Embedded Security (Summer Term 2018)
Dr. Stefan Nürnberger
CISPA Helmholtz Center i.G. / Saarland University

Exercise #1 (20 Points)

Date due: May 15., 23:59 am

Submission Instructions

- The solutions must be submitted via CMS as a PDF.
- Put names and matriculation numbers of all students who authored the solutions **on the PDF itself**.
- Explain how you solved each task (otherwise 0 points).
- If not stated otherwise, code must be written in *C*, *C++* or *Python*.
- Additional files (e.g. source code) should be added to a .zip (with the PDF solution)
- Label your solutions corresponding to the exercises on the task sheet.
- Handwritten solutions are accepted, but they have to be readable; otherwise points might be deducted.
- If you have questions that might leak deduced information, write a mail to the tutors or visit the office hours instead of creating a post in the forum.

1 The Mysterious Chip (9 points)

A friend of yours works at the stock exchange. He tells you that they recently found a mysterious chip attached to their network switches that might manipulate stock if it received the right command from an insider. He wants you to inspect that chip – but he cannot give you the original because the police is already investigating it. He managed to copy the firmware of the chip and gave it to you.

Your task is to put that firmware on your ATtiny and retrieve the secret that the chip computes and stores in its internal EEPROM memory.

(a) To achieve that task, you have to use the Arduino as an in-circuit serial programmer (ICSP). ICSP is a common way to flash the content (code, data) of a microcontroller while it is already built into a circuit, e.g., already in a sold product. ICSP is an easy way to manufacture products because the bare chips are soldered onto PCBs and the software is flashed into the chip in a later step using ICSP[1].

i. To get started, hook up the Arduino as Master and the ATtiny as slave as shown in Figure 1.
In this setup, the Arduino is able to flash (=overwrite!) the ATtiny's content. To do that, the Arduino needs the ISP software installed on it to work as a programmer for the ATtiny.

[1]The ICSP protocol uses MISO (Master Input Slave Output) and MOSI (Master Output Slave Input) to communicate while flashing. The data is synchronised using SCK (Serial Clock). The receiving chip will only accept to be flashed while its reset line (RST) is pulled low. You dont have to understand all this. The Arduinos software will take care of all this.

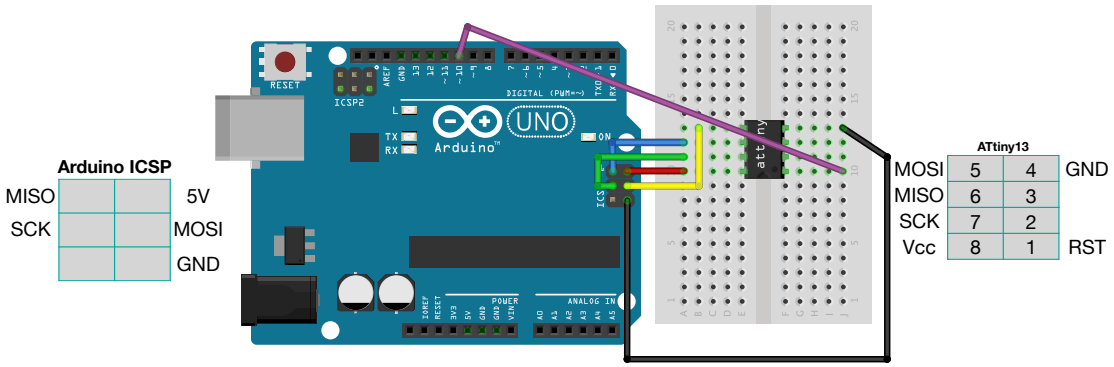

Figure 1: Arduino used as In-Circuit Programmer (ISP) to program the ATtiny

ii. First, install the ISP firmware to the Arduino by issuing the following command:

```
avrdude -p atmega328p -c arduino -P /dev/ttyUSB0 -b 115200
-U flash:w:task1/ArduinoISP.hex
```

You have to replace `/dev/ttyUSB0` with your Arduino's serial port. The file `task1/ArduinoISP.hex` is the ISP software that is to be flashed into the Arduino.

If that step was successful, you can then write the ATtiny's firmware through the Arduino ISP using the following command:

```
avrdude -c avrisp -P /dev/ttyUSB0 -p t13 -b 19200
-U flash:w:task1/ex1_?????-?????.ino.hex
```

Replace the `?????` with the appropriate file name that matches your group's matriculation numbers. Please note that the secrets are personalised and you'll get 0 points of you use somebody else's secrets.

If that step worked, the ATtiny now runs the secret firmware discovered at the stock exchange.

(5 points) iii. Now it's your turn to get the EEPROM memory from the ATtiny13 to check what secret it stored. The EEPROM can be retrieved using

```
avrdude -c avrisp -P /dev/ttyUSB0 -p t13 -b 19200
-U eeprom:r:dump_eep.hex:r
```

The file `dump_eep.hex` now contains the entire 64 bytes of ATtiny EEPROM memory. Now its up to you to find a way to decode the `.hex` file format and find the secret. Please put the secret in your solution.

(2 points) (b) The discovered secret is surrounded by other data. Check if the address of the secret is deterministic and if the other data changes? How did you do that and what was the result?

(2 points) (c) Now that you know how .hex files are built, have a look at the original `ex1_??????-?????.ino.hex` file. Is the secret already visible in there? If yes, where? If no,

give an example of how you would avoid a secret to show up.

(+4 bonus) (d) You can convert the Intel HEX format firmware for the ATtiny to binary and then inspect the instructions using a disassembler, e.g. `https://onlinedisassembler.com/odaweb`. Can you see how the secret is handled and how the address in EEPROM is calculated?

2 Power-Hungry (11 points)

Your friend has discovered another chip that intruders used to open a door. He sent you that chip's firmware as well (`/task2`).

He also told you that this time, the chip was receiving external data using two wires connected to it. These two wires transmit a binary signal using `CLK` and `DATA`. If the signal encodes a secret password, the chip would open the door to which it was connected.

(a) First install the new firmware (`/task2/attiny-sidechannel.hex`) using the ICSP method from before. When your ATtiny runs the new code, disconnect all wires but the `GND` and `5V`. You'll need those wires to send a digital signal from your Arduino in the next step:

Your task is to find the byte sequence that they used to open the door.

i. Connect the Arduino as depicted in Figure 2:

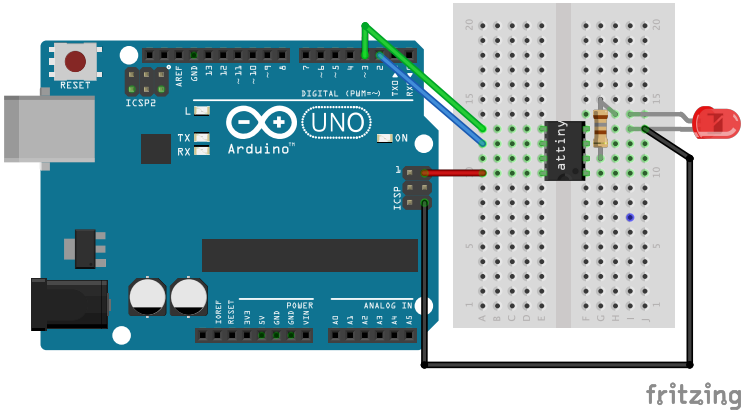

Figure 2: Arduino can send bits using two wires: CLK and DATA. The ATtiny has a feedback LED connected through a 180 Ohm resistor.

The CLK (ATtiny pin 6) and DATA (ATtiny pin 5) are both driven by the Arduino. When the ATtiny is waken up from standby mode (by a high CLK line), it waits for the clock line to go low again. When a falling edge is detected, it reads out the data line (interpreting a high level as 1 and a low level as 0); to receive another bit the clock line needs to go high (and then low) again.

(1 point)

ii. The LED connected to pin 2 will light up every time it receives the byte `0xFE`. This is for you to test the successful setup of your communication. Use the provided file `/task2/task2-arduino-power-analysis.zip` in Platform.IO as a starting point. It can already send data. Add the appropriate code to send `0xFE` (turn on the LED) and check if it works.

(2 points)

iii. Apart from the special `0xFE` byte, each byte is processed as follows:

```
void processPassword()
{
  for(int i = 0; i < strlen(password); i++) {
    byte b = receive_byte();
    if (b == 0xFE) {
      digitalWrite(PIN_LED, HIGH);
    }
    if (b != password[i]) {
      goto_sleep();
    }
  }
}
```

Analyse the code and see which side channels you could exploit. Name them and explain why.

(b) To analyse power consumption, prepare the Arduino and the ATtiny as follows:

i. Arduino: Add a 120 Ohm (brown-red-brown) resistor to the GND line of the ATtiny to be able to measure the current on Analog input 0 (A0) of the Arduino (see Figure 3).

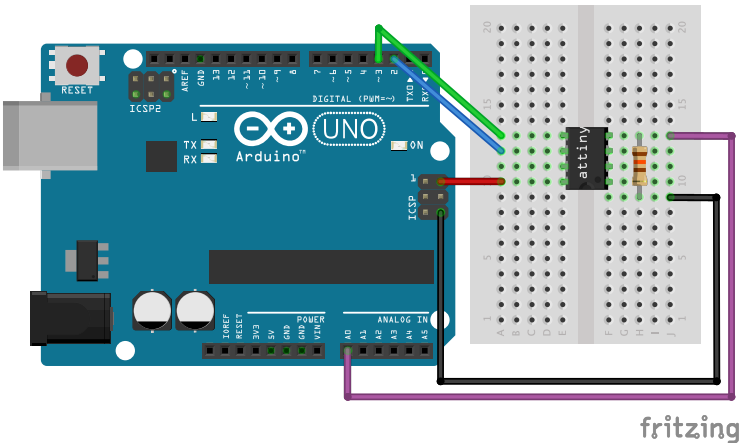

Figure 3: Current measurement using a resistor in series with the ATtiny's GND. The LED is removed in order not to disturb the current reading.

The purple wire[2] will feed the voltage after the resistor back the A0, thereby measuring the voltage drop across the resistor when the ATtiny consumes current. This voltage can then be read by the Arduino's ADC (analog to digital converter).

(1 point) ii. Find out how accurate the Arduino Uno's ADC is (e.g. how many bits) and which lowest number and highest number corresponds to which voltage.

(1 point) iii. Then check the provided Platform.IO source code for the Arduino and find out which maximum voltage the A0 input expects and to which values (integers) the different measured voltages will map.

iv. ATtiny: The current consumption of a processor increases with the clock speed. The clock speed of the ATtiny can be set to up to 9.6 MHz. This is done by setting internal fuses. Use the following command to do so:

```
avrdude -c avrisp -P /dev/YOURPORT -b 19200 -p t13
-U flash:w:task2/attiny-sidechannel.hex
-U lfuse:w:0x7a:m -U hfuse:w:0xff:m
```

(c) Everything is set up now.

(1 point) i. Explain how you think you can find the expected secret using the voltage measurement on A0.

(5 points) ii. Extend the provided Arduino code to smartly iterate over the 9-character key that the ATtiny expects. Can you find the key? How did you do it? What is the key?

[2] You can of course use any colour