# Criptografia Aplicada

## Criptografia assimétrica - RSA

LabSEC
Laboratório de Segurança em Computação

# Sumário

- Definições básicas de criptografia de chave pública
- Criptossistema RSA
- Corretude, Eficiência e Segurança
- RSA na prática

# Criptografia de chave pública (assimétrica)

- Uma das maiores evoluções na história da criptografia
- Baseada em funções matemáticas ao invés de substituições e permutações
- São assimétricas: utilizam duas chaves diferentes, o que tem consequências profundas nas áreas de confidencialidade, distribuição de chaves, autenticação, etc.

# Características

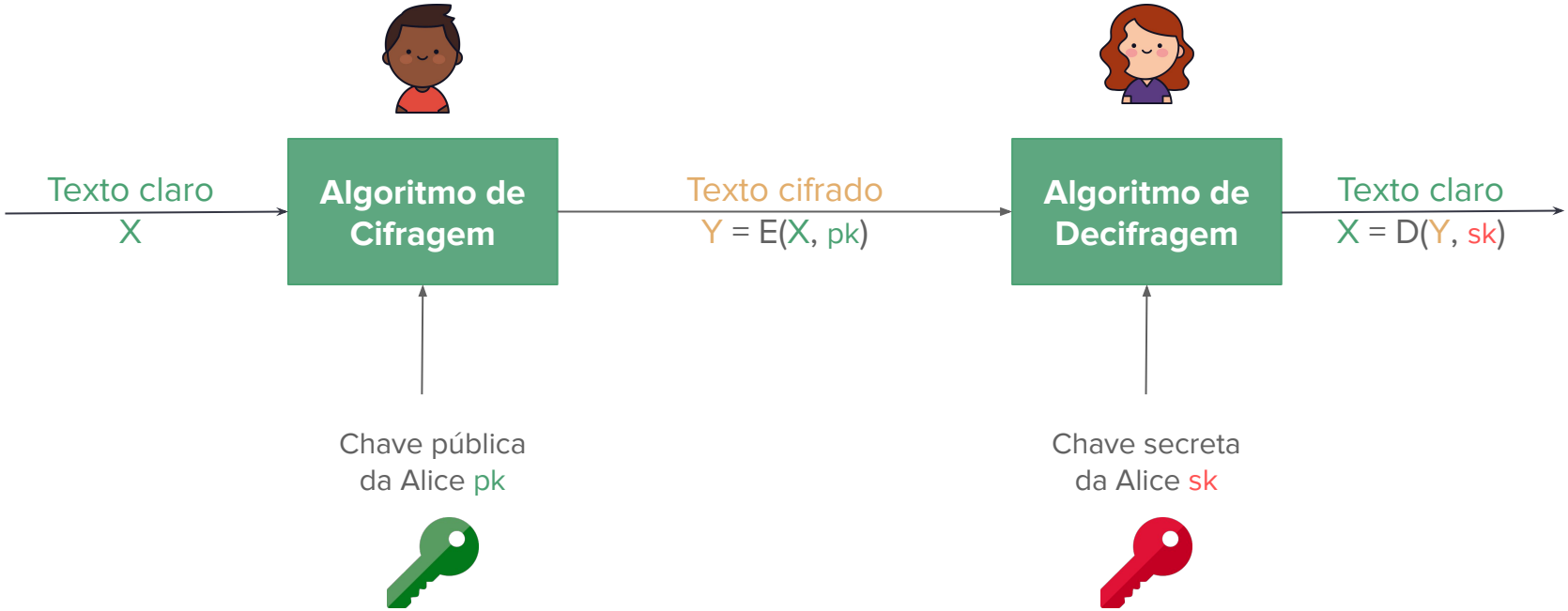
Um criptossistema de chave pública possui os seguintes ingredientes:

1. texto claro
2. algoritmo de cifragem
3. par de chaves (uma pública (pk) e outra privada (sk) ) 
4. texto cifrado
5. algoritmo de decifragem

# Características

- Cada usuário gera um par de chaves
  - Uma pública, disponibilizada em algum local público
  - Uma privada, que nunca deve ser compartilhada
- Se Bob quer enviar uma mensagem **confidencial** para Alice:
  - ele cifra a mensagem com a **chave pública** da Alice,
  - ao receber a mensagem, Alice decifra com sua **chave privada**.
  - Ela é a única que tem acesso à chave e pode decifrar a mensagem.

# Confidencialidade

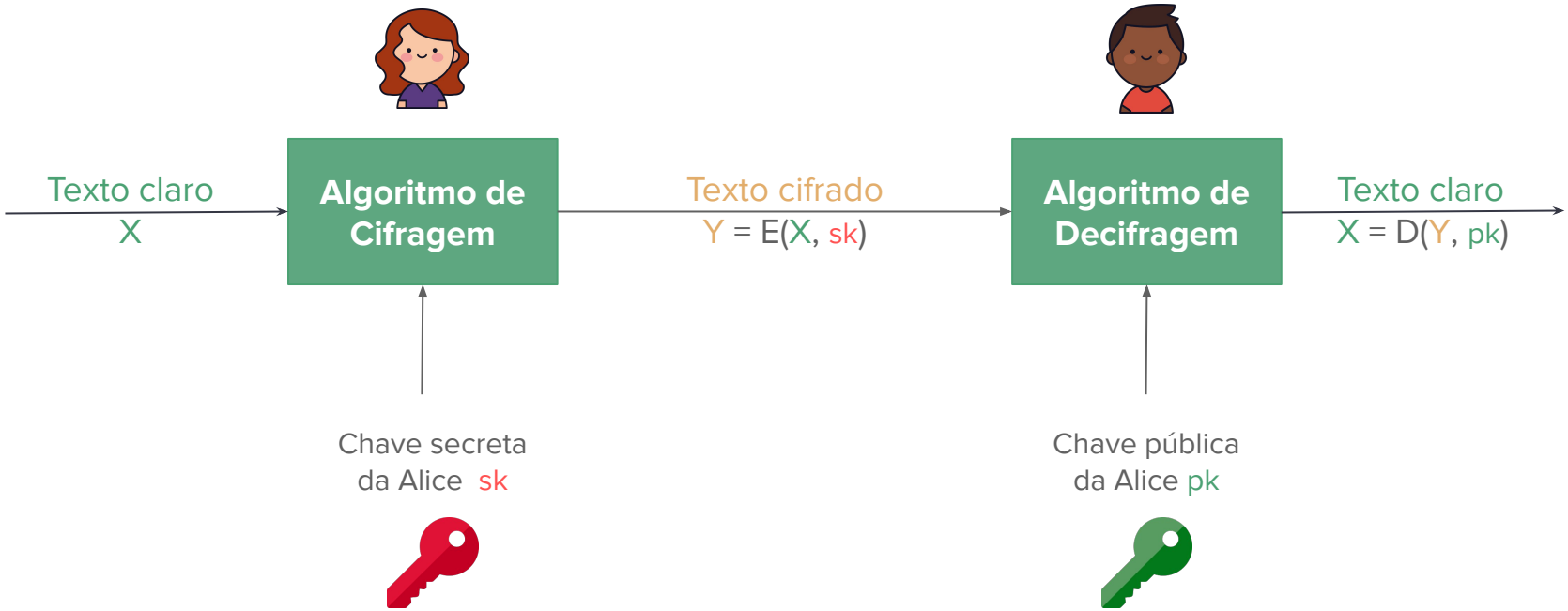


# Características

- Alguns algoritmos permitem que qualquer uma das chaves seja usada para cifragem, enquanto a outra é usada para a decifragem
- Podemos atingir outros objetivos, como a **autenticidade**.
  - A mensagem cifrada nesse caso serve como uma **assinatura digital**
- Além disso, alguns algoritmos permitem que emissor e receptor cooperem para executar um protocolo de **troca de chaves**.

- Se Alice quer convencer Bob da autenticidade de sua mensagem:
  - ela cifra a mensagem com sua **chave privada**,
  - Bob decifra com a **chave pública** da Alice.
  - Alice é a única que tem acesso à sua chave privada, então temos a certeza de que a mensagem veio dela.
  - É impossível alterar a mensagem sem ter acesso à chave privada da Alice, portanto garantimos também a integridade da mensagem.

# Autenticidade



# Aplicações

- Podemos classificar criptossistemas de chaves públicas em três categorias:
  - Cifragem/decifragem
  - Assinatura digital
  - Troca de chaves
- Alguns algoritmos servem para os três propósitos, outros apenas para um ou dois deles.

Table 9.3 Applications for Public-Key Cryptosystems

| Algorithm | Encryption/Decryption | Digital Signature | Key Exchange |
|---|---|---|---|
| RSA | Yes | Yes | Yes |
| Elliptic Curve | Yes | Yes | Yes |
| Diffie–Hellman | No | No | Yes |
| DSS | No | Yes | No |

Imagem: W. Stallings. *Cryptography and network security*. Cap 9.3

# Requisitos

- Computacionalmente fácil gerar um par de chaves
- Computacionalmente fácil para o remetente operar com a chave pública (pk)
  - c = E(m, pk)
- Fácil para o destinatário operar com a chave privada (sk)
  - m = D(c, sk) = D(E(m, pk), sk)
- Computacionalmente inviável que um adversário determine sk a partir de pk
- Computacionalmente inviável que um adversário recupere a m conhecendo pk e c
- Impossível forjar uma assinatura conhecendo a mensagem e pk

# Sumário

- Definições básicas de criptografia de chave pública
- **Criptossistema RSA**
- Corretude, Eficiência e Segurança
- RSA na prática

# O criptossistema RSA

- **1976**: Diffie e Hellman desafiaram a comunidade científica a criar um algoritmo que atendesse aos requisitos da criptografia de chave pública (assimétrica)
- **1977-1978**: Um dos primeiros criptossistemas foi desenvolvido por Rivest, Shamir, e Adleman, o RSA.
- **Hoje**: o RSA é a abordagem de propósito geral mais amplamente aceita e implementada para criptografia de chave pública


Imagem: http://www.ams.org/notices/200307/comm-turing.pdf

# Overview

- Necessidade de teoria de números
  - Utiliza números primos *p* e *q*
  - Cálculos em $\mathbb{Z}_n$, onde n = pq
  - uma chave é a inversa multiplicativa da outra
    - chave pública = {e,n}, chave privada = {d,n}
  - segurança baseada na dificuldade de fatorar *n*
- Cifragem e decifragem
  - $c = E(m, e) = m^e \bmod n$ (ou seja, c em $\mathbb{Z}_n$)
  - $m = D(c, d) = c^d \bmod n$
- Tamanho das chaves
  - o valor *n* tem entre 1024 e 2048 bits
  - cada primo tem entre 512 e 1024 bits

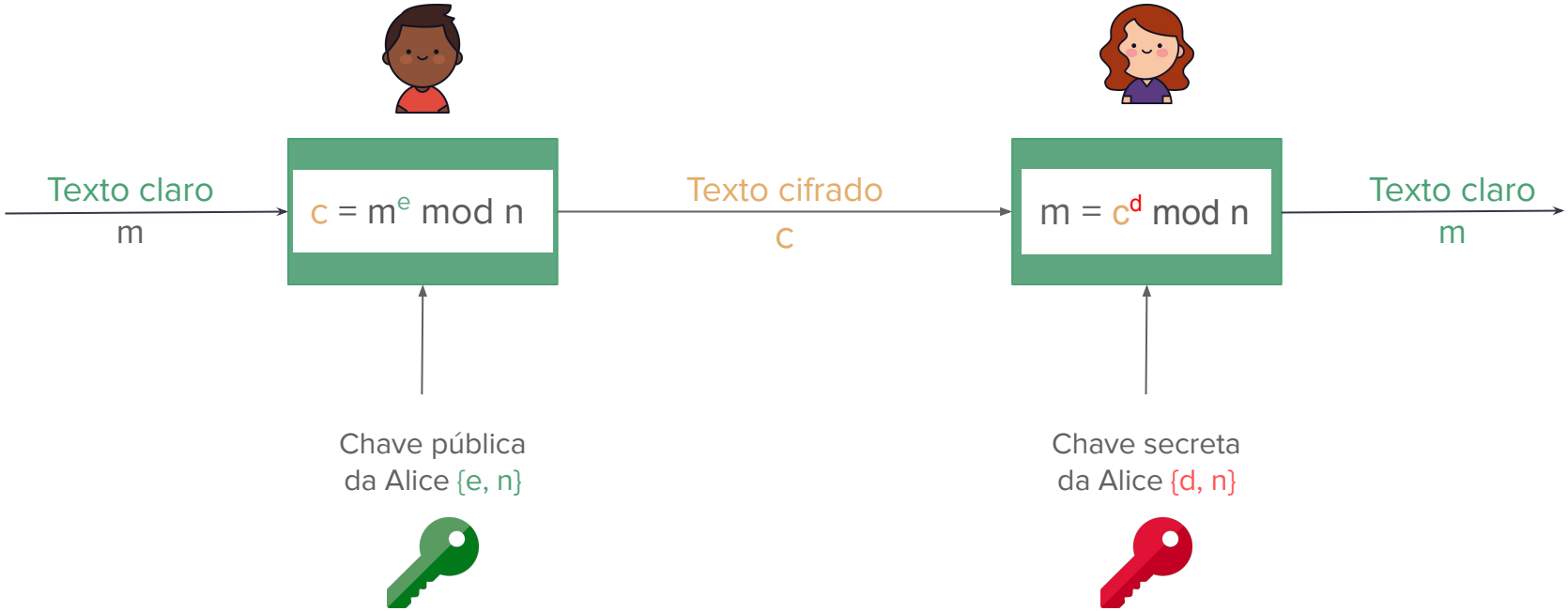
# Geração de chaves

- $p$, $q$ primos (valores secretos e escolhidos)
- $n = p.q$ (valor público e calculado)
- $\phi(n) = \phi(pq) = \phi(p)\phi(q) = (p-1)(q-1)$ (valor secreto, só quem sabe $p$ e $q$ consegue calcular)
- As chaves e e d são inversas multiplicativas módulo $\phi(n)$
  - Escolhemos o e tal que $mdc(\phi(n),e) = 1$ e $1 < e < \phi(n)$ (garantimos que e terá inversa)
  - Calculamos d tal que $ed \equiv 1 \pmod{\phi(n)}$ (ou seja, $ed \bmod \phi(n) = 1$ ou ainda $ed = k\phi(n) + 1$)
  - Ou seja, d é a inversa multiplicativa de e módulo $\phi(n)$ (encontramos com Algoritmo de Euclides Estendido)

LabSEC
Laboratório de Segurança em Computação

cpqd

# Cifragem e Decifragem



# Exemplo

**Geração de Chaves**

- $p = 17$ e $q = 11$
- $n = p \times q = 17 \times 11 = 187$
- $\phi(n) = (p - 1)(q - 1) = 16 \times 10 = 160$
- Escolhemos $e = 7$
  - $mdc(160, 7) = 1$
  - $1 < 7 < 160$
- Calculamos $d = 23$ (euclides estendido)
  - $ed = 161 \equiv 1 \pmod{160}$
- Chave Pública = {7, 187}
- Chave Privada = {23, 187}

- **Cifragem**
  - Texto Claro = 88
  - $c = m^e = 88^7 \bmod 187 = 11$
- **Decifragem**
  - $c^d \bmod 187 = 11^{23} \bmod 187 = 88$

LabSEC
Laboratório de Segurança em Computação

cpqd

# Sumário

- Definições básicas de criptografia de chave pública
- Criptossistema RSA
- **Corretude, Eficiência e Segurança**
- RSA na prática

# Requisitos

1. É possível encontrar valores para e, d, n tal que $m^{ed} \bmod n = m$ para todo $m<n$.
2. É relativamente fácil calcular $m^e \bmod n$ e $c^d \bmod n$ para para todo $m<n$.
3. É inviável determinar d dados e e n.

# Corretude (requisito 1)

- Utilizando o teorema de Euler e inversas multiplicativas.
- **Lembrete:**
  - **Cifragem:** $c = m^e \bmod n$
  - **Decifragem:** $m = c^d \bmod n$
  - note que $c^d = (m^e)^d$
- **Detalhes matemáticos:**
  - como $e.d \bmod \phi(n) = 1$, podemos escrever como $e.d = k\phi(n) + 1$ **(inversa multiplicativa módulo ϕ(n))**
  - então, $c^d = (m^e)^d = m^{k\phi(n)+1} = (m^{\phi(n)})^k \times m$
  - Pelo **teorema de Euler**, temos $m^{\phi(n)} \equiv 1 \pmod n$
  - então, $c^d \equiv (m^{\phi(n)})^k \times m \equiv m \pmod n$
  - portanto, a decifragem $m^{ed} \bmod n$ é capaz de corretamente obter a mensagem original para qualquer $m < n$.

# Aspectos computacionais (requisito 2)

- Cifragem e Decifragem utilizam exponenciação modular
- **Opção 1**: exponenciação feita nos inteiros e depois aplicar o módulo
  - valores intermediários **muito** grandes
  - **Exemplo**: $x^{11} = x \times x \times x \times x \times x \times x \times x \times x \times x \times x \times x$
- **Opção 2**: utilizar a propriedade **[(a mod n) × (b mod n)] mod n = (a × b) mod n**
  - valores intermediários reduzidos
  - calcular a exponenciação utilizando *repeated squares*
    - $x^{11} = x \times x^2 \times x^8$ (note que $x^2 = x \times x$, $x^4 = x^2 \times x^2$ e $x^8 = x^4 \times x^4$)
  - calculamos $x$ mod n, $x^2$ mod n, $x^4$ mod n, $x^8$ mod n
  - depois calculamos $[(x \bmod n) \times (x^2 \bmod n) \times (x^8 \bmod n)] \bmod n$



Mais detalhes em : W. Stallings. *Cryptography and network security*. Cap 9.2

# Aspectos computacionais (requisito 2)

- Escolhas de e para eficiência:
  - 65537 = $2^{16}+1$ ou 17 ou 3
  - Valores com apenas dois bits “1”
  - garantem aceleração na multiplicação
- *d* precisa ser grande para evitar força bruta
- Gerar chaves pode ser demorado
  - precisamos do teste de primalidade várias vezes em um número muito grande
  - Ao fixar e, podemos precisar recalcular p,q algumas vezes



Mais detalhes em : W. Stallings. *Cryptography and network security*. Cap 9.2

# Segurança (requisito 3)

- **Força bruta**: tentar todas as possibilidades de chave privada
- **Ataques matemáticos**: diversas técnicas, com objetivo de, por exemplo, fatorar *n* de maneira mais eficiente
- **Timming attacks:** explora variações de tempo na execução de uma operação criptográfica
- **Ataques e hardware**: induzir erros de hardware em um módulo criptográfico para descobrir informações sobre a chave
- **Chosen ciphertext attack:** o atacante consegue acesso à mensagem original correspondente a uma cifra para descobrir informações sobre a chave

A chance de sucesso depende da implementação do algoritmo e da escolha dos parâmetros vulneráveis.

# Atacando o RSA matematicamente

- Existem 3 formas:
  1. Fatorar n e descobrir p e q
     - permite o cálculo de $\phi(n) = (p - 1)(q - 1)$ e a determinação de $d \equiv e^{-1} \pmod{\phi(n)}$
  2. Determinar $\phi(n)$ diretamente, sem a necessidade de p e q.
     - assim como antes, permite a determinar d.
  3. Determinar d diretamente.
- Criptoanalistas tem focado na fatoração de n.
  - que é muito difícil quando p e q são grandes o suficiente.
- RSA Laboratories lançou desafios de fatoração com tamanhos variados de *n*.
- Técnicas envolvem algoritmos elaborados
  - **Ex**: quadratic sieve, number field sieve.

**Table 9.5** Progress in RSA Factorization

| Number of Decimal Digits | Number of Bits | Date Achieved |
|---|---|---|
| 100 | 332 | April 1991 |
| 110 | 365 | April 1992 |
| 120 | 398 | June 1993 |
| 129 | 428 | April 1994 |
| 130 | 431 | April 1996 |
| 140 | 465 | February 1999 |
| 155 | 512 | August 1999 |
| 160 | 530 | April 2003 |
| 174 | 576 | December 2003 |
| 200 | 663 | May 2005 |
| 193 | 640 | November 2005 |
| 232 | 768 | December 2009 |

Imagem: W. Stallings. *Cryptography and network security*. Cap 9.2

# RSA e os computadores quânticos

- O algoritmo de Shor é um algoritmo quântico desenvolvido em 1994, capaz de fatorar números inteiros grandes de forma eficiente
- Com ele, também seria possível calcular o logaritmo discreto eficientemente
- Se um computador quântico suficientemente grande for construído, esses algoritmos podem "quebrar" tanto o DH quando o RSA
- Em um *draft* sobre a transição para criptografia pós-quântica, o NIST recomenda o desuso do RSA de 2048 bits após 2030 e proíbe o uso após 2035.

# Sumário

- Definições básicas de criptografia de chave pública
- Criptossistema RSA
- Corretude, Eficiência e Segurança
- **RSA na prática**

# Aplicações do RSA

- **SSL/TLS**: usa RSA na autenticação da identidade de sites para o estabelecimento de conexões seguras;
- **Assinaturas Digitais**: Utilizadas para validar a autenticidade e integridade de documentos eletrônicos;
- **Distribuição Segura de Chaves**: usado em criptografia híbrida, para proteger a transmissão de chaves simétricas;
- entre muitas outras!

# Considerações importantes

- Criptografia de chave pública não tornou a criptografia simétrica obsoleta
  - por causa do overhead computacional da criptografia de chave pública, a criptografia de chave simétrica muito dificilmente será abandonada
- Criptografia de chave pública não é mais segura que a simétrica
  - segurança depende do tamanho da chave e do esforço computacional necessário para quebrar uma cifra. Um esquema não é superior que o outro
- Distribuição de chaves não se tornou trivial com criptografia de chaves públicas
  - ainda são necessários protocolos que podem envolver uma autoridade central.

# Atividade: cifragem e decifragem RSA com openssl

- Gere a chave privada RSA e salve ela cifrada com o AES (não esqueça a senha!)

```
openssl genrsa -aes256 -out seunome.privada.pem 2048
```

- Visualize a chave gerada:

```
openssl pkey -in seunome.privada.pem -text
```

- Derive a chave pública a partir da chave privada

```
openssl rsa -pubout -in seunome.privada.pem -out seunome.publica.pem
```

- Visualize a chave gerada:

```
openssl pkey -pubin -in seunome.publica.pem -text
```

- **Pergunta**: qual a diferença de informações informadas ao visualizar a chave pública e a chave privada?

# Atividade: cifragem e decifragem RSA com openssl

- Crie um arquivo de texto

```
echo "Esta é uma mensagem confidencial" > msg.txt
```

- Cifre o arquivo de texto com a chave pública

```
openssl pkeyutl -encrypt -inkey thais.publica.pem -pubin -in msg.txt -out secretMsg.bin
```

- Decifre o arquivo com a chave privada

```
openssl pkeyutl -decrypt -inkey thais.privada.pem -in secretMsg.bin -out msgDecifrada.txt
```

- **Pergunta**: Quem pode cifrar mensagens? E quem pode decifrar?

# Referências


- W. Stallings. *Cryptography and network security*. 7a edição.
  - Princípios de criptossistemas de chave pública: 9.1
  - RSA: 9.2
- D. Stinson e M. Paterson. *Cryptography: Theory and Practice*. 4a edição.
  - Introdução a criptossistemas de chave pública: 6.1
  - RSA: 6.3
- NIST Internal Report 8547
- imagens: Flaticon.com