Universidade de Brasília - UnB
Departamento de Ciência da Computação
Disciplina: Segurança Computacional - CIC0201
Professor: Dr. João José Costa Gondim
Período: 2023/2 - Turma: 2

Alunos: Eder de Amaral Amorim - Matrícula: 170140636
Tais Alves Oliveira - Matrícula: 190117176

Trabalho de Implementação 1 - Cifra de Vigenère

Descritivo

O presente trabalho abordou o desenvolvimento de um algoritmo que explora a Cifra de Vingenère, focando em duas partes principais, sendo:

- **Parte I - Cifrador/decifrador**
    - **Cifrador**

Permite ao usuário a cifragem de textos, em português ou inglês, com a cifra de Vingenère. Para a cifragem, é necessário que o utilizador submeta o texto a ser cifrado e escolha uma chave.

O algoritmo inicia-se adequando o tamanho da chave que foi definida pelo usuário ao mesmo tamanho da mensagem a ser cifrada. Após isso, o cifrador faz um loop *for()* percorrendo um intervalo de *'a'* a *'z'*, verificando se existem letras maiúsculas e convertendo-as, caso haja, em minúsculas por meio da função *lower()*.

Dentro do laço *for()*, cada caractere é convertido em um valor numérico da tabela ASCII, por meio da função *ord()*. O caractere correspondente na chave também será convertido. Em seguida, os valores serão subtraídos do valor de 'a' (que na tabela ASCII corresponde ao número 97), para fazer uma espécie de mapeamento para todas as letras. Posteriormente a isso, os valores são somados e o resultado da soma é reduzido com o módulo de 26 (que é a quantidade de letras do alfabeto), para garantir que a cifra esteja dentro no intervalo de *'a'* a *'z'.*

Por fim, o valor do resultado obtido nos passos anteriores é novamente convertido, porém, dessa vez para letras, usando a função *chr()* e assim obtemos a letra cifrada. O processo é repetido até que toda a mensagem seja cifrada.

De forma geral, o processo de cifragem ocorre por meio do deslocamento das letras da mensagem original pela letra correspondente usando a chave. O cifrador retornará um criptograma definido por meio do algoritmo da cifra de Vingenère.

- **Decifrador**

Por sua vez, o decifrador é responsável por decodificar o texto que foi originalmente codificado com a Cifra de Vingenère, por meio da operação inversa. Nesse caso o usuário entrará com o criptograma anteriormente gerado e a senha que foi definida por ele. O programa fará a decifração e retornará o texto que foi originalmente submetido antes da cifragem.

Primeiramente, assim como no processo de cifragem, a chave fornecida pelo usuário é adequada ao mesmo tamanho da mensagem que será decifrada. Após isso, cada caractere da mensagem, *'text[i]',* e da chave, *chave[i-j],* serão convertidos em valores numéricos por meio da função *ord().* Um loop *for()* percorre o intervalo de *'a'* a *'z'*, verificando e convertendo eventuais letras maiúsculas em minúsculas, caso haja, por meio da função *lower()*.

Após isso é aplicada o algoritmo para decifração, que começa pela subtração do valor numérico da letra e o valor numérico correspondente na chave. Após isso, ocorre a adição de 26 ao resultado da subtração, para garantir um resultado positivo ao fazer o módulo de 26 (com o operador *%26*). Isso garante um valor válido no intervalo que corresponde à quantidade de letras do alfabeto.

Finalmente, o valor numérico que corresponde a letra 'a' é adicionado ao resultado. O valor obtido será transformado em uma letra, por meio da função *'chr()'*. O processo é repetido e os valores vão sendo transformados novamente em caracteres por meio da função *'chr()'* e concatenadas na variável *'textdecifrado'* que será retornada ao final, por meio de *'return textdecifrado'.*

- **Parte II - Ataque de recuperação de senha por análise de frequência**

O código tem como principal objetivo recuperar a senha, utilizada para cifrar uma mensagem, por meio da análise de frequência dos caracteres e, assim, decifrar as mensagens criptografadas. Inicialmente, o código limpa a mensagem a ser decifrada, excluindo caracteres que não são cifrados, tais como espaços e pontuações, e convertendo todas as letras para minúsculas. Esse processo é realizado através de expressões regulares, garantindo que a análise subsequente seja mais precisa e eficiente.

A partir do texto filtrado, o código então identifica trigramas, que são conjuntos de três caracteres contíguos, e calcula a distância entre as ocorrências repetidas destes trigramas. Esse passo é crucial, pois a distância entre os trigramas repetidos pode ser usada para estimar o tamanho da chave que foi utilizada para cifrar a mensagem original.

O código, então, analisa as frequências de tais distâncias para identificar padrões e inferir o tamanho mais provável da chave. Esta análise é feita comparando as distâncias encontradas com os múltiplos conhecidos das frequências de trigramas em textos não cifrados, facilitando a identificação do tamanho da chave.

Para isso o código organiza as possíveis chaves baseadas em suas frequências em ordem decrescente e assume inicialmente que o tamanho mais provável da chave é igual ao tamanho do primeiro (e mais frequente) conjunto de trigramas repetidos.

É uma premissa reconhecida que chaves de maior extensão frequentemente se revelam divisíveis por suas contrapartes menores. Para ilustrar, se depararmos com uma chave de tamanho 8, é provável que os tamanhos 4 e 2 se manifestem com frequências elevadas. No entanto, observa-se que quando existe correlação a diferença entre as frequências são constantes. Portanto o código analisa se o fator subsequente é divisível pelo fator em análise e se a discrepância entre as frequências é consistente, de forma a entregar a maior chave possível.

Com o tamanho da chave em mãos, o código prossegue para calcular a frequência dos caracteres em cada segmento do texto cifrado, que é obtido dividindo o texto pelo tamanho estimado da chave. Esta frequência de caracteres é

então comparada com a frequência conhecida dos caracteres em um idioma específico, seja português ou inglês, facilitando a identificação da chave real.

Finalmente, com a chave identificada, é possível decifrar a mensagem original, completando assim o processo de criptoanálise. Durante esse processo, o código também lida com erros, especialmente quando o tamanho da cifra é insuficiente para determinar o tamanho da chave, e imprime uma mensagem de erro informando o usuário sobre o problema.