# E-fólio A | Folha de resolução para E-fólio

**UNIDADE CURRICULAR:** Segurança em redes e computadores

**CÓDIGO:** 21181

**DOCENTE:** Henrique São Mamede

**A preencher pelo estudante**

**NOME:** Vitor Manuel Metrogos Frango

**N.º DE ESTUDANTE:** 1802925

**CURSO:** Licenciatura em Engenharia Informática

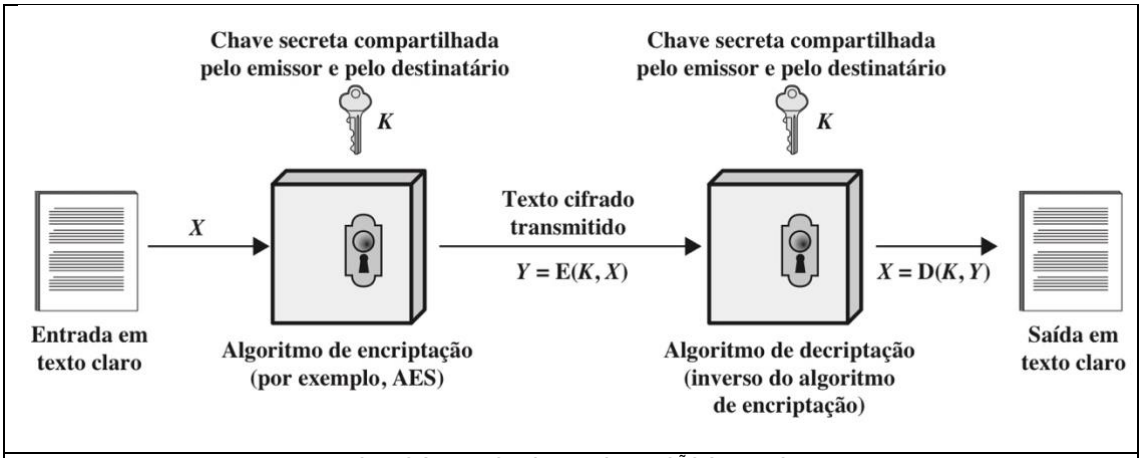
**DATA DE ENTREGA:** 17 de Novembro de 2022

**TRABALHO / RESOLUÇÃO:**

A encriptação simétrica, ou encriptação de chave única, era o único tipo em uso antes do desenvolvimento da encriptação por chave pública na década de 1970.Este método, dos dois tipos de encriptação (encriptação simétrica e encriptação assimétrica) continua sendo de longe o mais usado. A encriptação simétrica utiliza apenas uma chave que precisa ser aplicada para criptografar a mensagem numa ponta e para recuperá-la na outra. Este tipo de criptografia era usado em mensagens durante a Segunda Guerra Mundial, por exemplo. A chave é um tipo de código que permite reconstruir a mensagem original. A desvantagem para este método é a troca da chave entre os dois interlocutores, o processo deve ser feito em segurança de forma que uma terceira pessoa não tenha conhecimento da mesma, toda a segurança é refletida na chave.

O modelo de cifra simétrica possui cinco itens:

1. Texto claro (plain texto) é a mensagem ou dados originais inteligíveis que servem de entrada do algoritmo de encriptação
2. Algoritmo de encriptação realiza diversas substituições e transformações no texto claro
3. Chave secreta é também uma entrada para o algoritmo de encriptação sendo a mesma de um valor independente do texto claro e do algoritmo
4. Texto cifrado é a mensagem produzida como saída do algoritmo de encriptação, este texto depende do texto claro e da chave secreta
5. Algoritmo de decriptação é basicamente o algoritmo de encriptação executado de modo inverso




MODELO SIMPLIFICADO DA ENCRIPTAÇÃO SIMETRICA

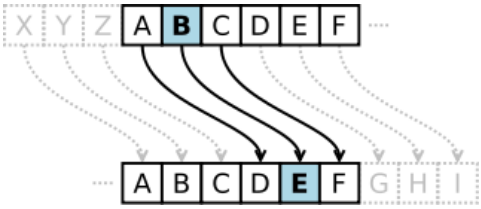
X=texto pleno / E= Algoritmo de encriptação / K = chave / Y = texto cifrado

Técnicas de encriptação clássicas

- Cifra de Cesar
- Cifras monoalfabeticas
- Cifra Playfair
- Cifra de Hill
- Cifras polialfabeticas (de Vigenere )

Para este trabalho escolhi o algoritmo cifra de Cesar por ser, provavelmente, o menos complicado de ser implementado.

A cifra de Cesar é uma das mais simples e conhecidas técnicas de criptografia que consiste numa cifra de substituição e envolve substituir cada letra do alfabeto por aquela que fica três posições adiante




A ação de uma cifra de César é mover cada letra do alfabeto um número de vezes fixo abaixo no alfabeto. Este exemplo está com uma troca de três, então o B no texto normal se torna E no texto cifrado

O algoritmo pode ser expresso da forma abaixo representada, em que casa letra p é substituída pela letra do texto cifrado C

$$C = E(3, p) = (p + 3) mod\ 26$$

No entanto o deslocamento pode ser de qualquer magnitude, ficando o algoritmo geral de encriptação de César da seguinte forma (em que k só permite ate 25 posições)

$$C = E(k, p) = (p + k) mod\ 26$$

E o algoritmo geral de desencriptação de César da seguinte forma

$$p = D(k, C) = (C - k) mod\ 26$$

Para implementação recorri à linguagem de programação C desenvolvendo o respetivo código através do CLion da JetBrains. Como referido anteriormente, para a **Cifra Simétrica** escolhi a mais simples e também a mais antiga das cifras clássicas de substituição - **Cifra de Cesar** - Stallings, W. (2017). Cryptography and Network Security: Principles and Practice. 7th Edition (Global Edition), Prentice Hall. Ripta

O código foi desenvolvido em dois ficheiros independentes “encripta.c” e “decripta.c” para o algoritmo de encriptação e desencriptação respetivamente

Inicialmente é solicitado ao utilizador a introdução de uma mensagem a encriptar bem como a introdução de uma chave numérica após estes dois passos sequenciais o algoritmo irá efetuar as seguintes ações:

1. Identificar o primeiro caracter dentro da string
2. Achar a sua posição dentro do alfabeto ( A -> Z e/ou a -> z)
3. Associar ao respetivo caracter o valor da chave introduzida ( caso a posição seja “posição + chave > 26” volta ao inicio do alfabeto
4. Repetir o passo 3 até ao final da mensagem (string)
5. Criar uma string e apresentar o resultado

Para desencriptar a mensagem obtida o algoritmo correra da mesma forma mas com a diferença em que o passo 3 será “posição – chave”

Código de encriptação

```
#include<stdio.h>

int main()
{
    char msg[100], ch;
    int i, chave;


    printf("Escreva a mensagem a encriptar: \n ");
    gets(msg);

    printf("Escreva uma chave numerica entre 1 e 25 : ");
    scanf("%d", &chave);

    // precorre a string e avalia se todos os caracvteres estao entre
a e z ou A e Z

    for(i = 0; msg[i] != '\0'; ++i){
        ch = msg[i];
        if(ch >= 'a' && ch <= 'z'){
            ch = ch + chave;
            if(ch > 'z'){
                ch = ch - 'z' + 'a' - 1;
            }
            msg[i] = ch;
        }
        else if(ch >= 'A' && ch <= 'Z'){
            ch = ch + chave;
            if(ch > 'Z'){
                ch = ch - 'Z' + 'A' - 1;
            }
            msg[i] = ch;
        }
    }
    printf("A mensagem encriptada é: %s", msg);
    return 0;
}
```

Código de desencriptação

```
#include<stdio.h>

int main()
{
    char msg[100], ch;
    int i, chave;
    printf("Escreva a mensagem a desencriptar: ");
    gets(msg);
```

```
    printf("Escreva uma chave numerica entre 1 e 25: ");
    scanf("%d", &chave);
    for(i = 0; msg[i] != '\0'; ++i){
        ch = msg[i];
        if(ch >= 'a' && ch <= 'z'){
            ch = ch - chave;
            if(ch < 'a'){
                ch = ch + 'z' - 'a' + 1;
            }
            msg[i] = ch;
        }
        else if(ch >= 'A' && ch <= 'Z'){
            ch = ch - chave;
            if(ch < 'A'){
                ch = ch + 'Z' - 'A' + 1;
            }
            msg[i] = ch;
        }
    }
    printf("A mensagem decifrada é: %s", msg);
    return 0;
}
```

A encriptação assimétrica também conhecida como criptografia publica é baseada em 2 tipos de chaves de segurança, uma privada e a outra publica. Elas são usadas para cifrar mensagens e também para verificar a identidade do usuário, ou seja, a chave privada é usada para decifrar mensagens enquanto a publica é utilizada para cifrar um conteúdo. Este sistema garante a privacidade dos utilizadores e aumenta a confiabilidade na troca de dados visto que o número de pessoas com acesso à chaves privada é muito reduzido.

Escolhi o algoritmo RSA por ser um dos mais seguros e também por ter sido o primeiro algoritmo a possibilitar criptografia e assinatura digital e uma das grandes inovações em criptografia de chave publica.

O seu funcionamento envolve, como referido atras, uma chave publica e uma privada, e a encriptação da mensagem sendo a sua implementação da seguinte forma:

Solicita se a introdução de dois números primos p e q, quanto maior for o número mais difícil será de fatorizar, ambos os valores de p e q que após multiplicados obtem-se o valor de 'n' que é um valor que faz parte das duas chaves.

De seguida calculamos o totiente de p q (variável 'tot' no código) através da formula φ(pq) = (p-1)(q-1) de acordo com o manual da UC. Após este cálculo passamos à criação das duas chaves (variáveis 'e' e 'd' no código) bem com a transformação dos caracteres elevando o seu valor em ASCII por 'e' e faz se a operação modular com 'n'. Para desencriptar a mensagem usamos o 'd' na operação modular.

```
#include<stdio.h>
#include<stdlib.h>
#include<math.h>
#include<string.h>


    long int
p,q,n,t,flag,e[100],d[100],temp[100],j,tot[100],en[100],i;
```

```c
char        msg[100];
int         primo(long int);


void        ce();
long int    cd(long int);
void        encripta();
void        desencript();
int         main()
{

    // Solicita introdução de 2 numeros primos
    printf("\nEscreva um numero primo\n");
        scanf("%ld",&p);
        flag=primo(p);
            if(flag==0)
            {
                printf("\nNão é numero primo\n");
                exit(1);
            }
    printf("\nEscreva outro numero primo \n");
    scanf("%ld",&q);

    flag=primo(q);
    if(flag==0||p==q)
    {
        printf("\nNão é numero primo\n");
        exit(1);
    }
    printf("\nEscreva uma mensagem sem espaços\n");
    fflush(stdin);
    scanf("%s",msg);

    // calcula o numero de totiente (co primos)
    for(i=0;msg[i]!=NULL;i++)
        tot[i]=msg[i];
    n=p*q;
    t=(p-1)*(q-1);
    ce();

    // valores para a chave publica e privada
    printf("\nValores possiveis para  'e' e 'd' \n");
    for(i=0;i<j-1;i++)
        printf("\n%ld\t%ld",e[i],d[i]);
        encripta();
        desencript();
    return 0;
}

int primo(long int pr)
{
    int i;
    j=sqrt(pr);
    for(i=2;i<=j;i++)
    {
        if(pr%i==0)
            return 0;
    }
    return 1;
}
```

```
void ce()
{
    int k;
    k=0;
    for(i=2;i<t;i++)
    {
        if(t%i==0)
            continue;
        flag=primo(i);
        if(flag==1&&i!=p&&i!=q)
        {
            e[k]=i; flag=cd(e[k]);
            if(flag>0)
            {
                d[k]=flag;
                k++;
            }
            if(k==99)
                break;
        }
    }
}

long int cd(long int x)
{
    long int k=1;
    while(1)
    {
        k=k+t;
        if(k%x==0)
            return(k/x);
    }
}
void encripta()
{
    long int pt,ct,key=e[0],k,len;
    i=0;
    len=strlen(msg);
    while(i!=len)
    {
        pt=tot[i];
        pt=pt-96;
        k=1;
        for(j=0;j<key;j++)
        {
            k=k*pt;
            k=k%n;
        }
        temp[i]=k;
            ct=k+96;
            en[i]=ct;
        i++;
    }
    en[i]=-1;

    printf("\nA mesangem encriptada é: \n");
    for(i=0;en[i]!=-1;i++)
        printf("%c",en[i]);
}
void desencript()
{
```

```
        long int pt,ct,key=d[0],k;
        i=0;

        while(en[i]!=-1)
        {
            ct=temp[i];
            k=1;
            for(j=0;j<key;j++)
            {
                k=k*ct;
                k=k%n;
            }
                pt=k+96;
                tot[i]=pt;
                i++;
        }
        tot[i]=-1;

        printf("\nA mensagem desencriptada é: \n");
        for(i=0;tot[i]!=-1;i++)
            printf("%c",tot[i]);
    }
```

REFERÊNCIAS BIBLIOGRÁFICAS:


Stallings, William (2007) Cryptography and Network Security: Principles and Practice. 7th Edition (Global Edition), Prentice Hall.

https://pt.wikipedia.org/wiki/Cifra_de_César

https://pplware.sapo.pt/internet/criptografia-simetrica-e-assimetrica-qual-a-diferenca/

https://acervolima.com/cifras-simetricas-tradicionais/

http://cecead.com/assuntos/disciplinas/seguranca-da-informacao/aula-03-seguranca-da-informacao/

https://stackoverflow.com/search?q=rsa