Quiz 3
(Deadline April 11, 2025)

Problem 1

Description (2.5 pts)

In traditional cryptography, SHA functions (like SHA-256 or SHA-3) are primarily used for hashing and digital signatures. However, SHA-3 (based on the Keccak sponge construction) is more flexible than traditional hash functions due to its sponge-based design. This unique structure allows SHA-3 to be adapted to other cryptographic purposes beyond hashing.

Requirements

1. **Comparison of SHA-256 and SHA-512/256:**
   - Compare the structure and security properties of SHA-256 and SHA-512-truncated-to-256-bits.
   - Do they provide the same level security?
   - Which one is better, and why? Provide a detailed explanation.
2. **Exploring SHA as an Encryption Function:**
   - Can SHA functions (especially SHA-3/SHAKE) be adapted for symmetric encryption?
   - What makes the sponge construction in SHA-3 suitable (or unsuitable) for encryption?
3. **(Extra) Implementation:**
   - Implement a simple encryption/decryption system using SHAKE128 or SHAKE256 in Python or Go:
     - Derive a key from your password or input.
     - Your implementation should generate a keystream for encryption purposes.
     - Develop your own method to transform plaintext into ciphertext using this stream.

Evaluation

- Comparison of SHA-256 and SHA-512/256: **1.5 points**
- Exploring SHA as an Encryption Function: **1 point**
- (Extra) Implementation: **0.5 point**

Problem 2

Description (3 pts)

The Index of Coincidence (IC) is a fundamental tool in cryptanalysis. It helps analysts determine whether a text is in a natural language, encrypted with a simple substitution cipher, or highly randomized.

The IC is computed using the following formula:

$$IC = \frac{\sum f_i(f_i - 1)}{N(N-1)}$$

Requirements

1. **Understanding IC:**
   - Provide a brief explanation of f_i and N.
2. **Program Implementation:**
   - Write a program in Python or Go to compute the Index of Coincidence for a given ciphertext.
   - Analyze the IC value and determine the probable language of the text.

   Ciphertext:

   ```
   WKHUH DUHWZ RZDBV RIFRQ VWUXF WLQJD VRIWZ DUHGH VLJQR QHZDB
   LVWRP DNHLW VRVLP SOHWK DWWKH UHDUH REYLR XVOBQ RGHIL FLHQF
   LHVDQ GWKHR WKHUZ DBLVW RPDNH LWVRF RPSOL FDWHG WKDWW KHUHD
   UHQRR EYLRX VGHIL FLHQF LHVWK HILUV WPHWK RGLVI DUPRU HGLII LFXOW
   ```

   (hint: ignore the spaces)
3. **IC Value Comparison:**
   - Compare the IC values of English, Chinese, and randomly generated text.
   - Explain the differences and their significance in cryptanalysis.
4. **(Extra) Decryption Challenge:**
   - Attempt to decrypt the ciphertext.
   - Describe your decryption process and methodology.

Evaluation

- Understanding IC: **0.5 point**
- Program implementation: **1.5 point**
  - Please include your code in the submitted ZIP file.
- IC value comparison: **1 point**
- (Extra) Decryption challenge: **0.5 points**
  - Please provide the whole plaintext; otherwise, you will not receive credit for this part.

Problem 3

Description (4.5 pts)

You are given a ciphertext (download from e3) encrypted using the **Vigenère Cipher** with an unknown key. Your task is to develop a program in **Python** or **Go** to break the encryption automatically.

A frequency table is provided. You can get freqMap in the "Reference" page (Last page).

Requirements

1. **Key Length Estimation:**
   - Use Index of Coincidence (IC) analysis to determine the most probable key length.

   (hint 1: The IC value of English is approximately 0.068.)
   (hint 2: The key length is between 1 and 8, i.e., $1 <$ key length < 8.)

2. **Key Recovery:**
   - Apply the Chi-Square test to identify the key using English letter frequency analysis.
   - Use the following Chi-Square formula:

$$X^2 = \sum \frac{(O - E)^2}{E}$$

   (hint: Try all possible key candidates and select the one with the lowest Chi-Square value.)

3. **Decrypt the ciphertext:** Use the recovered key to decrypt the text and specify which article the plaintext is.
   (You don't need to submit or take the whole screenshot of the plaintext, just give the title of the target article.)

Evaluation

- Key length estimation: **1.5 points**
- Key recovery: **1.5 points**
- Decrypt the ciphertext: **1.5 points**
  - All you need to submit is ONE program, though it may consist of multiple files depending on your code structure. You do not need to separate each part into multiple programs. However, you must clearly describe the implementation for each stage according to the question numbers.

Submission Guidelines

1. Upload a **single ZIP file** named `<student_id>.zip` to the new e3 platform, with the following structure:

   - `<student_id>.zip`
     - `<student_id>.pdf`
     - `problem1` (optional, for extra credit implementation)
       - `main.go` or `main.py`
     - `problem2`
       - `main.go` or `main.py`
       - (additional codes...)
     - `problem3`
       - `main.go` or `main.py`
       - (additional codes...)

   (You may include any additional code files, as long as they follow this file structure.)

   (**Incorrect file structure or incorrect file names will result in a 2% deduction.**)

2. Present your solutions in **numerical order**, clearly labeling each problem.

Grading Policy

1. This quiz comprises three problems:

   - Problem 1: 2.5 points
   - Problem 2: 3 points
   - Problem 3: 4.5 points

2. **Late Submission Penalty:** A penalty of **0.5 points per day** will be applied for late submissions, up to a maximum of 20 days. Beyond 20 days, late submissions will be assigned zero.

3. All quizzes are **mandatory**. Failure to submit any quiz will result in an automatic failing grade for the course.

Reference

Frequency Map

Go:

```go
var freqMap = map[rune]float64{
	'A': 0.082, 'B': 0.015, 'C': 0.028, 'D': 0.043, 'E': 0.13,
	'F': 0.022, 'G': 0.02, 'H': 0.061, 'I': 0.07, 'J': 0.0015,
	'K': 0.0077, 'L': 0.04, 'M': 0.024, 'N': 0.067, 'O': 0.075,
	'P': 0.019, 'Q': 0.00095, 'R': 0.06, 'S': 0.063, 'T': 0.091,
	'U': 0.028, 'V': 0.0098, 'W': 0.024, 'X': 0.0015, 'Y': 0.02,
	'Z': 0.00074,
}
```

Python:

```python
freqMap = {
    'A': 0.082, 'B': 0.015, 'C': 0.028, 'D': 0.043, 'E': 0.13,
    'F': 0.022, 'G': 0.02, 'H': 0.061, 'I': 0.07, 'J': 0.0015,
    'K': 0.0077, 'L': 0.04, 'M': 0.024, 'N': 0.067, 'O': 0.075,
    'P': 0.019, 'Q': 0.00095, 'R': 0.06, 'S': 0.063, 'T': 0.091,
    'U': 0.028, 'V': 0.0098, 'W': 0.024, 'X': 0.0015, 'Y': 0.02,
    'Z': 0.00074
}
```