar using Gzip
- **tar cjf *file.tar.bz2*** – create a tar with Bzip2 compression
- **tar xjf *file.tar.bz2*** – extract a tar using Bzip2
- **gzip *file*** – compresses *file* and renames it to *file.gz*
- **gzip -d *file.gz*** – decompresses *file.gz* back to *file*

Network

- **ping *host*** – ping *host* and output results
- **whois *domain*** – get whois information for *domain*
- **dig *domain*** – get DNS information for *domain*
- **dig -x *host*** – reverse lookup *host*
- **wget *file*** – download *file*
- **wget -c *file*** – continue a stopped download

Installation

Install from source:
./configure
make
make install
- **dpkg -i *pkg.deb*** – install a package (Debian)
- **rpm -Uvh *pkg.rpm*** – install a package (RPM)

Shortcuts

- **Ctrl+C** – halts the current command
- **Ctrl+Z** – stops the current command, resume with **fg** in the foreground or **bg** in the background
- **Ctrl+D** – log out of current session, similar to **exit**
- **Ctrl+W** – erases one word in the current line
- **Ctrl+U** – erases the whole line
- **Ctrl+R** – type to bring up a recent command
- **!!** - repeats the last command
- **exit** – 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

```c
#include <stdio.h>

int sum(int x, int y) {
  return (int)(x+y);
}

int main(void) {
  int a, b, c;

  a = 3;
  b = 2;

  c = sum(a, b);

  printf("The sum of a + b = %d + %d = %d\n", a, b, c);

  return 0;
}
```

All code in 1 file

demo_v1.c

```c
#include <stdio.h>

int sum(int x, int y) {
  return (int)(x+y);
}

int main(void) {
  int a, b, c;

  a = 3;
  b = 2;

  c = sum(a, b);

  printf("The sum of a + b = %d + %d = %d\n", a, b, c);

  return 0;
}
```

main( )

sum( )

All code in 1 file

demo_v1.c

```c
#include <stdio.h>

int sum(int x, int y) {
  return (int)(x+y);
}

int main(void) {
  int a, b, c;

  a = 3;
  b = 2;

  c = sum(a, b);

  printf("The sum of a + b = %d + %d = %d\n", a, b, c);

  return 0;
}
```

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

```c
lib.h"

int ma
  int a, b;

  a = 3;
  b = 2;

  c = sum(a, b);

  printf("The sum of a + b = %d + %d = %d\n", a, b, c);

  return 0;
}
```

demo_v2_lib.c

```c
#include "demo_v2_lib.h"

int sum(int x, int y) {
  return (int)(x+y);
}
```

demo_v2_lib.h

```c
#include <stdio.h>

int sum(int x, int y);
```

All code in 3 files

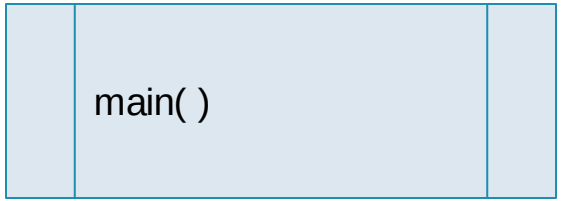
```c
lib.h"

int ma
  int a, b;

  a = 3;
  b = 2;

  c = sum(a, b);

  printf("The sum of a + b = %d + %d = %d\n", a, b, c);

  return 0;
}
```

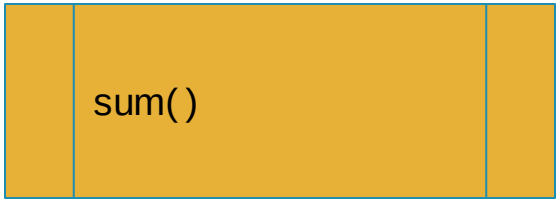
demo_v2_lib.c

```c
#include "demo_v2_lib.h"

int sum(int x, int y) {
  return (int)(x+y);
}
```

demo_v2_lib.h

```c
#include <stdio.h>

int sum(int x, int y);
```





All code in 3 files

```c
lib.h"

int ma
  int a, b;

  a = 3;
  b = 2;

  c = sum(a, b);

  printf("The sum of a + b = %d + %d = %d\n", a, b, c);

  return 0;
}
```

demo_v2_lib.c

```c
#include "demo_v2_lib.h"

int sum(int x, int y) {
  return (int)(x+y);
}
```

demo_v2_lib.h

```c
#include <stdio.h>

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 !!

```c
#include <stdio.h>

#include "demo_v2_lib.h"

int main(void) {
  int a, b, c;

  a = 3;
  b = 2;

  c = sum(a, b);

  printf("The sum of a + b = %d + %d = %d\n", a, b, c);

  return 0;
}
```

```c
#include "demo_v2_lib.h"

int sum(int x, int y) {
  return (int)(x+y);
}
```

```c
#include <stdio.h>

int sum(int x, int y);
```

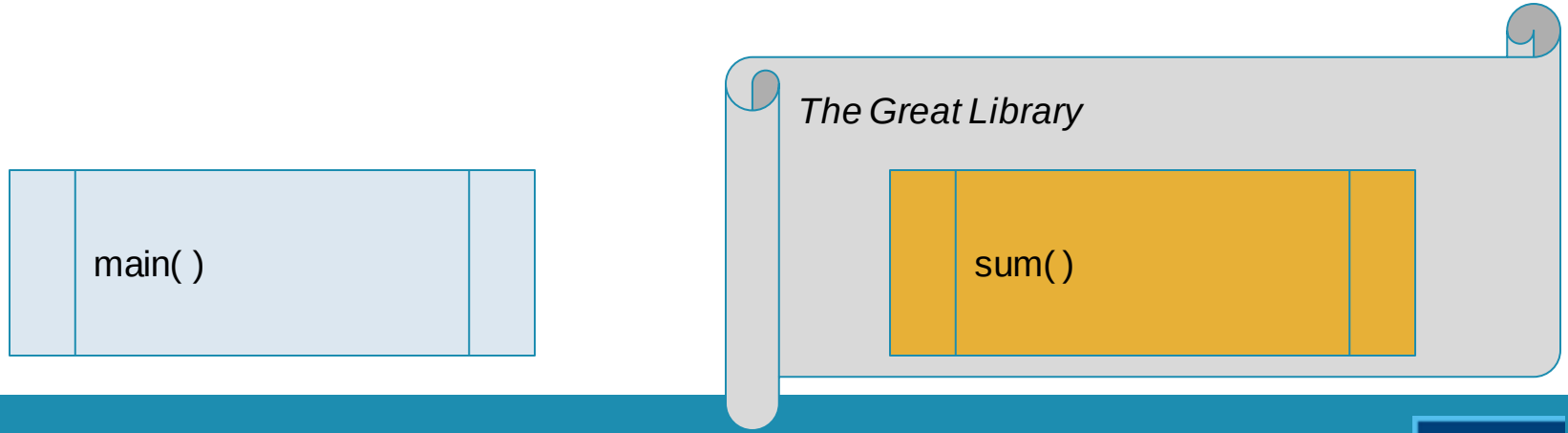
Using a library

code is unaltered !!

```c
#include <stdio.h>

#include "demo_v2_lib.h"

int main(void) {
  int a, b, c;

  a = 3;
  b = 2;

  c = sum(a, b);

  printf("The sum of a + b = %d + %d = %d\n", a, b, c);

  return 0;
}
```

```c
#include "demo_v2_lib.h"

int sum(int x, int y) {
  return (int)(x+y);
}
```

```c
#include <stdio.h>

int sum(int x, int y);
```



Using a library

code is unaltered !!

```c
#include <stdio.h>

#include "demo_v2_lib.h"

int main(void) {
  int a, b, c;

  a = 3;
  b = 2;

  c = sum(a, b);

  printf("The sum of a + b = %d + %d = %d\n", a, b, c);

  return 0;
}
```

```c
#include "demo_v2_lib.h"

int sum(int x, int y) {
  return (int)(x+y);
}
```

```c
#include <stdio.h>

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

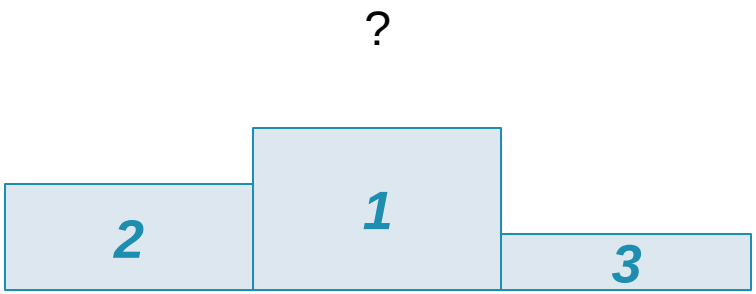
- Ongoing competition: https://csrc.nist.gov/projects/lightweight-cryptography/

- 10 finalists

?

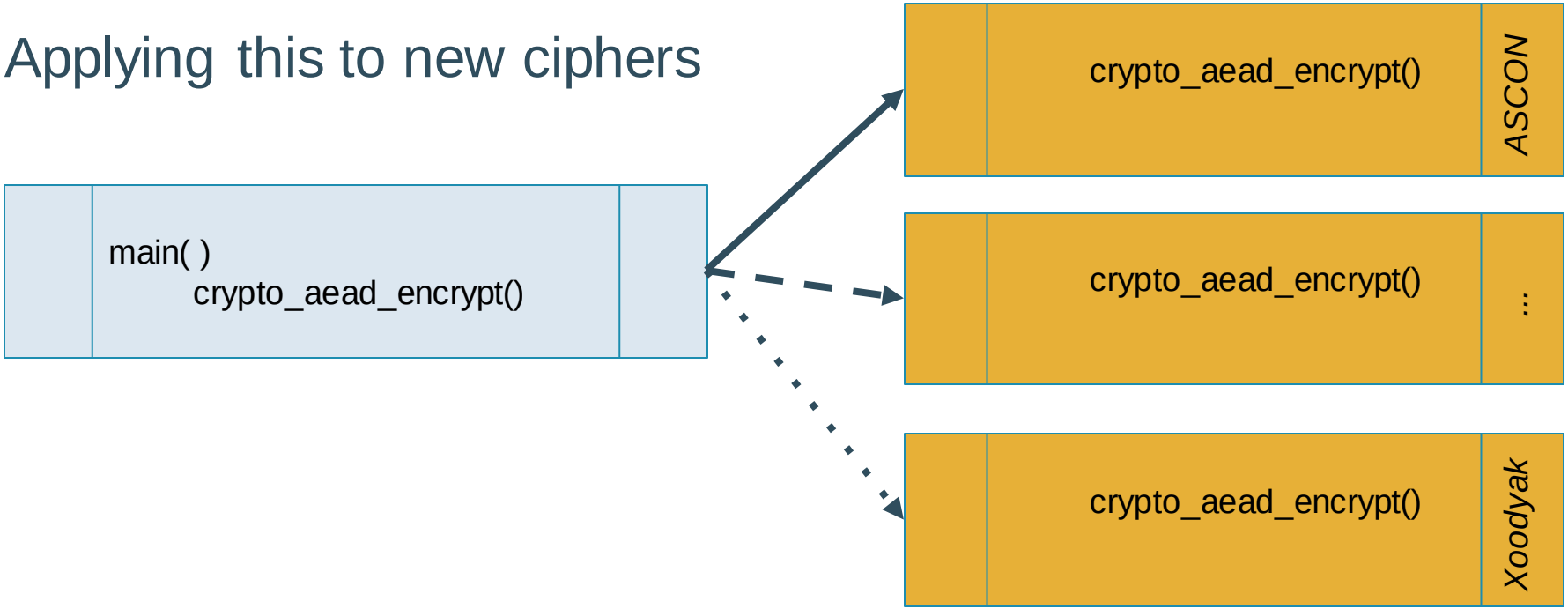
Setting

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



A little help

Applying this to new ciphers



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