

Class activity

A hands-on exercise from the class-activities appendix. The full text also appears in Appendix A of the book.

Accompanies *Computer Security: The Foundations*, Krerk Piromsopa, Ph.D., 2026.
<https://www.cp.eng.chula.ac.th/~krerk/books/ComputerSecurity/>

Authorized use only

These exercises involve real attack techniques. Carry them out only against systems you own, or that you have written permission from the owner to test. Unauthorized access, scanning or interception is a criminal offence under Thailand's Computer Crime Act and its equivalents elsewhere, whether or not any damage results. If you are unsure whether a target is in scope, ask your instructor before you start.

Activity 6: Fundamental of Cryptography

Overview

In this activity you will learn the basics of encryption through three exercises, each designed to illustrate a core cryptographic concept.

Tools required:

- ImageMagick
- OpenSSL
- A programming language of your choice (Python recommended)

Exercises

1. **Encryption and Statistical Analysis.** Although encryption is primarily designed to preserve confidentiality and integrity of data, the mechanism itself is vulnerable to brute-force (statistical) analysis: the more ciphertext we observe, the easier it becomes to crack. In this exercise you are asked to crack the following ciphertext.

PRCSOFQX FP QDR AFOPQ CZSPR LA JFPALQSKR.
QDFP FP ZK LIU BROJZK MOLTR0E.

- (a) Count the frequency of each letter. List the top three most frequent characters.
- (b) Knowing that this is English, what are commonly used two- and three-letter words? Does that knowledge provide a hint for cracking the text?
- (c) Crack the ciphertext and record how long it took.
- (d) Explain your cracking process.
- (e) If you know that the encryption scheme is a cipher disc (monoalphabetic substitution), does that allow you to crack it faster?
- (f) Draw a cipher disc for the given text.

- (g) Write a simple Python program that cracks a Caesar cipher using brute force. Explain the design and demonstrate your program. (You may use an English dictionary to validate results.)
2. **Cryptanalysis on Symmetric Encryption.** The Vigenère cipher is a more complex version of the Caesar cipher: it is a polyalphabetic substitution.
- (a) Review the *Kasiski examination* and explain how it is used to attack Vigenère. Friedrich Kasiski published the method in 1863, the first published break of a polyalphabetic cipher; Charles Babbage had reached the same result years earlier but never published it. Your explanation should cover why repeated sequences in the ciphertext betray the key length, what the greatest common divisor of the distances between those repetitions tells you, and why—once the key length is known—the problem collapses into several independent instances of Exercise 1.
 - (b) Based on the Confederate Cipher Disc, explain how it can be used to cipher data.
 - (c) If the key is the word CAT, how many cipher discs do we need? Analyze the security level of Vigenère compared with that of the Caesar cipher.
 - (d) Write a Python program for ciphering data with Vigenère.
3. **Modes in Block Cipher.** Block cipher is designed to introduce more randomness within each block. However, when every block uses the same key without chaining or feedback, patterns can leak. This exercise demonstrates the weakness of Electronic Code Book (ECB) mode.
- (a) Find a bitmap image larger than 2000×2000 pixels. Convert it to the portable bitmap format (PBM) with ImageMagick:


```
$ convert image.jpg -resize 2000x2000 org.pbm
```
 - (b) The NetPBM format stores a two-line header (format identifier and pixel dimensions) followed by binary pixel data. Remove the header before encryption so that it is not encoded:


```
$ cp org.pbm org.x
$ vi org.x          # delete the first two lines (P4 and "2000 2000")
```
 - (c) Encrypt the header-stripped file with OpenSSL using AES-256-ECB (no padding, no salt):

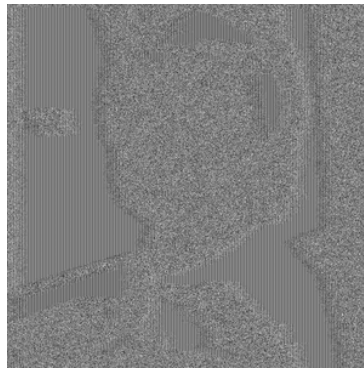
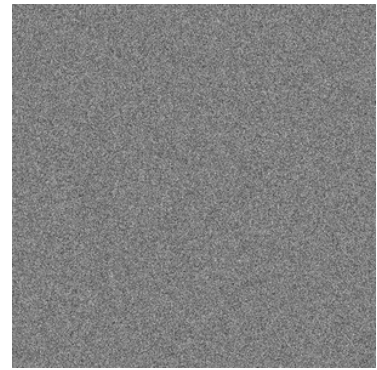

```
$ openssl enc -aes-256-ecb -in org.x -nosalt \
-out enc.x
```
 - (d) Restore the header and view the result:


```
$ cp enc.x enc.pbm
$ vi enc.pbm      # add back:  P4\n2000 2000\n
```
 - (e) Repeat with a mode that uses an initialization vector, chaining, or feedback (e.g. AES-256-CBC) and compare the two results.
 - (f) What does the result reveal about the weaknesses of ECB mode? Provide your analysis.

If correct, your ECB-encrypted image should still show the outline of the original content, while the CBC-encrypted image should appear as random noise.



(a) Original image

(b) AES-256-ECB encrypted
(outline still visible)(c) AES-256-CBC encrypted
(random noise)

ECB mode leaks structure; CBC mode does not.

4. Encryption Protocol — Digital Signature.

- (a) Measure the performance of SHA-1, RC4, Blowfish, and DSA using OpenSSL. Outline your experimental design.
- (b) Compare the performance and security provided by each method.
- (c) Explain the mechanism underlying a Digital Signature: how does it combine the strengths and weaknesses of each encryption scheme?

Hint (OpenSSL command-line reference):

```
# List available algorithms
$ openssl list cipher-algorithms

# Encrypt a file
$ openssl enc -ciphername [options] -e -in infile -out outfile \
  -K key -iv IV -nopad -nosalt
```

From *Computer Security: The Foundations*, Kerk Piromsopa, Ph.D.
Reproduce and adapt freely for non-commercial classroom instruction, with attribution. Full terms: LICENSE.txt in the student package.