Segurança Computacional Implementação da Cifra de *Vigenère*

Oseias Romeiro Magalhães / 211036123 Prof. João Gondim

11 de dezembro de 2022

Resumo

Neste trabalho de Segurança Computacional, foi explorado a cifra de *Vigenère* e o desenvolvimento de um programa composto por duas partes: o cifrado/decifrador e o ataque de recuperação de senha por análise de frequência.

1 Introdução

A cifra de *Vigenère* é um método de criptografia que usa uma série de diferentes cifras de César baseadas em letras de uma senha.

- Cifra de César: Consiste em incrementar ou decrementar cada letra por um número fixo, movimentando em N (chave) posições.[1]
- Cifra de Vigenère: Já na Cifra de Vigenère, a cada posição é atribuído um incremento diferente, dependendo da posição do keystream, sendo o keystream a repetição das letras de uma chave até alcançar o tamanho da mensagem a ser cifrada.[2]

2 Ambiente

Para o desenvolvimento desse projeto, foi utilizado a linguagem *Python* na versão 3.10.6, utilizando apenas os módulos padrão da linguagem.

3 Descrição

3.1 Cifrador/Decifrador

O cifrador recebe uma senha e uma mensagem que é cifrada segundo a cifra de *Vigenère*, gerando um criptograma, enquanto o decifrador recebe uma senha e um criptograma que é decifrado segundo a cifra de *Vigenère*, recuperando uma mensagem.[2]

```
class Cipher(Help):
    SYMBOLS: str
    SYMBOLS_SIZE: int

def __init__(self, symbols) -> None:
    self.SYMBOLS = symbols
    self.SYMBOLS_SIZE = len(symbols)

def encode(self, letter:str, key_index:int) -> str:
    return self.SYMBOLS[((self.SYMBOLS.find(letter) + key_index) % self.SYMBOLS_SIZE)]

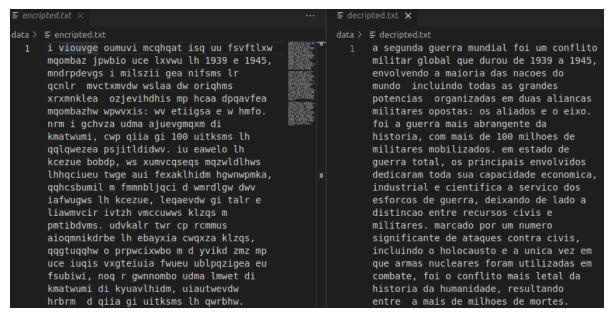
def decode(self, letter:str, key_index:int) -> str:
    return self.SYMBOLS[((self.SYMBOLS.find(letter) - key_index) % self.SYMBOLS_SIZE)]
```

```
class Viginere(Cipher):
   def init (self, symbols:str=string.ascii lowercase) -> None:
        super().__init__(symbols)
   def gen(self, key:str, message:str) -> str:
       key size = len(key)
        return "".join(
           key[i % key size]
            for i in range(0, len(message))
   def enc(self, keystream:str, message:str) -> str:
        self.vrfy key(message, keystream)
        return "".join(
           self.encode(l, self.SYMBOLS.find(keystream[i]))
           if (l in self.SYMBOLS) else l
            for i,l in self.enumerate(message, self.SYMBOLS)
   def dec(self, keystream:str, ciphertext:str) -> str:
        self.vrfy key(ciphertext, keystream)
        return "".join(
           self.decode(l, self.SYMBOLS.find(keystream[i]))
            if (l in self.SYMBOLS) else l
            for i,l in self.enumerate(ciphertext, self.SYMBOLS)
```

3.2 Recuperação de senha

O ataque de recuperação de senha por análise de frequência: recebe uma mensagens cifrada que é utilizada para recuperar a senha geradora do *keystream*, usado na cifração e então decifradas.[3]

```
unb@desktop:~/vigenere_cipher$ python3 main.py ideia -e
saved: data/encripted.txt
unb@desktop:~/vigenere_cipher$ python3 main.py ideia -d
saved: data/decripted.txt
unb@desktop:~/vigenere_cipher$ python3 main.py -b pt
possible key sizes: [2, 3, 5, 6, 4]
choose: 5
key: ideia
```



4 Conclusão

Assim, foi implementado o cifrador e decifrador utilizando a cifra de *Vigenère* e foi mostrado uma de suas vulnerabilidades ao descobrir a chave utilizada a partir de uma mensagem cifrada.

Referências

- [1] Daniel Adornes. *Quebrando a Cifra de Vigenère*. https://informatabrasileiro.blogspot.com/2013/04/quebrando-cifra-de-vigenere.html.
- [2] Udacity. Vigenere Clpher. https://youtu.be/SkJcmCaHqS0.
- [3] Wikipédia. Frequência de letras. https://pt.wikipedia.org/wiki/FrequÃłncia_de_letras.