Computer Security
Dept. of Computer Engineering,
Chulalongkorn University.

Activity IV - Fundamental of Cryptography

Created by : Krerk Piromsopa, Ph.D

Overviews

In this activity, we will learn the basics of encryption. There are 3 exercises in this activity. Each exercise is designed to let you learn the concepts of cryptography.

We will need:

- Imagemagick
- OpenSSL
- One of your favorite programming languages.

You are welcome to do this exercise with any programming language. If you have no preference, use python.

Exercises

1. (Cryptanalysis) Though encryption is primarily designed to preserve confidentiality and integrity of data, the mechanism itself is vulnerable to brute force (statistical analysis). In other words, the more we see the encrypted data, the easier we can hack it. In this exercise, you are asked to crack the following cipher text. Please provide the decrypted result and explain your strategy in decrypting this text.

Cipher text

PRCSOFQX FP QDR AFOPQ CZSPR LA JFPALOQSKR. QDFP FP ZK LIU BROJZK MOLTROE.

   a. Count the frequency of letters. List the top three most frequent characters.
   b. Knowing that this is English, what are commonly used three-letter words and two-letter words. Does the knowledge give you a hint on cracking the given text?
   c. Cracking the given text. Measure the time that you have taken to crack this message.
   d. Create a simple python program for cracking the Caesar cipher text using brute force attack. Explain the design and demonstrate your software. (You may use an English dictionary for validating results.)

2. (Cryptanalysis on Symmetric Encryption) Vigenère is a complex version of the Caesar cipher. It is a polyalphabetic substitution.

    a. Please review **Kasiski examination**[1] Explain how it can be used to attack Vigenère.

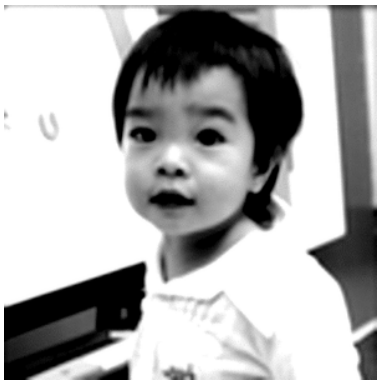
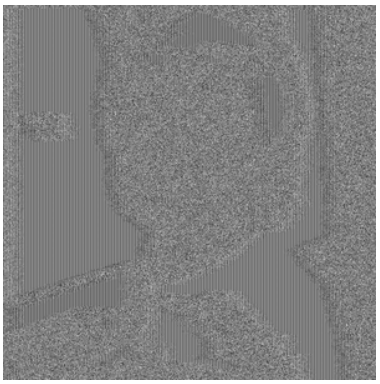
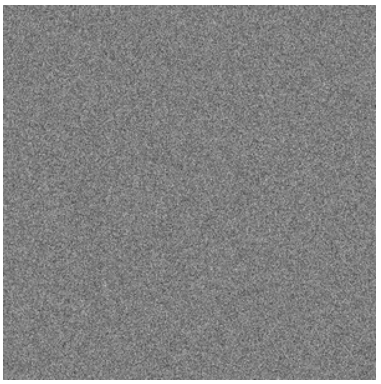
3. (Mode in Block Cipher) Block Cipher is designed to have more randomness in a block. However, an individual block still utilizes the same key. Thus, it is recommended to use a cipher mode with an initial vector, chaining or feedback between blocks. This exercise will show you the weakness of **E**lectronic **C**ode **B**ook mode which does not include any initial vector, chaining or feedback.
    a. Find a bitmap image that is larger than 2000x2000 pixels. Note that you may resize any image. To simplify the pattern, we will change it to bitmap (1-bit per pixel) using the portable bitmap format (pbm). In this example, we will use imagemagick for the conversion.
    ```
    $ magick convert  image.jpg -resize 2000x2000 org.pbm
    ```
    b. The NetPBM[2] format is a naive image format.The first two lines contain a header (format and size in pixel). Depending on the format, the pixels can be represented in either binary and ascii. For our exercise, we prefer binary. However, we first have to take out the header to prevent the encryption from encoding the header. To do so, use your text editor (eg. vi, notepad) to take out the first two lines.
    ```
    $ cp org.pbm org.x
    $ vi org.x
    P4
    2000 2000
    KR)B@HD◆◆@H◆
    ```

    c. Encrypt the file with OpenSSL[3] with any block cipher algorithm in ECB mode (no padding and no salt).
    ```
    $ openssl enc -aes-256-ecb  -in org.x -nosalt \
          -out enc.x
    ```

    d. Pad the header back and see the result.
    ```
    $ cp enc.x enc.pbm
    $ vi enc.pbm
    P4
    2000 2000
    KR)B@HD◆◆@
    ```
    e. You may try it with other modes with IV, chaining, or feedback and compare the result.
    f. What does the result suggest about the mode of operation in block cipher? Please provide your analysis.

[1] See wikipedia for more details. https://en.wikipedia.org/wiki/Kasiski_examination
[2] See wikipedia for more details. https://en.wikipedia.org/wiki/Netpbm
[3] For details on command-line arguments, see https://wiki.openssl.org/index.php/Enc

If you got it all right, the result should be like this.

Original	AES-256-ECB	AES-256-CBC
		

4. (Encryption Protocol - Digital Signature)
    a. Measure the performance of a hash function (sha1), RC4, Blowfish and DSA. Outline your experimental design.
    (Please use OpenSSL for your measurement)
    b. Comparing performance and security provided by each method.
    c. Explain the mechanism underlying Digital Signature. How does it combine the strength and weakness of each encryption scheme?

Hint: (OpenSSL command line)

```
# List algorithms
$ openssl list cipher-algorithms
# To encrypt
$ openssl enc -ciphername [options] -e -in filename -out filename \
      -K key -iv IV -nopad -nosalt
```