UNIVERSIDADE FEDERAL DE SANTA CATARINA
CENTRO TECNOLÓGICO
INE - DEPARTAMENTO DE INFORMÁTICA E ESTATÍSTICA

Atividade Prática 1

Aluno: Diego da Silva Marques - 12200625
Disciplina: INE5429 - Segurança em Computação
Professor: Ricardo Custódio
Data: 19/03/2017

Cifra de César

- **Descrição**:

A Cifra de César é uma das mais simples técnicas de criptografia conhecidas. É uma cifra do tipo de substituição monoalfabética, onde um caractere é substituído pela “rotação” do mesmo em torno do alfabeto previamente definido por um número X, onde X é definido como a chave do algoritmo. Caso a letra “A” seja escolhida, levando em consideração o alfabeto de A-Z, com uma rotação de número 3, o resultado da cifra será a letra “D” (pois A + 3 caracteres do alfabeto = D).

- **Implementação (Python)**:

```
#!/usr/bin/python2

ALPHABET_LIMIT = 90
PLAINTEXT = 'DIEGO'
ROTATION = 3

def cipher(text, rotation):
    cipher_text = ''
    for char in text:
        ciphered = chr(ord(char) + rotation - 26) if (ord(char.upper()) + rotation) >
ALPHABET_LIMIT else chr(ord(char) + rotation)
        cipher_text += ciphered

    return cipher_text

def main():
    print 'Ceasar Cipher'
    print 'Plain Text: %s' % PLAINTEXT
    print 'Rotation: %d' % ROTATION
    print 'Ciphered Text: %s' % cipher(PLAINTEXT, ROTATION)

if __name__ == '__main__':
    main()
```

- **Resultado**:

```
Ceasar Cipher
Plain Text: DIEGO
Rotation: 3
Ciphered Text: GLHJR
```

Cifra PlayFair

- **Descrição**:

A cifra PlayFair é uma cifra de substituição Polialfabética que cifra pares de letras, visando dificultar a análise de frequência. Nesta cifra é montada uma tabela 5x5 contendo a chave escolhida como entrada do começo da tabela, e o resto da tabela é preenchido com os caracteres presentes no alfabeto que não estavam presentes na chave. A partir da tabela, os caracteres do texto que deseja ser cifrado são analisados em pares e substituídos através de um algoritmo que faz uma ligação entre suas linhas e colunas.

- **Implementação (Python)**:

```
#!/usr/bin/python2

KEY = 'INFOSEC'
PLAINTEXT = 'DIEGO'
# J is omitted, assuming I=J
FULL_ALPHABET = 'ABCDEFGHIKLMNOPQRSTUVWXYZ'

def getPlayFairTable(key):
    # Remove duplicate characters from key
    key_chars = list(set(key))

    remains = [c for c in FULL_ALPHABET if c not in key]
    alphabet = list(key) + remains

    # Remove J from alphabet
    alphabet = [c for c in alphabet if c != 'J']

    table = ['' for i in xrange(5)]
    table[0] = alphabet[0:5]
    table[1] = alphabet[5:10]
    table[2] = alphabet[10:15]
    table[3] = alphabet[15:20]
    table[4] = alphabet[20:25]

    return table

def getDigraphs(text):
    i = 0
    digs = []
    while i < len(text) - 1:
        dig = text[i:i+2]
        # Add an 'X' to repeated characters
        if dig[0] == dig[1]:
            dig = [dig[0], 'X']
        digs.append(list(dig))
        i += 2
```

```
    # Even number padding
    if len(text) % 2 != 0:
        digs.append([text[-1], 'X'])

    return digs

def getLocation(table, char):
    # Make J = I
    if char == 'J':
        return getLocation(table, 'I')

    for i in xrange(len(table)):
        for j in xrange(len(table[i])):
            if table[i][j] == char:
                return (i, j)

def encryptDigraphs(table, digraphs):
    result = ''
    for dig in digraphs:
        row1, col1 = getLocation(table, dig[0])
        row2, col2 = getLocation(table, dig[1])
        if row1 == row2:
            result += table[row1][col1 + 1] + table[row2][col2 + 1]
        elif col1 == col2:
            result += table[row1 + 1][col1] + table[row2 + 1][col2]
        else:
            result += table[row1][col2] + table[row2][col1]

    return result


def cipher(text, key):
    table = getPlayFairTable(key)
    digraphs = getDigraphs(text)
    ciphered_text = encryptDigraphs(table, digraphs)
    return ciphered_text

def main():
    print 'PlayFair Cipher'
    print 'Plain Text: %s' % PLAINTEXT
    print 'Key: %s' % KEY
    print 'Ciphered Text: %s' % cipher(PLAINTEXT.upper(), KEY.upper())

if __name__ == '__main__':
    main()
```

- **Letras Retiradas/Combinadas:**
    - J=I
    - Letras Iguais em sequência
        - Segunda letra = X
    - Preenchimento para formar apenas pares = X

- **Resultado**:

```
PlayFair Cipher
Plain Text: DIEGO
Key: INFOSEC
Ciphered Text: ESGPFY
```

Cifra de Vigenère

- **Descrição**:

A cifra de Vigenère é uma cifra de substituição polialfabética que utiliza uma chave e executa diversas Cifras de César baseadas nos caracteres contidos na chave, onde cada caractere da chave corresponde a rotação que será feita no caractere de índice equivalente (considerando as palavras como uma lista de caracteres) no texto a ser cifrado.

- **Implementação (Python)**:

```
#!/usr/bin/python2

from itertools import cycle

KEY = 'INFOSEC'
PLAINTEXT = 'DIEGO'

def cipher(text, key):
    result = ''
    # Zip gets two lists and iterates through it combining all elements with
    # the same index on the other list.
    #
    # Cycle iterates through a list going back to the start
    # when the end is reached, just like is needed on Vigenere Algorithm.
    for comb in zip(text, cycle(key)):
        if ord(comb[0]) >= ord('A') and ord(comb[0]) <= ord('Z'):
            result += chr(ord('A') + ((ord(comb[0]) + ord(comb[1])) % 26))

    return result

def main():
    print 'Vigenere Cipher'
    print 'Plain Text: %s' % PLAINTEXT
    print 'Key: %s' % KEY
    print 'Ciphered Text: %s' % cipher(PLAINTEXT.upper(), KEY.upper())

if __name__ == '__main__':
    main()
```

- **Letras Retiradas/Combinadas:** Qualquer caractere não-alfabético
- **Resultado**:

```
Vigenere Cipher
```

```
Plain Text: DIEGO
Key: INFOSEC
Ciphered Text: LVJUG
```