Criptografia

Pedro Quaresma
Departamento de Matemática, Universidade de Coimbra
3001-454 COIMBRA, PORTUGAL
pedro@mat.uc.pt

Elsa Lopes
Núcleo de Estágio Pedagógico
Lic. Matemática, F.C.T.U.C.
Escola B. 2, 3 c/ Sec. José Falcão, Miranda do Corvo
elsalopes80@sapo.pt

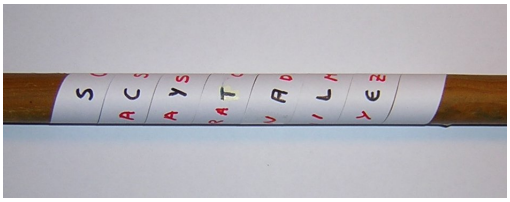
1 Introdução

A necessidade de proteger os canais de comunicação entre pessoas de uma mesma comunidade vem desde os primórdios da civilização, a ideia de não só proteger os meios de comunicação mas também de proteger o próprio conteúdo da mensagem, através da cifração da mensagem é também muito antiga. O Imperador Romano Júlio César (100 – 44 a.C.) desenvolveu uma cifra simples para poder comunicar com os seus Generais: na mensagem original cada letra é «deslocada» três posições para a direita, considerando-se que o alfabeto se fecha sobre si próprio, isto é, que após a última letra vem a primeira; o receptor da mensagem só tem que «deslocar» cada letra três posições para a esquerda para obter a mensagem original.

A cifração de mensagens foi-se tornando um processo cada vez mais sofisticado, passando pelas máquinas Enigma [3] usadas pelo exército alemão aquando da Segunda Guerra Mundial, até aos nossos dias com as transacções electrónicas na *Internet*. Na actual *Sociedade da Informação*, em que cada vez mais as pessoas comunicam através da *Internet*, um meio de comunicação muito exposto, a importância da criptografia é enorme, só através da cifração das comunicações é que podemos garantir a confidencialidade da informação que queremos transmitir.

2 O Surgimento da Criptografia

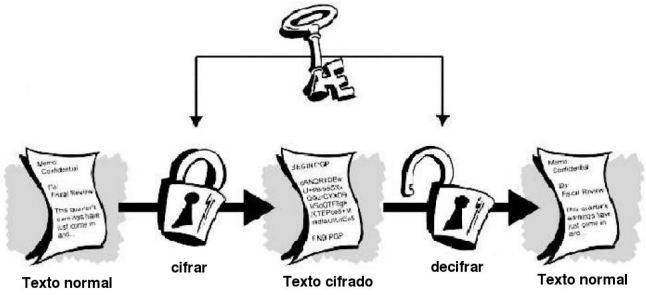
O surgimento da criptografia (do Grego: kryptós, oculto + graph, r. de graphein, escrever) deve ter sido quase que simultâneo com o da escrita [7]. Os Espartanos, em 400 a.C., desenvolveram um sistema muito curioso, num bastão enrolava-se uma tira de couro, após isso escrevia-se a mensagem na tira de couro, o acto de desenrolar a tira do bastão cifrava a mensagem, a qual só podia ser decifrada tornando a enrolar a tira num bastão de diâmetro semelhante.



Em contraponto com este método puramente mecânico a cifra de Júlio César implicava um algoritmo de cifração. Um sistema criptográfico é então um conjunto de técnicas que nos permitem tornar incompreensível uma dada mensagem, de modo que só o verdadeiro destinatário da mesma a consiga decifrar, obtendo dessa forma o texto original.

2.1 Sistemas Criptográficos Simétricos

Os primeiros sistemas criptográficos inventados eram do tipo *criptografia simétrica*, ou de *chave secreta*. Sistemas em que existe uma só *chave de cifração*, e em que os processos de cifração e de decifração são simétricos[1].



No caso do algoritmo de Júlio César estamos perante um algoritmo mono-alfabético aditivo, isto é, no processo de cifração só é utilizado um alfabeto, e basta somar ou subtrair três ao código numérico de cada letra do alfabeto.

Qual será o significado da frase?

```
D FKDYH WHP GH VHU PDQWLGD VHFUHWD
```

Embora se possam desenvolver sistemas mais sofisticados, nomeadamente os métodos poli-alfabéticos multiplicativos [6] este tipo de sistema tem sempre dois problemas de base que limitam a sua capacidade de proteger a informação:

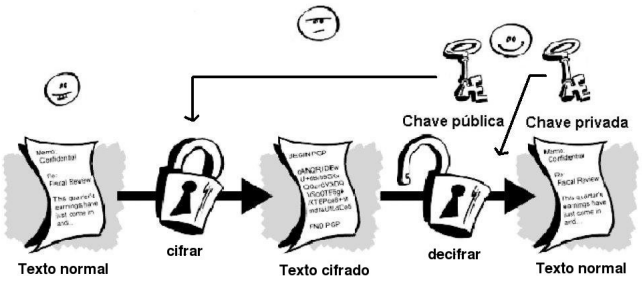
- a chave de cifração tem de ser do conhecimento de toda a organização «amiga», e tem de ser mantida secreta de todas as organizações inimigas. Quanto maior for a complexidade da organização «amiga» mais difícil é de verificar esta condição;

[1] Esta imagem e a seguinte (os quais foram por nós modificadas) foram retiradas do texto (disponível na *Internet*), Francisco Gomes Milagres, «Segurança da Informação: Estratégias para Neutralizar o Inimigo», Universidade do Estado de Minas Gerais, Faculdade de Informática de Passos, 21 de Maio de 2003.

- a despeito de algoritmos mais sofisticados, o estudo das línguas naturais, a sua construção frásica, a frequência relativa das diferentes letras do alfabeto, entre outras características, permitem obter muita informação que pode depois ser usada para quebrar o código de cifração.

2.2 Sistemas Criptográficos Assimétricos

Surgem então em cena os sistemas de *criptografia assimétrica* ou de *chave pública*. Sistemas em que o processo de cifração usa uma *chave pública*, mas em que o processo de decifração usa uma chave diferente, dita *chave privada*.



Este tipo de sistema resolve os dois problemas acima expostos:

- a chave privada é do conhecimento de uma única entidade, o receptor da mensagem, mantê-la secreta é assim muito mais fácil.
- os algoritmos desenvolvidos são bastante mais complicados de quebrar do que os anteriores.

No que se segue vamos descrever um dos algoritmos actualmente usados. Esperamos conseguir convencer o leitor da maior dificuldade existente em quebrar um código deste tipo quando em contraponto com os anteriores métodos. Queremos no entanto referir dois pontos: primeiro que as implementações apresentadas usam estruturas de dados simples, comummente encontradas nas linguagens de programação, o que leva a que não seja possível dificultar muito a tarefa do «inimigo»; por outro lado no exemplo que iremos apresentar mais à frente a cifração é feita carácter a carácter o que não é o caso das implementações em uso na *Internet* as quais usam blocos de caracteres como forma de evitar o estudo linguístico da mensagem cifrada.

3 O Algoritmo RSA

Um sistema assimétrico muito usado na actualidade é o assim designado *sistema de criptografia RSA* [2, 4], o qual obtém o seu nome das iniciais dos seus três autores. Vamos de seguida descrevê-lo, apresentando também uma sua implementação desenvolvida no sistema de programação numérica *Octave*[2].

[2]Octave, `www.octave.org`, é um sistema de programação numérica de distribuição gratuita, compatível com o *MatLab*.

Num sistema de criptografia assimétrica é então necessário possuir programas para:

- gerar as chaves pública e privada (secreta), C_p e C_s;
- cifrar as mensagens $A_{C_p} : M \longrightarrow A_{C_p}(M)$;
- decifrar as mensagens $A_{C_s} : M \longrightarrow A_{C_s}(M)$;

Para que estejamos perante um sistema de criptografia e não perante um simples sistema de baralhação de mensagens, os programas de cifração e decifração têm de ser funções inversas, isto é, tem-se de verificar que:

$$A_{C_s}(A_{C_p}(M)) = M \qquad A_{C_p}(A_{C_s}(M)) = M$$

O sistema RSA vai usar resultados conhecidos da Teoria dos Números para poder assegurar uma grande segurança no processo de decifração, não será de estranhar portanto que surja a necessidade de trabalhar com números primos.

3.1 Geração das Chaves

As chaves pública e privada vão ser constituídas, cada uma delas, por um par de números inteiros, os quais vão depois constituir o âmago dos processos de cifração e decifração.

Começa-se por escolher dois números primos p e q, deles obtêm-se $n = pq$.

De seguida determina-se a função φ de Euler para n, $\varphi(n)$ dá-nos o número de naturais inferiores ou iguais a n e que são primos com n. A função de Euler é uma função de inteiros em inteiros que, entre outras, tem as seguintes propriedades [2].

Teorema 1 *Se m e n forem dois números naturais, primos entre si, tem-se que $\varphi(mn) = \varphi(m)\varphi(n)$.*

Teorema 2 *Um número natural p é primo se e só se $\varphi(p) = p - 1$.*

Temos então que $\varphi(n) = \varphi(p)\varphi(q) = (p-1)(q-1)$, ou seja o cálculo de $\varphi(n)$ é, dado a escolha de p e q, muito fácil de efectuar.

O próximo passo é o de escolher um natural e, tal que, $1 < e < \varphi(n)$ e que seja primo relativo com $\varphi(n)$. O par (e, n) é a chave pública do código RSA.

Finalmente determina-se um natural d que seja o inverso multiplicativo de e, módulo $\varphi(n)$, ou seja deve-se verificar a seguinte igualdade $d\,e = 1 (\text{mod } \varphi(n))$. O par (d, n) é a chave privada do código RSA.

A obtenção dos números primos p e q pode ser feita recorrendo a um dos muitos algoritmos para a obtenção de números primos, por exemplo o *Crivo de Eratóstenes* [5, pag. 278].

O algoritmo para a criação das chaves é o seguinte:

```
##Determinação das Chaves Pública e Privada:
##
## -> p,q dois números primos.
```

```
## <- (e,n) e (d,n), as chaves públicas e privadas.
function chaves(p, q)
  n = p*q;
  fi = (p-1)*(q-1);
  e = 2;
  k = 0;
  do
    e = e+1;
  until (gcd(fi,e) == 1)
  achou = false;
  while (!achou)
    d = (1 + (k * fi))/e;
    if ( d == round(d))
      achou = true;
    else
      k = k+1;
    endif
  endwhile
  printf("\n A chave pública é (%d, %d). \n", e, n);
  printf("\n A chave privada é (%d, %d). \n \n", d, n);
endfunction
```

Para os valores de $p = 11$ e $q = 23$ ter-se-ia:

```
octave>  chaves(11,23)

 A chave pública é (3, 253).

 A chave privada é (147, 253).
```

O Algoritmo de cifração RSA é:

$$C = A_{C_p}(M) = M^e (\text{mod } n)$$

e o algoritmo de decifração é:

$$M = A_{C_s}(C) = C^d (\text{mod } n)$$

Em *Octave* temos:

```
##Cifrar a Mensagem Digital Original:
##
## -> (e,n), chave pública
##    m, mensagem a cifrar (vector de inteiros)
## <- x, mensagem cifrada (vector de inteiros)
function x=cifrar(e, n, m)
```

```
  t=columns(m);
  for i=1:t
    x(i) = mod((m(i))^e, n);
  endfor
endfunction

##Decifrar a Mensagem Cifrada:
##
## -> (d,n), chave pública
##    c, mensagem a decifrar (vector de inteiros)
## <- modl, mensagem decifrada (vector de inteiros)
function modl = decifrar(d, n, c)
  t=columns(c);
  for i=1:t
    modl(i) = 1;
    j=1;
    while (j <= d)
      modl(i) = mod((c(i))*modl(i), n);
      j = j+1;
    endwhile
  endfor
endfunction
```

Sendo que nesta nossa implementação simplificada do algoritmo RSA a mensagem é primeiro convertida de um vector de caracteres num vector de naturais, de seguida cifrada, depois decifrada e finalmente convertida de novo num vector de caracteres[3].

3.2 Validação do Sistema RSA

Como já dissemos antes é necessário verificar se estamos perante um sistema de criptografia válido, isto é, temos que verificar que [2]:

$$A_{C_s}(A_{C_p}(M)) = A_{C_p}(A_{C_s}(M)) = M^{ed} (\text{mod } n)$$

Para o desenvolvimento da demonstração são necessários alguns resultados auxiliares.

Definição 1 (Congruência módulo n) *Sejam $n \in \mathbb{N}$ e $a, b \in \mathbb{Z}$, então a e b dizem-se* congruentes módulo n *se tiverem o mesmo resto na divisão por n, denota-se tal facto por $a \equiv b (\text{mod } n)$.*

Decorre da definição que se $a \equiv b (\text{mod } n)$ então $a = b + kn$, para um dado $k \in \mathbb{Z}$.

Teorema 3 (Teorema de Euler) *Sejam $a \in \mathbb{Z}$ e $n \in \mathbb{N}$. Se $\text{mdc}(a, n) = 1$, então $a^{\varphi(n)} \equiv 1 (\text{mod } n)$.*

[3] O programa Octave contendo todas as funções referidas no texto pode ser obtido em `http://www.mat.uc.pt/~pedro/cientificos/Cripto/`

Teorema 4 (Pequeno Teorema de Fermat) *Se n é um número primo, então $a^{n-1} \equiv 1(\text{mod } n)$, para todo o $a \in \mathbb{Z}$ tal que* $\text{mdc}(a, n) = 1$

Teorema 5 (Sistema de Criptografia RSA) *Sendo (e, n) e (d, n) as chaves pública e privada respectivamente do Sistema de Criptografia RSA verifica-se então que:*

$$(m^e)^d (\text{mod } n) = m$$

para qualquer inteiro m, com $0 \leq m < n$.

Demonstração

Da definição de e e d tira-se que $ed \equiv 1(\text{mod } \varphi(n))$ existe então um $k \in \mathbb{Z}$ tal que $ed = 1 + k\varphi(n)$, ou seja:

$$ed = 1 + k(p-1)(q-1),\ k \in \mathbb{Z}$$

donde

$$(m^e)^d = m^{ed} = m^{1+k(p-1)(q-1)} = m(m^{(p-1)(q-1)})^k$$

segue-se que

$$(m^e)^d \equiv m(m^{(p-1)})^{(q-1)k} \equiv m(\text{mod } p)$$

Se p não é um divisor de m esta congruência é uma consequência do Pequeno Teorema de Fermat. Caso contrário a asserção é trivial dado que ambos os membros da equação são congruentes com 0 mod p.

De forma análoga ter-se-ia que:

$$(m^e)^d \equiv m(\text{mod } q)$$

Dado que p e q são números primos distintos e dado que se assume que $0 \leq m < n$, obtém-se

$$(m^e)^d \equiv m(\text{mod } n)$$

□

3.3 Como «Quebrar» o Código RSA

Por quebrar um código entende-se o acto de conseguir decifrar a mensagem sem que se tenha um prévio conhecimento da chave secreta. Para quebrar o código RSA basta descobrir o d, o qual pode ser obtido de e, de p e de q. O e pertence à chave pública, o p, e o q são factores primos de n, o qual é o outro elemento da chave pública. Ou seja para quebrar um sistema deste tipo basta factorizar n.

O problema reside então na factorização em números primos de um dado número natural n. Para valores de n suficientemente grandes esta tarefa é impraticável, mas isso é tema para outro artigo, até lá deixamos ao leitor da *Gazeta de Matemática* algumas pistas [1, 2, 7] e um pequeno desafio.

```
359394 185904 0 231105 382481 474195 382481 10935 75745 382481
185904 0 201637 382481 302441 522545 270765 382481 185904 0 185904
382481 265174 79985 0 365807 292080 66056 261188 75745 382481 371293
60839 185904 185904 265174 185904 0 90175 75745 75745 382481 185904
270765 522545 10935 66056 474195
```

sabendo que se usaram os algoritmos descritos acima, com uma cifração letra a letra (caracteres *ASCII* entre ' ' e '~' que correspondem a, ' '=0, '!'=1,...), e a nossa chave pública é $(5, 561971)$.

Referências

[1] D. Atkins, M. Graff, A. Lenstra, e P. Leyland. The magic words are squeamish ossifrage. In *ASIACRYPT 1994*, pages 263–277, 1994.

[2] Johannes Buchmann. *Introduction to Cryptography.* Springer-Verlag, New York, 2000.

[3] António Machiavelo. O que vem à rede .... *Gazeta de Matemática*, (147):14–15, Julho 2004.

[4] R. Rivest, A. Shamir, e L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.

[5] Pimenta Rodrigues, Pedro Pereira, e Manuela Sousa. *Programação em C++.* FCA, Editora de Informática LDA, 2 edition, 1998.

[6] Abraham Sinkov. *Elementary Cryptanalysis, a mathematical approach.* The Mathematical Association of America, 1966.

[7] Richard Spillman. *Classical and Contemporary Cryptology.* Prentice Hall, 2005.