DHBW
Duale Hochschule
Baden-Württemberg
CAS
Center for Advanced Studies

IT-Security Cryptography and Secure Communications

Exercise: **Block Ciphers**
Lecturer: *Prof. Dr. Michael Eichberg*
Version: 2023-10-19

Feistel Cipher

1. Implement a feistel chiper in the programming language of your choice (e.g., Java, Scala, Python, C, (JavaScript) ...) that enables you to:
    - encrypt and decrypt messages
    - encrypt blocks of 128 bits
    - easily exchange the function *f* to test the effect of *f* (depending on the language of your choice you can, e.g, use native higher order functions or a function pointer)
    - you can use a function that produces the round keys by simply shifting the key

        Note

        Don't worry about messages that are larger or smaller than the block size. This is not necessary to understand the impact of *f* or using a round key. Don't worry about a key that does not have the appropriate size. I.e., use a message and a key with the appropriate size.
2. What happens if *f* just returns *0x00* values (independent of the round key)?
3. What happens if *f* just returns *0x01* values (independent of the round key)?
4. What happens if *f* simply xors the respective half with the result of the shift of the key?
5. Test what happens when you change your message. In particular test what happens when the message just consists of *0x00* (and you use a "more reasonable" *f* function.)
6. Test what happens when you change your key. What happens in extrem cases (e.g., the password just consists of "0"s?

Solution

A naive Python implementation of the algorithm can be found here:

Jupyter Notbook with Solution

By adapting the above implementation and testing it, it will immediately beome apparent that the use of an inappropriate *f* function will lead to no security at all and that the design of such a function is really hard work. Additionally, it is necessry to consider all possible scenarios.