

Engineering Cryptographic Software

The Software assignment

Radboud University, Nijmegen, The Netherlands



Winter 2022

Background

Writing crypto software

1. Start with slow, potentially insecure, but functioning reference implementation in C
2. Remove main sources for timing leakage, i.e.,
 - ▶ remove secret-dependent branches
 - ▶ remove secretly indexed memory access
3. Profile the code, optimize most important routines
4. Typically use assembly for (micro-)architecture specific optimization

Background

Writing crypto software

1. Start with slow, potentially insecure, but functioning reference implementation in C
2. Remove main sources for timing leakage, i.e.,
 - ▶ remove secret-dependent branches
 - ▶ remove secretly indexed memory access
3. Profile the code, optimize most important routines
4. Typically use assembly for (micro-)architecture specific optimization

Typical minimal building blocks

1. Elliptic-curve Diffie-Hellman (ECDH) for key exchange
2. Some streamcipher for bulk data encryption
3. Some symmetric authenticator (MAC)

The assignment

- ▶ Given C reference implementations of
 - ▶ ChaCha20 stream cipher,
 - ▶ Poly1305 authenticator, and
 - ▶ ECDH on Curve25519 in Edwards form,
- ▶ produce optimized implementations for the ARM Cortex-M4

For details see `ecsw2022-assignment.pdf` in Brightspace or at
[https://cryptojedi.org/peter/teaching/ecsw2022/
ecsw2022-assignment.pdf](https://cryptojedi.org/peter/teaching/ecsw2022/ecsw2022-assignment.pdf)

Getting started: Target platform



STM32F407

- ▶ ARM Cortex-M4
- ▶ 32-bit architecture
- ▶ 192 KiB RAM
- ▶ 1 MiB Flash
- ▶ 168 MHz
- ▶ 24 MHz for benchmarking

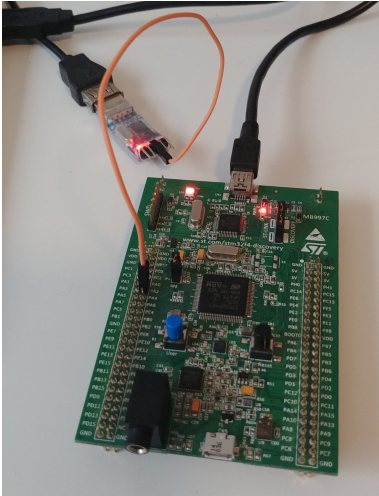
Getting started: Setting up toolchain

- ▶ **Option 1:** Using virtual machine image (recommended)
 - ▶ Ubuntu 22.04
 - ▶ Everything you need pre-installed
 - ▶ First steps in the next slides
- ▶ **Option 2:** Install toolchain on your own Linux
 - ▶ Tutorial: <https://github.com/joostrijneveld/STM32-getting-started>
 - ▶ Depending on your OS, we might not be able to help you

Getting started: Setting up toolchain

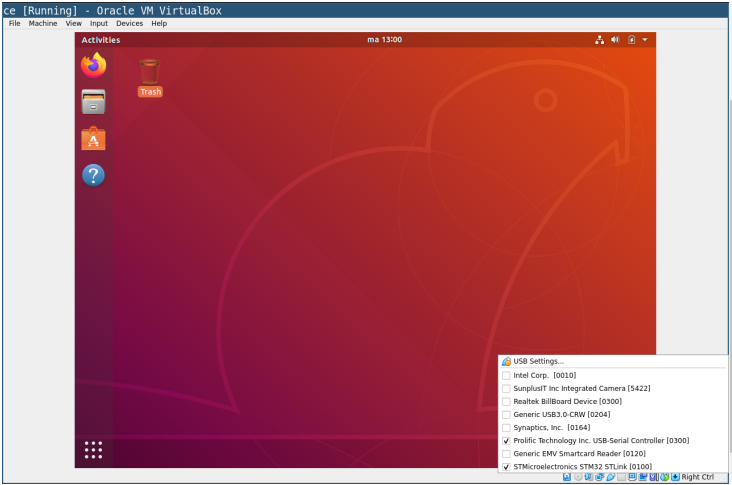
- ▶ Install Virtualbox: <https://www.virtualbox.org/>
- ▶ Download and import image:
<https://sos.cs.ru.nl/teaching/ecsw2022/ecsw2022.ova>
- ▶ Start VM and login with
Username: `ecsw`
Password: `ecsw`
- ▶ `/home/ecsw/ecsw2022-assignment/` contains the assignment

Getting started: Connecting discovery board



- ▶ Connect USB cable to your machine
 - ▶ Used for flashing and as power supply
- ▶ Connect PA2 pin with RXD pin of UART-USB connector
 - ▶ Used for receiving serial output
 - ▶ You may also connect GND with GND

Getting started: Mapping USB devices to VM

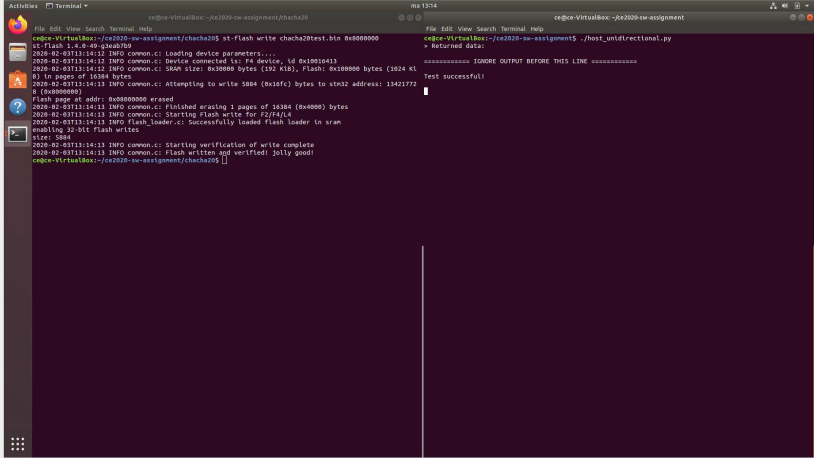


- ▶ Map board and UART-USB connector into the virtual machine

Getting started: Flashing software and receiving output

- ▶ Compile libopencm3 library
`cd ~/ecsw2022-assignment/libopencm3`
`make`
- ▶ Compile binary (e.g., test for chacha20)
`cd ~/ecsw2022-assignment/chacha20`
`make`
- ▶ Flash binary to the board
`st-flash write chacha20test.bin 0x8000000`
- ▶ Receive output
`cd ~/ecsw2022-assignment/`
`./host_unidirectional.py`

Getting started: Flashing software and receiving output



The image shows two terminal windows side-by-side in a virtual machine environment. The left window shows the execution of a flash command, and the right window shows the execution of a host script to receive data from the device.

```
ce@ce-VirtualBox: ~/ce2020-sw-assignment/chacha20$ st-Flash write chacha20test.bin 0x80000000
10-Flash 1.4.0-49-g36a3709
2020-02-03T11:14:12 INFO common.c: Loading device parameters....
2020-02-03T11:14:12 INFO common.c: Device connected is: F4 device, id 0x10010413
2020-02-03T11:14:12 INFO common.c: SRAM size: 0x30000 bytes (192 KiB), Flash: 0x100000 bytes (1024 Ki
0) in pages of 10384 bytes
2020-02-03T11:14:13 INFO common.c: Attempting to write SBB4 (0x10fc) bytes to stn32 address: 13421772
0 (0x00000000)
Flash page at addr: 0x00000000 erased
2020-02-03T11:14:13 INFO common.c: Finished erasing 1 pages of 16384 (0x4000) bytes
2020-02-03T11:14:13 INFO common.c: Starting Flash write for F2/F4/L4
2020-02-03T11:14:13 INFO Flash_loader.c: Successfully loaded flash loader in sram
enabling 32-bit Flash writes
size: SBB4
2020-02-03T11:14:13 INFO common.c: Starting verification of write complete
2020-02-03T11:14:13 INFO common.c: Flash written and verified! Jolly good!
ce@ce-VirtualBox: ~/ce2020-sw-assignment/chacha20$
```

```
ce@ce-VirtualBox: ~/ce2020-sw-assignment$ ./host_unidirectional.py
> Returned data:
===== IGNORE OUTPUT BEFORE THIS LINE =====
Test successful!
```

Distribution of Hardware

Pick up boards on Thursday, Nov. 10

- ▶ Ronny and Ciske will hand out boards
- ▶ One board per group; possibly one board per person
- ▶ Return board after the end of the course
- ▶ **One person** in charge of returning each board