Capítulo 3

Criptografia assimétrica

Este capítulo apresenta uma introdução aos algoritmos de criptografia assimétricos, ou seja, que usam chaves distintas para cifrar e decifrar informações. Este texto não tem a mínima pretensão de ser completo sobre esse vasto tema; leitores em busca de uma abordagem mais profunda e completa devem procurar livros específicos sobre criptografia.

3.1 O acordo de chaves de Diffie-Hellman-Merkle

Um dos principais problemas no uso da criptografia simétrica para a criação de um canal de comunicação segura é a troca de chaves, ou seja, o estabelecimento de um segredo comum entre os interlocutores. Caso eles não estejam fisicamente próximos, criar uma senha secreta comum, ou substituir uma senha comprometida, pode ser um processo complicado e demorado.

O protocolo de troca de chaves de Diffie-Hellman-Merkle (*Diffie-Hellman-Merkle Key Exchange Protocol*) [Schneier, 1996; Stallings, 2011] foi proposto em 1976. Ele permite estabelecer uma chave secreta comum entre duas entidades distantes, mesmo usando uma rede insegura. Um atacante que estiver observando o tráfego de rede não poderá inferir a chave secreta a partir das mensagens em trânsito capturadas. Esse protocolo é baseado em aritmética inteira modular e constitui um exemplo muito interessante e didático dos mecanismos básicos de funcionamento da criptografia assimétrica.

Considere-se um sistema com três usuários: Alice e Bob[1] são usuários honestos que desejam se comunicar de forma confidencial; Mallory é uma usuária desonesta, que tem acesso a todas as mensagens trocadas entre Alice e Bob e tenta descobrir seus segredos (ataque de interceptação).

A troca de chaves proposta por Diffie-Hellman-Merkle ocorre conforme os passos do esquema a seguir. Sejam p um número primo e g uma raiz primitiva[2] módulo p:

[1] Textos de criptografia habitualmente usam os nomes Alice, Bob, Carol e Dave para explicar algoritmos e protocolos criptográficos, em substituição às letras A, B, C e D. Outros usuários são frequentes, como Mallory (M), que é uma usuária maliciosa (atacante).

[2] Uma raiz primitiva módulo p é um número inteiro positivo g com certas propriedades específicas em relação a p usando aritmética modular. Mais precisamente, um número g é uma raiz primitiva módulo p se todo número n coprimo de p é congruente a uma potência de g módulo p.

passo	Alice	Mallory	Bob
1	escolhe p e g	$\xrightarrow{(p,g)}$	recebe p e g
2	escolhe a secreto		escolhe b secreto
3	$A = g^a mod\ p$		$B = g^b mod\ p$
4	envia A	$\xrightarrow{A}$	recebe A
5	recebe B	$\xrightarrow{B}$	envia B
6	$k = B^a mod\ p = g^{ba} mod\ p$		$k = A^b mod\ p = g^{ab} mod\ p$
7	$m' = \{m\}_k$	$\xrightarrow{m'}$	$m = \{m'\}_k^{-1}$

Como $g^{ba} mod\ p = g^{ab} mod\ p = k$, após os passos 1–6 do protocolo Alice e Bob possuem uma chave secreta comum k, que pode ser usada para cifrar e decifrar mensagens (passo 7). Durante o estabelecimento da chave secreta k, a usuária Mallory pôde observar as trocas de mensagens entre Alice e Bob e obter as seguintes informações:

- O número primo p
- O número gerador g
- $A = g^a mod\ p$ (aqui chamado *chave pública* de Alice)
- $B = g^b mod\ p$ (aqui chamado *chave pública* de Bob)

Para calcular a chave secreta k, Mallory precisará encontrar a na equação $A = g^a mod\ p$ ou b na equação $B = g^b mod\ p$. Esse cálculo é denominado *problema do logaritmo discreto* e não possui nenhuma solução eficiente conhecida: a solução por força bruta tem complexidade exponencial no tempo, em função do número de dígitos de p; o melhor algoritmo conhecido tem complexidade temporal subexponencial.

Portanto, encontrar a ou b a partir dos dados capturados da rede por Mallory torna-se impraticável se o número primo p for muito grande. Por exemplo, caso seja usado o seguinte número primo de Mersenne[3]:

$$p = 2^{127} - 1 = 170.141.183.460.469.231.731.687.303.715.884.105.727$$

o número de passos necessários para encontrar o logaritmo discreto seria aproximadamente de $\sqrt{p} = 13 \times 10^{18}$, usando o melhor algoritmo conhecido. Um computador que calcule um bilhão (10^9) de tentativas por segundo levaria 413 anos para testar todas as possibilidades!

Apesar de ser robusto em relação ao segredo da chave, o protocolo de Diffie-Hellman-Merkle é suscetível a ataques do tipo *man-in-the-middle* (ataque de modificação).

[3]Um *número primo de Mersenne* é um número primo de forma $N_m = 2^m - 1$ com $m \geq 1$. Esta família de números primos tem propriedades interessantes para a construção de algoritmos de criptografia e geradores de números aleatórios.

Se Mallory puder modificar as mensagens em trânsito, substituindo os valores de p, g, A e B por valores que ela escolher, ela poderá estabelecer uma chave secreta $Alice \rightleftharpoons Mallory$ e outra chave secreta $Mallory \rightleftharpoons Bob$, sem que Alice e Bob percebam. Há versões modificadas do protocolo que resolvem este problema [Stamp, 2011].

3.2 Criptografia assimétrica

O protocolo de acordo de chaves de Diffie-Hellmann (Seção 3.1) revolucionou a criptografia em 1976, ao criar a família de **criptossistemas assimétricos**. Os algoritmos assimétricos se caracterizam pelo uso de um par de chaves complementares: uma **chave pública** kp e uma **chave privada** kv. Uma informação cifrada com uma determinada chave pública só poderá ser decifrada através da chave privada correspondente, e vice-versa[4]. A Figura 3.1 ilustra o funcionamento básico da criptografia assimétrica.

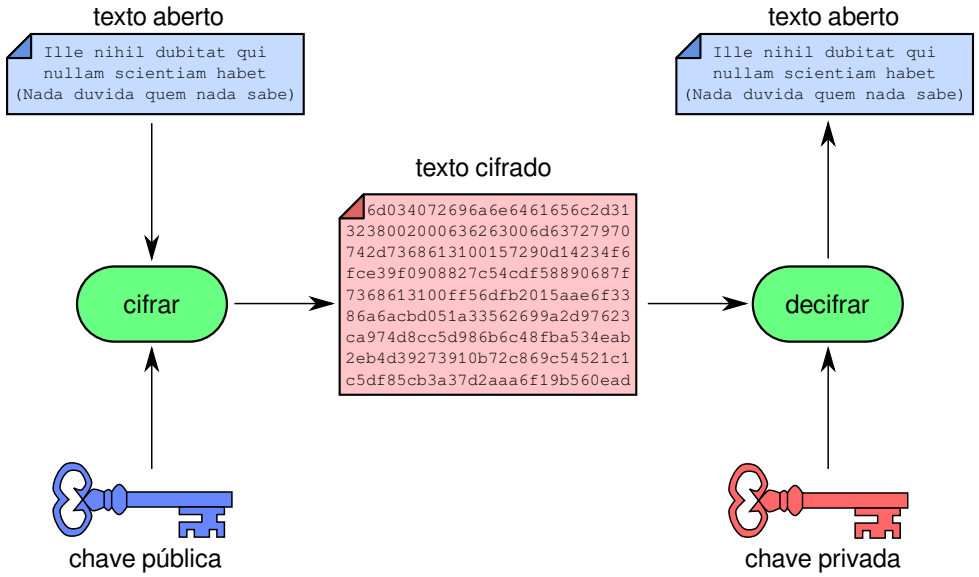

Figura 3.1: Criptografia assimétrica.

Considerando uma chave pública kp e sua chave privada correspondente kv, temos:

$$\{\{x\}_{kp}\}_k^{-1} = x \iff k = kv$$
$$\{\{x\}_{kv}\}_k^{-1} = x \iff k = kp$$

ou seja

$$x \xrightarrow{kp} x' \xrightarrow{kv} x \quad \text{e} \quad x \xrightarrow{kp} x' \xrightarrow{k \neq kv} y \neq x$$
$$x \xrightarrow{kv} x' \xrightarrow{kp} x \quad \text{e} \quad x \xrightarrow{kv} x' \xrightarrow{k \neq kp} y \neq x$$

[4]Como bem observado pelo colega Diego Aranha (Unicamp), nem todos os algoritmos assimétricos têm chaves reversíveis, ou seja, o vice-versa não é aplicável a todos os algoritmos assimétricos.

Essas equações deixam claro que as chaves pública e privada estão fortemente relacionadas: para cada chave pública há uma única chave privada correspondente, e vice-versa. Como o próprio nome diz, geralmente as chaves públicas são amplamente conhecidas e divulgadas (por exemplo, em uma página Web ou um repositório de chaves públicas), enquanto as chaves privadas correspondentes são mantidas em segredo por seus proprietários. Por razões óbvias, não é possível calcular a chave privada a partir de sua chave pública.

Além do algoritmo de *Diffie-Hellman*, apresentado na Seção 3.1, outros criptossistemas assimétricos famosos são o RSA (*Rivest-Shamir-Adleman*), que é baseado na fatoração do produto de número primos, e o ElGamal, baseado no cálculo de logaritmos discretos [Stamp, 2011].

Um exemplo prático de uso da criptografia assimétrica é mostrado na Figura 3.2. Nele, a usuária Alice deseja enviar um documento cifrado ao usuário Bob. Para tal, Alice busca a chave pública de Bob previamente divulgada em um chaveiro público (que pode ser um servidor Web, por exemplo) e a usa para cifrar o documento que será enviado a Bob. Somente Bob poderá decifrar esse documento, pois só ele possui a chave privada correspondente à chave pública usada para cifrá-lo. Outros usuários poderão até ter acesso ao documento cifrado, mas não conseguirão decifrá-lo.

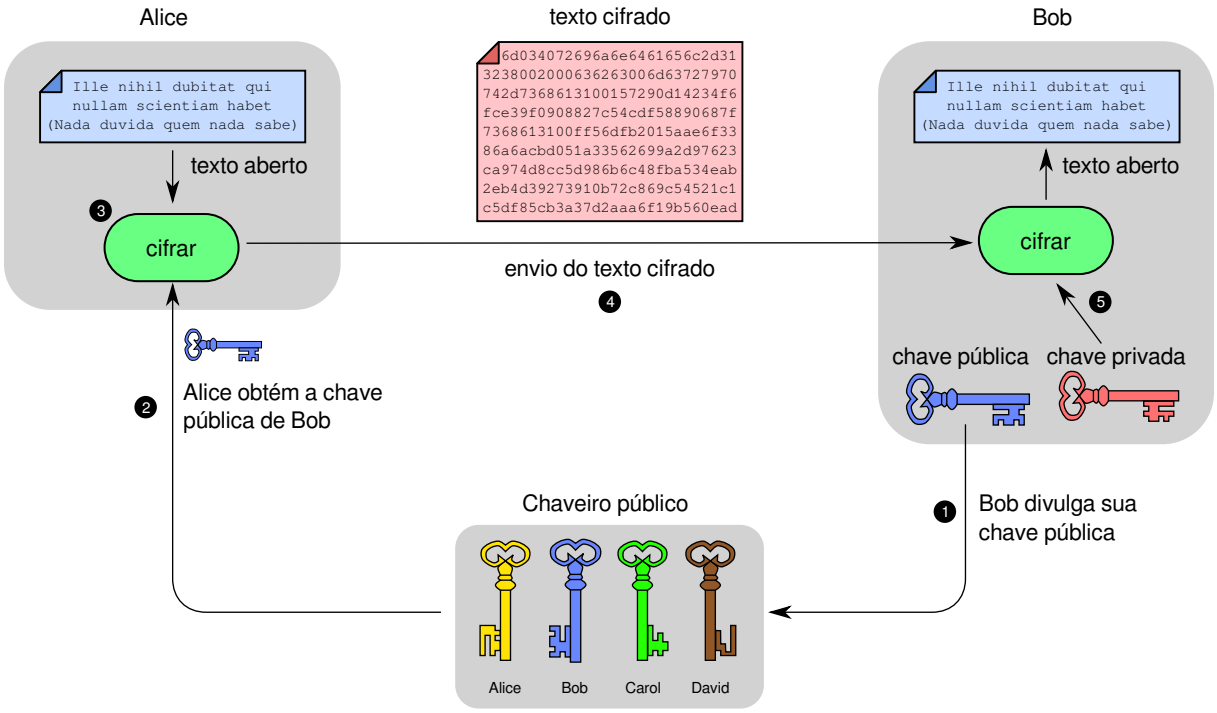

Figura 3.2: Exemplo de uso da criptografia assimétrica.

A criptografia assimétrica também pode ser usada para identificar a autoria de um documento. Por exemplo, se Alice criar um documento e cifrá-lo com sua chave privada, qualquer usuário que tiver acesso ao documento poderá decifrá-lo e lê-lo, pois a chave pública de Alice está publicamente acessível. Todavia, o fato do documento poder ser decifrado usando a chave pública de Alice significa que ela é a autora legítima do mesmo, pois só ela teria acesso à chave privada que foi usada para cifrá-lo. Esse mecanismo é usado na criação das *assinaturas digitais* (Seção 4.2).

A Tabela 3.1 traz uma análise comparativa das principais características dos cifradores simétricos e assimétricos.

Cifrador	**Simétrico**	**Assimétrico**
Característica das chaves	Uma única chave para cifrar e decifrar.	Chaves complementares para cifrar e decifrar.
Tamanho das chaves	Pequena (AES: 64 a 256 bits).	Grande (RSA: 2.048 a 15.360 bits).
Tamanho dos dados	Qualquer (podem ser tratados em blocos ou em fluxo).	No máximo o tamanho da chave, menos alguns bytes de *padding*.
Velocidade	Alta (centenas de MBytes/s em um PC típico).	Baixa (centenas de KBytes/s em um PC típico).
Uso	Cifragem de grandes quantidades de dados (tráfego de rede, arquivos, áudio, etc).	Cifragem de pequenas quantidades de dados (troca de chaves, assinaturas digitais).
Exemplos	RC4, A/51, DES, 3DES, AES.	Diffie-Hellman, RSA, ElGamal, ECC (Curvas Elípticas).

Tabela 3.1: Quadro comparativo de famílias de cifradores.

3.3 Criptografia híbrida

Embora sejam mais versáteis, os algoritmos assimétricos costumam exigir muito mais processamento que os algoritmos simétricos equivalentes. Além disso, eles necessitam de chaves bem maiores que os algoritmos simétricos e geralmente só podem cifrar informações pequenas (menores que o tamanho da chave). Por isso, muitas vezes os algoritmos assimétricos são usados em associação com os simétricos. Por exemplo, os protocolos de rede seguros baseados em TLS (*Transport Layer Security*), como o SSH e HTTPS, usam criptografia assimétrica somente durante o início de cada conexão, para definir uma chave simétrica comum entre os dois computadores que se comunicam. Essa chave simétrica, chamada *chave de sessão*, é então usada para cifrar/decifrar os dados trocados entre os dois computadores durante aquela conexão, sendo descartada quando a sessão encerra.

O esquema a seguir ilustra um exemplo de criptografia híbrida: a criptografia assimétrica é usada para definir uma chave de sessão comum entre dois usuários. Em seguida, essa chave de sessão é usada para cifrar e decifrar as mensagens trocadas entre eles, usando criptografia simétrica.

Passo	Alice	canal	Bob	significado
1	$k = random()$			sorteia uma chave secreta k
2	$k' = \{k\}_{kp(Bob)}$			cifra a chave k usando $kp(Bob)$
3	$k'' = \{k'\}_{kv(Alice)}$			cifra k' usando $kv(Alice)$
4	$k'' \longrightarrow \cdots$	k''	$\cdots \longrightarrow k''$	envia a chave cifrada k''
5			$k' = \{k''\}^{-1}_{kp(Alice)}$	decifra k'' usando $kp(Alice)$
6			$k = \{k'\}^{-1}_{kv(Bob)}$	decifra k' usando $kv(Bob)$, obtém k
7	$m' = \{m\}_k$			cifra mensagem m usando k
8	$m' \longrightarrow \cdots$	m'	$\cdots \longrightarrow m'$	envia mensagem cifrada m'
9			$m = \{m'\}^{-1}_k$	decifra a mensagem m' usando k

Inicialmente, Alice sorteia uma chave secreta simétrica k (passo 1) e a cifra com a chave pública de Bob (passo 2), para que somente ele possa decifrá-la (garante confidencialidade). Em seguida, ela cifra k' com sua chave privada (passo 3), para que Bob tenha certeza de que a chave foi gerada por Alice (garante autenticidade). Em seguida, A chave duplamente cifrada k'' é enviada a Bob (passo 4), que a decifra (passos 5 e 6) e resgata a chave secreta k. Agora, Alice e Bob podem usar a chave de sessão k para trocar mensagens cifradas entre si (passos 7 a 9).

Se Mallory estiver capturando mensagens no canal de comunicação, ela terá acesso somente a k'' e m' (e às chaves públicas de Alice e Bob), o que não a permite descobrir a chave de sessão k nem a mensagem aberta m.

3.4 O algoritmo RSA

O RSA é um algoritmo de criptografia assimétrico proposto em 1977 por Ron Rivest, Adi Shamir e Leonard Adleman, pesquisadores do MIT [Rivest et al., 1978]. O nome RSA provém das iniciais dos autores. RSA é o algoritmo de cifragem assimétrica mais usado atualmente, sendo amplamente empregado na assinatura digital de certificados, arquivos e outros documentos na Internet.

Os passos básicos do algoritmo são a geração do par de chaves (pública/privada), a cifragem e a decifragem, explicados a seguir.

3.4.1 Fundamentos matemáticos

O algoritmo RSA é baseado na dificuldade de fatoração do produto de dois números primos de grandes dimensões [Stamp, 2011]: dados dois números primos p e q de grandes dimensões, é fácil calcular $n = p \times q$, mas é muito difícil calcular os fatores p e q a partir de n.

Assim como diversos outros algoritmos criptográficos, o RSA é baseado em aritmética modular. Para entendê-lo, precisamos de alguns conceitos matemáticos:

A aritmética modular opera sobre *corpos finitos*, ou seja, conjuntos finitos de números inteiros aos quais podem ser aplicadas as operações aritméticas básicas (soma, subtração, multiplicação, divisão). Dado um inteiro positivo p, o conjunto de inteiros módulo p definido como $\mathbb{Z}_p = \{0, 1, 2, \ldots, p-1\}$ é um exemplo de corpo finito, ou corpo de Galois.

Na aritmética modular, operandos e resultados pertencem ao corpo finito. Por exemplo:

$$5_{12} + 9_{12} = 14_{12} = 2_{12}$$

pois 14 mod 12 = 2

Em aritmética modular, uma equação geralmente admite várias soluções:

$$x \bmod 4 = 3 \Rightarrow x \equiv 3, 7, 11, 15, \ldots$$

Inverso multiplicativo:

$$x \cdot x^{-1} = 1$$

3.4.2 Geração das chaves

A primeira etapa do algoritmo RSA é a geração do par de chaves pública/privada. Para tal, são efetuados os seguintes passos:

1. Sortear dois números primos aleatórios p e q. Para melhores resultados, devem ser escolhidos números de magnitude similar. A forma mais usual de escolher p e q é sortear números aleatórios grandes e testar sua primalidade, usando testes como o teste probabilístico de Miller-Rabin (probabilístico) [Rabin, 1980].

2. A partir dos primos p e q, calcular o **módulo** n:

$$n = p \times q$$

3. Da mesma forma, calcular o totiente de Carmichael (*lambda*) de p e q:

$$\lambda(n) = mmc(p-1, q-1)$$

onde $mmc(a, b)$ é o mínimo múltiplo comum entre a e b.

4. De posse dos elementos anteriores, pode-se calcular o expoente e da chave pública:

   - escolher e tal que $1 < e < \lambda(n)$ e $mdc(e, \lambda(n)) = 1$
   - onde $mdc(a, b)$: máximo divisor comum
   - portanto, e e $\lambda(n)$ devem ser **coprimos**, ou seja, eles não têm nenhum fator em comum (exceto o 1)

5. Finalmente, pode-se calcular o expoente d da chave privada:
   - $d \equiv e^{-1} \bmod \lambda(n)$
   - ou seja, $d \cdot e = 1 \bmod \lambda(n)$
   - d é o **inverso multiplicativo** de e, calculado usando o Algoritmo de Euclides estendido
6. As chaves pública k_p e privada k_v são formadas pelo módulo n e seus respectivos expoentes e e d:

$$\begin{aligned} k_p &= \{e, n\} \\ k_v &= \{d, n\} \end{aligned}$$

3.4.3 Cifragem e decifragem

A operação de cifragem ($m \rightarrow c$) usa $\{e, n\}$:

$$c \equiv m^e \bmod n$$

Operação de decifragem ($c \rightarrow m$) usa $\{d, n\}$:

$$m \equiv c^d \equiv (m^e)^d \bmod n$$

No mundo real:

- p e q têm centenas de dígitos
- Operações com inteiros de precisão arbitrária
- exponenciações são **muito** pesadas

3.4.4 Exemplo em pequena escala

1. Escolher dois primos: $p = 61$ e $q = 53$
2. Calcular o módulo: $n = p \cdot q = 61 \cdot 53 = 3233$
3. Calcular o totiente: $\lambda(3233) = mmc(60, 52) = 780$
4. Escolher expoente $1 < e < 780$, e coprimo de 780
   - $e = 17$
   - 17 é primo e não é divisor de 780
5. Calcular expoente d tal que $d \cdot e = 1 \bmod \lambda(n)$
   - $d = 413$
   - $413 \cdot 17 \bmod \lambda(n) = 1$
6. Chave pública $\{n, e\}$: $\{3233, 17\}$

7. Chave privada $\{n, d\}$: $\{3233, 413\}$

8. Usando as chaves:

Mensagem aberta:	$m = 65$ (letra "A" em ASCII)
Cifrando ($m \rightarrow c$):	$c \equiv m^e \bmod n = 65^{17} \bmod 3233 = 2790$
Mensagem cifrada:	$c = 2790$
Decifrando ($c \rightarrow m$):	$m \equiv c^d \bmod n = 2790^{413} \bmod 3233 = 65$
Mensagem decifrada:	$m = 65$

Exercícios

1. Alice precisa enviar a imagem ISO de um CD confidencial a seus amigos Bob, Carol e David. Como o arquivo é muito grande, ela o carregou em um servidor de arquivos acessível remotamente. Contudo, esse servidor pode ser invadido e as comunicações entre eles podem ser capturadas. Como Alice pode cifrar o arquivo ISO de forma que somente Bob, Carol e David possam abri-lo, cada um com sua própria chave?

Referências

M. O. Rabin. Probabilistic algorithm for testing primality. *Journal of Number Theory*, 12 (1):128 – 138, 1980.

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

B. Schneier. *Applied cryptography: protocols, algorithms, and source code in C, 2nd edition*. Wiley, 1996.

W. Stallings. *Cryptography and Network Security – Principles and Practice, 4th edition*. Pearson, 2011.

M. Stamp. *Information Security - Principles and Practice, 2nd edition*. Wiley, 2011.