Capítulo 27

Fundamentos de criptografia

Este capítulo apresenta uma introdução às técnicas clássicas de criptografia frequentemente usadas em sistemas operacionais e redes de computadores. 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.

27.1 Terminologia

O termo "criptografia" provém das palavras gregas *kryptos* (oculto, secreto) e *graphos* (escrever). Assim, a criptografia foi criada para codificar informações, de forma que somente as pessoas autorizadas pudessem ter acesso ao seu conteúdo. Conceitualmente, a criptografia faz parte de um escopo mais amplo de conhecimento:

Criptografia: técnicas para codificar/decodificar informações, ocultando seu conteúdo de pessoas não autorizadas;

Criptanálise: conjunto de técnicas usadas para "quebrar" uma criptografia, expondo a informação ocultada por ela;

Criptologia: área geral, englobando criptografia e criptanálise.

Criptossistema: conjunto de algoritmos/mecanismos para realizar um tipo específico de criptografia.

As técnicas criptográficas são extensivamente usadas na segurança de sistemas, para garantir a confidencialidade e integridade dos dados. Além disso, elas desempenham um papel importante na autenticação de usuários e recursos.

Alguns conceitos fundamentais para estudar as técnicas criptográficas são [Menezes et al., 1996]:

Texto aberto: a mensagem ou informação a codificar (x);

Texto cifrado: a informação codificada de forma a ocultar seu conteúdo (x');

Chave: informação complementar, necessária para cifrar ou decifrar as informações (k);

Cifrar: transformar o texto aberto em texto cifrado ($x \xrightarrow{k} x'$) ;

Decifrar: transformar o texto cifrado em texto aberto ($x' \xrightarrow{k} x$) ;

Cifrador: mecanismo responsável por cifrar/decifrar as informações;

No restante deste texto, a operação de cifragem de um conteúdo aberto x usando uma chave k e gerando um conteúdo cifrado x' será representada por $x' = \{x\}_k$ e a decifragem de um conteúdo x' usando uma chave k será representada por $x = \{x'\}_k^{-1}$.

27.2 Cifradores, chaves e espaço de chaves

Uma das mais antigas técnicas criptográficas conhecidas é o *cifrador de César*, usado pelo imperador romano Júlio César para se comunicar com seus generais. O algoritmo usado nesse cifrador é bem simples: cada caractere do texto aberto é substituído pelo k-ésimo caractere sucessivo no alfabeto. Assim, considerando $k = 2$, a letra "A" seria substituída pela letra "C", a letra "R" pela "T", e assim por diante.

Usando esse algoritmo, a mensagem secreta "Reunir todos os generais para o ataque" seria cifrada da seguinte forma:

mensagem aberta:	`REUNIR TODOS OS GENERAIS PARA O ATAQUE`
mensagem cifrada com $k = 1$:	`SFVOJS UPEPT PT HFOFSBJT QBSB P BUBRVF`
mensagem cifrada com $k = 2$:	`TGWPKT VQFQU QU IGPGTCKU RCTC Q CVCSWG`
mensagem cifrada com $k = 3$:	`UHXQLU WRGRV RV JHQHUDLV SDUD R DWDTXH`

Para decifrar uma mensagem no cifrador de César, é necessário conhecer a mensagem cifrada e o valor de k utilizado para cifrar a mensagem, que é a *chave criptográfica*. Caso essa chave não seja conhecida, ainda é possível tentar "quebrar" a mensagem cifrada testando todas as chaves possíveis, o que é conhecido como análise exaustiva ou "ataque de força bruta". Considerando o cifrador de César e somente letras maiúsculas, a análise exaustiva é trivial, pois há somente 26 valores possíveis para a chave k (as 26 letras do alfabeto).

O número de chaves possíveis em um algoritmo de cifragem é conhecido como o seu **espaço de chaves** (*keyspace*). Em 1883, muito antes dos computadores eletrônicos, o criptólogo francês Auguste Kerckhoffs enunciou um princípio segundo o qual "*o segredo de uma técnica criptográfica não deve residir no algoritmo em si, mas no espaço de chaves que ela provê*". Obedecendo esse princípio, a criptografia moderna se baseia em algoritmos públicos, extensivamente avaliados pela comunidade científica, para os quais o espaço de chaves é extremamente grande, tornando inviável qualquer análise exaustiva, mesmo por computador.

Um bom exemplo de aplicação do princípio de Kerckhoffs é dado pelo algoritmo de criptografia AES (*Advanced Encryption Standard*), adotado como padrão pelo governo americano. Usando chaves de 128 bits, esse algoritmo oferece um espaço de chaves com 2^{128} possibilidades, ou seja, 340.282.366.920.938.463.463.374.607.431.768.211.456 chaves diferentes... Se pudéssemos testar um bilhão (10^9) de chaves por segundo, ainda assim seriam necessários 10 sextilhões de anos para testar todas as chaves possíveis!

27.3 O cifrador de Vernam-Mauborgne

O cifrador de Vernam-Mauborgne foi proposto em 1917 por Gilbert Vernam, engenheiro da ATT, e melhorado mais tarde por Joseph Mauborgne. Neste criptossistema, a cifragem de um texto aberto x com b bits de comprimento consiste em realizar uma operação XOR (OU-exclusivo, $\oplus$) entre os bits do texto aberto e os bits correspondentes de uma chave k de mesmo tamanho:

$$x' = x \oplus k, \text{ ou seja, } \forall i \in [1...b], x'_i = x_i \oplus k_i$$

Como a operação XOR é uma involução (pois $(x \oplus y) \oplus y = x$), a operação de decifragem consiste simplesmente em reaplicar essa mesma operação sobre o texto cifrado, usando a mesma chave:

$$x = x' \oplus k, \text{ ou seja, } \forall i \in [1...b], x_i = x'_i \oplus k_i$$

O exemplo a seguir ilustra este cifrador, usando caracteres ASCII. Nele, a mensagem x ("TOMATE") é cifrada usando a chave k ("ABCDEF"):

x (texto)	T	O	M	A	T	E
k (chave)	A	B	C	D	E	F
x (ASCII)	84	79	77	65	84	69
k (ASCII)	65	66	67	68	69	70
x (binário)	01010100	01001111	01001101	01000001	01010100	01000101
k (binário)	01000001	01000010	01000011	01000100	01000101	01000110
$x' = x \oplus k$	00010101	00001101	00001110	00000101	00010001	00000011
x' (ASCII)	21	13	14	5	17	3

Para decifrar a mensagem x', basta repetir a operação $\oplus$ com a mesma chave:

x'	00010101	00001101	00001110	00000101	00010001	00000011
k	01000001	01000010	01000011	01000100	01000101	01000110
$x' \oplus k$	01010100	01001111	01001101	01000001	01010100	01000101
(ASCII)	84	79	77	65	84	69
(texto)	T	O	M	A	T	E

A Figura 27.1 ilustra a aplicação do cifrador de Vernam-Mauborgne sobre uma imagem. A chave é uma imagem aleatória com as mesmas dimensões da imagem a cifrar; a operação de OU-exclusivo deve ser aplicada entre os pixels correspondentes na imagem e na chave.

Apesar de extremamente simples, o cifrador de Vernam-Mauborgne é considerado um criptossistema inquebrável, sendo comprovadamente seguro se a chave utilizada for realmente aleatória. Entretanto, ele é pouco usado na prática, porque exige uma chave k do mesmo tamanho do texto a cifrar, o que é pouco prático no caso de mensagens longas. Além disso, essa chave deve ser mantida secreta e deve ser usada uma única vez (ou seja, um atacante não deve ser capaz de capturar dois textos distintos

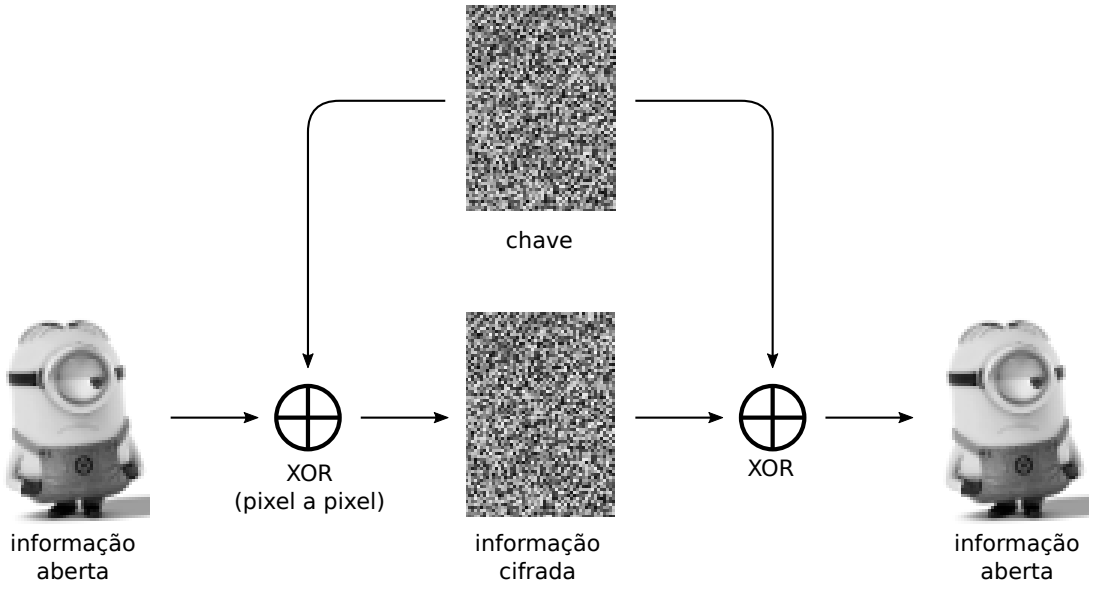

Figura 27.1: Aplicação do cifrador de Vernam sobre uma imagem.

cifrados com a mesma chave). O requisito de uso único da chave levou este cifrador a ser chamado também de *One-Time Pad* [1].

[1] O cifrador conhecido como *One-Time Pad* consiste em uma melhoria do algoritmo original de G. Vernam, proposto por J. Mauborgne pouco tempo depois.

27.4 Criptografia simétrica

De acordo com o tipo de chave utilizada, os algoritmos de criptografia se dividem em dois grandes grupos: *algoritmos simétricos* e *algoritmos assimétricos*. Nos **algoritmos simétricos**, a mesma chave k é usada para cifrar e decifrar a informação. Em outras palavras, se usarmos uma chave k para cifrar um texto, teremos de usar a mesma chave k para decifrá-lo. Essa propriedade pode ser expressa em termos matemáticos:

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

Os cifradores de César e de Vernam são exemplos típicos de cifradores simétricos simples. Outros exemplos de cifradores simétricos bem conhecidos são:

- DES (*Data Encryption Standard*): criado pela IBM nos anos 1970, foi usado amplamente até o final do século XX. O algoritmo original usa chaves de 56 bits, o que gera um espaço de chaves insuficiente para a capacidade computacional atual. A variante 3DES (*Triple*-DES) usa chaves de 168 bits e é considerada segura, sendo ainda muito usada.
- AES (*Advanced Encryption Standard*): algoritmo simétrico adotado como padrão de segurança pelo governo americano em 2002. Ele pode usar chaves de 128, 192 ou 256 bits, sendo considerado muito seguro. É amplamente utilizado na Internet e em programas de cifragem de arquivos em disco.
- A5/1, A5/2, A5/3: algoritmos de criptografia simétrica usados em telefonia celular GSM, para cifrar as transmissões de voz.

A Figura 27.2 ilustra o esquema básico de funcionamento de um sistema de criptografia simétrica para a troca segura de informações. Nesse esquema, a chave simétrica deve ter sido previamente compartilhada entre quem envia e quem recebe a informação.

Os criptossistemas simétricos são muito rápidos e bastante eficientes para a cifragem de grandes volumes de dados, como arquivos em um disco rígido ou o tráfego em uma conexão de rede. Entretanto, se a informação cifrada tiver de ser enviada a outro usuário, a chave criptográfica secreta usada terá de ser transmitida a ele através de algum meio seguro, para mantê-la secreta. Esse problema é conhecido como *o problema da distribuição de chaves*, e será discutido na Seção 27.5.

27.4.1 Cifradores de substituição e de transposição

De acordo com as operações usadas para cifrar os dados, os cifradores simétricos podem ser baseados em operações de *substituição* ou de *transposição*. Os **cifradores de substituição** se baseiam na substituição de caracteres por outros caracteres usando tabelas de substituição (ou *alfabetos*). Esses cifradores podem ser **monoalfabéticos**, quando usam uma única tabela de substituição, ou **polialfabéticos**, quando usam mais de uma tabela.

O cifrador de César é um exemplo trivial de cifrador de substituição monoalfabético. Outro exemplo dessa família, bem conhecido na cultura popular, é a linguagem *Alien*, usada em episódios da série de TV *Futurama*, cuja tabela é apresentada na Figura 27.3.

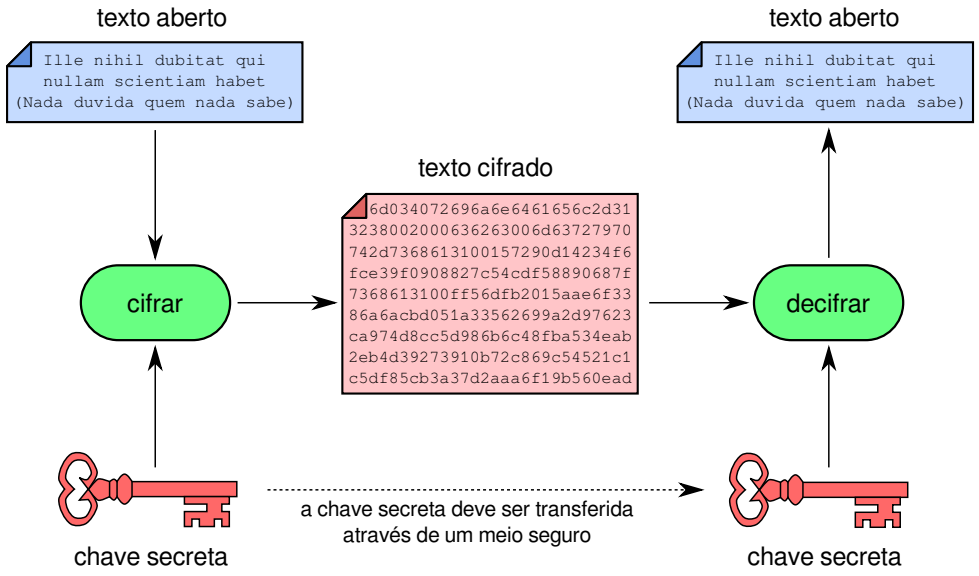

Figura 27.2: Criptografia simétrica.

Figura 27.3: Linguagem *Alien* da série *Futurama*.

Os cifradores de substituição polialfabéticos operam com mais de uma tabela de substituição de caracteres. Um exemplo clássico de cifrador polialfabético é o cifrador de Vigenère, que foi inicialmente proposto por Giovan Battista Bellaso em 1553 e refinado por Blaise de Vigenère no século XIX. Trata-se de um método de cifragem que combina vários cifradores de César em sequência. As operações de cifragem/decifragem usam uma tabela denominada *tabula rasa*, apresentada na Figura 27.4.

Nesse cifrador, para cifrar uma mensagem, primeiro se escolhe uma palavra-chave qualquer, que é repetida até ter o mesmo comprimento da mensagem. Em seguida, cada caractere da mensagem original é codificado usando um cifrador de substituição específico, definido pela linha da *tabula rasa* indicada pela letra correspondente da palavra-chave.

Um exemplo de cifragem usando a palavra-chave “bicicleta” é indicado a seguir. Nele, pode-se observar que a letra “M” da mensagem aberta, combinada à letra “T” da palavra-chave, gera a letra “F” na mensagem cifrada.

Mensagem aberta	ATACARE**M**OS AO AMANHECER DE SEXTA-FEIRA
Palavra-chave	BICICLE**T**AB IC ICLETABIC IC LETAB ICICL
Mensagem cifrada	BBCKCCI**F**OT IQ IOLRAEDMT LG DIQTB-NGQTL

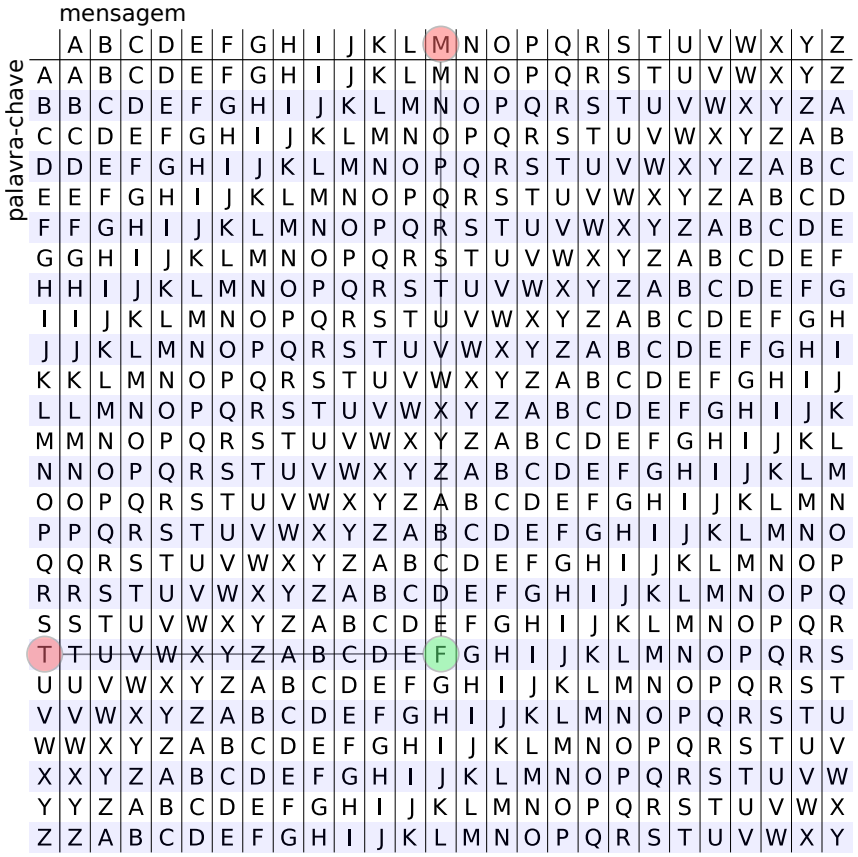

Figura 27.4: Tabula rasa do cifrador de Vigenère.

No cifrador de Vigenère, uma senha com n caracteres distintos irá usar n linhas distintas da *tabula rasa*. Considerando um alfabeto com 26 letras, teremos um espaço de chaves de 26^n, o que é bastante respeitável para a época em que o esquema foi usado.

Os cifradores de substituição podem ainda ser **monográficos** ou **poligráficos**, conforme cifrem caracteres individuais ou grupos de caracteres. Os cifradores de César e de Vigenère são monográficos. O cifrador *Playfair*, que trata o texto aberto em grupos de duas letras, é um bom exemplo de cifrador poligráfico.

Por outro lado, os **cifradores de transposição** (ou permutação) têm como mecanismo básico a troca de posição (ou embaralhamento) dos caracteres que compõem uma mensagem, sem substituí-los. O objetivo básico da operação de transposição é espalhar a informação aberta em toda a extensão do texto cifrado.

Um exemplo clássico de cifrador de transposição é o algoritmo *Rail Fence*, no qual os caracteres da mensagem aberta são distribuídos em várias linhas de uma "cerca" imaginária. Por exemplo, considerando a mesma mensagem do exemplo anterior ("Atacaremos ao amanhecer de sexta-feira") e uma cerca com 4 linhas ($k = 4$), teríamos a seguinte distribuição de caracteres na cerca:

```
A . . . . . E . . . . . A . . . . . C . . . . . E . . . . . I . .
. T . . . R . M . . . O . M . . . E . E . . . S . X . . . E . R .
. . A . A . . . O . A . . . A . H . . . R . E . . . T . F . . . A
. . . C . . . . . S . . . . . N . . . . . D . . . . . A . . . . .
```

A mensagem cifrada é obtida ao percorrer a cerca linha após linha: `AEACEITRMOMEESXERAAOAAHRETFACSNDA`. É interessante observar que os caracteres da

mensagem cifrada são os mesmos da mensagem aberta; uma análise de frequência dos caracteres poderá auxiliar a inferir qual a língua usada no texto, mas será pouco útil para identificar as posições dos caracteres na mensagem original.

Algoritmos de cifragem por substituição e por transposição raramente são utilizados isoladamente. Cifradores simétricos modernos, como o 3DES, AES e outros, são construídos usando vários blocos de substituição e de transposição aplicados de forma alternada, para aumentar a resistência do criptossistema [Stamp, 2011].

27.4.2 Cifradores de fluxo e de bloco

De acordo com a forma de agrupar os dados a cifrar, os cifradores simétricos podem ser classificados em dois grandes grupos: os *cifradores de fluxo* e os *cifradores de bloco*. Os **cifradores de fluxo** (*stream ciphers*) cifram cada byte da mensagem aberta em sequência, produzindo um byte cifrado como saída. Por essa característica sequencial, esses cifradores são importantes para aplicações de mídia em tempo real, como VoIP (voz sobre IP) e comunicações em redes celulares. Exemplos típicos de cifradores de fluxo incluem o RC4, usado até pouco tempo atrás nas redes sem fio, e o A5/1, usado para cifrar fluxo de voz em telefones GSM.

A maioria dos cifradores de fluxo funciona de forma similar: um fluxo contínuo de bytes aleatórios (*keystream*) é gerado por um bloco PRNG (*Pseudo-Random Number Generator*) a partir de uma semente que é a chave simétrica, ou calculada a partir dela. Cada byte desse fluxo é combinado com um byte do fluxo de dados aberto, para produzir um byte do fluxo cifrado. A combinação entre os fluxos geralmente é feita usando a operação XOR, de forma similar ao cifrador de Vernam. No lado do receptor, o mesmo bloco PRNG, inicializado com a mesma semente, refaz a operação XOR para decifrar o fluxo de dados. A figura 27.5 ilustra esse funcionamento.

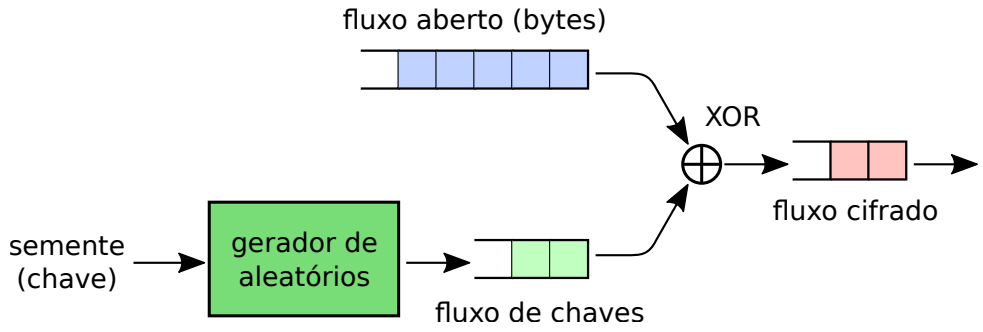

Figura 27.5: Funcionamento básico de um cifrador por fluxo

Como seu próprio nome diz, os **cifradores de bloco** (*block ciphers*) cifram os dados em blocos de mesmo tamanho, geralmente entre 64 e 128 bits. Os dados a serem cifrados são divididos em blocos e o algoritmo de cifragem é aplicado a cada bloco, até o final dos dados. Caso o último bloco não esteja completo, bits de preenchimento (*padding*) são geralmente adicionados para completá-lo. A operação em blocos provê a estes algoritmos uma maior eficiência para cifrar grandes volumes de dados, como tráfego de rede e arquivos em disco. Exemplos comuns de cifradores de bloco incluem o AES (*Advanced Encryption Standard*) e o DES/3DES (*Data Encryption Standard*).

O *modo de operação* de um cifrador de blocos define a forma como algoritmo percorre e considera os blocos de dados a cifrar. Esse modo de operação pode ter

um impacto significativo na segurança do algoritmo. O modo de operação mais simples, chamado ECB (de *Electronic Codebook*), consiste em aplicar o mesmo algoritmo sobre os blocos de dados abertos em sequência, obtendo os blocos de dados cifrados correspondentes. Esse modo de operação está ilustrado na Figura 27.6.

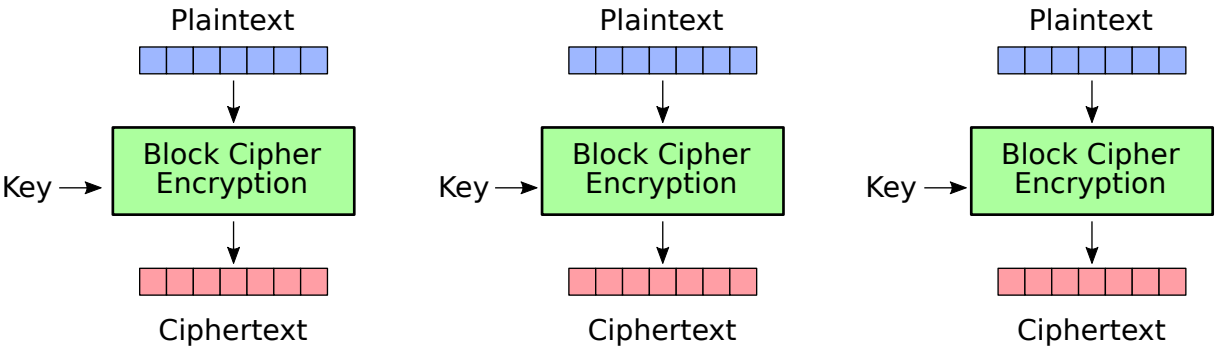

Figura 27.6: Cifragem por blocos em modo ECB [Wikipedia, 2018]

O modo de operação ECB preserva uma forte correlação entre os trechos da mensagem aberta e da mensagem cifrada, o que pode ser indesejável. A figura 27.7 demonstra o efeito dessa correlação indesejada na cifragem por blocos em modo ECB aplicada aos pixels de uma imagem.

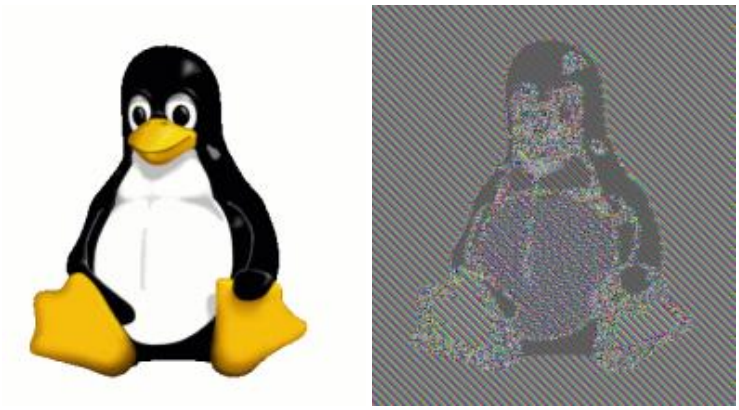

Figura 27.7: Cifragem por blocos em modo ECB: imagem aberta (à esquerda); imagem cifrada (à direita) [Wikipedia, 2018]

Para evitar a correlação direta entre blocos da entrada e da saída, cifradores de bloco modernos usam modos de operação mais sofisticados, que combinam blocos entre si. Um modo de operação bem simples com essa característica é o CBC (*Cipher Block Chaining*), no qual a saída do primeiro bloco é combinada (XOR) com a entrada do segundo bloco e assim por diante, como ilustrado na Figura 27.8. Nessa figura, o IV (*Initialization Vector*) corresponde a um bloco aleatório adicional, que deve ser conhecido ao cifrar e decifrar a mensagem. A Figura 27.9 demonstra o efeito do modo de operação CBC sobre a cifragem em blocos de uma imagem.

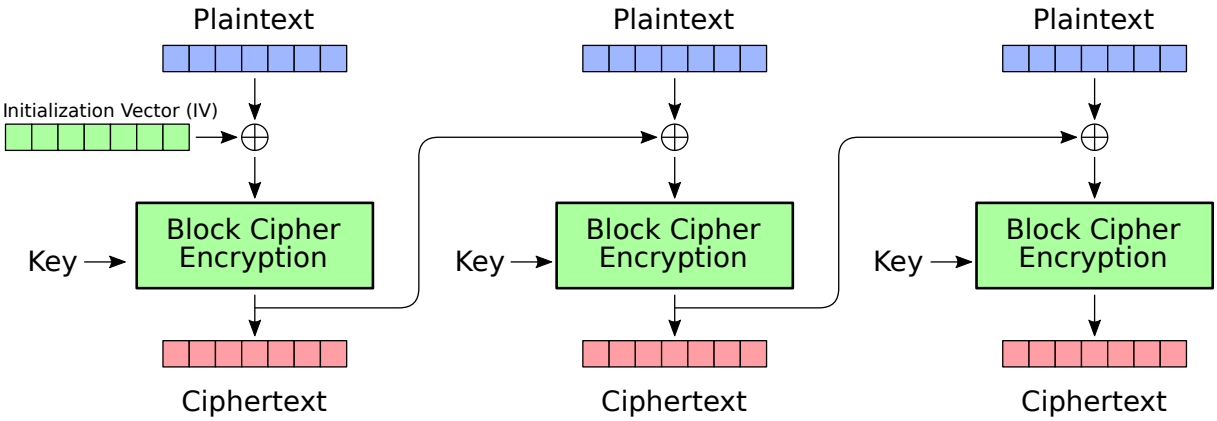

Figura 27.8: Cifragem por blocos em modo CBC [Wikipedia, 2018]

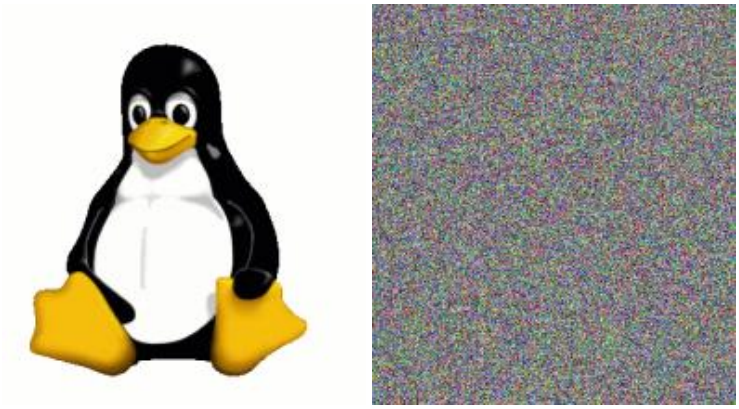

Figura 27.9: Cifragem por blocos em modo CBC: imagem aberta (à esquerda); imagem cifrada (à direita) [Wikipedia, 2018]

27.5 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[2] são usuários honestos que desejam se comunicar de forma confidencial; Mallory é uma usuária desonesta,

[2]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).

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[3] 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[4]:

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

[3] 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.

[4] 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.

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). 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].

27.6 Criptografia assimétrica

O protocolo de acordo de chaves de Diffie-Hellmann (Seção 27.5) 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[5]. A Figura 27.10 ilustra o funcionamento básico da criptografia assimétrica.

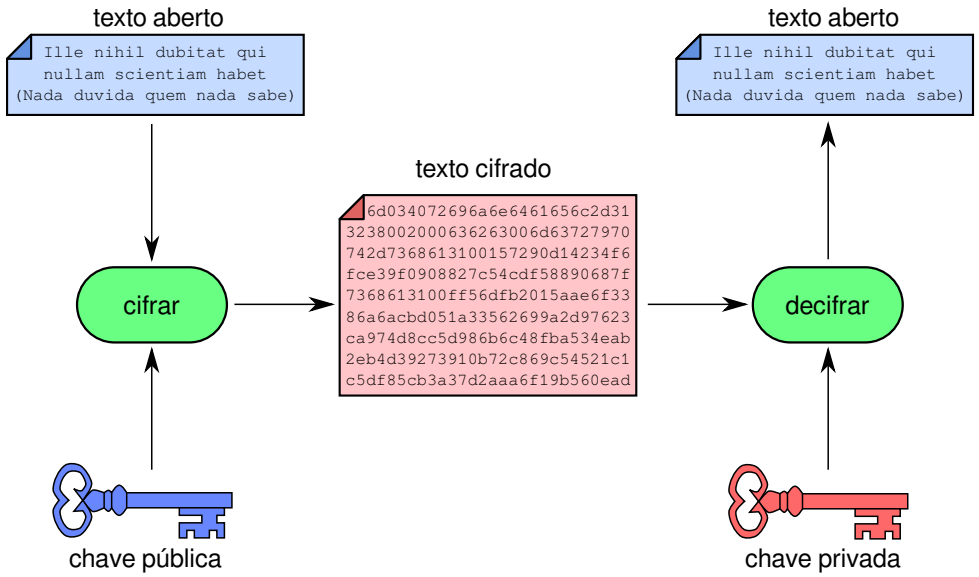

Figura 27.10: Criptografia assimétrica.

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

[5]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.

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

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 27.5, 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 27.11. 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.

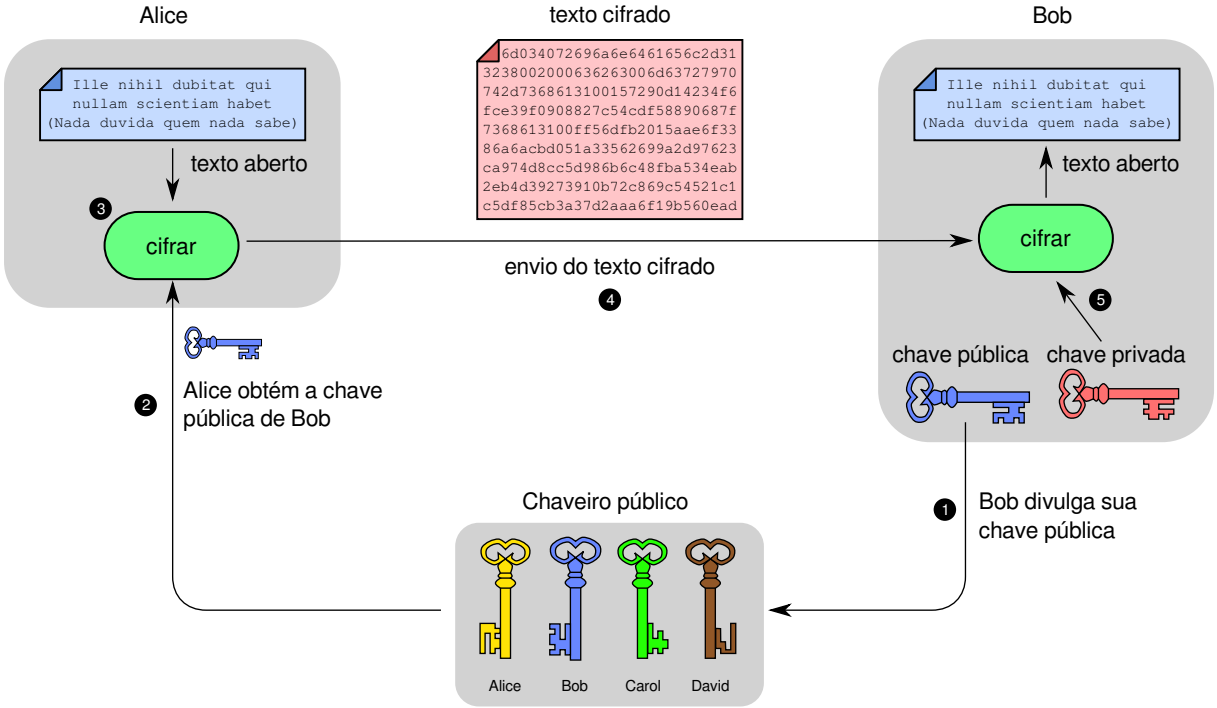

Figura 27.11: Exemplo de uso da criptografia assimétrica.

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 27.9).

A Tabela 27.1 traz uma análise comparativa das principais características dos cifradores simétricos e assimétricos.

Cifrador	**Simétrico**	**Assimétrico**
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 27.1: Quadro comparativo de famílias de cifradores.

27.7 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.

27.8 Resumo criptográfico

Um *resumo criptográfico* (*cryptographic hash*) [Menezes et al., 1996] é uma função $y = hash(x)$ que gera uma sequência de bytes y de tamanho pequeno e fixo (algumas dezenas ou centenas de bytes) a partir de um conjunto de dados x de tamanho variável aplicado como entrada. Os resumos criptográficos são frequentemente usados para identificar unicamente um arquivo ou outra informação digital, ou para atestar sua integridade: caso o conteúdo de um documento digital seja modificado, seu resumo também será alterado.

Em termos matemáticos, os resumos criptográficos são um tipo de *função unidirecional* (*one-way function*). Uma função $f(x)$ é chamada unidirecional quando seu cálculo direto ($y = f(x)$) é rápido, mas o cálculo de sua inversa ($x = f^{-1}(y)$) é impossível ou computacionalmente inviável. Um exemplo clássico de função unidirecional é a fatoração do produto de dois números primos muito grandes, como os usados no algoritmo RSA. Considerando a função $f(p, q) = p \times q$, onde p e q são inteiros primos, calcular $y = f(p, q)$ é simples e rápido, mesmo se p e q forem grandes. Entretanto, fatorar

y para obter de volta os primos p e q pode ser computacionalmente inviável, se y tiver muitos dígitos[6].

Os algoritmos de resumo criptográfico mais conhecidos e utilizados atualmente são os da família SHA (*Secure Hash Algorithms*). Os algoritmos MD5 e SHA1 foram muito utilizados, mas se mostraram inseguros a colisões e não devem mais ser utilizadom para aplicações criptográficas; seu uso continua viável em outras aplicações, como a detecção de réplicas de arquivos [Menezes et al., 1996; Stamp, 2011]. No Linux, comandos como `md5sum` e `sha1sum` permitem calcular respectivamente resumos criptográficos de arquivos:

```
~:> md5sum livro.*
371f456d68720a3c0ba5950fe2708d37  livro.pdf
d4a593dc3d44f6eae54fc62600581b11  livro.tex

~:> sha1sum livro.*
9664a393b533d5d82cfe505aa3ca12410aa1f3b7  livro.pdf
d5bd8d809bb234ba8d2289d4fa13c319e227ac25  livro.tex

~:> sha224sum livro.*
36049e03abf47df178593f79c3cdd0c018406232a0f300d872351631  livro.pdf
edac4154fe0263da86befa8d5072046b96b75c2f91764cc6b5b2f5c0  livro.tex

~:> sha256sum livro.*
c5fc543d1758301feacdc5c6bfd7bc12ef87036fbc589a902856d306cb999d50  livro.pdf
da5075006c6f951e40c9e99cef3218d8a2d16db28e746d0f4e4b18cf365a8099  livro.tex
```

Uma boa função de resumo criptográfico deve gerar sempre a mesma saída para a mesma entrada ($hash(m_1) = hash(m_2) \iff m_1 = m_2$) e saídas diferentes para entradas diferentes (($hash(m_1) \neq hash(m_2) \iff m_1 \neq m_2$). No entanto, como o número de bytes do resumo é pequeno, podem ocorrer *colisões*: duas entradas distintas m_1 e m_2 gerando o mesmo resumo ($m_1 \neq m_2$ mas $hash(m_1) = hash(m_2)$). Idealmente, uma função de *hash* criptográfico deve apresentar as seguintes propriedades:

Determinismo: para uma dada entrada m, a saída é sempre a mesma.

Rapidez: o cálculo de $x = hash(m)$ é rápido para qualquer m.

Resistência à pré-imagem: dado um valor de x, é difícil encontrar m tal que $x = hash(m)$ (ou seja, a função é difícil de inverter).

Resistência à colisão: é difícil encontrar duas mensagens quaisquer $m_1 \neq m_2$ tal que $hash(m_1) = hash(m_2)$.

Espalhamento: uma modificação em um trecho específico dos dados de entrada m gera modificações em várias partes do resumo $hash(m)$.

Sensibilidade: uma pequena mudança nos dados de entrada m (mesmo um só bit) gera mudanças significativas no resumo $hash(m)$.

[6] Em 2014, um grupo de pesquisadores conseguiu fatorar o inteiro $2^{1199} - 1$ (que tem 361 dígitos), em um projeto que consumiu cerca de 7.500 anos-CPU.

27.9 Assinatura digital

Os algoritmos de criptografia assimétrica e resumos criptográficos previamente apresentados permitem efetuar a *assinatura digital* de documentos eletrônicos. A assinatura digital é uma forma de verificar a autoria e integridade de um documento, sendo por isso o mecanismo básico utilizado na construção dos *certificados digitais*, amplamente empregados para a autenticação de servidores na Internet.

Em termos gerais, a assinatura digital de um documento consiste de um resumo digital do mesmo, cifrado usando a chave privada de seu autor (ou de quem o está assinando). Sendo um documento d emitido pelo usuário u, sua assinatura digital $s(d, u)$ é definida por:

$$s(d, u) = \{\ hash(d)\ \}_{kv(u)}$$

onde $hash(x)$ é uma função de resumo criptográfico conhecida, $\{x\}_k$ indica a cifragem de x usando uma chave k e $kv(u)$ é a chave privada do usuário u. Para verificar a validade da assinatura, basta calcular novamente o resumo $r' = hash(d)$ e compará-lo com o resumo obtido da assinatura, decifrada usando a chave pública de u ($r'' = \{s\}^{-1}_{kp(u)}$). Se ambos forem iguais ($r' = r''$), o documento foi realmente assinado por u e está íntegro, ou seja, não foi modificado desde que u o assinou [Menezes et al., 1996].

A Figura 27.12 ilustra o processo de assinatura digital e verificação de um documento. Os passos do processo são:

1. Alice divulga sua chave pública kp_a em um repositório acessível publicamente;
2. Alice calcula o resumo digital r do documento d a ser assinado;
3. Alice cifra o resumo r usando sua chave privada kv_a, obtendo uma assinatura digital s;
4. A assinatura s e o documento original d, em conjunto, constituem o documento assinado por Alice: $[d, s]$;
5. Bob obtém o documento assinado por Alice ($[d', s']$, com $d' = d$ e $s' = s$ se ambos estiverem íntegros);
6. Bob recalcula o resumo digital $r' = hash(d')$ do documento, usando o mesmo algoritmo empregado por Alice;
7. Bob obtém a chave pública kp_a de Alice e a usa para decifrar a assinatura s' do documento, obtendo um resumo r'' ($r'' = r$ se s foi realmente cifrado com a chave kv_a e se $s' = s$) ;
8. Bob compara o resumo r' do documento com o resumo r'' obtido da assinatura digital; se ambos forem iguais ($r' = r''$), o documento foi assinado por Alice e está íntegro, assim como sua assinatura.

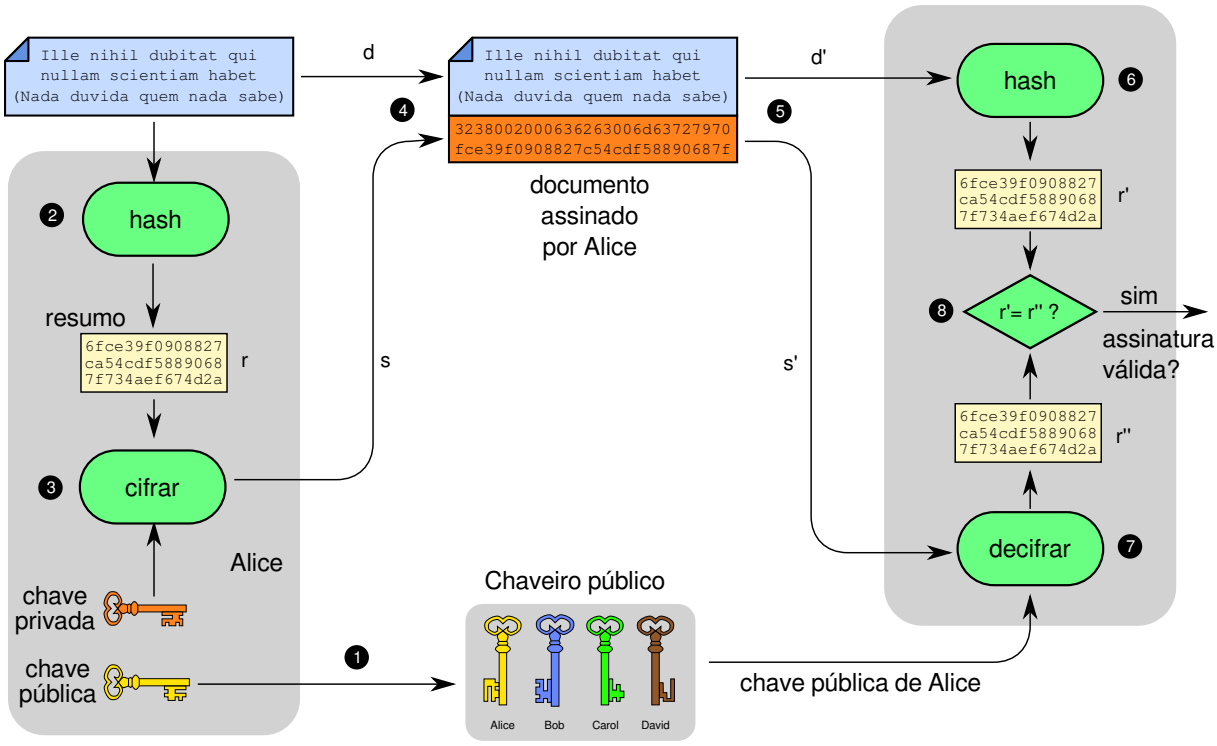

Figura 27.12: Assinatura e verificação de uma assinatura digital.

27.10 Certificado de chave pública

A identificação confiável do proprietário de uma chave pública é fundamental para o funcionamento correto das técnicas de criptografia assimétrica e de assinatura digital. Uma chave pública é composta por uma mera sequência de bytes que não permite a identificação direta de seu proprietário. Por isso, torna-se necessária uma estrutura complementar para fazer essa identificação. A associação entre chaves públicas e seus respectivos proprietários é realizada através dos *certificados digitais*. Um certificado digital é um documento digital assinado, composto das seguintes partes [Menezes et al., 1996]:

- Identidade do proprietário do certificado (nome, endereço, e-mail, URL, número IP e/ou outras informações que permitam identificá-lo unicamente)[7];
- Chave pública do proprietário do certificado;
- Identificação da entidade que emitiu/assinou o certificado;
- Outras informações, como período de validade do certificado, algoritmos de criptografia e resumos utilizados, etc.;
- Uma ou mais assinaturas digitais do conteúdo, emitidas por entidades consideradas confiáveis pelos usuários do certificado.

[7]Deve-se ressaltar que um certificado pode pertencer a um usuário humano, a um sistema computacional ou qualquer módulo de software que precise ser identificado de forma inequívoca.

Dessa forma, um certificado digital "amarra" uma identidade a uma chave pública. Para verificar a validade de um certificado, basta usar a chave pública da entidade que o assinou. Existem vários tipos de certificados digitais com seus formatos e conteúdos próprios, sendo os certificados PGP e X.509 aqueles mais difundidos [Mollin, 2000]. Os certificados no padrão X509 são extensivamente utilizados na Internet para a autenticação de chaves públicas de servidores Web, de e-mail, etc. Um exemplo de certificado X.509, destacando sua estrutura básica e principais componentes, é apresentado na Figura 27.13.

```
Certificate Data:
   Version: 3 (0x2)
   Serial Number: 05:f1:3c:83:7e:0e:bb:86:ed:f8:c4:9b
   Issuer: C=BE, O=GlobalSign nv-sa, CN=GlobalSign Extended Validation CA-SHA256-G3
   Validity                                                          informações básicas
      Not Before: Feb  7 12:41:03 2017 GMT
      Not After : May  9 23:59:59 2018 GMT
   Subject: businessCategory=Private Organization/serialNumber=00.000.000/7297-44/
            jurisdictionC=BR, C=BR, ST=Distrito Federal, L=Brasilia/
            street=ST STN SN QD 716 CONJ C EDIF SEDE IV ANDAR 1 ASA NORTE,   proprietário
            OU=DITEC, O=Banco do Brasil S.A., CN=www2.bancobrasil.com.br     do certificado
   Subject Public Key Info:
      Public Key Algorithm: rsaEncryption
         Public-Key: (2048 bit)
         Modulus:                                                    chave pública
            00:db:4a:0e:92:da:5b:f3:38:3f:d5:63:9d:6d:f9:            do proprietário
            91:6c:16:fc:24:84:28:e8:aa:86:aa:9c:a3:aa:1a:            do certificado
            2e:b6:09:74:6a:f8:1e:31:4a:60:81:0f:ac:76:59:
            ... (linhas omitidas)
            8e:0b
         Exponent: 65537 (0x10001)
   X509v3 extensions:
      X509v3 Key Usage: critical                                     campos opcionais
         Digital Signature, Key Encipherment
      Authority Information Access:
         CA Issuers - URI:http://secure.globalsign.com/cacert/gsextendvalsha2g3r3.crt
         OCSP - URI:http://ocsp2.globalsign.com/gsextendvalsha2g3r3
      X509v3 Extended Key Usage:
         TLS Web Server Authentication, TLS Web Client Authentication
Signature Algorithm: sha256WithRSAEncryption
   94:8e:14:c6:38:30:78:77:80:fc:92:f1:5b:8b:72:6a:b6:b6:
   95:66:c5:7b:ba:be:51:a4:b8:8a:f5:37:0a:4a:74:4d:82:27:          assinatura do emissor
   ... (linhas omitidas)
   b6:44:e8:8c
```

Figura 27.13: Certificado digital no padrão X.509.

27.11 Infraestrutura de chaves públicas

Todo certificado deve ser assinado por alguma entidade considerada confiável pelos usuários do sistema. Essas entidades são normalmente denominadas *Autoridades*

Certificadoras (AC ou CA – *Certification Authorities*). Como as chaves públicas das ACs devem ser usadas para verificar a validade de um certificado, surge um problema: como garantir que uma chave pública realmente pertence a uma dada autoridade certificadora?

A solução para esse problema é simples: basta criar um certificado para essa AC, assinado por outra AC ainda mais confiável. Dessa forma, pode-se construir uma estrutura hierárquica de certificação, na qual a AC mais confiável (denominada "AC raiz") assina os certificados de outras ACs, e assim sucessivamente, até chegar aos certificados dos servidores, usuários e demais entidades do sistema. Uma estrutura de certificação dessa forma se chama *Infraestrutura de Chaves Públicas* (ICP ou PKI - *Public-Key Infrastructure*). Em uma ICP convencional (hierárquica), a chave pública da AC raiz deve ser conhecida de todos e é considerada íntegra [Mollin, 2000].

A Figura 27.14 traz um exemplo de infraestrutura de chaves públicas hierárquica. A chave pública AC raiz (vermelha) é usada para assinar os certificados das chaves verde e azul, e assim por diante. Reciprocamente, o certificado de chave roxo depende da confiança na chave azul, que por sua vez depende da confiança na chave vermelha. A sequência de certificados *roxo* → *azul* → *vermelho* é chamada de **cadeia de certificação** ou *cadeia de confiança*.

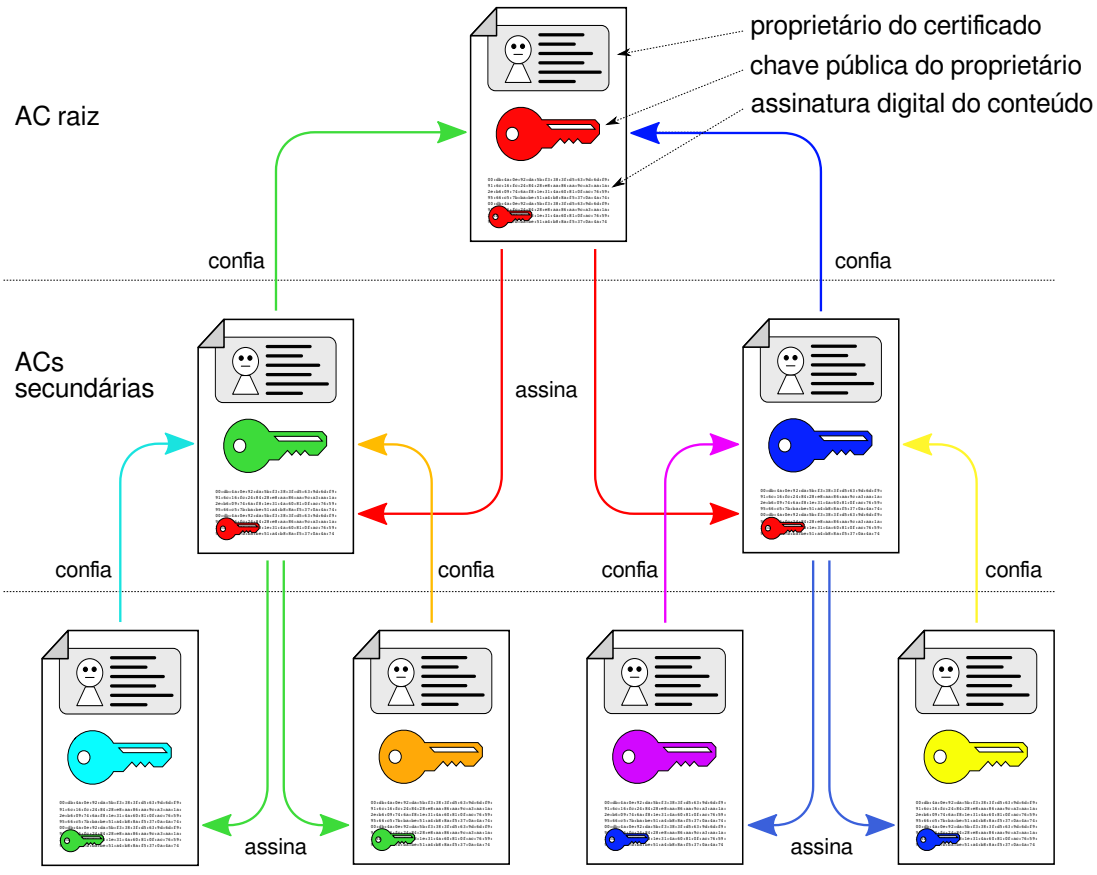

Figura 27.14: Infraestrutura de chaves públicas hierárquica.

O campo *Validity* de um certificado X509 (ver Figura 27.13) diz respeito ao seu prazo de validade, ou seja, o período de tempo em que o certificado é considerado válido. Entretanto, em algumas situações pode ser necessário **revogar certificados** antes do prazo final de validade. Casos típicos de revogação envolvem o vazamento da chave

privada do usuário ou da de alguma autoridade certificadora na cadeia de certificação que valida o certificado, ou então situações mais banais, como a cessação de atividade da empresa proprietária do certificado ou mudanças na finalidade do certificado (campos opcionais `Key Usage` na Figura 27.13).

Existem dois mecanismos básicos para revogar certificados: as CRLs - *Certificate Revocation Lists* são listas de certificados revogados mantidas pelas autoridades certificadoras, que pode ser descarregadas pelo software cliente através de um acesso HTTP. Contudo, em autoridades certificadoras populares, as CRLs podem conter muitos certificados e se tornar muito grandes. Por isso, mais recentemente foi definido o OCSP - *Online Certificate Status Protocol*, que permite ao software consultar junto à CA o status de um certificado digital específico.

Exercícios

1. Na série de TV Futurama (escrita por Matt Groening, o mesmo autor dos Simpsons) é usada uma escrita cifrada denominada *Alien Language*, mostrada na Figura 27.3. Explique qual o tipo de criptografia empregado na *Alien Language* e indique qual o tamanho do espaço de chaves da mesma.

2. O texto em português a seguir foi cifrado usando o cifrador de César. Encontre o **texto original** e a **chave** usada para cifrá-lo; explique seu procedimento.

   ```
   Kjqne fvzjqj vzj ywfsxkjwj t vzj
   xfgj j fuwjsij t vzj jsxnsf.
   Htwf Htwfqnsf.
   ```

   Para facilitar seu trabalho, a tabela a seguir traz a frequência de caracteres típica de textos na língua portuguesa:

letra	freq%	letra	freq%	letra	freq%	letra	freq%	letra	freq%
A	14,6	B	1,04	C	3,88	D	4,99	E	12,6
F	1,02	G	1,30	H	1,28	I	6,18	J	0,40
K	0,02	L	2,78	M	4,74	N	5,05	O	10,7
P	2,52	Q	1,20	R	6,53	S	7,81	T	4,34
U	4,63	V	1,67	W	0,01	X	0,21	Y	0,01
Z	0,47								

3. Use o cifrador de Vigenère para cifrar a mensagem secreta “Encontramos aliens” usando a palavra-chave “missao”.

4. 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?

5. Recentemente foi noticiado na imprensa que certificados digitais emitidos pela Autoridade Certificadora holandesa *DigiNotar* haviam sido falsificados e estavam sendo usados por um governo do oriente médio para monitorar as comunicações de seus cidadãos. Considerando o certificado falso do serviço de e-mails do *Google* (`mail.google.com`), explique:

   (a) Neste contexto, em que consiste um certificado falso?

   (b) Qual a utilidade de um certificado falso na interceptação de comunicações?

   (c) Por que somente os usuários do navegador *Chrome* (produzido pelo próprio *Google*) detectaram o certificado falso, enquanto usuários de outros navegadores não perceberam nada?

6. O provedor de conteúdo TOL (*Tabajara OnLine*) decidiu implementar um novo mecanismo de segurança em suas páginas web. Esse mecanismo consiste em adicionar uma etiqueta oculta (*HTML tag*) em cada página, contendo o nome do autor (*name*), a data de produção (*date*) e uma assinatura digital *s*. Essa assinatura é constituida pelo hash criptográfico do nome do autor e da data (*hash*(*name* + *date*)), cifrado usando a chave privada do autor da página. O conteúdo da página Web em si não é cifrado. As chaves públicas dos autores registrados podem ser obtidas em `http://www.tol.com.br/pubkeys.html`.

   Responda:

   (a) Que objetivo tinham em mente os proponentes desse mecanismo?

   (b) Esse esquema é seguro? Por que?

   (c) Se o esquema não for seguro, indique um possível ataque ao mesmo; caso seja seguro, explique por que esse mesmo ataque não funcionaria.

Referências

A. Menezes, P. Van Oorschot, and S. Vanstone. *Handbook of Applied Cryptography*. CRC Press, 1996.

R. A. Mollin. *An Introduction to Cryptography*. CRC Press, Inc., Boca Raton, FL, USA, 2000. ISBN 1584881275.

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.

Wikipedia. Wikipedia online enciclopedia. `http://www.wikipedia.org`, 2018.