Exercises for Module 3

The exercises for the third module are about block and stream ciphers. Live-coding scripts and selected solutions will be posted on the Github repository for the course: https://github.com/henningth/Applied-Cryptography-2024

Bytearrays

Exercise 1: This exercise is about Python bytearrays. Solutions are on the Github page.

(a): Begin by defining an empty bytearray, call it barr1. Also, define a bytes-object with contents “Python 3 programming”. Call it barr2.

(b): Copy the contents of barr2 into barr1, except that the “3” should be a “2”. In other words, the resulting bytearray should read: “Python 2 programming” (Hint: Like lists, you can slice bytearrays: for example, use barr1[0:4] to get the first 4 bytes from barr1.)

(c): Compute the number of bytes in barr1 and barr2. (Hint: this is like computing the number of elements in a list)

(d): What is the value and type of the fifth byte in barr1?

(e): Use a Python-builtin function to find the ASCII character of the fifth byte in barr1?

(f): Use a Python-builtin function to print the hexadecimal representation of the bytearray barr1.

Block Ciphers

The following two exercises use the live-coding of AES-CTR.

Exercise 2: Begin by encrypting a string of your own choosing using AES-CTR. Then, change one byte in the ciphertext, and try to decrypt this modified ciphertext using the same key as before. Does it work, and if so, what is the (modified?) plaintext?

Also change one byte in the key, can you decrypt using the modified key?

Exercise 3: Generate a random ciphertext and decrypt it using the same key and nonce as in the previous exercise. Does it work? Why/why not?

Stream Ciphers

Exercise 4: In this exercise, we are working with the stream cipher ChaCha20 in Python. See https://cryptography.io/en/latest/hazmat/primitives/symmetric-encryption/#cryptography.hazmat.primitives.ciphers.algorithms.ChaCha20 for usage and an example.

(a): Encrypt the plaintext “Hello world” (without quotes) using the stream cipher ChaCha20. Be sure to generate a key and nonce properly. Print the ciphertext in the console, what do you see?

(b): Decrypt the ciphertext obtained in part (a), and check that the original plaintext and decrypted ciphertext are equal (why do we want to check this?)

(c): Change one byte of the ciphertext and decrypt it using the same key and nonce as in part (a). What do you observe?

(d): Change one byte of the key and decrypt the ciphertext obtained in part (b). What do you see?

If you have finished the above exercises, you can look at the ones from Set 1 in Cryptopals: https://cryptopals.com/sets/1. You can start with challenge 2 and afterwards, challenge 3.