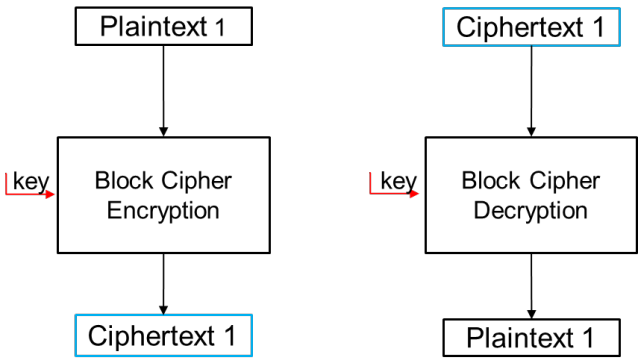
The University of Newcastle
School of Information and Physical Sciences

SENG2250/6250 System and Network Security
Week 4 Lab

PART 1: Block Cipher Modes of Operation

Block ciphers are limited by their input size, that is, they can only encrypt plaintext up to n bits in length. So, to address this issue, block cipher modes of operation were created. Modes of operation are a set of algorithms that use the block cipher as a "black box" and perform other operations so that they can encrypt plaintexts of any length.

Electronic Code Book (ECB): The ECB mode of operation is relatively simple: It divides the plaintext into "blocks" of the same size as the input to the block cipher in use (padding one of the blocks if they cannot be divided evenly), it then individually encrypts each plaintext, and finally concatenates the results to form the ciphertext.



In the following questions, we will study Electronic Code Book (ECB), including highlighting its limitations which motivated the development of more advanced modes of operation. We will use Data Encryption Standard (DES) in ECB mode for encryption via a DES calculator such as https://emvlab.org/descalc/. We will use the following plaintext (P) and key (K):

P=120647265616D2E2054686174206F6E65206461792065766572792070657273 6F6E20696E2074686973206E6174696F6E2077696C6C20636F6E74726F6C207468656 9722 06F776E2064657374696E792E2041206E6174696F6E206F66207468652074727 56C7920667265652C2064616D6D6974

K = 6E616E6F6D63686E

Q1: How many blocks are there in the plaintext?

Q2: Does the plaintext need padding? If so, add a padding. Explore the different methods of padding, what are their advantages and disadvantages.

Q3: Encrypt the first block of plaintext and show the result.

Q4: Encrypt the plaintext and show the full ciphertext.

Part 2: Programming

In this part, we will explore the limitations of the ECB mode of operation by developing a program that encrypts an image file and displays the result.

Q5: Develop a program that encrypts logo.jpg on Canvas and displays the result. The sample code files EncryptImage.java and encrypt_image.py on Canvas show how to read image files, encrypt using DES in ECB mode, and then shows how to display an array of bytes as an image.

Q6: Compare the result to the original image, do they still look similar? What would a mode of operation need to do to mitigate this issue?

PART 3: Avalanche Effect

In this section, we will explore where the DES cryptosystem exhibits the avalanche effect. The avalanche effect is a property of block ciphers wherein a minor alteration in the plaintext results in a significant change in the resulting ciphertext.

Like earlier, we will use the DES calculator such as https://emvlab.org/descalc/ and our initial plaintext (P) and key (K) will be:

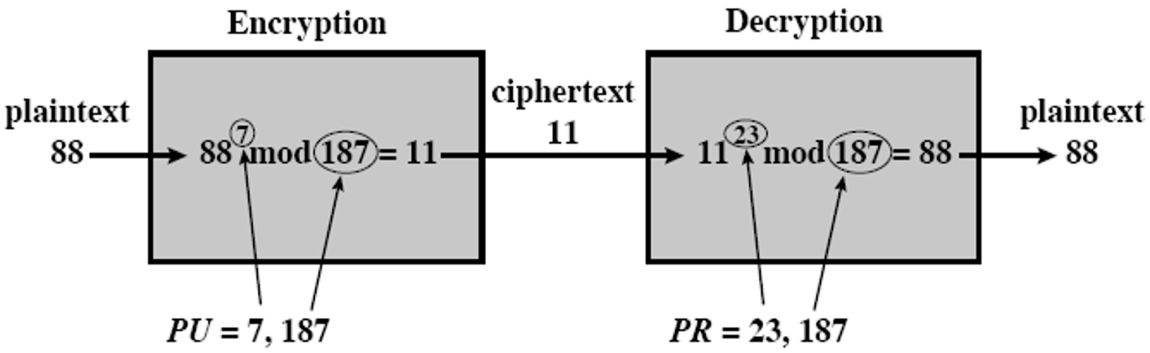
P_0 = 120647265616D2E2
K = 6E616E6F6D63686E

Q7: Find 3 unique single bit flips of P0, for reference in the following question call them P_1, P_2, and P_3 respectively.

Q8: Encrypt P_0, P_1, P_2, and P_3, for reference in the following question call them C_0, C_1, C_2, and C_3 respectively.

Q9: How many bits are different between each C_i? Do you think DES provides the avalanche effect? Why or why not?

PART 4: Asymmetric Cryptography - RSA



- **Algorithm**
  - Choose $n = p \times q$, where p and q are primes.
  - Calculate $\varphi(n) = (p-1)(q-1)$.
  - Choose $1 \leq e < \varphi(n)$ such that $\gcd(e, \varphi(n)) = 1$.
  - Compute d from $(e \times d) \, mod \, \varphi(n) = 1$.
- **Public Key:** (e, n)
- **Private Key:** (d, p, q)

In the RSA scheme with the public key (e, n) = (5, 35), answer the following questions.

Q10: Encrypt the plaintext M = 12.

Q11: Break the cipher by finding p, q and d, where n = p×q and (d, n) is the private key.

Q12: Decrypt the ciphertext found in Q1 and verify that your RSA cryptosystem works with the chosen public and private key pair (e, d).