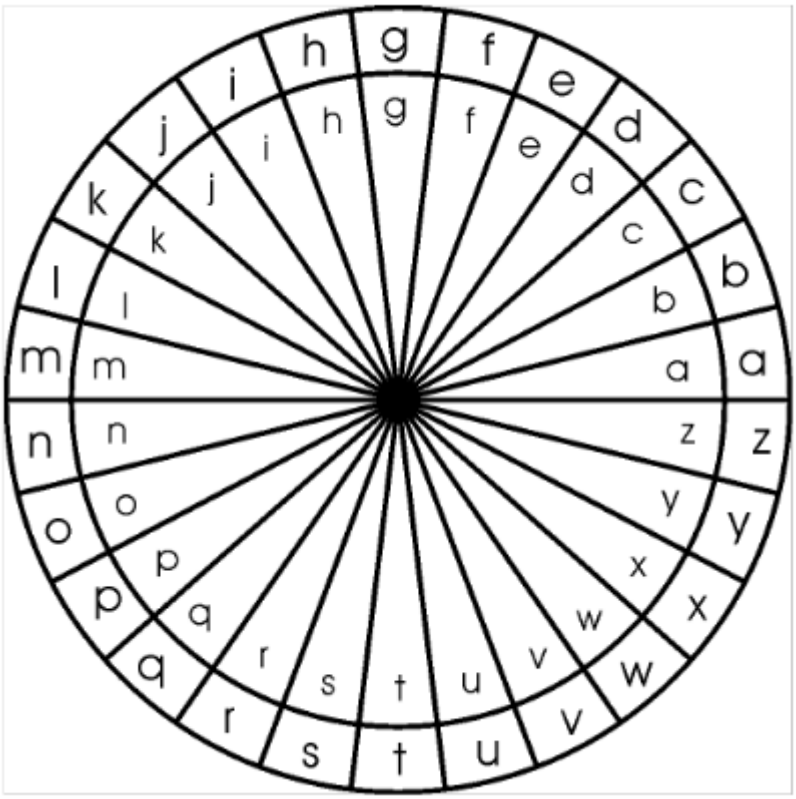
1 Criptografia

Criptografia de chave secreta: quando remetente e destinatário concordam um código.

Exemplo: "Código de Caesar"



O que está escríto?

D OLJHLUD UDSRVD PDUURP VDOWRX VREUH R FDFKRUUR FDQVDGR[1]

Desvantagens:

Os dois precisam conhecer o código, então

1. pode ser roubado mais facilmente,
2. precisa concordar com antecedência ou enviar separadamente.

Criptografia de chave pública: quando o algorítmo de codificação é público, mas o de decodificação é conhecido apenas pelo destinatário (ou viceversa).

IDEIA GERAL:

- Função de codificação **publica** P
- Função de decodificação **secreta** S:
  que seja a inversa da P, isto é, $P(S(m) = m$ e $S(P(m)) = m$

> **P e S devem ser tais que mesmo quem conhece P não consegue descobrir S.**

Envio de texto criptografado:
qualquer um envia $c = P(m)$, o dono do secreto é o único que pode ler $m = S(c) = S(P(m))$

Envio de texto assinado:
o dono do secreto envia $c = S(m)$, qualquer um pode ler $m = P(c) = P(S(m))$ e terá a certeza que quem enviou foi o dono do secreto.

Envio de texto criptografado e assinado:
Sejam
- P_A e S_A as chaves pública e secreta de A.
- P_B e S_B as chaves pública e secreta de B.

Se A envia $S_A(P_B(m))$ todos podem obter $P_B(m))$, mas só B pode decriptar e saberá que foi A quem enviou.

Se A envia $P_B(S_A(m))$ só B pode obter $S_A(m)$, da qual ele obtém m com a certeza que foi A quem enviou.

Como devem ser P e S?

- agir sobre conjuntos "grandes", para não poder inverter "construindo a tabela inversa"
- P fácil de aplicar (pelo menos com computador padrão) mas difícil de inverter (mesmo usando supercomputadores!)

Exemplos:

- $P : Z_d \to Z_d : P(m) = (m + a)(mod\ d)$:
  trivial: $S : Z_d \to Z_d : S(c) = (c - a)(mod\ d)$
- $P : Z_d \to Z_d : P(m) = (m * a)(mod\ d)$:
  fácil: $S : Z_d \to Z_d : S(c) = (c * a^{-1})(mod\ d)$, onde a^{-1} calcula via Alg. de Euclide Estendido (mesmo se d muito grande).

O que sim vai servir será a operação de **potência a módulo d**!
$P : Z_d \to Z_d : P(m) = (m^e)(mod\ d)$: com e e d oportunos (e d da ordem de 10^{300}).

---

2 Precodificação

Dado $d \in \mathbb{N}$ da ordem de 10^{300}, assumiremos como dado um código (público) que permita transformar uma mensagem em um (ou mais) número $m \in \mathbb{Z}_d$, e vice-versa.

O problema da criptografia se torna então o de determinar as funções $P, S : \mathbb{Z}_d \to \mathbb{Z}_d$ com as propriedades descritas acima. [1]

[1] A LIGEIRA RAPOSA MARROM SALTOU SOBRE O CACHORRO CANSADO

3 O RSA

Sistema inventado por **Ron Rivest, Adi Shamir, Leonard Adleman (77)**

Sejam

- p, q dois primos grandes (da ordem de 10^{150}) (**secretos**)
- $n = pq$ (**público**), $\phi = (p-1)(q-1)$ (**secreto**)
- $e \in Z_\phi \setminus \{1\}$ (**público**) invertível ($MCD(e, \phi) = 1$)
- $f = e^{-1}$ em Z_ϕ (**secreto**)

Observações:

- sabendo ϕ e n não seria difícil calcular p e q.
- sabendo p, q calcula-se ϕ, com isso para achar um e invertível e seu inverso usa-se o Alg de Eucl ext. (computacionalmente fácil)
- sabendo só (n, e) descobrir um qualquer entre p, q, ϕ, f equivale a fatorar (computacionalmente difícil)

CODÍFICA: usando a **chave pública** (n, e)
dada a mensagem $m \in Z_n$, a codífica é $c = m^e \ (mod\ n)$:

$$P : Z_n \to Z_n : m \mapsto m^e \ (mod\ n)$$

DECODÍFICA: usando a **chave secreta** (n, f)
dada a mensagem codificada $c \in Z_n$ a decodífica é $d = c^f \ (mod\ n)$,

$$S : Z_n \to Z_n : c \mapsto c^f \ (mod\ n)$$

Teorema 3.1 (RSA).
$S = P^{-1}$

4 Algoritmos de cálculo

Cálculo de $a * b$ para fatores grandes:

Suponha que podemos armazenar e fazer operações diretamente com variáveis que são inteiros de S bits (por exemplo S=32 para unsigned int in C)

Para armazenar dois naturais $a, b < 2^{TS}$ precisamos $2T$ variáveis $a_0, .., a_{T-1}, b_0, .., b_{T-1}$ de forma que

$$a = \sum_{i=0}^{T-1} a_i 2^{Si}, \qquad b = \sum_{i=0}^{T-1} b_i 2^{Si}$$

Então

$$ab = \left(\sum_{i=0}^{T-1} a_i 2^{Si}\right)\left(\sum_{j=0}^{T-1} b_j 2^{Sj}\right) = \sum_{i,j=0}^{T-1} a_i b_j 2^{S(i+j)} = \ldots$$

O número de produtos de int que precisamos fazer é $\sim T^2$.

Cálculo de $a^b \ (mod\ c)$:

Suponha

- $b \in \mathbb{N}$ é feito de k dígitos binários: $b = \sum_{i=0}^{k-1} b_i\, 2^i = \sum_{\substack{i=0,..,k-1 \\ b_i=1}} 2^i$
- $a, c \in \mathbb{N}$ com $a < c < 2^{TS}$

(1) ponha $a_1 = a$

(P) calcule $a_2 = a^2 \ (mod\ c)$

(P) calcule $a_{2^i} = a_{2^{i-1}}^2 \ (mod\ c)$, até quando $i = k-1$

(kP) calcule $\prod_{i=0}^{k-1} (a_{2^i})^{b_i} \ (mod\ c) = \prod_{\substack{i=0,..,k-1 \\ b_i=1}} a_{2^i}$

cada passo (P) envolve um produto de dois números "grandes"

o passo (kP) envolve tantos passos análogos a (P) quantos são os dígitos 1 de b

O número de produtos de int que precisamos fazer é $\sim kT^2$.

5 Procurando primos*

(*COUTINHO, S.C., Números Inteiros e Criptografia RSA)

Teorema 5.1 (Teorema dos números primos, 1896)**.**
Seja

$$\pi(N) = \textit{o número de primos menores que } N.$$

Então

$$\lim_{N\to\infty} \frac{\pi(N)}{\left(\frac{N}{\ln(N)}\right)} = 1$$

Isso significa que **a probabilidade de N ser primo é cerca de** $\frac{1}{\ln(N)}$.

Precisamos um teste para verificar se um número é primo, já que tentar fatorá-lo é computacionalmente inviável.

5.1 Pseudoprimos

Observação: Pelo Peq. T. de Fermat (contrapositiva)

Se existe $w \in \mathbb{Z}_d \setminus \{0\}$ tal que $w^{d-1} \ (mod\ d) \neq 1$ então d é composto.

Definições:

- Chamamos $w \in \mathbb{Z}_d \setminus \{0\}$ de **testemunha** (do fato que d é composto) se $w^{d-1} \ (mod\ d) \neq 1$
- Se d é composto, $1 < b < d-1$ e $b^{d-1} \ (mod\ d) = 1$, dizemos que **d é pseudoprimo com respeito à base** b

Para provar que d é composto, em vez de procurar por divisores de d podemos procurar por testemunhas.

quanto vale $2^{d-1} \ (mod\ d)$? `pseudo2.txt`

- 2 é testemunha para todos os compostos entre 3 e 99
- $341 = 11 \cdot 31$ é pseudoprimo com respeito à base 2, mas não com respeito à base 3!! (ou seja, 2 não é testemunha mas 3 sim!)
- $561 = 3 \cdot 11 \cdot 17$ é pseudoprimo com respeito a toda base prima com 561. Ou seja, as unícas testemunhas são os não invertíveis. Numeros assim são ditos **de Carmichael**)

5.2 Teste de Miller - Rabin:

Observação:

Seja $d = 2^k q + 1$ com q ímpar, $k \geq 1$.

Se d é primo então

- $b^{2^k q} \ (mod\ d) = 1$, logo a sequência

$$b^{2^i q} \ (mod\ d) = 1 : \quad i = 0, .., k$$

termina por 1.

Também vale um dos seguintes (TMR):

- a sequência começa por 1 e logo vale sempre 1
- a sequência vale $d - 1$ antes do primeiro 1

Se (TMR) não acontecer então d é composto.

Definição

- Se d é composto, $1 < b < d - 1$ e a sequência $b^{2^i q} \ (mod\ d)$ satisfaz (TMR) dizemos que **d é fortemente pseudoprimo com respeito à base** b.

O **Test de Miller-Rabin** verifica (TMR) para certa base b:

- se (TMR) vale então d é primo ou fortemente pseudoprimo com respeito à base b
- se (TMR) não vale então d é composto

Pseudoprimos fortes existem, porem vale o seguinte

Teorema 5.2 (Teorema de Rabin, 1980)**.** *Seja d ímpar. Se (TMR) vale para mais que $d/4$ bases entre 1 e $d-1$,*
então d é primo.

Isso significa que

- não existem compostos que satisfaçam (TMR) para todo b invertível em $\mathbb{Z}_d$,
- se (TMR) vale para uma base b, a probabilidade que d seja composto é no máximo $1/4$ (prob que a base escolhida seja uma das $d/4$ do teor) ...
  se (TMR) vale para k bases distintas, a probabilidade que d seja composto é no máximo $1/4^k$
  se (TMR) vale para 10 bases distintas, a probabilidade que d seja composto é no máximo $1/4^{10} \sim 10^{-6}$

Logo **testando algumas vezes podemos chegar a uma probabilidade alta quanto quisermos que d seja primo.**

Curiosidade: entre 1 e 10^9 existem

- 50'847'534 primos
- 5597 pseudoprimos com respeito à base 2
- 1272 pseudoprimos com respeito às bases 2 e 3
- 685 pseudoprimos com respeito às bases 2, 3 e 5
- 646 números de Carmichael
- 1282 pseudoprimos fortes com respeito à base 2

5.3 Rotina para escolha dos primos para RSA

- escolha o numero de dígitos para n, divida elas entre p e q de forma não igual
- escolha N aleatoriamente do tamanho desejado para p e considere cerca de $3\ln(N)$ números a seguir
- procure divisores pequenos para eliminar a maioria deles
- use o Teste de Miller- Rabin nos que sobram com um número de bases que deixe a probabilidade de erro pequena quanto queira.
- se nenhum número passa o test tente de novo com outro N (improvável)
- uma vez encontrado o candidato ainda faça tentativas de fatorá-lo... para maior segurança.
- repita para q

Observe que a probabilidade de obter um pseudoprimo forte que não é primo pode ser deixada pequena, mas seria nula apenas testando com um quarto das bases, o que é inviável!