Trabalho 1 de Segurança Computacional
Cifra de Vigenere

Matheus Guaraci Lima Bouças Alves
Dep. de Ciência da Computação
Universidade de Brasília
Brasília, Brasil
180046951@aluno.unb.br

Aline Mitiko Otsubo
Dep. de Ciência da Computação
Universidade de Brasília
Brasília, Brasil
170004601@aluno.unb.br

***Abstract*—Publicada em 1586, a cifra de Vigenère é um exemplo de cifragem polialfabética de substituição. Seu funcionamento é dado pela aplicação de diversas instâncias independentes da cifra de César (KATZ e LINDELL, 2014). Durante muitos anos, a cifra de Vigenère foi considerada inquebrável, mas o desenvolvimento das técnicas de criptoanálise provaram o contrário (apesar dela ser segura quando a chave tem o mesmo tamanho do texto a ser cifrado). Com a utilização do método de Kasiski e a análise de frequência, é possível descobrir a chave juntamente com o seu tamanho.**

I. Introdução

Alan Konheim define a criptografia como a arte e ciência de tornar a comunicação incompreensível para todos, exceto o destinatário desejado. Essa é uma definição que se aplica muito bem para os métodos de cifragem históricas iniciais, como a cifra de César e a cifra monoalfabética de substituição. A cifra de Vigenère aponta para uma expansão dessa definição, pois além de tornar incompreensível, ela tenta assegurar que a mensagem seja ilegível, tentando proteger a mensagem de ataques como a análise de frequência (que era a forma que decifravam as cifras de substituição).

Diferente da cifra de César, cujo número de chaves possíveis para cifrar um texto era 25, as cifras de Vigenère e a monoalfabética de substituição, possuem um número muito alto de possibilidades de chave. Sendo mais específico, a primeira possui 26^t possibilidades, sendo 't' o número de caracteres da chave. A segunda, possui 26! possibilidades de chave.

Com a cifra de Vigenère, mesmo ela tendo um espaço de possibilidades de chave maior que a monoalfabética de substituição (dependendo do tamanho da chave, pois o espaço da cifra de Vigenère cresce, enquanto o espaço da outra é fixo), percebemos que ela não é segura. Ou seja, por mais que seja inviável atacar por força bruta, e que seja necessário um espaço grande de chaves, ele não é suficiente para tornar o método de cifragem seguro.

II. Metodologia

A linguagem utilizada para implementar as soluções dos problemas de cifração/decifração e ataque de recuperação de senha foi o C++. Para a realização dos testes foram utilizados arquivos texto contendo mensagens cifradas em inglês e português.

III. Implementação

A. Cifrador e Decifrador

A função *Cripto*, soluciona a primeira parte do trabalho que consiste em criar um cifrador e decifrador. A partir de uma mensagem e senha passadas na entrada da implementação, a função realiza a cifração/decifração caractere por caractere baseada no conceito da cifra de Vigenere, como visto na seção I.

Caso a escolha do usuário seja cifrar a mensagem, é aplicado o seguinte método algébrico:

$$Ci = (Pi + Ki) \mod 26 \tag{1}$$

Nesta expressão, *Ci* representa o caractere cifrado, *Pi* representa o caractere da mensagem e *Ki* representa o caractere da senha. O operador módulo 26 é usado para calcular o resto da divisão de (Pi + Ki) por 26, garantindo que o resultado esteja dentro do limite do alfabeto.

Por outro lado, caso o usuário decida decifrar, é aplicado:

$$Pi = (Ci - Ki + 26) \mod 26 \tag{2}$$

Os caracteres especiais contido na mensagem foram apenas replicados na variável de saída.

B. Ataque de Recuperação de Senha

Ao implementar a parte do ataque foi utilizado o método Kasiski para encontrar o melhor candidato ao tamanho da senha, enquanto que para a recuperação da senha foi utilizado análise de frequência.

Para implementar o método Kasiski foi utilizado três funções auxiliares e uma principal. Inicialmente, a função *PosicoesTrigramas* filtra os trigramas repetidos no texto cifrado e registra suas posições em um vetor.

Em seguida, a função *CalculaEspacos* calcula os espaços(distância) entre as posições dos trigramas repetidos.

Com base nos espaços encontrados, é utilizado *ObtemFatores* para buscar todos fatores. Esses são considerados os candidatos ao tamanho da senha.

A função *EncontraTamanhosSenha* é utilizada para realizar a chamada das funções expostas acima. Os fatores candidatos com valores entre 2 e 19 são armazenados em um vetor. Em seguida, a função conta quantas vezes cada tamanho candidato aparece e retorna um mapa que associa os tamanhos candidatos a suas ocorrências.

Após descobrirmos o tamanho da chave, separamos o texto cifrado em grupos. A quantidade de grupos é definida pelo tamanho da chave. Na imagem abaixo, podemos observar que cada caractere do texto cifrado pode ser organizado de forma a ter uma posição correspondente a um caractere da chave.

```
ciphertext: zeq cig kqy ssp zep dib oqg xly
key:        key key key key key key key key
position:   123 123 123 123 123 123 123 123
```

Fig. 1. Exemplo de uma reorganização de um texto cifrado com relação a chave.

Neste exemplo, a chave possui tamanho 3, definindo que teremos 3 grupos. O grupo 1 será formado por todos os caracteres que ficaram acima da letra 'k', ocupando a posição 1. Os outros 2 grupos são formados de maneira análoga.

Dessa forma, conseguimos fazer a análise de frequência em cima dos caracteres que pertencem a um certo grupo. Pois todos eles foram cifrados com o mesmo caractere, como é feito na cifra de César.

Cada caractere aparece uma certa quantidade de vezes no grupo, a frequência deles é definida pela quantidade de aparições dividida pelo tamanho do grupo. Nosso código gera uma tabela associando cada caractere com a sua frequência. Cada grupo possui uma tabela. Essas tabelas são listadas pelo método 'ListaDeTabelas'.

Com as tabelas, pegamos cada uma, multiplicamos as frequências das letras listadas na tabela com as frequências de cada letra de um idioma (no nosso caso, inglês ou português) e fazemos um somatório. Nós também rotacionamos a tabela do grupo para gerar 26 somatórios. Cada somatório é colocado em uma lista e ao final selecionamos o maior deles. O índice do somatório selecionado é o índice de um caractere da chave no alfabeto. No método 'EncontraSenha', fazemos este processo para todas as tabelas listadas anteriormente, obtendo então todos os caracteres da chave.

IV. Conclusão

Este trabalho abordou a cifra de Vigenere, uma cifra polialfabética de substituição que expandiu a definição de criptografia ao tentar tornar as mensagens não apenas incompreensíveis, mas também ilegíveis. A cifra de Vigenere oferece um espaço de chaves muito maior em comparação com a cifra de César, tornando-a aparentemente mais segura.

No entanto, o trabalho demonstrou que a cifra de Vigenere não é totalmente segura. Por meio do método Kasiski e da análise de frequência, é possível descobrir o tamanho da chave e, subsequentemente, a própria chave.

A maior dificuldade dos integrantes foi entender o funcionamento dos métodos utilizados no ataque. E a linguagem escolhida para a realização do trabalho também tornou a implementação mais complexa.

References

[1] SCHNEIER, Bruce. Applied Cryptography, Second Edition: Protocols, Algorthms, and Source Code in C. 2.ed. New york: John Wiley & Sons, 1996.

[2] KATZ, Jonathan; LINDELL, Yehuda. Introduction to modern cryptography. 2.ed. Boca Raton: Chapman & Hall/CRC, 2014.

[3] KONHEIM, Alan. Cryptography:A Primer. New York: John Wiley & Sons, Inc, 1981.