CSCE 4050/5050 Applications of Cryptography

Programming Project 1 - Exhaustive key search

Due on March 3 (Fri) at 11:59pm

Exhaustive search is a generic attack on computationally secure ciphers.

Project Task: Write a program that runs an exhaustive search attack on a block cipher.
As an example, we choose AES-128 cipher in the randomized counter mode (CTR) with a key space effectively limited to **24 bits**. Specifically, the key has the following format (in hex): 80 00 … 00 XX XX XX, where "X" denotes an arbitrary hex number. In other words, the first 13 bytes of the key are fixed to be zeroes (except that the leading bit is "1"), and the last 3 bytes can be arbitrary
(so that they need to be checked by the exhaustive search).

Your initial task is to find the key, which matches the provided three plaintext-ciphertext pairs. The plaintexts are given as the files "m1.txt", "m2.txt", and "m3.txt", the respective ciphertexts are in the files "c1.bin", "c2.bin", and "c3.bin", and the respective nonces are in the files "nonce1.bin", "nonce2.bin", and "nonce2.bin". Specifically, the plaintext files contain some parts of HTTP headers, so they can be obtained by an adversary eavesdropping in a user's communication.

Now, suppose that the adversary obtained a ciphertext of some important message whose contents is not known. This ciphertext (which we will call a "challenge ciphertext") is placed in the file "c_c.bin" and the respective nonce is in "nonce_c.bin". Your final task is to use the key obtained in the previous step to learn this important message.

All the above files are enclosed in the ZIP archive "challenge.zip" and posted on the assignment page. Also enclosed in this archive are the Python program, which was used to generate the above mentioned files ("setup_demo.py") and a file "util_demo.py" contained some functions.
The code from these files may be re-used for your project. This program utilized the PyCryptodomex package (https://pypi.org/project/pycryptodomex/).

Feel free to use the programming language of your choice. Python is recommended.
For Python, the package PyCryptodomex (https://pypi.org/project/pycryptodomex/) is recommended.

Tip: Implement the (straightforward) algorithm described in Lecture 4-2.

Organization: This project allows group work. One project report per group should be submitted by any of the group members. Both group members will receive the same grade.

Submission requirements:

- The project report (2-3 pages) must contain a short description of the task (exhaustive key search), a description of your program and the algorithms which it implements, and screenshots demonstrating that your program works according to the project task. Also, the report must contain the key, which was found, and the message obtained by decrypting the challenge ciphertext.
- The source code, the executable(s) if any, and all other related files must be enclosed as a ZIP archive. The source code must be properly commented. Your program which implements

exhaustive search must write the key, which was found, to a file. This file must be included into your submission. In addition, you must write a short program, which reads the challenge ciphertext (and nonce) as well as the key from the respective files, and it must print the decrypted message.

Submission tip: It is convenient for the instructor, if your submission consisted of two files: a project report (in DOCX or PDF format), and all other files in a ZIP archive.

Remark 1: Failure to provide any of the above listed items will result in reduced grades.

Remark 2: It is students' responsibility to demonstrate that the implementation is working correctly. In cases when the code cannot be run, screenshots placed in the report and comments placed in the code will serve as evidence that the code is working, and that the authorship is attributed properly.

Rubric:

- The code is worth 80 points, specifically:
  - The code is working: 70 points.
    - Either the instructor is able to run it, or they are convinced via inspecting the code and the screenshots provided in the report.
  - The code is properly commented: 10 points.
- The project report is worth 20 points.
  - A detailed description of the task (exhaustive key search) and the implemented algorithm must be provided. The report must be understandable without prior knowledge of cryptography.

Remark: CSCE 4050 (undergraduate) students receive 5 bonus points, as long as the project submission is made.