Crypto libraries introduction

Milan Brož
xbroz@fi.muni.cz

PV181, FI MUNI, Brno

Open source cryptographic libraries

- Linux environment (with OpenSSL3) – up to you:
  - ssh to aisa.fi.muni.cz
  - Debian / VirtualBox VM (see course materials)
  - Your own distro – need to install development env.:
    - libgcrypt: Fedora: **libgcrypt-devel**; Debian/Ubuntu: **libgcrypt20-dev**
    - OpenSSL:Fedora: **openssl-devel**; Debian/Ubuntu: **libssl-dev**
    - libsodium:Fedora: **libsodium-devel**; Debian/Ubuntu: **libsodium-dev**
- All examples in C language
- We will use only free open-source tools and libraries
- 2x Home assignments (10 points each)

Lab environment, git and VirtualBox image (optional)

- Optional VM install
  - Unpack zip archive from IS
  - Open VirtualBox (click **blue** icon – config file)
  - Login and password is **pv181**
    (same for **sudo** and **root** password)
  - In pc181 home is a script to clone examples
- Examples on gitlab (always **git pull** for updates)

```
git clone https://gitlab.fi.muni.cz/xbroz/pv181.git
make clean; make; ./example
```

- Check that you can **compile and run** examples

```
1_rng_gcrypt, 1_rng_openssl, 1_rng_sodium
```

Cryptographic libraries
Goals for this lab

- Crypto libraries and API / abstraction
- More practical and implementation view
- Why legacy code, compatibility and standards
- Coding practices – in C language
- Defensive approach: **It will fail, be prepared for it :-)**

Why not use a modern language with garbage collection and functional programming and free massages after lunch?
Here’s the answer: Pointers are real. They’re what the hardware understands. Somebody has to deal with them.
You can’t just place a LISP book on top of an x86 chip and hope that the hardware learns about lambda calculus by osmosis.
- James Mickens, https://www.usenix.org/system/files/1311_05-08_mickens.pdf

Why implementation matters

- It works, but …
- How many possible bugs do you see?

```
/* Read a key from Linux RNG */
#include <string.h>
#include <unistd.h>
#include <fcntl.h>

int main(int argc, char *argv[])
{
   int fd;
   char key[32];

   fd = open("/dev/random", O_RDONLY);
   read(fd, key, 32);
   close(fd);
   /* Do something with the key[] */
   memset(key, 0, 32);
   return 0;
}
```

Practically oriented books

- *Jean-Phillipe Aumasson*
  Serious Cryptography:
  A Practical Introduction
  to Modern Encryption (2017)

- *Ferguson, Schneier, Kohno*
  Cryptography Engineering:
  Design Principles and Practical
  Applications (2010)

- *David Wong*
  Real-World Cryptography (2021)