Final Projects

Elements of Applied Data Security

Alex Marchioni – alex.marchioni@unibo.it

Final Project

- Final project consists in:
  - **Implementation** in Python of one of the proposed final projects.
    - You can also propose a different project, but it must be previously approved by me.
    - You must communicate your choice to me, when you choose the project you are going to work, send me an email
  - **Description** with a report or a brief presentation
    - You can write your report/presentation as notebook together with the implementation

Final Project

- You can **work in groups** (max 4 students), but assessment is individual.
- Assessment consists in a **10/15 minutes discussion**. Discussion takes place on the exam date, but projects must be sent in advance (at least one day).
- Mark is **Pass**/**Fail**. Pass is required to access to the theory exam
- **No deadline**. Just send the project at least one day before the exam.

Final Project

Report/Presentation must contain:

- A title
- A brief contextualization or theoretical explanation of the implemented method.
- A detailed description of the implementation and the choices you made
- A demonstration that your implementation works (comparison with libraries, exhaustive example, etc..)
- Conclusion – just a brief recap of what you have done.

Project proposals

Project 1 - LFSR

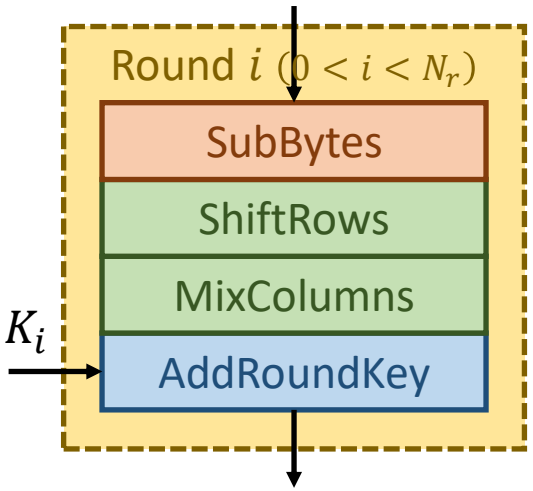
Known plaintext attack to a stream cipher based on an LFSR.

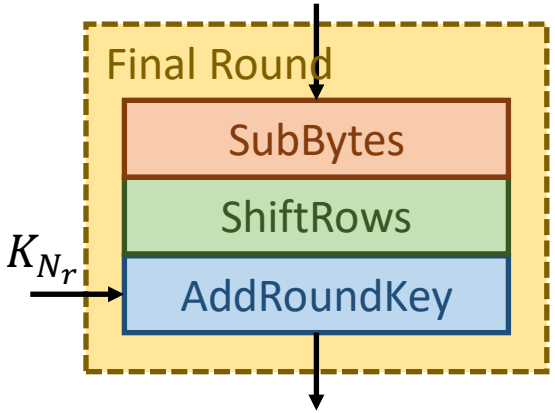
- You are given:
  - a **ciphertext** (proj1_ciphertext.bin) encrypted with a stream cipher that uses an LFSR as a pseudo random bit generator.
  - the **first part of the plaintext** (proj1_known-plaintext.txt) .
- The objective is to **decrypt the rest of the ciphertext**.

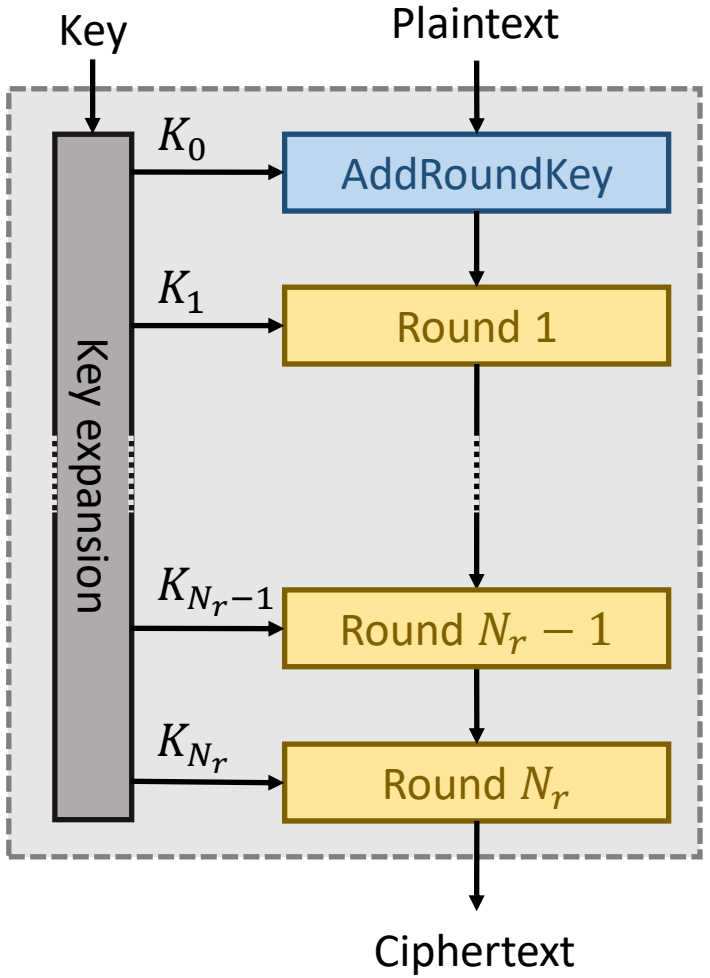
Note that the ciphertext has been produced by using the classes and functions showed during the lessons and available on virtuale and GitHub.

Project 2 - AES

Implementation the **AES encoder** with key length 128 bit.







Project 3 - RSA

Implementation of the **RSA** public-key cryptosystem (n, e, d)

- Key generation
  - Generation of two big prime numbers p and q for the factorization of n. (Usage of the Miller-Robin property for primality test).
  - Choice of e and computation of d. (Usage of the Extended Euclidean algorithm)
- Encryption and decryption
  - Exponentiation of plain/ciphertext

Project 4 - Hash

Use Monte Carlo Simulation to compute the **average collision time** for two different **Hash** functions H_{16} and H_{32}.

- H_{16} takes the output of MD5 and splits it into 16 bits words that are xored to yield a 16 bits output.
- H_{32} takes the output of MD5 and splits it into 32 bits words that are xored to yield a 32 bits output.

You can use the library pycriptodome for the MD5 primitive.