MonieCrypt Cipher Report

Sara Isaid , Saja Matar

Original Mono Weaknesses

The problem with the original algorithm is it's easily breakable. Brute forcing the key takes a factorial complexity of 26! , while using frequency analysis attack would take only linear time complexity to break the cipher.

The New Algorithm

The new algorithm solved the previous problem by using public key encryption methods on top of the original substitution in monoaphabatic.

Logic:

- Key Generation
  Two keys are used in the process, one as input to the original mono cipher, and another used for public encryption.

  - First Key : generated using the original cipher letter approach and is considered a symmetric key used for both encryption and decryption.
  - Second Key : generated using two large prime numbers (p,q) as following
    * n = p.q
    * totient = (p-1).(q-1)
    * choose an e, where gcd(e, totient) =1 and (2< e <totient)

    then use the **e,n** pair as the public key for encryption.

    * for the private key (d), use modular multiplicative inverse of e mod totient

    then use the **d,n** pair as the private key for decryption.

- Encryption Process
  There are two rounds of encryption, the First uses the original algorithm with the first generated key. Second uses e and n with the following equation

$$Enc(m) = m^e \ mod \ n$$

  where m is the plaintext.

- Decryption Process
  There are two rounds of decryption. The first uses d and n with the following equation, where c is the ciphered text

$$Dec(c) = c^d \ mod \ n$$

  then decrypt the output again using the first generated key with the original mono algorithm.

Time Efficiency and Possible Attacks

Original Algorithm

The time taken to encrypt or decrypt a message depends on the length of the plain text, so the complexity is O(I), where I is the length of the message. And the time taken to crack it using brute force is 26!, which is the time taken to try all possible keys. However, to crack it using frequency analysis, it would take O(n) time.

- Encryption / Decryption = O(I) ,where I = Plain text length
- Brute Force= 26!
- Frequency analysis attack = Linear time depending on the plain text length

New Algorithm

The time here is divided on two rounds, the first is similar to the original one which is O(I), where I is the plain text length. The second part is O(log n), which is the time taken for modular exponentiation.

- Encryption/Decryption = O(I)+ O(log n)
- Brute Force attack + n Factorization = 26! + $O(e\sqrt{ln(n).ln(ln(n))}$
- Frequency Analysis + n Factorization = O(I) + $O(e\sqrt{ln(n).ln(ln(n))}$

Services

For the public key, we recommend a size of 2048 to make the factorization attack harder since a 1024 key can be factorized in a matter of weeks. and when using such scheme (that is the public key), we will achieve

- confidentiality : comes from the encryption process.
- non-repudiation : comes from the digital signature (encrypting with the private key)