EECS 388 Discussion 3

Homework 1 & 2

Homework 1: Rational Paranoia

- Apply the security mindset to four different scenarios
  - Identify critical assets
  - Think like an attacker
  - Think like a defender

Homework 1: Rational Paranoia

- You are developing and deploying a self-checkout system for Kroger
  - What assets are important for you to protect?
  - What security threats will you choose to defend against?
  - What countermeasures can you justify, in terms of costs and benefits?

Homework 1: Rational Paranoia

- You are grading homework submissions for a class of 100+ students
  - What assets are important for you to protect?
  - What security threats will you choose to defend against?
  - What countermeasures can you justify, in terms of costs and benefits?

Homework 2: Cryptanalysis

- Part 1: Crack a message encrypted with the Vigenère cipher

- Part 2: Perform a statistical analysis of letter frequencies in plaintexts and ciphertexts

Caesar Cipher

- Replace each letter in the plaintext with a letter a fixed number of places down the alphabet
- For an encryption key *k*:
  - Encryption: $\boldsymbol{c_i} := (\boldsymbol{p_i} + \boldsymbol{k}) \bmod 26$
  - Decryption: $\boldsymbol{p_i} := (\boldsymbol{c_i} - \boldsymbol{k}) \bmod 26$

Caesar Cipher Example

- Encrypt a message using the key ***k*** = 5
  - Plaintext: `attackatdawn`
  - Shift: `+55555555555`
  - Ciphertext: `FYYFHPFYIFBS`
- Decrypt the message
  - Plaintext: `FYYFHPFYIFBS`
  - Shift: `-55555555555`
  - Ciphertext: `attackatdawn`

Breaking the Caesar Cipher

- Brute force
  - How many possible keys are there?
- Frequency analysis
  - English letters aren’t used with the same frequency
  - Compare letter frequency distribution of ciphertext with that of the language of the plaintext

Caesar Cipher in Python

- Write a Python script that takes a message and a key as input and outputs the message encrypted with the Caesar cipher for that key
- Useful functions
  - ord(c) - Returns the ASCII value of that character
    - ord(‘A’) returns 65
  - chr(n) - Returns the character for that ASCII value
    - chr(65) returns ‘A’

Vigenère Cipher

- Called “le chiffre indéchiffrabe”
  - The indecipherable cipher
- Use a sequence of Caesar ciphers determined by the letters of a key
- For a key ***k*** of length ***n***:
  - Encryption: $\boldsymbol{c_i} := (\boldsymbol{p_i} + \boldsymbol{k_{i \bmod n}}) \bmod 26$
  - Decryption: $\boldsymbol{p_i} := (\boldsymbol{c_i} - \boldsymbol{k_{i \bmod n}}) \bmod 26$

Vigenère Cipher Example

- Encryption using the key ***k*** = abcde = 01234
  - Plaintext: `attackatdawn`
  - Shift: `+012340123401`
  - Ciphertext: `AUVDGKBVGEWO`
- Decryption
  - Plaintext: `AUVDGKBVGEWO`
  - Shift: `-012340123401`
  - Ciphertext: `attackatdawn`

Breaking the Vigenère Cipher

- Kasiski Method
  - Look for repeated strings in the ciphertext
  - Distance between occurrences is likely a multiple of the key length
  - Determine the distance between multiple repeated strings to narrow down results
  - Find common factors in distances between strings

Breaking the Vigenère Cipher

- Example:
  - Plaintext: `decipherthein`**`decipher`**`able`
  - Key: `cryptanalysiscryptanalysi`
  - Ciphertext: **`FVAXIHRR`**`EFWQF`**`FVAXIHRR`**`LZDM`
  - Distance of 13 between substrings
    - Key length is likely 1 or 13
- After determining the key length, treat ciphertext as a series of Caesar ciphers

Frequency Analysis

- The population variance of a finite population ***X*** of size ***N*** and mean $\boldsymbol{\mu}$ is given by

$$\mathrm{Var}(X) = \frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2$$

Frequency Analysis in Python

- Write a Python script that takes a sequence of numbers as input and outputs the population variance

$$\mathrm{Var}(X) = \frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2$$