Assignment 4 -

Breaking A Synchronous Stream Cipher

with Reused Keystream

Security Tools Lab 2

Hand out : 5-Mar-2019

Hand in: 13-Mar-2019

1. **Objectives**

By the end of this lab, you should be able to:

- Gain an understanding of synchronous stream ciphering
- Understand that keys should never be reused for stream ciphering
- Attack a ciphertext with reused keystream

System

- Ubuntu Linux machine/VM

Notes

Download all needed files from Edimension.

2.

You are given several files that contain ciphertexts (and in one case a corresponding plaintext), which were encrypted by an unknown stream cipher. Your main goal is to recover the secret string (i.e., seed) that was used to generate the keystream. The secret string has form **INS{...23 characters...}**

Note that all the files were encrypted with the same keystream (i.e., using the same input key), resulting into reuse of keystream, which is the first security vulnerability you should exploit. Then, you may consider XOR function used for encryption/decryption, which has 2 inputs: keystream and plaintext/ciphertext.

3.

Leverage the fact that the keystream is being reused in order to obtain the first part of keystream.

With this keystream, you may (partially) decrypt the remaining files and see the first part of the cipher that was used in this laboratory.

You will see that a secret seed is passed as an argument to the cipher and function next() generates an unlimited keystream using its previous output. Thanks to that you can recover the full encrypted files – full super_cipher.py and hint.gif.

Recovered picture outlines you how the cipher was used and demonstrates what you have to do with the function next() that generates keystream in order to obtain the secret seed.

4.

In your solution, write a python3 script that will recover the secret seed used for generating the stream cipher. Hint: “transform” the function next() and realize that triplets of bits are mapped to a single bit (with significant overlap). Here, you will exercise bit operations in python, such as bit shifts, AND, OR.

5.

Use python3 language and hand in one zipped file on Edimension. The archive will include
1) solution.py, containing your programmed solution
2) doc.pdf with your write-up
3) bash script install.sh, which will install all non-standard dependencies required for running of your solutions. This script will be run as the first.

The solution will be tested on contemporary Linux Ubuntu machine.

Rubrics : 4.5 for solution & 1.5 for report quality (Total marks = 6)

# Submission is individual though you can discuss with your classmates. Solution will be tested for plagiarism.