Universidade Regional de Blumenau - FURB
Centro de Ciências Exatas e Naturais - CCEN
Departamento de Sistemas e Computação - DSC

Sistemas Distribuídos

MARCOS RODRIGO MOMO
marcos.rodrigomomo@gmail.com

Blumenau, outubro 2024.

Conceitos Básicos

- Três propriedades da segurança da informação:
  - Confidencialidade;
  - Integridade;
  - Disponibilidade;

Conceitos Básicos

- Confidencialidade:
  - Protege o conteúdo;
  - Apenas lê quem tem direito;
  - Protege por grau de sigilo;

Conceitos Básicos

- Integridade:
  - Modificação durante o trânsito;
  - Informação não pode ser alterada;
  - Informação igual a original;
  - Apenas quem tem direito pode modificar;

Conceitos Básicos

- Disponibilidade:
  - A informação deve estar disponível;
  - Quando quem tem direito deseja acessar;
  - Exceto em situações previstas, como manutenção.

Segurança em SD

- É Um tema muito importante
- Precisa estar presente em todas as partes do sistema
- Uma única falha de projeto em relação à segurança pode inutilizar todas as outras medidas de segurança

Segurança em SD

A política de segurança

- Não faz sentido construir todos os tipos de

mecanismos de segurança em um sistema sem

antes definir a política de segurança

- Na política de segurança descreve a

Segurança em SD

Pode ser dividida em duas partes:

– Uma trata da comunicação entre usuários ou

processos que possivelmente residem em

máquinas diferentes

– O principal mecanismo para garantir a

comunicação segura é o uso de canal seguro

Segurança em SD

A outra parte diz respeito à autorização

– Objetiva assegurar que um processo receba

somente os direitos de acesso a recursos de

um sistema distribuído no qual está habilitado

- Controle de acesso
- Gerência de segurança

Segurança em SD

Tanto na implementação de canais seguros como no controle de acesso requerem mecanismos:

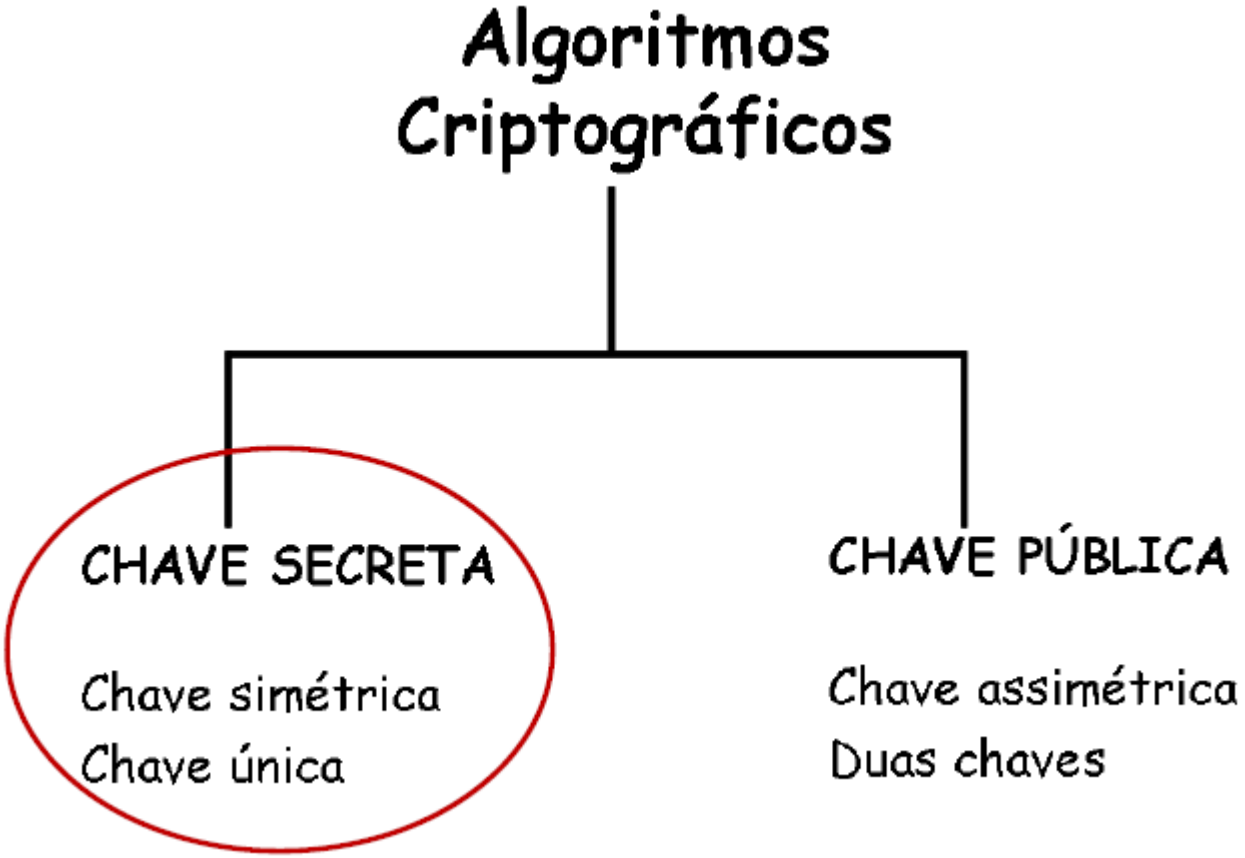
- Para distribuir chaves criptográficas
- Para adicionar e remover usuários
- Gerenciar grupos

Criptografia

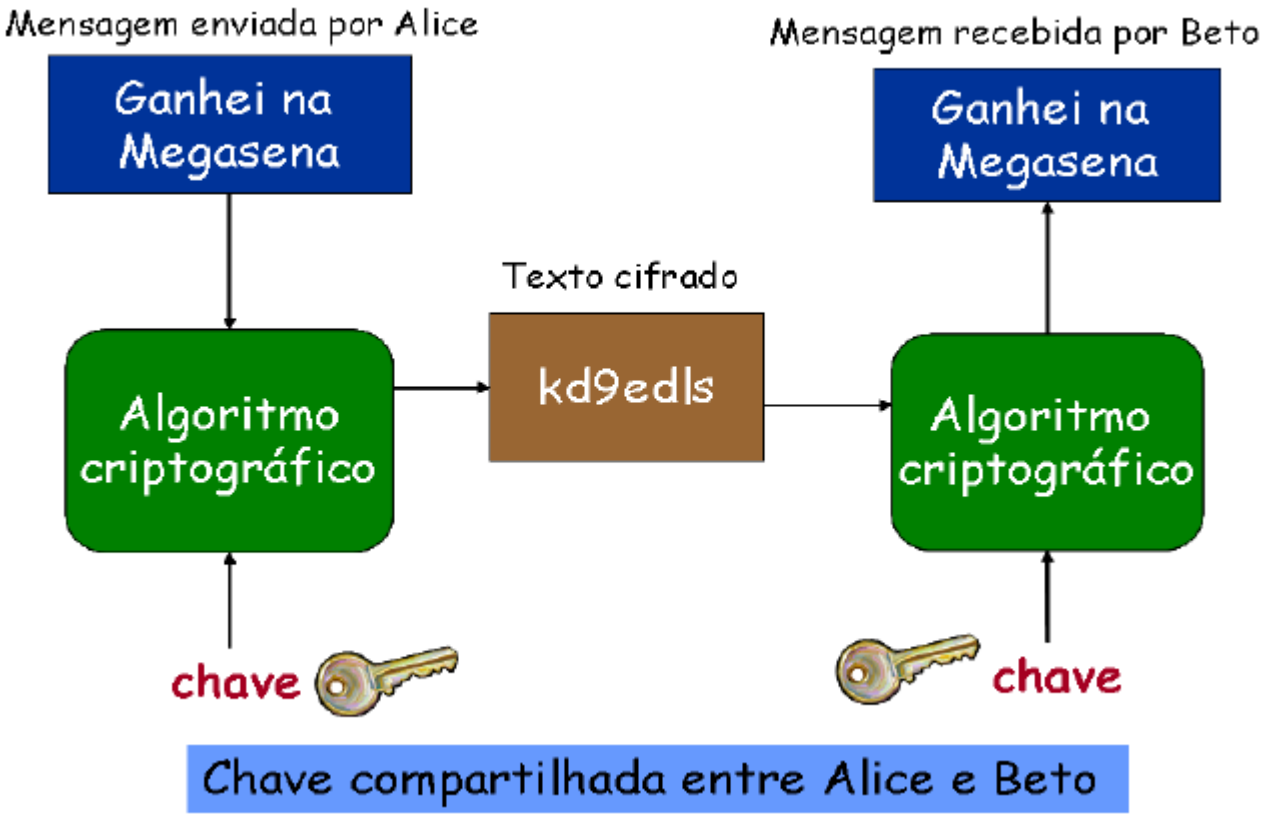
Fundamental para a segurança de computadores

- Transforma dados em algo que um atacante não possa entender
- É a forma de implementar a confidencialidade de dados
- Permite saber se os dados transmitidos foram modificados
- Oferece suporte para verificações de integridade

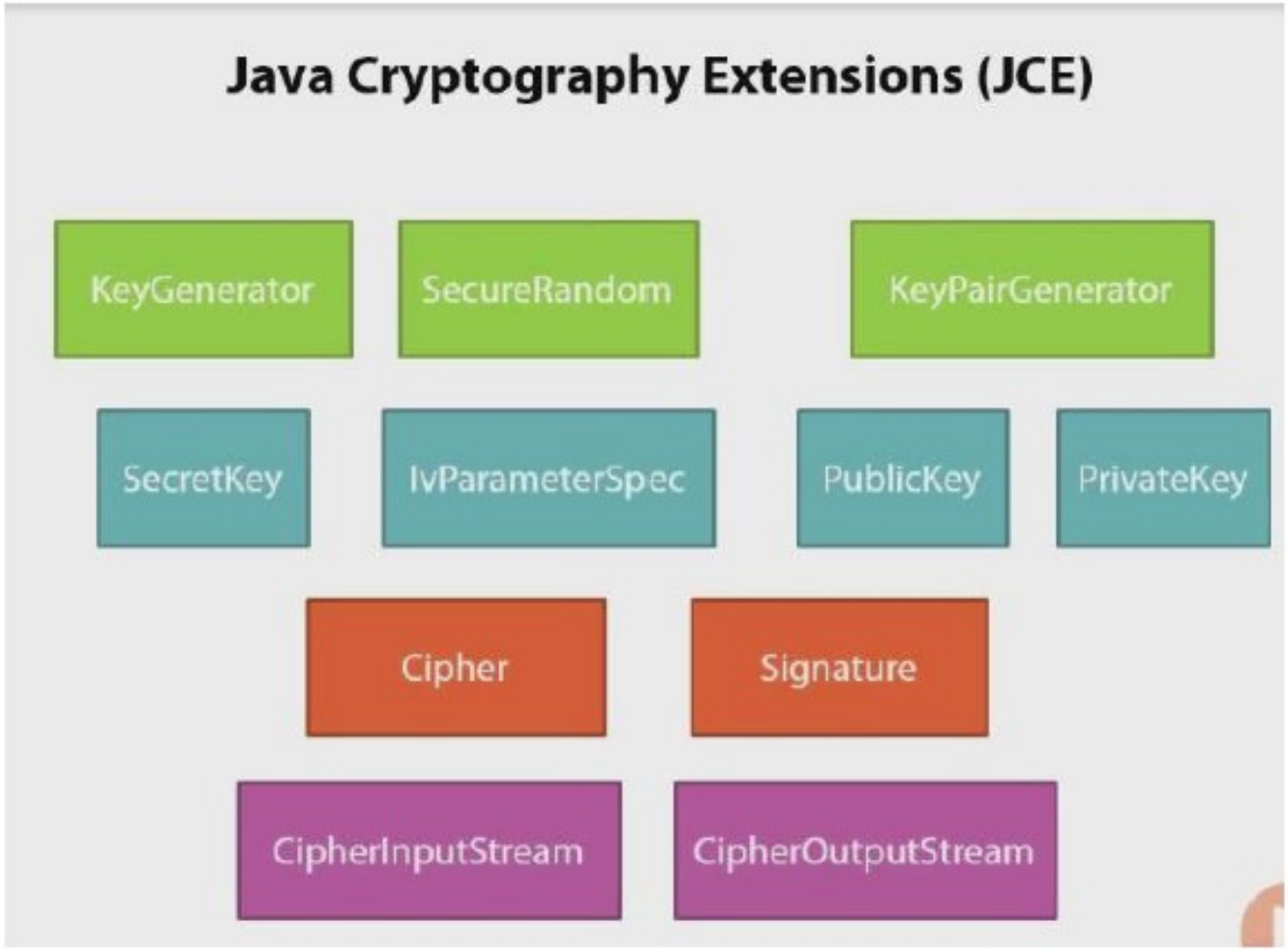
Criptografia



Criptografia



Criptografia



Criptografia simétrica

Algoritmos simétricos a chave de cifragem é igual a chave de decifragem

- A transformação é dada caractere por caractere ou bit a bit.
- A execução é mais rápida se comparada a criptografia de chave assimétrica

Algoritmos de criptografia simétrica

Mais conhecidos: *IDEA*, *TwoFish*, *BlowFish*, *Serpent*, *DES*, *AES*, *RC5*, *RC6*

- Também conhcecidos como criptografia "*de bloco*"
- Nesse tipo de criptografia são processados blocos de informação de uma só vez, concatenando-os no final do processo.

Algoritmos de criptografia simétrica

Outros sistemas criptográficos, por exemplo: o *RC4* e *OTP* que são chamados criptografia "*De fluxo*"

- Nesse tipo de criptografia é processado cada bit da mensagem individualmente (processamento bit a bit).

Esquema básico da criptografia simétrica

1. Criação da chave

```
KeyGenerator keygenerator = KeyGenerator.getInstance("RC4");
SecretKey chaveDES = keygenerator.generateKey();
System.out.println("A chave criada foi" + chaveDES);
```

2. Criação do objeto Cipher e Cifrar

```
Cipher cifraDES;
cifraDES = Cipher.getInstance("RC4");
cifraDES.init(Cipher.ENCRYPT_MODE, chaveDES);
```

Esquema básico da criptografia simétrica

3. Configuração do objeto

```
// Texto plano

    byte[] textoPlano = "Exemplo do texto plano".getBytes();
    System.out.println("Texto [Formato de Byte] : " +
    textoPlano); System.out.println("Texto plano: " + new
    String(textoPlano));


// Texto encriptado

    byte[] textoEncriptado = cifraDES.doFinal(textoPlano);
    System.out.println(“O Texto foi Encriptado : " +
    textoEncriptado);
```

Atividades

Baixar a classe CriptografarDescriptograr.jar

- **OTP, AES e RC4**

1)Estudar o código;

2)Implementar uma aplicação com algoritmo de criptografia simétrica, onde recebe uma mensagem via console e exiba na saída:

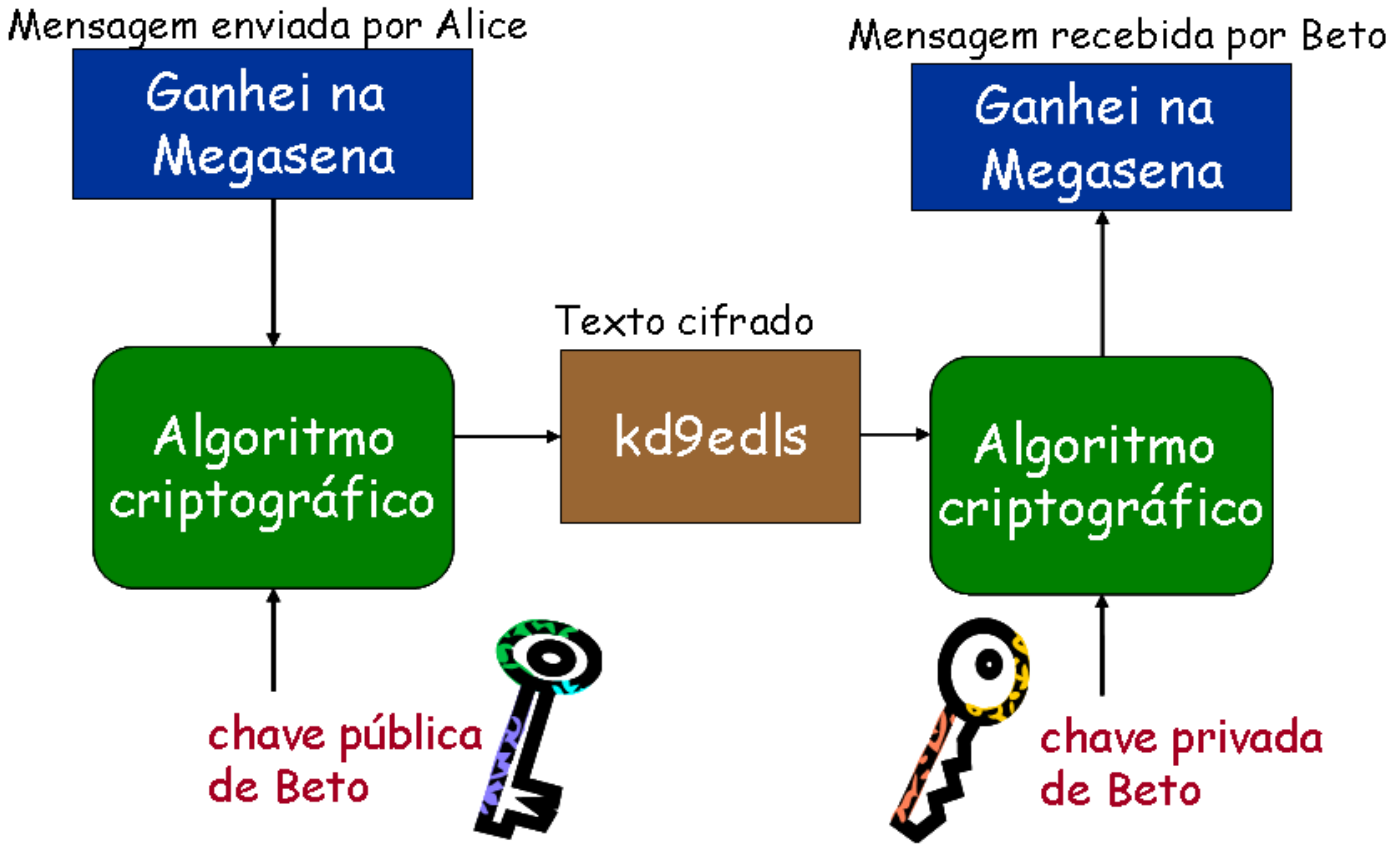
a)A chave criada (observe que a chave é criada randomicamente);

b)O texto plano no formato de bytes;

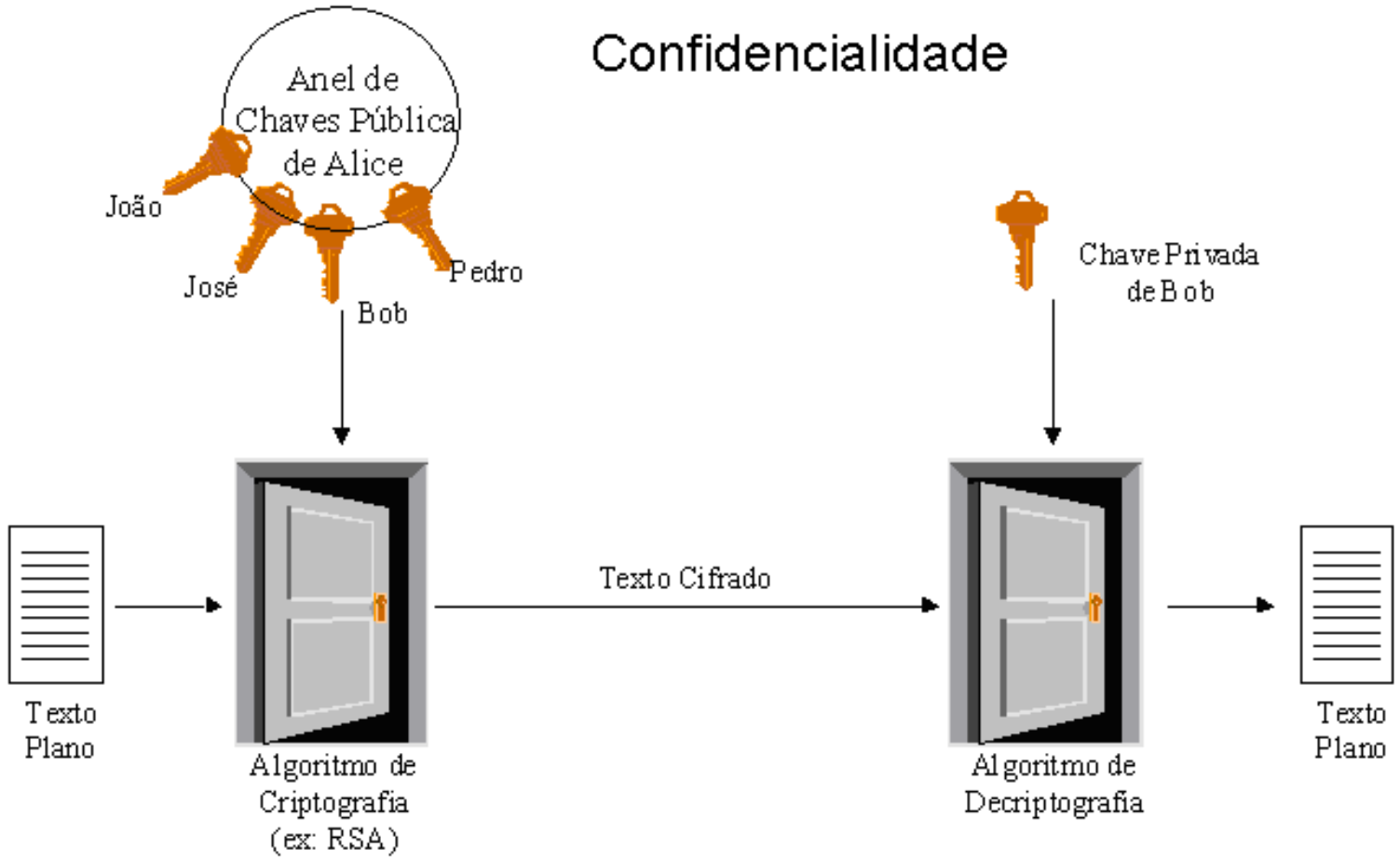
c)O texto cifrado,

d)O texto descifrado

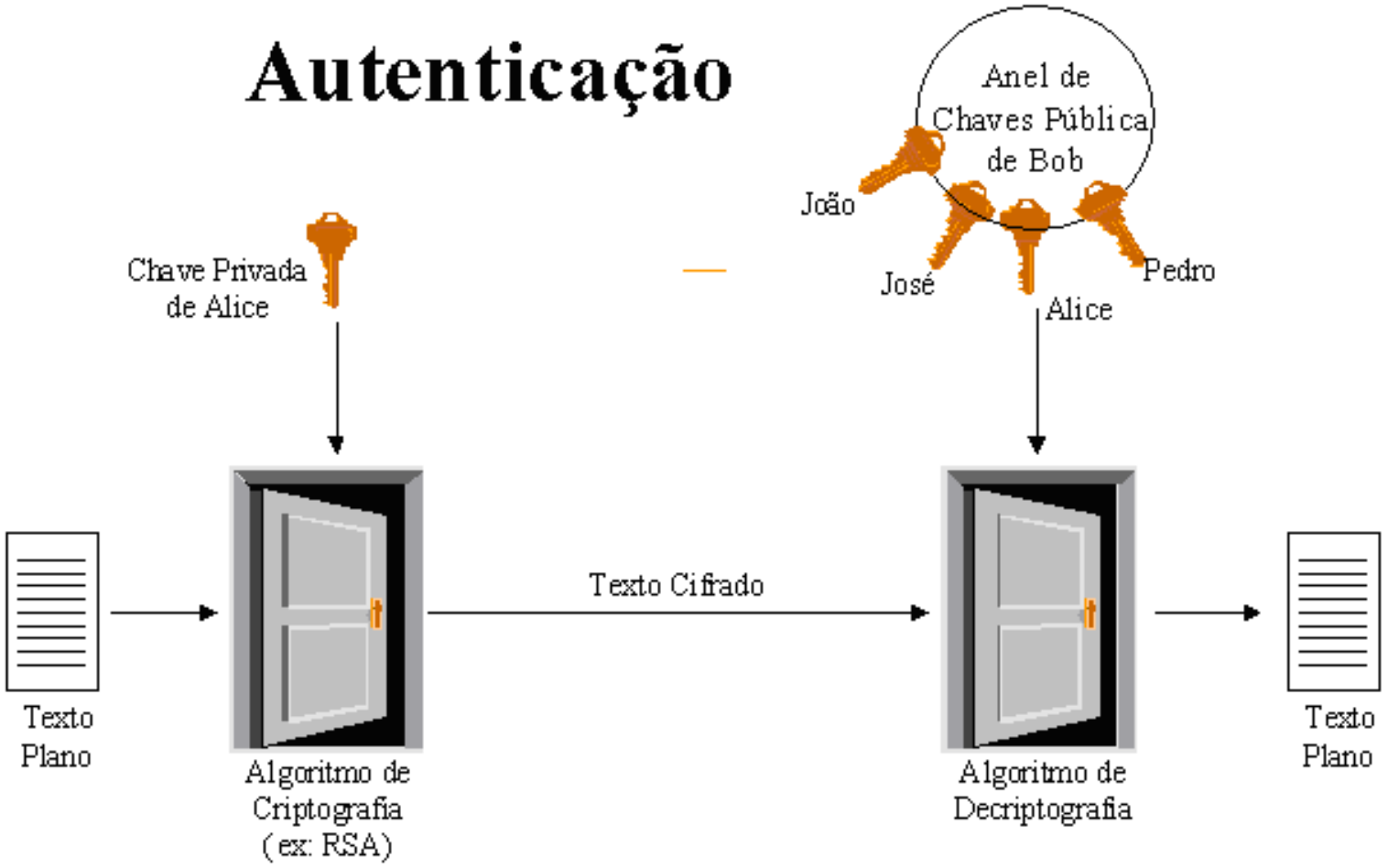
Criptografia Assimétrica



Criptografia Assimétrica



Criptografia Assimétrica



Algoritmos Assimétricos

- Dois algoritmos mais conhecidos:
  - RSA e ElGamal;
- Algoritmos RSA:
- É o mais usado comercialmente;
- Cifra blocos de tamanho variado = n;

Algoritmo RSA

- O par de chaves é derivado de n;
- n é um número muito grande;
- n é resultado de dois números primos muito grandes = p & q;
- p & q devem ter mais de 100 dígitos cada um;

Algoritmo RSA

- Um invasor pode conhecer a chave pública e o número n;
- Mas não conhece p & q;
- Logo ele não consegue gerar a chave privada;

Algoritmo RSA

- Escolher dois números primos grandes (> 10^100) *p* e *q*
- Calcular *n = p * q*
- Escolher um número "*e*" : 1<e<ϕ(n) de forma que "e" e ϕ(n) sejam coprimos
- Calcular d

  d= (2*ϕ(n)+1)/e
- Publicar (*n*, *e*) – chave pública
- Manter (*n*, *d*) – chave privada e "*p*" "*q*" em segredo

Algoritmo RSA

- Escolher dois números primos grandes (> 10^100) p e q
- Calcular $n = p * q$
- Escolher um número "e" relativamente primo com $(p - 1) * (q - 1)$
- Calcular d de forma que $e * d = 1 \bmod (p - 1) * (q - 1)$, isto é, d = e-1 mod $(p - 1) * (q - 1)$
- Publicar (n, e) – chave pública, manter (n, d) – chave privada – e p, q em segredo

Algoritmo RSA

- KU = {e, n}
- KR = {d, n}
- Cifrar: M^e mod n
- Decifrar: C^d mod n

- Invasor não consegue descobrir “d” a partir de “e” e “n”

Algoritmo RSA

- p= 7 e q = 17;
- n = 119;
- Totiente de n = 96;
- e relativamente primo a 96 = 5;
- d = 77;
- KU = {5, 119}
- KR = {77,119}

Algoritmo RSA

- KU = {5, 119}
- KR = {77, 119}
- M = 19
- Cifrar: 19^5 mod 119 = 66
- C = 66
- Decifrar: 66^77 mod 119 = 19
- Obs: na prática a chave é bem maior, mais de 130 dígitos;

Testar exemplo...

- Exemplo Criptografia com o algoritmo RSA

Hash e algoritmos

- Funções hash, ou *message digests* ou funções one-way;
- Recebe uma mensagem de tamanho variável como entrada e na saída, produz um resumo de tamanho fixo
- Essa saída é chamada de *Message Digest,* totalmente incompreensível
- Função hash: y = f(x);
- y é facilmente calculado;
- x é computacionalmente complexo;

Hash e algoritmos

- Propriedades:
  - A partir do resumo não deve ser possível encontrar-se a entrada;
  - Não deve ser possível encontrar uma entrada que gere um resumo específico;
  - Computacionalmente, deve ser inviável encontrar duas mensagens que tenha o mesmo resumo (colisão)
  - O mapeamento de um *hash* deve ser totalmente aleatório
  - A alteração de um bit que seja na mensagem original deve produzir um *hash distinto (efeito avalanche)*

Hash e algoritmos

- Usos das funções *hash*
  - Autenticação
  - Integridade de mensagens
  - Assinaturas de mensagens

- Detecção de dados duplicados
- Verificação de corrupção de arquivos
- Assinatura única de arquivo

Hash e algoritmos

- Alguns algoritmos são: MD5, SHA-1, SHA-2 ou SHA-256;

- SHA é o padrão do NIST;
- SHA-224, 256, 384 e 512;

Hash MD5

- MD5 – *Message Digest 5*
- Algoritmo bastante usado
- Produz um valor de *Hash* de 32 caracteres
- São hexadecimais de 128 bits
- Código fonte está na RFC 1321
- Exemplo de função *Hash* gerada:
  - C40ab71cc70e95b83d4d52a09e210a7f

Vulnerabilidades do MD5

- Ataques de colisão
  - Tipo de ataque onde se tenta encontrar duas entradas que produz o mesmo valor de *hash* de saída
- Para armazenar senha
  - Uso de *Rainbow Tables*
    - Reverter um *hash* e descobrir a string que o originou

Atividade

- Pesquisar na internet por ferramentas de funções *Hash*:
  - Gerar *Hash* MD5
  - Comprove o efeito avalanche
- Através de ferramentas para computar e comprovar *hashes* criptográficos
  - Verifique a integridade de um arquivo transferido

Testar exemplo...

- Exemplo Criptografia com Hash utilizando o algoritmo SHA-1