Universidade de Brasília

Departamento de Ciência da Computação

Projeto 1 - Segurança Computacional

Autores:

João Gabriel Ferreira Saraiva 18/0103016

João Francisco Gomes Targino 18/0102991

Brasília

11 de agosto de 2021

1 Introdução

Este primeiro projeto de Segurança Computacional consiste em duas partes, a primeira na realização de um cifrador/decifrador explorando a cifra de Vigenère, e a segunda sendo um ataque de recuperação de senha, tanto em português como também em inglês.

Na primeira parte apenas foi necessário um conhecimento de Cifra de César. Já na segunda foi preciso pesquisar um pouco sobre o Método de Kasiski. O projeto foi feito utilizando a linguagem C++ e estava rodando no Sistema Operacional Ubuntu.

2 Primeiros Passos

Primeiramente deve-se ressaltar que para todas as 4 funcionalidades do programa foi feita a função **tratartex**, que consiste em um método que pega a mensagem que será mandada pelo usuário e vai fazer a filtragem dessa mensagem, retirando caracteres especiais e espaços guardando-os em um vetor com as posições que eles deverão ser mostrados ao final da execução, e além disso a função também é responsável por transformar todos os caracteres da string em caracteres minúsculos para que ao decorrer do programa seja mais fácil transformar nossos caracteres.

3 Primeira Parte

3.1 Cifrador

Para o cifrador foi preciso conhecimento sobre Cifra de César, pois a cifra de Vigenère é feita usando uma série de cifras de César, então vendo que o cálculo para codificar uma cifra de césar era:

$$C = (x + n) mod 26$$

sendo **C** a letra resultado da cifra, **x** a letra do texto original e **n** a letra da chave. E além desse cálculo quando necessário caso a soma das duas letras seja maior do que o número de letras do alfabeto, deve-se diminuir 26 do resultado total. Então apenas foi feito o cálculo anterior para todas as letras da mensagem até que ao terminar a mensagem criptografada é mostrada na tela.

3.2 Decifrador

Para o decifrador foi usada a mesam lógica anterior, com pequenas diferenças de cálculo, decodifica-se uma cifra de César da seguinte maneira:

$$C = (x - n) mod 26$$

sendo **C** a letra resultado da cifra, **x** a letra do texto original e **n** a letra da chave. E ao invés de subtrairmos 26 do resultado total caso o mesmo passe dessa vez devemos ver se o resultado final é um número negativo e então somarmos 26 se necessário. Então apenas foi feito o cálculo anterior para todas as letras da mensagem até que ao terminar a mensagem original é mostrada na tela.

4 Parte 2 - Ataques

Para os ataques tanto em português como em inglês foram usadas as mesmas funções, sendo a única diferença o uso de vetores diferentes para testarmos a ocorrência das letras ao comparar com o texto que teremos para descobrir a chave.

Para essa parte foi estudado sobre o Método de Kasiski que foi usado da seguinte forma:

4.1 Uso do Método de Kasiski

Depois que o usuário digita a frase para o ataque, esta frase é tratada e logo em seguida é executada a primeira parte do Método, são separados blocos de 2 a 5 letras e contadas a repetição de cada um desses blocos, e após pegar todos os blocos é usado somente aquele que se repete mais dentro do texto.

Com o bloco de maior recorrência no texto agora deve-se fazer o passo 2, que é contar a distância entre cada uma dessas repetições daquele bloco em específico, para então fazermos o passo 3 que com esses números das distância em mãos foi preciso ver qual o divisor de 1 até 25 era mais comum entre eles, e essa resposta é o tamanho da chave(passo 4) para decifrarmos esse texto.

Para o passo 5 utilizando o valor achado no passo anterior, que será chamado de **K**, devemos fazer **K** segmentos de caracteres, sendo que cada segmento conterá partes do texto de distância também igual a **K**. Por exemplo, se achamos que nossa chave tem tamanho 3 e o texto tamanho 9, então

deveremos fazer 3 sequencias de caracteres, a sequência 1 com os caracteres=1,4,7, a sequência 2 com os caracteres=2,5,8 e a sequência 3 com os caracteres=3,6,9.

Para o passo 6 foi pega cada uma dessas sequências individualmente, pois, cada uma delas será decifrada por um caractere para então formarmos a chave, e com cada uma das sequências é realizada a análise de frequência de caracteres, e então essa frequência dependendo da opção escolhida no menu interativo será comparada com a frequência das letras da língua que estaremos tentando quebrar a cifra.

Cada letra da chave é procurada separadamente, o texto inicial é separado por n textos, onde n é o tamanho da chave que foi achado anteriormente. Após o texto ser separado, é medido a frequência de cada letra do alfabeto com a frequência da língua escolhida, onde calculamos usando:

$$VarDif = VarDif + ((Freqletra(x)) - (FreqSupostaletra(x)))$$

Onde, quanto mais próximo *VarDif* for de 0, mais maior a chance desse ser o alfabeto correto, lembrando que isso é testado para (x+1, x+2, ... x+n).

Após isso temos o numero de vezes que temos que "arrastas"o alfabeto para o lado.

E então depois de achada a chave, a mesma é mandada para a função de descriptografar juntamente com o texto recebido e é mostrado na tela o provável texto original descriptografado.

E para o passo 7 não foi feita implementação, esse passo consiste em verificar o texto e então caso o resultado não seja algo legível, deve-se tentar voltar ao terceiro passo descartando o valor achado e usando o segundo que mais divide todas as distâncias. Mas essa parte poderia ter sido facilmente implementada apenas colocando uma pergunta ao final do ataque perguntando ao usuário se a mensagem parece legível para a linguagem que o mesmo escolheu, se não apenas seria descartado o valor usado e testaríamos o próximo.

5 Conclusão

O trabalho foi bem complicado pela questão do tempo, foi um tempo muito apertado para a entrega devido a dificuldade do trabalho e o momento em que estamos vivendo. Tirando as dificuldades ligadas ao tempo, o trabalho foi muito bom para o aprendizado, afinal colocamos em pratica algumas coisas vistas em aula.

Tiveram alguns erros no decorrer do trabalho, grande parte deles foi arrumado, mas ficou um que não conseguimos arrumar, vimos apenas depois dele pronto, quando testamos chaves com 2 caracteres e texto pequeno, ele tem um erro na parte de ataque, onde o numero da chave é trocado por um numero maior e assim não conseguimos recuperar o texto original, ás vezes também tem a chance da chave sair duplicada, mas a decifração sai correta.