Bacharelado em Ciência da Computação

CCMP3079 Segurança de Redes de Computadores

Atividade Cap. 02

Prof. Sérgio Mendonça

Nome Completo: Luan Valentino Sampaio Marques

Data: 17/10/2023

Universidade Federal do Agreste de Pernambuco
Av. Bom Pastor s/n - Boa Vista
55292-270 Garanhuns/PE
T +55 (87) 3764-5500
http://www.ufape.edu.br

Questões

Questões retiradas do livro-texto da disciplina.

1. Responda (de forma objetiva) as questões a seguir:

   (a) **Quais são os elementos essenciais de uma cifra simétrica?**
   Solução: Os elementos essenciais de uma cifra simétrica são a chave secreta compartilhada entre as partes envolvidas e o algoritmo de encriptação/decriptação.

   (b) **Quais são as duas funções básicas usadas nos algoritmos de encriptação?**
   Solução: Substituição e transposição.

   (c) **Qual é a diferença entre uma cifra de bloco e uma cifra de fluxo?**
   Solução: Uma cifra de bloco encripta dados em blocos de tamanho fixo (por exemplo, 64 ou 128 bits), enquanto uma cifra de fluxo encripta dados um bit ou byte de cada vez.

   (d) **Quais são as duas técnicas gerais para atacar uma cifra?**
   Solução: Criptoanálise e ataque de força bruta.

   (e) **Quais são os dois problemas com o one-time pad?**
   Solução: A necessidade de uma chave verdadeiramente aleatória do mesmo tamanho da mensagem e a dificuldade de distribuir e gerenciar essas chaves de forma segura.

   (f) **O que é uma cifra de transposição?**
   Solução: Uma cifra que rearranja os caracteres na mensagem original de acordo com uma regra específica, sem alterar os caracteres em si.

   (g) **O que é esteganografia?**
   Solução: A prática de ocultar a existência de uma mensagem, escondendo dados dentro de outros arquivos, como imagens, áudio ou vídeo, de forma que não seja perceptível.

2. **Generalização da Cifra de César: Cifra de César Afim**

   Uma generalização da cifra de César, conhecida como cifra de César afim, tem a seguinte forma: a cada letra de texto claro p, substitua-a pela letra de texto cifrado C:

   $$C = E([a, b], p) = (ap + b) \mod 26$$

   Um requisito básico de qualquer algoritmo de encriptação é que ele seja um para um. Ou seja, se $p \neq q$, então $E(k, p) \neq E(k, q)$. Caso contrário, a decriptação é impossível, pois mais de um caractere de texto claro é mapeado no mesmo caractere de texto cifrado. A cifra de César afim não é um-para-um para todos os valores de a. Por exemplo, para $a = 2$ e $b = 3$, então

   $$E([a, b], 0) = E([a, b], 13) = 3.$$

   (a) **Existem limitações sobre o valor de b? Explique por que sim ou por que não.**
   Solução: Não, não existem limitações sobre o valor de b. O valor de b realiza apenas um deslocamento aditivo no resultado da multiplicação ap. O valor de

b não afeta a propriedade de ser um-para-um da função de encriptação. A propriedade de ser um-para-um depende exclusivamente do valor de a. Para que a cifra de César afim seja um-para-um, é necessário que a função $ap \mod 26$ seja uma bijeção. Isso é garantido quando a é coprimo com 26, mas o valor de b pode ser qualquer número inteiro entre 0 e 25.

(b) **Determine quais valores de a não são permitidos.**
Solução: Os valores de a que não são permitidos são aqueles que não são coprimos com 26. Em outras palavras, se a e 26 compartilham algum fator comum além de 1, então a não é permitido. Para que $ap \mod 26$ seja uma função um-para-um, a deve ter um inverso multiplicativo módulo 26. Isso só acontece se a for coprimo com 26, ou seja, $\gcd(a, 26) = 1$. Portanto, os valores de a não permitidos são aqueles que compartilham os fatores primos de 26, que são 2 e 13. Assim, os valores de a não permitidos são múltiplos de 2 ou 13: 2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22 e 24.

(c) **Ofereça uma afirmação geral sobre quais valores de a são e não são permitidos. Justifique-a.**
Solução: Os valores de a que são permitidos são aqueles que são coprimos com 26, ou seja, $\gcd(a, 26) = 1$. Para que a função de encriptação $(ap + b) \mod 26$ seja um-para-um, a deve ter um inverso multiplicativo módulo 26. Isso significa que a e 26 não podem ter nenhum divisor comum além de 1. Em outras palavras, a deve ser coprimo com 26. Os fatores primos de 26 são 2 e 13, então qualquer a que não seja múltiplo de 2 ou 13 será coprimo com 26. Portanto, os valores permitidos de a são: 1, 3, 5, 7, 9, 11, 15, 17, 19, 21, 23 e 25. Esses valores garantem que a cifra de César afim será um-para-um e, portanto, permitirá uma decriptação correta.

3. (a) Encriptação da mensagem usando a cifra de Hill

1. **Preparação da mensagem**:

   - Mensagem original: "meet me at the usual place at ten rather than eight oclock"
   - Mensagem sem espaços: "meetmeattheusualplaceattenratherthaneightoclock"
   - Adiciona 'X' no final para obter um número par de letras: "meetmeattheusualplaceattenratherthaneightoclockx"

2. **Conversão para números**:

$$\begin{array}{llll} m=12, & e=4, & e=4, & t=19, \\ m=12, & e=4, & a=0, & t=19, \\ t=19, & h=7, & e=4, & u=20, \\ s=18, & u=20, & a=0, & l=11, \\ p=15, & l=11, & a=0, & c=2, \\ e=4, & a=0, & t=19, & t=19, \\ e=4, & n=13, & r=17, & a=0, \\ t=19, & h=7, & a=0, & n=13, \\ e=4, & i=8, & g=6, & h=7, \\ t=19, & o=14, & c=2, & l=11, \\ o=14, & c=2, & k=10, & x=23 \end{array}$$

3. **Agrupar em pares e multiplicar pela chave**: Para cada par $\begin{bmatrix} p1 \\ p2 \end{bmatrix}$:

$$\begin{bmatrix} C1 \\ C2 \end{bmatrix} = \begin{bmatrix} 9 & 4 \\ 5 & 7 \end{bmatrix} \begin{bmatrix} p1 \\ p2 \end{bmatrix} \mod 26$$

- "me" (12, 4):

$$\begin{bmatrix} 9 & 4 \\ 5 & 7 \end{bmatrix} \begin{bmatrix} 12 \\ 4 \end{bmatrix} = \begin{bmatrix} (9 \cdot 12 + 4 \cdot 4) \\ (5 \cdot 12 + 7 \cdot 4) \end{bmatrix} \mod 26 = \begin{bmatrix} 124 \\ 104 \end{bmatrix} \mod 26 = \begin{bmatrix} 20 \\ 0 \end{bmatrix}$$

  "20" é "U" e "0" é "A".

Repetindo o processo para todos os pares:

```
"me" -> UA
"et" -> HW
"me" -> UA
"at" -> ND
"th" -> IP
"eu" -> XL
"su" -> MH
"al" -> PH
"pl" -> QH
"ac" -> KR
"et" -> HW
"te" -> XE
"nr" -> KL
"at" -> ND
"he" -> PW
"an" -> JX
"ei" -> XC
"gh" -> PY
```

```
"to" -> YQ
"cl" -> DL
"oc" -> OC
"kx" -> UP
```

4. **Mensagem cifrada**:

```
UAHWUA NDIPXLM HPHQHK RHWXEK LNDPWJ XCXPY YQDL OCUP
```

(b) Decriptação da mensagem cifrada

Para decriptar a mensagem cifrada, precisamos da matriz inversa da chave usada:

1. **Encontrar a matriz inversa da chave**:
   - Matriz chave: $\begin{bmatrix} 9 & 4 \\ 5 & 7 \end{bmatrix}$
   - Determinante: $\det(A) = (9 \cdot 7 - 5 \cdot 4) \mod 26 = 17$
   - Inverso de 17 módulo 26 é 23.
   - Matriz adjunta: $\begin{bmatrix} 7 & -4 \\ -5 & 9 \end{bmatrix} = \begin{bmatrix} 7 & 22 \\ 21 & 9 \end{bmatrix}$
   - Inversa da matriz: $A^{-1} = 23 \begin{bmatrix} 7 & 22 \\ 21 & 9 \end{bmatrix} \mod 26 = \begin{bmatrix} 11 & 14 \\ 1 & 17 \end{bmatrix}$

2. **Decriptar cada par de letras**: Para cada par $\begin{bmatrix} C1 \\ C2 \end{bmatrix}$:

$$\begin{bmatrix} p1 \\ p2 \end{bmatrix} = \begin{bmatrix} 11 & 14 \\ 1 & 17 \end{bmatrix} \begin{bmatrix} C1 \\ C2 \end{bmatrix} \mod 26$$

   Exemplos:

   - "UA" (20, 0):

$$\begin{bmatrix} p1 \\ p2 \end{bmatrix} = \begin{bmatrix} 11 & 14 \\ 1 & 17 \end{bmatrix} \begin{bmatrix} 20 \\ 0 \end{bmatrix} = \begin{bmatrix} 220 \\ 20 \end{bmatrix} \mod 26 = \begin{bmatrix} 12 \\ 4 \end{bmatrix}$$

     "12" é "M" e "4" é "E".

   Repetindo o processo para todos os pares:

```
UA -> ME
HW -> ET
UA -> ME
ND -> AT
IP -> TH
XL -> EU
MH -> SU
PH -> AL
```

```
QH -> PL
KR -> AC
HW -> ME
XE -> AT
KL -> TE
ND -> NR
PW -> AN
JX -> TH
XC -> EI
PY -> GH
YQ -> TO
DL -> CL
OC -> OC
UP -> KX
```

3. **Mensagem decifrada**:

```
meetmeattheusualplaceattenratherthaneightoclockx
```

Removendo o 'x' adicional, a mensagem original é:

```
meet me at the usual place at ten rather than eight oclock
```

4. Encriptação e Decriptação usando a Cifra de César

```python
def cifra_cesar(texto, chave, modo='encriptar'):
    resultado = ""
    for char in texto:
        if char.isalpha():
            shift = chave if modo == 'encriptar' else -chave
            base = ord('A') if char.isupper() else ord('a')
            resultado += chr((ord(char) - base + shift) % 26 + base
                )
        else:
            resultado += char
    return resultado

# Exemplo de uso
texto_original = "meet me at the usual place at ten rather than
    eight oclock"
chave = 3
texto_encriptado = cifra_cesar(texto_original, chave, modo='
    encriptar')
print("Texto Encriptado:", texto_encriptado)

texto_decriptado = cifra_cesar(texto_encriptado, chave, modo='
    decriptar')
print("Texto Decriptado:", texto_decriptado)
```

Listing 1: Programa para Cifra de César

5. Ataque de Frequência em uma Cifra Aditiva

```
import string
from collections import Counter

def ataque_frequencia(texto_cifrado, n=10):
    frequencia_ingles = {
        'e': 12.02, 't': 9.10, 'a': 8.12, 'o': 7.68, 'i': 7.31, 'n'
            : 6.95, 's': 6.28, 'h': 6.09,
        'r': 5.99, 'd': 4.25, 'l': 4.03, 'u': 2.76, 'c': 2.23, 'm':
             1.92, 'f': 1.82, 'y': 1.60,
        'w': 1.54, 'g': 1.43, 'p': 1.29, 'b': 1.11, 'v': 1.00, 'k':
             0.77, 'x': 0.15, 'z': 0.07,
        'j': 0.15, 'q': 0.10
    }

    def calcular_score(texto):
        texto = texto.lower()
        contagem = Counter(texto)
        score = sum((contagem.get(c, 0) / len(texto)) *
            frequencia_ingles.get(c, 0) for c in frequencia_ingles)
        return score

    resultados = []
    for chave in range(26):
        texto_tentativa = cifra_cesar(texto_cifrado, chave, modo='
            decriptar')
        score = calcular_score(texto_tentativa)
        resultados.append((chave, texto_tentativa, score))

    resultados.sort(key=lambda x: x[2], reverse=True)
    return resultados[:n]

# Exemplo de uso
texto_cifrado = "phhw␣ph␣dw␣wkh␣xvxdo␣sodfh␣dw␣whq␣uhdwk␣wkdq␣hljkw
    ␣rorfn"
resultados = ataque_frequencia(texto_cifrado, n=10)
for i, (chave, texto, score) in enumerate(resultados, 1):
    print(f"{i}.␣Chave:␣{chave},␣Texto:␣{texto},␣Score:␣{score}")
```

Listing 2: Programa para Ataque de Frequência

6. Encriptação e Decriptação usando a Cifra de Hill 2×2

```
import numpy as np

def cifra_hill(texto, chave, modo='encriptar'):
    def texto_para_numeros(texto):
        return [ord(char) - ord('A') for char in texto.upper() if
            char.isalpha()]

    def numeros_para_texto(numeros):
        return ''.join(chr(num + ord('A')) for num in numeros)

    def criar_matriz_chave(k):
        return np.array(k).reshape(2, 2)

    def multiplicar_matriz(matriz, vetor):
        return np.dot(matriz, vetor) % 26
```

```
    def inversa_matriz(matriz):
        det = int(np.round(np.linalg.det(matriz)))
        det_inv = pow(det, -1, 26)
        matriz_adjunto = np.array([[matriz[1, 1], -matriz[0, 1]],
            [-matriz[1, 0], matriz[0, 0]]])
        matriz_inv = (det_inv * matriz_adjunto) % 26
        return matriz_inv

    def cifra(texto, matriz_chave):
        numeros = texto_para_numeros(texto)
        if len(numeros) % 2 != 0:
            numeros.append(0)
        resultado = []
        for i in range(0, len(numeros), 2):
            par = np.array(numeros[i:i+2])
            cifrado = multiplicar_matriz(matriz_chave, par)
            resultado.extend(cifrado)
        return numeros_para_texto(resultado)

    matriz_chave = criar_matriz_chave(chave)
    if modo == 'encriptar':
        return cifra(texto, matriz_chave)
    else:
        matriz_chave_inv = inversa_matriz(matriz_chave)
        return cifra(texto, matriz_chave_inv)

# Exemplo de uso
texto_original = "meetmeattheusualplaceattenratherthaneightoclock"
chave = [9, 4, 5, 7]
texto_encriptado = cifra_hill(texto_original, chave, modo='
    encriptar')
print("Texto Encriptado:", texto_encriptado)

texto_decriptado = cifra_hill(texto_encriptado, chave, modo='
    decriptar')
print("Texto Decriptado:", texto_decriptado)
```

Listing 3: Programa para Cifra de Hill