Criptografia de chave pública

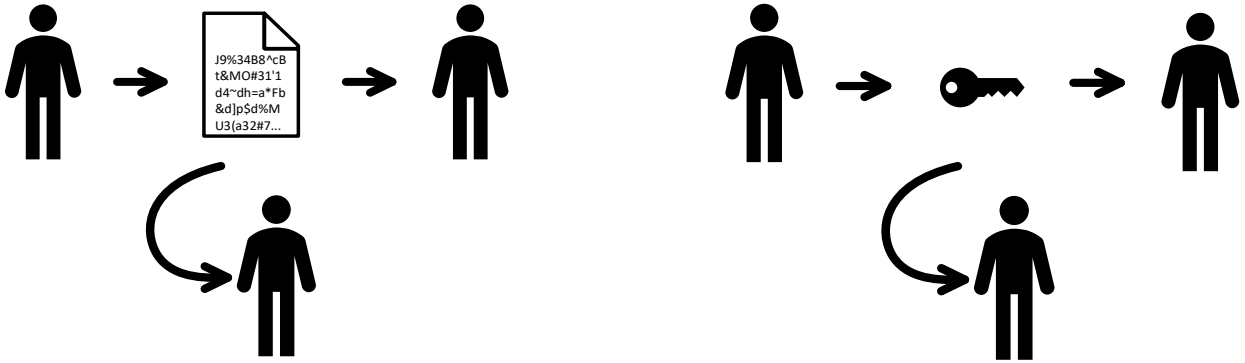
FURB
UNIVERSIDADE DE BLUMENAU

Bibliografia

- RSA (SISTEMA CRIPTOGRÁFICO). In: WIKIPÉDIA, a enciclopédia livre. Flórida: Wikimedia Foundation, 2019. Disponível em: <https://pt.wikipedia.org/w/index.php?title=RSA_(sistema_criptogr%C3%A1fico)&oldid=54535931>. Acesso em: 18 mar. 2019.

Motivação

- Para utilizar a criptografia de chave simétrica é preciso compartilhar a chave com o destinatário



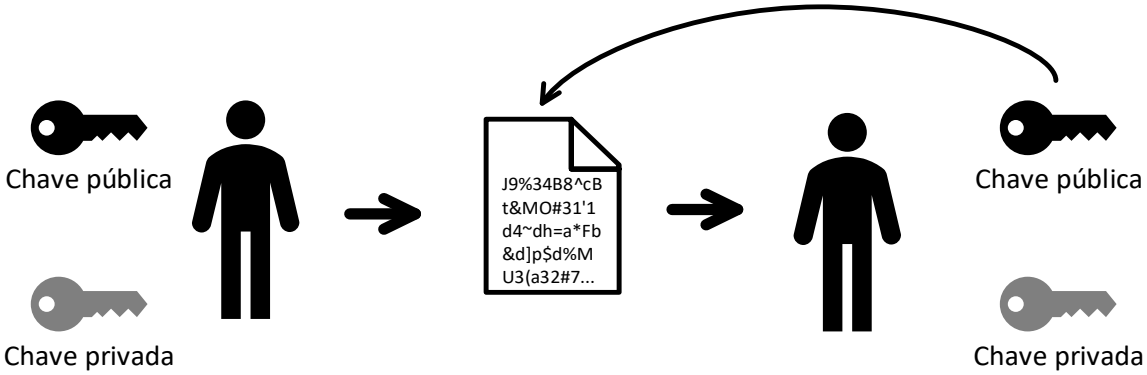
- Se o agente pode interceptar a mensagem, ele também pode interceptar a chave
- Esta dificuldade é conhecida como **problema de distribuição de chaves**:

  Como duas ou mais pessoas podem, de forma segura, enviar as chaves por meio de rotas inseguras?

Criptografia de chave assimétrica

- Na década de 1970 foi criada a criptografia de chave assimétrica
- Trata-se de um esquema que utiliza duas chaves que estão matematicamente relacionadas
  - São chaves “parceiras”: uma é usada para encriptar e a outra para decriptar
  - A chave que é usada para encriptar não é utilizada para decriptar, apenas a chave parceira correspondente pode decriptar
- Uma das chaves pode se tornar pública, enquanto que a outra deve ser privada
  - A criptografia de chave assimétrica também é conhecida como criptografia de chave pública

Criptografia de chave assimétrica



- Para criptografar dados, o emissor utiliza a chave pública do destinatário
- Somente quem possui a chave privada relacionada à pública pode decriptar os dados
  - Somente a chave privada deve ser mantida em segredo

Segurança da chave assimétrica

- Um algoritmo de chave assimétrica pode ser quebrado?

- Um algoritmo assimétrico pode ser quebrado determinando qual é sua chave privada
  - Cálculos matemáticos pode ser feitos para derivar a chave privada a partir da pública
  - O tempo está relacionado ao tamanho da chave. “Teoricamente é possível, mas extremamente difícil gerar a chave privada a partir da pública”

- Não existe algoritmo de chave assimétrica que não tenha fraquezas

Notas históricas

- O esquema de chave assimétrica foi proposto por Whitfield Diffie em 1976
  - A intenção era estabelecer comunicação segura por meio de linhas públicas.
  - O nome do esquema é conhecido como *Troca de chaves Diffie-Hellman*

- Em 1978, Ron Rivest, Adi Shamire e Len Adleman desenvolveram um algoritmo para a proposta de Whitfield, tornando-o conhecido como RSA.

RSA

- O algoritmo de chave pública RSA utiliza chaves de 1024 à 4096 bits.
  - Com 2.048 bits, é possível representar um número com até 617 dígitos.

- Existem três operações que são realizadas para trabalhar com o algoritmo RSA:
  - Gerar o par de chaves
  - Cifrar
  - Decifrar

O princípio do algoritmo RSA

- A segurança do algoritmo RSA é baseada no fato de que é fácil calcular o produto (n) de dois números primos grandes *(p* e *q),* entretanto, é muito difícil determinar, a partir de *n* os dois números primos *p* e *q* que produziram *n.*

O algoritmos RSA

Geração de chaves

1. **Escolher dois números primos randômicos: p e q.**

   Os números devem ser diferentes e grandes – Uso do *Teste de Fermat*

2. **Calcular $n = pq$**

   n é chamado de "módulo" da chave pública e da chave privada

3. **Calcular a função totiente: $\phi(n)$**

   Esta função tem como objetivo calcular a quantidade de coprimos que n possui (quantidade de números inferiores à n que são coprimos de n).

   Dois números são coprimos quando o máximo divisor comum (mdc) entre eles é 1, ou seja, quando o único divisor comum entre eles é 1.

   Exemplo: os números 20 e 21 são coprimos pois:

   - Os divisores de 20 são: **1**, 2, 4, 5, 10 e 20
   - Os divisores de 21 são: **1**, 3, 7 e 21.

O algoritmos RSA

Geração de chaves

4. **Escolher um número *e*** tal que:

$$\begin{cases} 1 < e < \phi(n) \\ mdc\big(e, \phi(n)\big) = 1 \end{cases}$$

e é chamado de “expoente da chave pública”

Nota: Frequentemente utiliza-se como e o valor 65537, por questões de performance para dispositivos pequenos.

5. **Calcular**: $d = \frac{1 + k\phi(n)}{e}$

d é chamado de “expoente da chave privada”

O algoritmo RSA

- A chave pública é composta de:
  - n (chamado de módulo)
  - o expoente e
- A chave privada é composta de:
  - n (módulo)
  - O expoente d

Chave pública

$n =$

```
c6 2c 3f 8c fe c2 95 d1 d9 11 55 ae 94 62 1d b4
f3 0d f2 22 ea a1 62 01 13 22 95 89 3c 0f 89 9f
5e f3 01 2c e8 45 3f d9 2f 99 90 37 4e fa 35 89
0b cf e4 83 cf 9e f7 28 92 a8 89 2b 0b 0b e8 f1
ec 00 f1 e9 30 6f ae 32 16 29 0c 64 71 48 b9 f6
d7 e5 73 db b0 4b be ab d8 a3 83 3f 34 1e 0d 03
d0 70 51 f1 40 df 11 f3 6c 29 6e 7d 5a a6 dc b1
c8 d8 13 1f 57 14 a0 ff 4e d7 de a9 ef 4a 9c b7
```

$e =$ 03

Chave privada

n = igual a da chave pública

$d =$

```
84 1d 7f b3 54 81 b9 36 90 b6 39 1f 0d 96 be 78
a2 09 4c 17 47 16 41 56 0c c1 b9 06 28 0a 5b bf
94 a2 00 c8 9a d8 d5 3b 75 11 0a cf 89 fc 23 b0
b2 8a 98 57 df bf 4f 70 61 c5 b0 c7 5c b2 9b 4a
c5 56 70 ff 91 e0 c9 e2 67 25 4e f7 d0 a5 f8 73
f5 ec 07 83 73 24 06 76 ed d8 1e e7 d2 f3 6c 3b
af 1c 0b 3e ba 33 e3 34 08 24 f3 b9 51 20 68 0d
ee a4 e3 e7 42 71 90 a6 20 5e 2e dc 2b 4c c0 db
```

Cifragem e decifragem

- Considerar que:
  - m – texto simples
  - c – texto cifrado
  - e – expoente da chave pública
  - d – expoente da chave privada
  - n – módulo

- Para cifrar (usar o algoritmo de *Euclides estendido*):
  - $m^{\mathrm{e}} \equiv c \bmod n$

- Para decifrar:
  - $c^{\mathrm{d}} \equiv m \bmod n$

Preenchimento

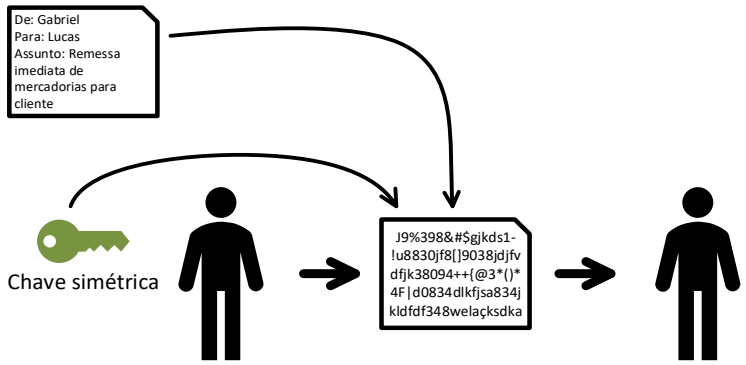
- O algoritmo RSA também utiliza esquema de preenchimento.
- O esquema típico é o definido por PKCS#1
  - Completa-se o bloco com o tamanho de bytes de dados reais e
  - Com bytes aleatórios

Criptografia de chave assimétrica

- Infelizmente os algoritmos de chave assimétrica são lentos
  - de 100 à 2500 vezes mais lentos que os algoritmos de chave simétrica

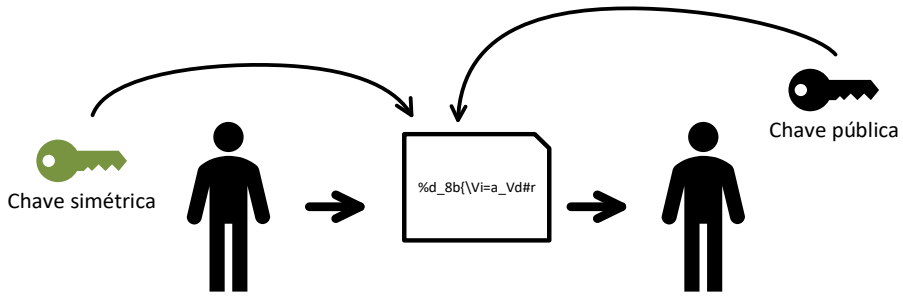
❶

Criptografa o documento com algoritmo simétrico



❷

Criptografa a chave simétrica usando criptografia assimétrica



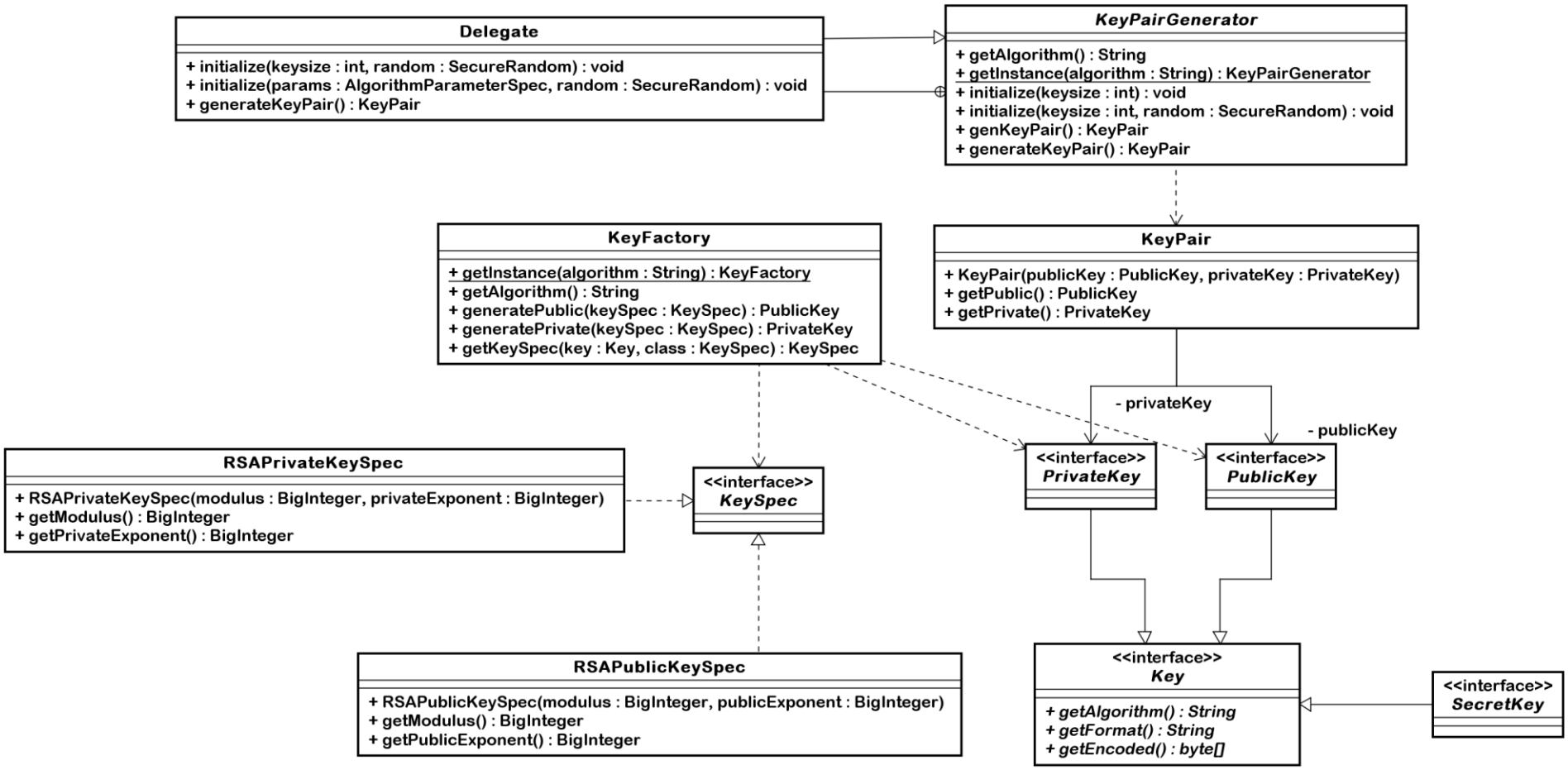
Criptografia de chave assimétrica

- Este processo, usado para encriptar dados em grande quantidade utilizando a criptografia de chave simétrica e para encriptar a chave simétrica com um algoritmo de chave assimétrica, é chamado de *envelope digital*
  - Portanto, com o envelope digital, o texto simples é uma chave.
  - Se um agente interceptar a mensagem que contém o documento (dados), precisará da chave simétrica
  - Se um agente interceptar a mensagem que contém a chave simétrica, precisará da chave privada (assimétrica) para decriptar a chave

Cifragem utilizando Java

FURB
UNIVERSIDADE DE BLUMENAU

Diagrama de classes



Gerando um par de chaves

- O fragmento abaixo gera um par de chaves usando o algoritmo RSA:

```
KeyPairGenerator kpg = KeyPairGenerator.getInstance("RSA");
kpg.initialize(1024);

KeyPair kp = kpg.genKeyPair();
System.out.println(kp.getPublic());
System.out.println(kp.getPrivate());
```

- Abaixo obtém-se, a partir de uma chave pública, o módulo (n) e o expoente (e)

```
KeyFactory fact = KeyFactory.getInstance("RSA");
RSAPublicKeySpec pks = fact.getKeySpec(kp.getPublic(), RSAPublicKeySpec.class);
System.out.println("Módulo: " + pks.getModulus());
System.out.println("Expoente: " + pks.getPublicExponent());
```

- Aqui obtém-se o módulo (n) e o expoente (d) da chave privada:

```
fact = KeyFactory.getInstance("RSA");
RSAPrivateKeySpec prks = fact.getKeySpec(kp.getPrivate(), RSAPrivateKeySpec.class);
System.out.println("Módulo: " + prks.getModulus());
System.out.println("Expoente (d): " + prks.getPrivateExponent());
```

Definido uma chave a partir do módulo e expoente

É possível gerar uma chave a partir do módulo e do expoente, como no exemplo abaixo:

```
BigInteger modulo = new
BigInteger("113117210048721295851652740750852492226584745077049794854564088974585921616560497170462543070408231633306125574814823284096570845260758042429994608488417001936775490676237737505266741689227531475899458994771076388889360627044812540854478253930685241391724111250797272589210665090243075086842837453726140103789");
BigInteger expoente = new
BigInteger("132194746748120360243497313183511487856235800012988751207883540207265143912787715005046403920877770737063279640127978322610988617704830225211915209181498337279755083294083910960638550351330131867768169878209519355200471425480978110941720671925471292420359171829858330342431187933884884509929604754838385957 13");
KeyFactory fact = KeyFactory.getInstance("RSA");
PrivateKey key = fact.generatePrivate( new RSAPrivateKeySpec(modulo, expoente) );
```

Cifrando com RSA

```
Cipher c = Cipher.getInstance("RSA");
c.init(Cipher.ENCRYPT_MODE, key);
byte[] textoSimples = "aqui texto simples...".getBytes();
byte[] textoCifrado = c.doFinal(textoSimples);
```