## Indian Institute of Information Technology Surat

****

# Lab Report on

# Network Security (CS 702) Practical

**Submitted by**

### [RAHUL KUMAR SINGH] (UI21CS44)

**Course Faculty**

### Dr. Reema Patel

## Department of Computer Science and Engineering

## Indian Institute of Information Technology Surat

## Gujarat-394190, India

**Aug-2024**

## Lab No: 2

## Aim:

To implement encryption, decryption, and cryptanalysis of Caesar Cipher using Java.

## Description:

Write the Menu driven Program for following Cipher and Cryptanalysis.

1. Encryption and Decryption of Ceaser Cipher

2. Cryptanalysis of Ceaser Cipher – Brute Force attack, Frequency Analysis.

Input: File - Large Plaintext (file.txt)

Output: File - Encoded Text (Cipher.txt)

Input File Name: Plaintext.txt

Encrypted File Name: Cipher.txt

Decrypted Filename: Recover.txt

## Code:

import javax.swing.\*;

import java.awt.event.\*;

import java.io.\*;

import java.nio.file.\*;

import java.util.\*;

public class CeaserCipherGUI {

public static void main(String[] args) {

JFrame frame = new JFrame("Ceaser Cipher Encryption and Decryption");

frame.setSize(400, 300);

frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

JButton encryptButton = new JButton("Encrypt (Ceaser Cipher)");

JButton decryptButton = new JButton("Decrypt (Ceaser Cipher)");

JButton bruteForceButton = new JButton("Cryptanalysis - Brute Force");

JButton freqAnalysisButton = new JButton("Cryptanalysis - Frequency Analysis");

encryptButton.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

performEncryption();

}

});

decryptButton.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

performDecryption();

}

});

bruteForceButton.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

performBruteForceAttack();

}

});

freqAnalysisButton.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

performFrequencyAnalysis();

}

});

JPanel panel = new JPanel();

panel.add(encryptButton);

panel.add(decryptButton);

panel.add(bruteForceButton);

panel.add(freqAnalysisButton);

frame.add(panel);

frame.setVisible(true);

}

public static void performEncryption() {

try {

String inputFileName = "Plaintext.txt";

String outputFileName = "Cipher.txt";

String content = new String(Files.readAllBytes(Paths.get(inputFileName)));

int key = Integer.parseInt(JOptionPane.showInputDialog("Enter the Key for Ceaser Cipher (1-25):"));

String encrypted = encryptCeaserCipher(content, key);

Files.write(Paths.get(outputFileName), encrypted.getBytes());

JOptionPane.showMessageDialog(null, "Encryption Completed! Encrypted file saved as " + outputFileName);

} catch (Exception ex) {

JOptionPane.showMessageDialog(null, "Error: " + ex.getMessage());

}

}

public static void performDecryption() {

try {

String inputFileName = "Cipher.txt";

String outputFileName = "Recover.txt";

String content = new String(Files.readAllBytes(Paths.get(inputFileName)));

int key = Integer.parseInt(JOptionPane.showInputDialog("Enter the Key for Decryption (1-25):"));

String decrypted = decryptCeaserCipher(content, key);

Files.write(Paths.get(outputFileName), decrypted.getBytes());

JOptionPane.showMessageDialog(null, "Decryption Completed! Decrypted file saved as " + outputFileName);

} catch (Exception ex) {

JOptionPane.showMessageDialog(null, "Error: " + ex.getMessage());

}

}

public static void performBruteForceAttack() {

try {

String inputFileName = "Cipher.txt";

String content = new String(Files.readAllBytes(Paths.get(inputFileName)));

StringBuilder results = new StringBuilder();

int probableKey = -1;

int maxReadableScore = Integer.MIN\_VALUE;

for (int key = 1; key < 26; key++) {

String decrypted = decryptCeaserCipher(content, key);

int readableScore = calculateReadableScore(decrypted);

results.append("Key ").append(key).append(": ").append(decrypted).append("\n");

if (readableScore > maxReadableScore) {

maxReadableScore = readableScore;

probableKey = key;

}

}

results.append("\nProbable Key: ").append(probableKey).append("\n");

JTextArea textArea = new JTextArea(results.toString());

JScrollPane scrollPane = new JScrollPane(textArea);

textArea.setLineWrap(true);

textArea.setWrapStyleWord(true);

scrollPane.setPreferredSize(new java.awt.Dimension(350, 200));

JOptionPane.showMessageDialog(null, scrollPane, "Brute Force Attack Results", JOptionPane.INFORMATION\_MESSAGE);

} catch (Exception ex) {

JOptionPane.showMessageDialog(null, "Error: " + ex.getMessage());

}

}

public static int calculateReadableScore(String text) {

String[] commonWords = {"the", "be", "to", "of", "and", "a", "in", "that", "have", "I"};

int score = 0;

for (String word : commonWords) {

if (text.contains(word)) {

score++;

}

}

return score;

}

// Fact 1: 'e' is the most frequent letter in the English language

public static void performFrequencyAnalysis() {

try {

String inputFileName = "Cipher.txt";

String content = new String(Files.readAllBytes(Paths.get(inputFileName)));

Map<Character, Integer> freqMap = new HashMap<>();

for (char c : content.toCharArray()) {

if (Character.isLetter(c)) {

c = Character.toLowerCase(c);

freqMap.put(c, freqMap.getOrDefault(c, 0) + 1);

}

}

List<Map.Entry<Character, Integer>> list = new ArrayList<>(freqMap.entrySet());

list.sort(Map.Entry.comparingByValue(Comparator.reverseOrder()));

char mostFrequentChar = list.get(0).getKey();

char assumedE = 'e';

int probableKey = mostFrequentChar - assumedE;

if (probableKey < 0) {

probableKey += 26;

}

StringBuilder results = new StringBuilder();

results.append("Most Frequent Character: ").append(mostFrequentChar).append("\n");

results.append("Probable Key (Based on Frequency Analysis): ").append(probableKey).append("\n");

JTextArea textArea = new JTextArea(results.toString());

JScrollPane scrollPane = new JScrollPane(textArea);

textArea.setLineWrap(true);

textArea.setWrapStyleWord(true);

scrollPane.setPreferredSize(new java.awt.Dimension(350, 200));

JOptionPane.showMessageDialog(null, scrollPane, "Frequency Analysis Results", JOptionPane.INFORMATION\_MESSAGE);

} catch (Exception ex) {

JOptionPane.showMessageDialog(null, "Error: " + ex.getMessage());

}

}

public static String encryptCeaserCipher(String text, int key) {

StringBuilder result = new StringBuilder();

for (int i = 0; i < text.length(); i++) {

char ch = text.charAt(i);

if (Character.isLetter(ch)) {

char base = Character.isLowerCase(ch) ? 'a' : 'A';

result.append((char) ((ch - base + key) % 26 + base));

} else {

result.append(ch);

}

}

return result.toString();

}

public static String decryptCeaserCipher(String text, int key) {

return encryptCeaserCipher(text, 26 - key);

}

}

## Output:

**Plaintext.txt**

In the vast and ancient world, where time flows like a river and the stars paint the sky, there existed a land of great beauty and mystery. This land, rich with rolling hills, deep forests, and winding rivers, was home to a people whose lives were intertwined with the natural rhythms of the earth. They cultivated the land, worshipped the sun and the moon, and passed down stories from one generation to the next. These stories spoke of heroes and gods, of love and betrayal, of battles won and lost. But among all these tales, one legend stood out. It was the tale of a hidden treasure, buried deep within the heart of the land. A treasure so vast and powerful that it could change the course of history. Many had searched for it, but none had found it. Some said it was guarded by a fierce dragon, others believed it was cursed. But despite the dangers, the lure of the treasure was too great. And so, the search continued, driven by hope, greed, and the desire for glory. As the years passed, the legend grew, taking on a life of its own. But in the end, it was not the treasure that mattered, but the journey itself.

**Cipher.txt**

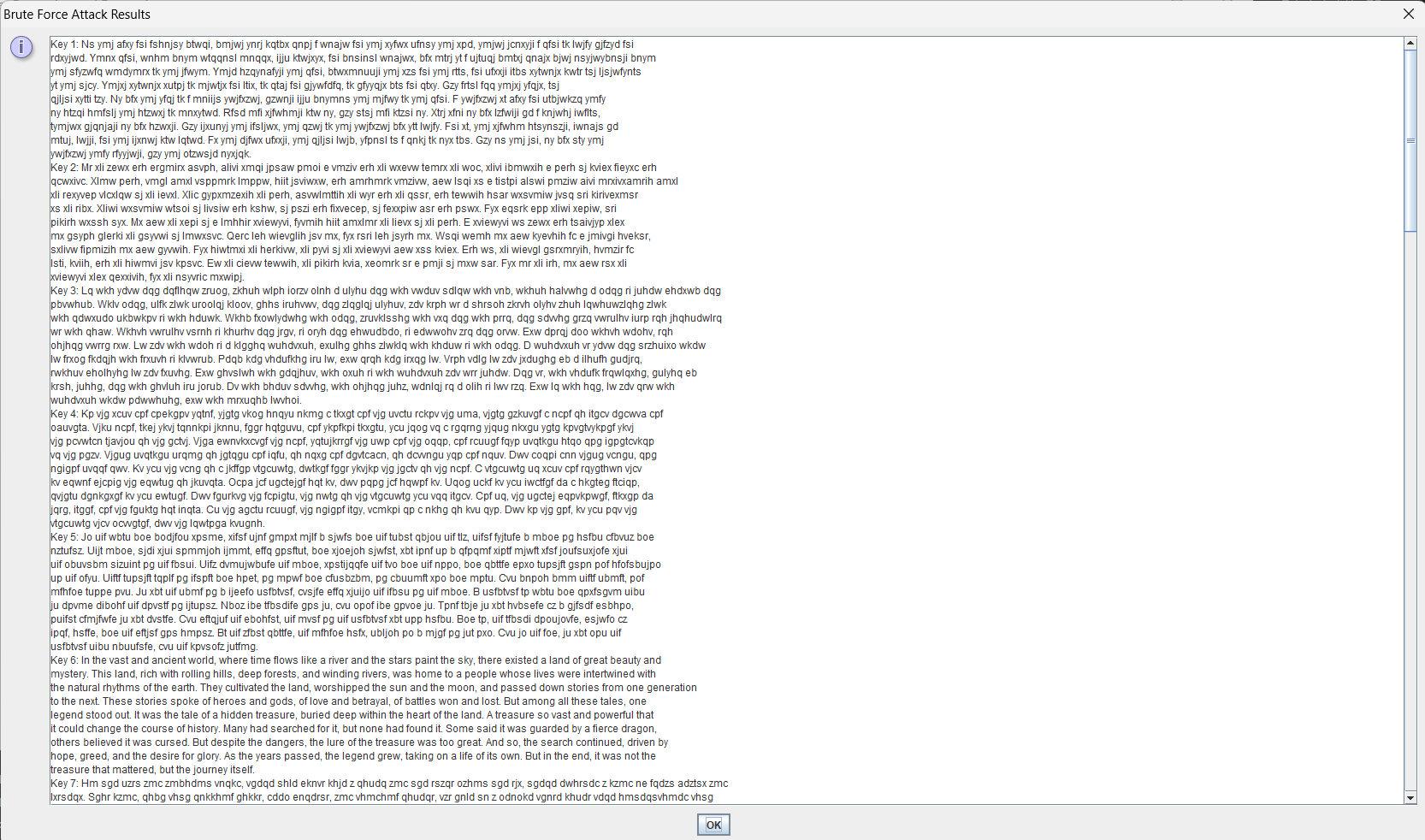
**Ot znk bgyz gtj gtioktz cuxrj, cnkxk zosk lrucy roqk g xobkx gtj znk yzgxy vgotz znk yqe, znkxk kdoyzkj g rgtj ul mxkgz hkgaze gtj seyzkxe. Znoy rgtj, xoin cozn xurrotm norry, jkkv luxkyzy, gtj cotjotm xobkxy, cgy nusk zu g vkuvrk cnuyk robky ckxk otzkxzcotkj cozn znk tgzaxgr xneznsy ul znk kgxzn. Znke iarzobgzkj znk rgtj, cuxynovvkj znk yat gtj znk suut, gtj vgyykj juct yzuxoky lxus utk mktkxgzout**

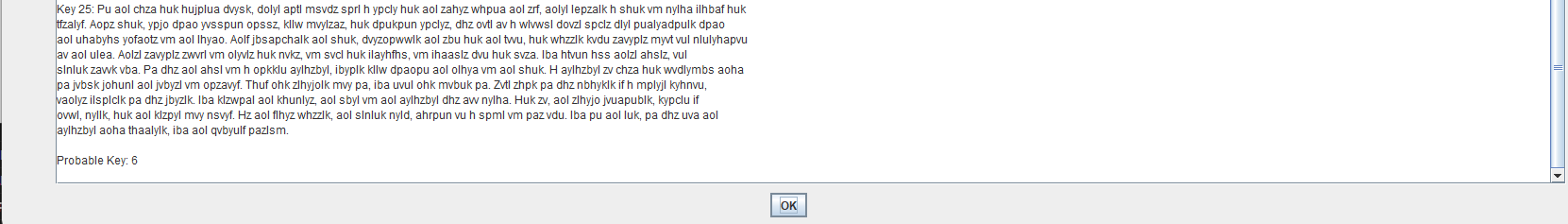
**zu znk tkdz. Znkyk yzuxoky yvuqk ul nkxuky gtj mujy, ul rubk gtj hkzxgegr, ul hgzzrky cut gtj ruyz. Haz gsutm grr znkyk zgrky, utk rkmktj yzuuj uaz. Oz cgy znk zgrk ul g nojjkt zxkgyaxk, haxokj jkkv coznot znk nkgxz ul znk rgtj. G zxkgyaxk yu bgyz gtj vuckxlar zngz oz iuarj ingtmk znk iuaxyk ul noyzuxe. Sgte ngj ykgxinkj lux oz, haz tutk ngj luatj oz. Yusk ygoj oz cgy magxjkj he g lokxik jxgmut, uznkxy hkrokbkj oz cgy iaxykj. Haz jkyvozk znk jgtmkxy, znk raxk ul znk zxkgyaxk cgy zuu mxkgz. Gtj yu, znk ykgxin iutzotakj, jxobkt he nuvk, mxkkj, gtj znk jkyoxk lux mruxe. Gy znk ekgxy vgyykj, znk rkmktj mxkc, zgqotm ut g rolk ul ozy uct. Haz ot znk ktj, oz cgy tuz znk zxkgyaxk zngz sgzzkxkj, haz znk puaxtke ozykrl.**

**Recover.txt**

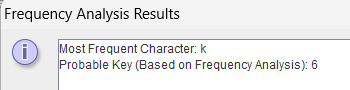
In the vast and ancient world, where time flows like a river and the stars paint the sky, there existed a land of great beauty and mystery. This land, rich with rolling hills, deep forests, and winding rivers, was home to a people whose lives were intertwined with the natural rhythms of the earth. They cultivated the land, worshipped the sun and the moon, and passed down stories from one generation to the next. These stories spoke of heroes and gods, of love and betrayal, of battles won and lost. But among all these tales, one legend stood out. It was the tale of a hidden treasure, buried deep within the heart of the land. A treasure so vast and powerful that it could change the course of history. Many had searched for it, but none had found it. Some said it was guarded by a fierce dragon, others believed it was cursed. But despite the dangers, the lure of the treasure was too great. And so, the search continued, driven by hope, greed, and the desire for glory. As the years passed, the legend grew, taking on a life of its own. But in the end, it was not the treasure that mattered, but the journey itself.

**Brute Force Attack**

****

****

**Frequency Analysis**

****

## Conclusion:

* Implementing Caesar Cipher encryption and decryption demonstrates fundamental cryptographic techniques and their applications.
* Brute force attack effectively identifies the correct key by testing all possible options.
* Frequency analysis helps determine the most probable key by analyzing character frequencies.