

# INFORMATION AND NETWORK SECURITY LAB

## Lab Manual 2 (Group-A) report

### Submitted To

Fazle Rabbi Rakib  
Lecturer  
Department of Software Engineering  
Shahjalal University of Science and  
Technology

### Submitted By

Mahmudul Hasan Mridul  
Reg No: 2020831005



Department of Software Engineering

Shahjalal University of Science and Technology

## Checkpoint 1:

**Objective:** Break a Caesar cipher through brute force attack

### Problem Statement

Given cipher text: **odroboewscdrolocdcwkbdkmyxdbkmdzvkdpybwyeddrobo**

**Task:** Write a program to decrypt this Caesar cipher and recover the original plaintext.

## Approach and Methodology

### Understanding Caesar Cipher

The Caesar cipher is a substitution cipher where each letter in the plaintext is shifted by a fixed number of positions in the alphabet. For example, with a shift of 3:

$$A \rightarrow D, B \rightarrow E, C \rightarrow F$$

### Attack Strategy: Brute Force

Since there are only 26 possible shifts (0-25) in the English alphabet, I employed a brute force attack approach:

1. Try all possible shifts - Iterate through shifts from 0 to 25
2. Decrypt with each shift - Apply reverse transformation for each shift value
3. Examine outputs - Manually identify which decrypted text makes sense in English

### Decryption Formula

For each character in the cipher:

$$\text{decrypted\_char} = ((\text{cipher\_char} - 'a' - \text{shift} + 26) \% 26) + 'a'$$

The + 26 ensures the result remains positive when wrapping around the alphabet.

### Result:

Trying all possible shifts:

Shift 0: odroboewscdrolocdcwkbdkmdzvkdpybwyeddrobo

Shift 1: ncqnandvrbcqnknbcbvjaclxwcailcyujcoxavxdccqnan

Shift 2: mbpmzmcuqabpmjmabauizbkwvbzikbxtibnwzuwcbbpmm

Shift 3: laolylbtpzaolilzazthyajvuayhjawshamvytvbaaoly

Shift 4: kznkxkasoyznkhkyzysgxziutzxgizvrgzluxsuazznkxk

Shift 5: jymjwjzrnxyjmjgjxyxrfwyhtsywfhuyuqfyktwrtzyymjwj

Shift 6: ixliviymwxlfiwxwqevxgsrxvegxtpexjsvqsyxylivi

Shift 7: hwkhuhxplvwkhehvwpduwfrqwudfwsodwiruprxwwkhuh

Shift 8: gvjgtgwokuvjgdguvuoctveqpvtcevrncvhqtoqwvvjgtg

Shift 9: fuijsfvnjtuifctutnbsudpousbduqmbugpsnpvuuisf

Shift 10: ethereumisthebestsmartcontractplatformoutthere

Shift 11: dsgdqdtlhrsgdadrsrlzqsbnmsqzbsokzsenqlntssgdqd

Shift 12: crfcpcskgqrfzczqrqkypramlrpyarnjyrdmpkmsrrfcpc

Shift 13: bqebobrfjpqebypqpjxoqzlkqoxzqmixonclojlrqebob

Shift 14: apdanaqieopdaxaopoiwnpykjpnywplhwpbknikqppdana

Shift 15: zoczmzphdnoczwznnonhvmoxjiomvxokgvoajmhjpoozczmz

Shift 16: ynbylyogcmnbyvymnmgulnwihnlwnjfunzilgionnbyly

Shift 17: xmaxkxnfblmaxuxlmftkmvhgmktvmietmyhkfhnmmmaxkx

Shift 18: wlzjwmeaklzwtklkesjlugfljsulhdsxgjegmllzjw

Shift 19: vkyvivldzjkyvsvkjdrktfekirtkgcrkwfidflkkyviv

Shift 20: ujuhukcyijxuruijcqhjsedjhqsjfbqjvehcekjjxuhu

Shift 21: tiwtgtjbxhiwtqthihbpgirdcigprieapiudgbdjiwtgt

Shift 22: shvsfsiawghvpsghgaofhqcbhfoqhdzohtcfacihhvsfs

Shift 23: rgurerhzhvfgurorfgfznegpbagenpgcyngsbezbhggurer

Shift 24: qftqdqgyueftqnqefeymdfoazfdmofbxmfradyagfftqdq

Shift 25: pespcpfxtdespmpdedxlcenzyeclneawleqzcxzfeespcp

## Checkpoint 2:

### Objective: Breaking Substitution Ciphers

**Substitution Cipher:** A substitution cipher replaces each letter of the alphabet with another letter. For example, every 'a' might become 'x', every 'b' might become 'k', and so on. Unlike Caesar cipher where all letters shift by the same amount, substitution cipher uses a random mapping, making it more complex to break.

**Approach:** I used frequency analysis to break both substitution ciphers.

**Frequency Analysis:** I used frequency analysis as the main technique to break these ciphers. This method exploits a fundamental weakness in substitution ciphers: they don't hide the frequency patterns of letters.

#### Step 1: Understanding Letter Frequencies

In English language, letters don't appear equally. Some letters are very common while others are rare:

1. Most common letters: E (12.22%), T (9.67%), A (8.05%), O (7.63%), I (6.28%)
2. Moderately common: N, S, H, R, D, L
3. Rare letters: J, Q, X, Z (less than 0.2% each).

#### Step 2: Analyzing the Ciphertext

First, I counted how many times each letter appears in the ciphertext:

1. Read through the entire ciphertext
2. Count each letter (ignoring spaces and punctuation)
3. Calculate the percentage for each letter

Sort letters from most frequent to least frequent

For example, if letter 'x' appears 120 times in a 1000-letter text, its frequency is 12%.

### **Step 3: Creating the Mapping**

This is the key step where we match cipher letters to English letters:

1. Take the most frequent cipher letter → map it to 'e' (most common in English)
2. Take the second most frequent cipher letter → map it to 't' (second most common)
3. Take the third most frequent cipher letter → map it to 'a' (third most common)

Continue this pattern for all 26 letters

**Logic:** If the ciphertext is English, the most frequent cipher letter is likely representing 'e', the second most frequent is likely 't', and so on.

### **Step 4: Applying the Mapping**

Once the mapping is created, apply it to decrypt:

1. Go through each letter in the ciphertext
2. Replace it with its corresponding plaintext letter from the mapping
3. Preserve spaces, punctuation, and capitalization

### **Step 5: Verification**

Read the decrypted text to check if it makes sense. If the frequency analysis was accurate (which happens with longer texts), the decrypted text should be readable English.

**Results:** Both ciphers were successfully decrypted using frequency-based substitution.

**Which cipher was easier to break?**

**Cipher 2 was easier to break.**

**Explanation:**

1. Cipher 2 has more letters than Cipher 1.
2. Longer texts provide more accurate frequency distribution that closely matches standard English patterns
3. With more data, statistical analysis becomes more reliable and the frequency-based mapping produces better results

The code was run on google colab and using Python language.