Técnicas clássicas de encriptação

02

TÓPICOS ABORDADOS

OBJETIVOS DE APRENDIZAGEM

APÓS ESTUDAR ESTE CAPÍTULO, VOCÊ SERÁ CAPAZ DE:

- ☑ Apresentar uma visão geral dos principais conceitos de criptografia simétrica.
- ☑ Explicar a diferença entre criptoanálise e ataque por força bruta.
- ☑ Entender a operação de uma cifra de substituição monoalfabética.
- ☑ Entender a operação de uma cifra polialfabética.
- ☑ Apresentar uma visão geral da cifra de Hill.
- ☑ Descrever a operação de uma máquina de rotor.

"Estou bastante familiarizado com todas as formas de escritas secretas, e eu mesmo sou autor de um monógrafo divertido sobre o assunto, no qual analiso cento e sessenta cifras separadas", disse Holmes.

— The Adventure of the Dancing Men, Sir Arthur Conan Doyle

A encriptação simétrica, também chamada de encriptação convencional ou encriptação de chave única, era o único tipo em uso antes do desenvolvimento da encriptação por chave pública na década de 1970. Esse continua sendo de longe o mais usado dos dois tipos de encriptação. A Parte 1 avalia diversas cifras simétricas. Neste capítulo, começaremos olhando um modelo geral para o processo de encriptação simétrica; isso nos permitirá entender o contexto dentro do qual os algoritmos são usados. Em seguida, examinaremos diversos algoritmos em uso antes da era do computador. Finalmente, estudaremos rapidamente uma técnica diferente, conhecida como esteganografia. Os capítulos 3 e 5 introduzem as duas cifras simétricas mais utilizadas: DES e AES.

Antes de começar, definiremos alguns termos. Uma mensagem original é conhecida como **texto claro** (ou *plaintext*), enquanto a mensagem codificada é chamada de **texto cifrado** (ou *ciphertext*). O processo de converter um texto claro em um texto cifrado é conhecido como **cifração** ou **encriptação**; restaurar o texto claro a partir do texto cifrado é **decifração** ou **decriptação**. Os muitos esquemas utilizados para a encriptação constituem a área de estudo conhecida como **criptografia**. Esse esquema é designado **sistema criptográfico** ou **cifra**. As técnicas empregadas para decifrar uma mensagem sem qualquer conhecimento dos detalhes de encriptação estão na área da **criptoanálise**, que é o que os leigos chamam de "quebrar o código". As áreas da criptografia e criptoanálise, juntas, são chamadas de **criptologia**.

2.1 MODELO DE CIFRA SIMÉTRICA

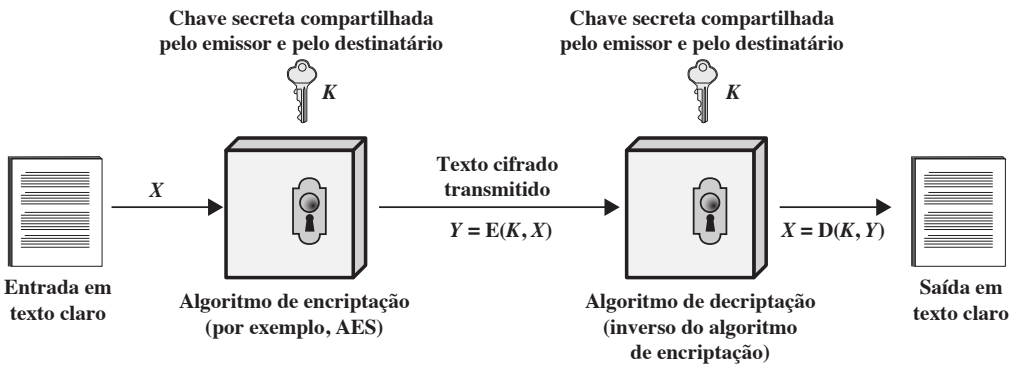
Um esquema de encriptação simétrica possui cinco itens (Figura 2.1):

- **Texto claro:** essa é a mensagem ou dados originais, inteligíveis, que servem como entrada do algoritmo de encriptação.
- **Algoritmo de encriptação:** realiza diversas substituições e transformações no texto claro.
- **Chave secreta:** também é uma entrada para o algoritmo de encriptação. A chave é um valor independente do texto claro e do algoritmo. O algoritmo produzirá uma saída diferente, dependendo da chave usada no momento. As substituições e transformações exatas realizadas pelo algoritmo dependem da chave.
- **Texto cifrado:** essa é a mensagem embaralhada, produzida como saída do algoritmo de encriptação. Ela depende do texto claro e da chave secreta. Para determinada mensagem, duas chaves diferentes produzirão dois textos cifrados distintos. O texto cifrado é um conjunto de dados aparentemente aleatório e, nesse formato, ininteligível.
- **Algoritmo de decriptação:** esse é basicamente o algoritmo de encriptação executado de modo inverso. Ele apanha o texto cifrado e a chave secreta e produz o texto claro original.

Existem dois requisitos para o uso seguro da encriptação simétrica:

1. Precisamos de um algoritmo de encriptação forte. No mínimo, gostaríamos que o algoritmo fosse tal que um oponente que conheça o algoritmo e tenha acesso a um ou mais textos cifrados seja incapaz de decifrar o texto cifrado ou descobrir a chave. Esse requisito normalmente é indicado de maneira mais forte: o oponente deverá ser incapaz de decriptar o texto cifrado ou descobrir a chave, mesmo que possua diversos textos cifrados com seus respectivos textos claros.
2. Emissor e receptor precisam ter obtido cópias da chave secreta de uma forma segura e mantê-la protegida. Se alguém conseguir descobrir a chave e o algoritmo, toda a comunicação usando essa chave poderá ser lida.

Figura 2.1 Modelo simplificado da encriptação simétrica.



Consideramos que é impraticável decriptar uma mensagem com base no texto cifrado, *mais* o conhecimento do algoritmo de encriptação/decriptação. Em outras palavras, não precisamos manter o algoritmo secreto, mas apenas a chave secreta. Essa característica da encriptação simétrica é o que a torna viável para uso generalizado. O fato de que o algoritmo não precisa ser mantido secreto significa que os fabricantes podem desenvolver, e realmente têm desenvolvido, implementações de chip de baixo custo com algoritmos de encriptação de dados. Esses chips são encontrados com facilidade e estão incorporados em diversos produtos. Com o uso da encriptação simétrica, o principal problema de segurança consiste de manter o sigilo da chave.

Vejamos mais de perto os elementos essenciais de um esquema de encriptação simétrica, usando a Figura 2.2. Uma origem produz uma mensagem em texto claro, $X = [X_1, X_2, \ldots, X_M]$. Os M elementos de X são letras em algum alfabeto finito. Tradicionalmente, o alfabeto consiste de 26 letras maiúsculas. Hoje, o alfabeto binário $\{0, 1\}$ em geral é utilizado. Para a encriptação, uma chave na forma $K = [K_1, K_2, \ldots, K_J]$ é gerada. Se isso acontecer na origem da mensagem, então ela também precisa ser fornecida ao destino por meio de algum canal seguro. Como alternativa, um terceiro poderia gerar a chave e oferecê-la com segurança à origem e ao destino.

Com a mensagem X e a chave de encriptação K como entradas, o algoritmo de encriptação produz o texto cifrado $Y = [Y_1, Y_2, \ldots, Y_N]$. Podemos escrever isso como:

$$Y = \mathrm{E}(K, X)$$

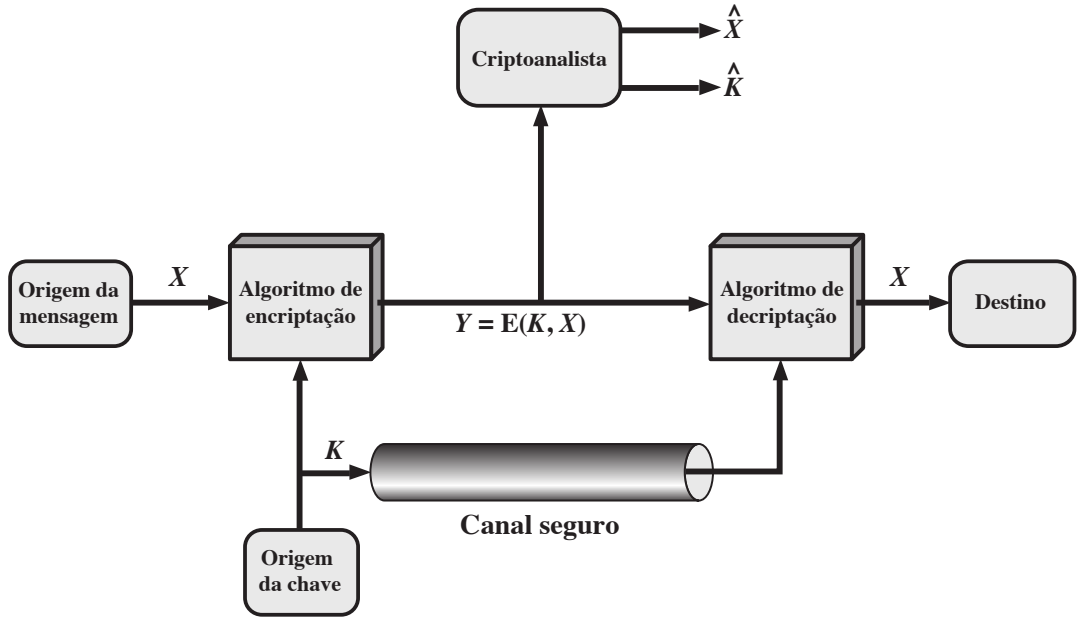
Essa notação indica que Y é produzido usando-se o algoritmo de encriptação E como função do texto claro X, com a função específica determinada pelo valor da chave K.

O receptor legítimo, de posse da chave, é capaz de inverter a transformação:

$$X = \mathrm{D}(K, Y)$$

Um oponente, observando Y, mas não tendo acesso a K ou X, pode tentar recuperar X ou K, ou ambos. Considera-se que o oponente conhece os algoritmos de encriptação (E) e decriptação (D). Se o oponente estiver interessado apenas nessa mensagem em particular, então o foco do ataque é recuperar X, gerando uma estimativa de texto claro $\hat{X}$. Normalmente, porém, o oponente está interessado em ser capaz de ler também mensagens futuras, quando se faz uma tentativa de recuperar K, gerando uma estimativa $\hat{K}$.

Figura 2.2 Modelo de criptossistema simétrico.



Criptografia

Os sistemas criptográficos são caracterizados ao longo de três dimensões independentes:

1. **O tipo das operações usadas para transformar texto claro em texto cifrado.** Todos os algoritmos de encriptação são baseados em dois princípios gerais: substituição, em que cada elemento no texto claro (bit, letra, grupo de bits ou letras) é mapeado em outro elemento, e transposição, em que os elementos no texto claro são rearranjados. O requisito fundamental é que nenhuma informação seja perdida (ou seja, que todas as operações sejam reversíveis). A maioria dos sistemas envolve vários estágios de substituições e transposições (sendo chamados de *sistemas de produto*).
2. **O número de chaves usadas.** Se tanto o emissor quanto o receptor utilizarem a mesma chave, o sistema é considerado de encriptação simétrica, de chave única, de chave secreta ou convencional. Se emissor e receptor usarem chaves diferentes, o sistema é considerado de encriptação assimétrica, de duas chaves ou de chave pública.
3. **O modo em que o texto claro é processado.** Uma *cifra de bloco* processa a entrada de um bloco de elementos de cada vez, produzindo um de saída para cada de entrada. Uma *cifra em fluxo* processa os elementos da entrada continuamente, proporcionando a saída de um elemento de cada vez.

Criptoanálise e ataque por força bruta

Em geral, o objetivo de atacar um sistema de encriptação é recuperar a chave em uso, em vez de simplesmente recuperar o texto claro a partir de um único texto cifrado. Existem duas técnicas gerais para o ataque a um esquema de encriptação convencional:

- **Criptoanálise:** os ataques criptoanalíticos utilizam-se da natureza do algoritmo, e talvez de mais algum conhecimento das características comuns ao texto claro, ou ainda de algumas amostras de pares de texto claro-texto cifrado. Esse tipo de ataque explora as características do algoritmo para tentar deduzir um texto claro específico ou a chave utilizada.
- **Ataque por força bruta:** o atacante testa todas as chaves possíveis em um trecho do texto cifrado, até obter uma tradução inteligível para o texto claro. Na média, metade de todas as chaves possíveis precisam ser experimentadas para então se obter sucesso.

Se algum dos tipos de ataque tiver sucesso na dedução da chave, o efeito é catastrófico: todas as mensagens futuras e passadas, encriptadas com essa chave, ficam comprometidas.

Primeiro, consideramos a criptoanálise, e depois os ataques por força bruta.

O Quadro 2.1 resume os diversos tipos de **ataques de criptoanálise**, baseados na quantidade de informação conhecida pelo criptoanalista. O cenário mais difícil surge quando a única informação disponível é *apenas o texto cifrado*. Em alguns casos, nem sequer o algoritmo de encriptação é conhecido, mas em geral podemos considerar que o oponente sabe qual é o algoritmo usado para a encriptação. Um ataque sob essas circunstâncias é a técnica de força bruta de testar todas as chaves possíveis. Se o espaço de chaves for muito grande, isso se torna impraticável. Assim, o oponente precisa contar com uma análise baseada apenas no texto cifrado, geralmente aplicando diversos testes estatísticos a ele. Para usar essa técnica, o oponente necessita ter alguma ideia geral do tipo de texto claro que está encoberto, como um texto em inglês ou francês, um arquivo EXE, um código fonte em Java, um arquivo de contabilidade, e assim por diante.

O ataque apenas com texto cifrado é o mais fácil de ser defendido, pois o oponente tem a quantidade mínima de informação para trabalhar. Em muitos casos, porém, o analista tem mais informações. Ele pode ser capaz de capturar uma ou mais mensagens de texto claro, além de suas encriptações. Ou então pode saber que certos padrões de texto claro aparecerão em uma mensagem. Por exemplo, um arquivo codificado no formato Postscript sempre começa com o mesmo padrão, ou pode ter um cabeçalho ou *banner* padronizado para uma mensagem de transferência eletrônica financeira, e assim por diante. Todos esses exemplos são de *texto claro conhecido*. Ciente disso, o analista pode ser capaz de deduzir a chave com base no modo como o texto claro conhecido é transformado.

Bastante relacionado ao ataque de texto claro conhecido é o que poderia ser chamado de ataque de palavra provável. Se o oponente estiver trabalhando com a encriptação de alguma mensagem de texto geral, ele talvez tenha pouco conhecimento do que está nela. Porém, se o oponente estiver atrás de alguma informação muito

Quadro 2.1 Tipos de ataque sobre mensagens encriptadas.

TIPO DE ATAQUE	CONHECIDO AO CRIPTOANALISTA
Apenas texto cifrado	■ Algoritmo de encriptação ■ Texto cifrado
Texto claro conhecido	■ Algoritmo de encriptação ■ Texto cifrado ■ Um ou mais pares de texto claro-texto cifrado produzidos pela chave secreta
Texto claro escolhido	■ Algoritmo de encriptação ■ Texto cifrado ■ Mensagem de texto claro escolhida pelo criptoanalista, com seu respectivo texto cifrado gerado com a chave secreta
Texto cifrado escolhido	■ Algoritmo de encriptação ■ Texto cifrado ■ Texto cifrado escolhido pelo criptoanalista, com seu respectivo texto claro decriptado produzido pela chave secreta
Texto escolhido	■ Algoritmo de encriptação ■ Texto cifrado ■ Mensagem de texto claro escolhida pelo criptoanalista, com seu respectivo texto cifrado produzido pela chave secreta ■ Texto cifrado escolhido pelo criptoanalista, com seu respectivo texto claro decriptado produzido pela chave secreta

específica, então partes da mensagem podem ser conhecidas. Por exemplo, se um arquivo de contabilidade estiver sendo transmitido, o oponente pode conhecer o posicionamento de certas palavras-chave no cabeçalho do arquivo. Em outro caso, o código fonte para um programa desenvolvido pela Empresa X poderia incluir uma nota de direito autoral em alguma posição padronizada.

Se o analista de alguma forma for capaz de fazer a origem inserir no sistema uma mensagem escolhida por ele, então o ataque de *texto claro escolhido* é possível. Um exemplo dessa estratégia é a criptoanálise diferencial, explicada no Capítulo 3. Em geral, se o analista for capaz de escolher as mensagens a encriptar, ele poderá escolher de forma deliberada padrões que talvez revelarão a estrutura da chave.

O Quadro 2.1 lista dois outros tipos de ataque: texto cifrado escolhido e texto escolhido. Estes são menos empregados como técnicas criptoanalíticas, mas são possíveis meios de ataque.

Somente algoritmos relativamente fracos não conseguem resistir a um ataque de texto cifrado. Em geral, um algoritmo de encriptação é projetado para aguentar a um ataque de texto claro conhecido.

Duas outras definições merecem ser comentadas. Um esquema de encriptação é **incondicionalmente seguro** se o texto cifrado gerado por ele não tiver informação suficiente para determinar exclusivamente o texto claro correspondente, não importa quanto texto cifrado esteja à disposição. Ou seja, é indiferente quanto tempo um oponente tem, ele não tem como decriptar o texto cifrado, simplesmente porque a informação exigida não está lá. Com a exceção de um esquema conhecido como *one-time pad* (descrito mais adiante neste capítulo), não existe algoritmo de encriptação que seja incondicionalmente seguro. Portanto, tudo o que os usuários de um algoritmo de encriptação podem se esforçar para obter é um algoritmo que atenda a um ou a ambos os critérios a seguir:

- O custo para quebrar a cifra ultrapassa o valor da informação encriptada.
- O tempo exigido para quebrar a cifra supera o tempo de vida útil da informação.

Um esquema de encriptação é considerado **computacionalmente seguro** se um desses dois critérios for atendido. O problema é que é muito difícil estimar a quantidade de esforço exigido para criptoanalisar textos cifrados com sucesso.

Todas as formas de criptoanálise para esquemas de encriptação simétricos são projetadas para explorar o fato de que rastros da estrutura ou do padrão do texto claro podem sobreviver à encriptação e ser discerníveis no texto cifrado. Isso se tornará mais claro à medida que examinarmos diversos esquemas de encriptação

simétrica neste capítulo. Na Parte 2, veremos que a criptoanálise para esquemas de chave pública partem de uma premissa fundamentalmente diferente, a saber, de que as propriedades matemáticas do par de chaves possibilitam que uma das duas chaves seja deduzida a partir da outra.

Um **ataque por força bruta** envolve a tentativa de cada chave possível até que seja obtida uma tradução inteligível de texto cifrado para texto claro. Em média, metade de todas as chaves possíveis precisa ser experimentada para se obter sucesso. Ou seja, se houver X chaves diferentes, um intruso descobriria a verdadeira após $X/2$ tentativas, em média. É importante observar que há mais coisas em um ataque por força bruta do que simplesmente testar todas as chaves possíveis. A menos que seja fornecido um texto claro conhecido, o analista deverá ser capaz de reconhecê-lo como tal. Se a mensagem for simplesmente texto claro em inglês, então o resultado aparece facilmente, embora a tarefa de reconhecer o inglês tenha que ser automatizada. Se a mensagem de texto foi compactada antes da encriptação, então o reconhecimento é mais difícil. E, se a mensagem for de algum tipo mais geral de dado, como um arquivo numérico, e este tiver sido compactado, o problema se torna ainda mais difícil de automatizar. Assim, para suplementar o método por força bruta, é preciso haver algum grau de conhecimento sobre o texto claro esperado, além de algum meio de distinguir automaticamente o texto claro de dados aleatórios.

2.2 TÉCNICAS DE SUBSTITUIÇÃO

Nesta seção e na próxima, examinaremos algumas técnicas de encriptação clássicas. Um estudo dessas técnicas nos permite ilustrar aquelas básicas da encriptação simétrica usadas hoje e os tipos de ataque criptoanalítico que precisam ser antecipados.

Os dois blocos de montagem básicos de todas as técnicas de encriptação são substituição e transposição. Vamos examiná-los nas duas seções seguintes. Por fim, discutiremos um sistema que combina substituição e transposição.

Uma técnica de substituição é aquela em que as letras do texto claro são substituídas por outras letras, números ou símbolos.[1] Se o texto claro for visto como uma sequência de bits, então a substituição envolve trocar padrões de bits de texto claro por padrões de bits de texto cifrado.

Cifra de César

O uso mais antigo que conhecemos de uma cifra de substituição, e o mais simples, foi feito por Júlio César. A cifra de César envolve substituir cada letra do alfabeto por aquela que fica três posições adiante. Por exemplo,

```
claro: meet me after the toga party
cifra: PHHW PH DIWHU WKH WRJD SDUWB
```

Observe que o alfabeto recomeça no final, de modo que a letra após Z é A. Podemos definir a transformação listando todas as alternativas, da seguinte forma:

```
claro: a b c d e f g h i j k l m n o p q r s t u v w x y z
cifra: D E F G H I J K L M N O P Q R S T U V W X Y Z A B C
```

Vamos atribuir um equivalente numérico a cada letra:

a	b	c	d	e	f	g	h	i	j	k	l	m
0	1	2	3	4	5	6	7	8	9	10	11	12

n	o	p	q	r	s	t	u	v	w	x	y	z
13	14	15	16	17	18	19	20	21	22	23	24	25

[1] Quando envolvem letras, as convenções a seguir são usadas neste livro. Texto claro está sempre em minúsculas; texto cifrado está em maiúsculas; os valores de chave estão em minúsculas e itálico.

Então, o algoritmo pode ser expresso da forma a seguir. Para cada letra em texto claro p, substitua-a pela letra do texto cifrado C:[2]

$$C = \mathrm{E}(3, p) = (p + 3) \bmod 26$$

Um deslocamento pode ser de qualquer magnitude, de modo que o algoritmo de César geral é

$$C = \mathrm{E}(k, p) = (p + k) \bmod 26 \quad \textbf{(2.1)}$$

onde k assume um valor no intervalo de 1 a 25. O algoritmo de decriptação é simplesmente

$$p = \mathrm{D}(k, C) = (C - k) \bmod 26 \quad \textbf{(2.2)}$$

Se for conhecido que determinado texto cifrado é uma cifra de César, então uma criptoanálise pela força bruta será facilmente realizada. Basta experimentar todas as 25 chaves possíveis. A Figura 2.3 mostra os resultados da aplicação dessa estratégia ao texto cifrado do exemplo. Nesse caso, o texto claro aparece na terceira fileira.

Três características importantes desse problema nos permitiram usar a criptoanálise pela força bruta:

1. Os algoritmos de encriptação e decriptação são conhecidos.
2. Existem apenas 25 chaves para experimentar.
3. A linguagem do texto claro é conhecida e facilmente reconhecível.

Figura 2.3 Criptoanálise por força bruta da cifra de César.

CHAVE	PHHW PH DIWHU WKH WRJD SDUWB
1	oggv og chvgt vjg vqic rctva
2	nffu nf bgufs uif uphb qbsuz
3	meet me after the toga party
4	ldds ld zesdq sgd snfz ozqsx
5	kccr kc ydrcp rfc rmey nyprw
6	jbbq jb xcqbo qeb qldx mxoqv
7	iaap ia wbpan pda pkcw lwnpu
8	hzzo hz vaozm ocz ojbv kvmot
9	gyyn gy uznyl nby niau julns
10	fxxm fx tymxk max mhzt itkmr
11	ewwl ew sxlwj lzw lgys hsjlq
12	dvvk dv rwkvi kyv kfxr grikp
13	cuuj cu qvjuh jxu jewq fqhjo
14	btti bt puitg iwt idvp epgin
15	assh as othsf hvs hcuo dofhm
16	zrrg zr nsgre gur gbtn cnegl
17	yqqf yq mrfqd ftq fasm bmdfk
18	xppe xp lqepc esp ezrl alcej
19	wood wo kpdob dro dyqk zkbdi
20	vnnc vn jocna cqn cxpj yjach
21	ummb um inbmz bpm bwoi xizbg
22	tlla tl hmaly aol avnh whyaf
23	skkz sk glzkx znk zumg vgxze
24	rjjy rj fkyjw ymj ytlf ufwyd
25	qiix qi ejxiv xli xske tevxc

[2] Definimos a mod n como sendo o resto quando a é dividido por n. Por exemplo, 11 mod 7 = 4. Veja, no Capítulo 4, uma discussão mais detalhada sobre aritmética modular.

Na maioria das situações de rede, podemos considerar que os algoritmos são conhecidos. O que geralmente torna a criptoanálise pela força bruta impraticável é o uso de um algoritmo que emprega um grande número de chaves. Por exemplo, o algoritmo *Triple DES*, examinado no Capítulo 6, utiliza uma chave de 168 bits, gerando um espaço de chave de 2^{168}, ou mais de $3{,}7 \times 10^{50}$ chaves possíveis.

A terceira característica também é significativa. Se a linguagem do texto claro for desconhecida, então a saída de texto claro pode não ser reconhecível. Além do mais, a entrada pode ser abreviada ou compactada de alguma maneira, novamente dificultando o reconhecimento. Por exemplo, a Figura 2.4 mostra uma parte de um arquivo de texto compactada usando um algoritmo chamado ZIP. Se esse arquivo for então encriptado com uma cifra de substituição simples (expandida para incluir mais do que apenas 26 caracteres alfabéticos), então o texto claro não poderá ser reconhecido quando for revelado na criptoanálise por força bruta.

Figura 2.4 Exemplo de texto compactado.

```
˜+Wμ"— Ω–O)≤4{∞‡¸ ë~Ω%ràu·¯Í ◊¯Z-
Ú≠2Ò#Åæð œ«q7¸Ωn·®3N◊Ú Œz'Y-ƒ∞Í[±Û_ èΩ,<NO¬±«ˇxã  Åä£èü3Å
x}ö§kºÂ
_yÍ ^ΔÉ] ¸¤ J/˚iTê&ı 'c<uΩ-
 ÄD(G WÄC~y_ïõÄW PÔı«ÎÜ†ç],¤¡ˇÌ^üÑπˇ≈ˇLˇ9OgflOˇ&Œ≤ ¬≤ ØÔ§˝:
 ˇŒ!SGqèvoˆ ú\¸S>h<-*6ø‡%x´˝|fiÓ#≈˜my‰ˇ≥ñP<,ƒi Áj  Å◊¿˝Zù-
Ω¨Õ¯6Œÿ{% „ΩÊó ¸ï π÷Âî˚úO2çSŷ´O-
2ÄflßiÍ /@ˆ”∏Kº ªPŒπ¸úéˆ´3∑ˇöˇÔZÌ"Y¬ŸΩœY> Ω+eô/˙<K£¿*÷˜"≤û˜
B ZøK˜Qßÿüƒ¸!ÒflÎzsS/]>ÈQ ü
```

Cifras monoalfabéticas

Com apenas 25 chaves possíveis, a cifra de César está longe de ser segura. Um aumento dramático no espaço de chave pode ser conseguido permitindo-se uma substituição arbitrária. Antes de prosseguir, definimos o termo *permutação*. Uma **permutação** é um conjunto finito de elementos S em uma sequência ordenada de todos os elementos de S, com cada um aparecendo exatamente uma vez. Por exemplo, se S = {a, b, c}, existem seis permutações de S:

abc, acb, bac, bca, cab, cba

Em geral, existem $n!$ permutações de um conjunto de n elementos, pois o primeiro deles pode ser escolhido de n maneiras, o segundo, de $n - 1$ maneiras, o terceiro, de $n - 2$ maneiras, e assim por diante.

Lembre-se da atribuição para a cifra de César:

```
claro: a b c d e f g h i j k l m n o p q r s t u v w x y z
cifra: D E F G H I J K L M N O P Q R S T U V W X Y Z A B C
```

Se, em vez disso, a linha "cifra" puder ser qualquer permutação dos 26 caracteres alfabéticos, então haverá 26! ou mais do que 4×10^{26} chaves possíveis. Isso significa 10 ordens de grandeza a mais do que o espaço de chave para DES, e evitaria qualquer técnica de força bruta para criptoanálise. Essa técnica é conhecida como **cifra por substituição monoalfabética**, pois um único alfabeto de cifra (mapeando do alfabeto claro para um cifrado) é utilizado por mensagem.

Porém, existe outra linha de ataque. Se o criptoanalista souber a natureza do texto claro (por exemplo, texto em inglês não compactado), então o analista poderá explorar as regularidades da linguagem. Para ver como essa criptoanálise poderia prosseguir, damos um exemplo parcial aqui, que é adaptado de outro em [SINK09]. O texto cifrado a ser resolvido é

```
UZQSOVUOHXMOPVGPOZPEVSGZWSZOPFPESXUDBMETSXAIZ
VUEPHZHMDZSHZOWSFPAPPDTSVPQUZWYMXUZUHSX
EPYEPOPDZSZUFPOMBZWPFUPZHMDJUDTMOHMQ
```

Como um passo inicial, a frequência relativa das letras poderá ser determinada e comparada com uma distribuição de frequência padrão para o inglês, como vemos na Figura 2.5 (com base em [LEWA00]). Se a mensagem fosse longa o suficiente, essa técnica talvez tivesse sucesso, mas, como essa é relativamente curta,

não podemos esperar uma combinação exata. De qualquer forma, as frequências relativas das letras no texto cifrado (em porcentagens) são as seguintes:

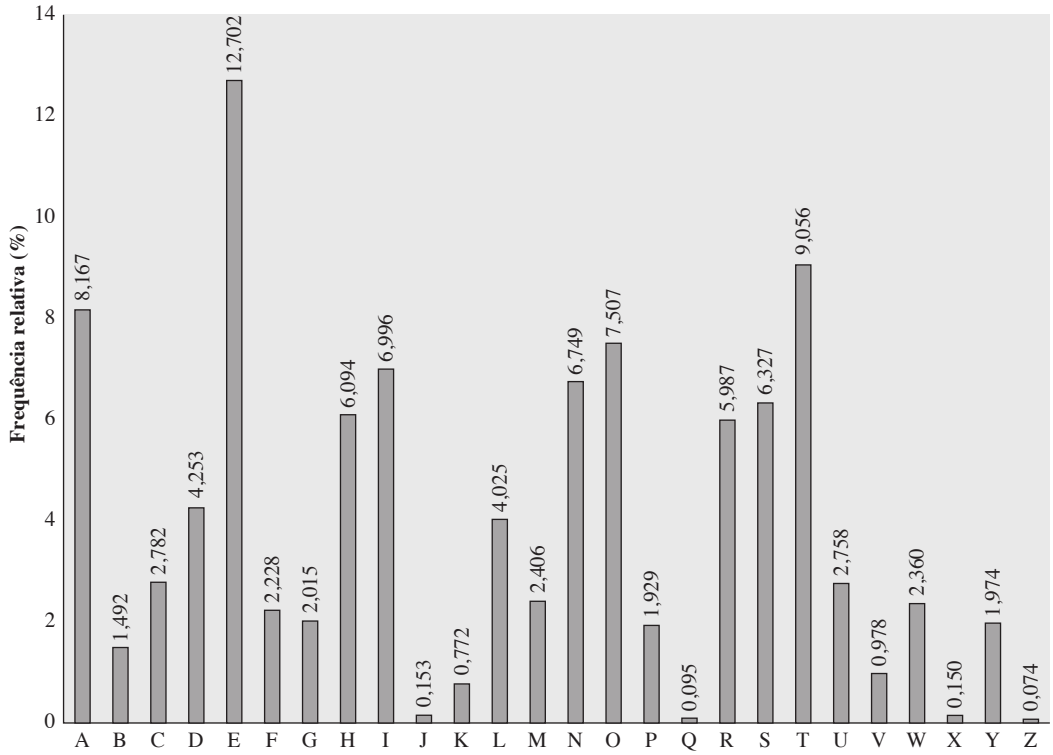
P 13,33	H 5,83	F 3,33	B 1,67	C 0,00
Z 11,67	D 5,00	W 3,33	G 1,67	K 0,00
S 8,33	E 5,00	Q 2,50	Y 1,67	L 0,00
U 8,33	V 4,17	T 2,50	I 0,83	N 0,00
O 7,50	X 4,17	A 1,67	J 0,83	R 0,00
M 6,67				

Comparando esses dados com a Figura 2.5, parece provável que as letras cifradas P e Z sejam equivalentes às letras do texto claro e e t, mas não se sabe ao certo quem é quem. As letras S, U, O, M e H são todas de frequência relativamente alta e é provável que correspondem às que estão no conjunto {a, h, i, n, o, r, s}. As letras com frequências mais baixas (a saber, A, B, G, Y, I, J) provavelmente estão incluídas no conjunto {b, j, k, q, v, x, z}.

Neste ponto, existem diversas maneiras de prosseguir. Poderíamos fazer algumas tentativas de atribuição e começar a preencher o texto claro para ver se ele tem um "esqueleto" pertinente a uma mensagem. Uma técnica mais sistemática é procurar outras regularidades. Por exemplo, talvez saber que certas palavras estão no texto. Ou então poderíamos procurar sequências repetidas de letras cifradas e tentar deduzir seus equivalentes no texto claro.

Uma ferramenta poderosa é examinar a frequência de combinações de duas letras, conhecidas como **digramas**. Uma tabela semelhante à Figura 2.5 poderia ser montada para mostrar a frequência relativa dos digramas. O mais comum deles é th. Em nosso texto cifrado, o digrama mais comum é ZW, que aparece três vezes. Assim, fazemos a correspondência de Z com t e de W com h. Depois, pela nossa hipótese anterior, podemos igualar P com e. Agora, observe que a sequência ZWP aparece no texto cifrado, e podemos traduzir essa sequência como "the". Esse é o trigrama (combinação de três letras) mais frequente em inglês, o que parece indicar que estamos no caminho certo.

Figura 2.5 Frequência relativa de letras no texto em inglês.



Em seguida, observe a sequência ZWSZ na primeira linha. Não sabemos que essas quatro letras compõem uma palavra completa, mas, se sim, ela tem a forma th_t. Nesse caso, S é igual a a.

Até aqui, então, temos

```
UZQSOVUOHXMOPVGPOZPEVSGZWSZOPFPESXUDBMETSXAIZ
 t a        e  e te  a that e e a       a
VUEPHZHMDZSHZOWSFPAPPDTSVPQUZWYMXUZUHSX
   e t   ta t ha e ee  a e  th    t  a
EPYEPOPDZSZUFPOMBZWPFUPZHMDJUDTMOHMQ
 e  e e tat  e   the   t
```

Somente quatro letras foram identificadas, mas já temos um bom pedaço da mensagem. A análise continuada das frequências, mais tentativa e erro, deverão facilmente gerar uma solução a partir desse ponto. O texto claro completo, com espaços incluídos entre as palavras, é o seguinte:*

```
it was disclosed yesterday that several informal but
direct contacts have been made with political
representatives of the viet cong in moscow
```

As cifras monoalfabéticas são fáceis de se quebrar porque refletem os dados de frequência do alfabeto original. Uma contramedida é oferecer vários substitutos, conhecidos como homófonos, para uma única letra. Por exemplo, a letra e poderia ser atribuída a diversos símbolos de cifra diferentes, como 16, 74, 35 e 21, com cada homófono usado em rodízio, ou aleatoriamente. Se o número de símbolos atribuídos a cada letra for proporcional à frequência relativa dela, então a informação de frequência de única letra é completamente extinta. O grande matemático Carl Friedrich Gauss acreditava ter criado uma cifra indecifrável usando homófonos. Porém, até mesmo com homófonos, cada elemento do texto claro afeta somente um elemento do texto cifrado, e padrões de múltiplas letras (por exemplo, frequências de digrama) ainda sobrevivem no texto cifrado, tornando a criptoanálise relativamente simples.

Dois métodos principais são usados nas cifras de substituição para reduzir a extensão da estrutura sobrevivente do texto claro no cifrado. Uma técnica é encriptar várias letras do texto claro, e a outra é usar vários alfabetos de cifra. Examinamos rapidamente cada uma delas.

Cifra Playfair

A cifra de encriptação de múltiplas letras mais conhecida é a Playfair, que trata os digramas no texto claro como unidades isoladas e as traduz para digramas de texto cifrado.[3]

O algoritmo Playfair é baseado no uso de uma matriz 5 × 5 de letras construídas usando uma palavra-chave. Aqui está um exemplo, solucionado por Lord Peter Wimsey em *Have His Carcase*, de Dorothy Sayers:[4]

M	O	N	A	R
C	H	Y	B	D
E	F	G	I/J	K
L	P	Q	S	T
U	V	W	X	Z

Nesse caso, a palavra-chave é *monarchy*. A matriz é construída com o preenchimento das letras da palavra-chave (menos duplicatas) da esquerda para a direita e de cima para baixo, e depois do restante da matriz com as outras letras na ordem alfabética. As letras I e J contam como uma só. O texto claro é encriptado com duas letras de cada vez, de acordo com as seguintes regras:

---

[3] Essa cifra, na realidade, foi inventada pelo cientista britânico Sir Charles Wheatstone em 1854, mas recebeu o nome de seu amigo Barão Playfair de St. Andrews, que a defendeu na agência estrangeira da Grã Bretanha.

[4] O livro fornece um relato fascinante de um ataque de palavra provável.

* N. do Ed.: "Foi revelado ontem que diversos contatos informais, mas diretos, tem sido feitos com representantes políticos dos vietcongues em Moscou."

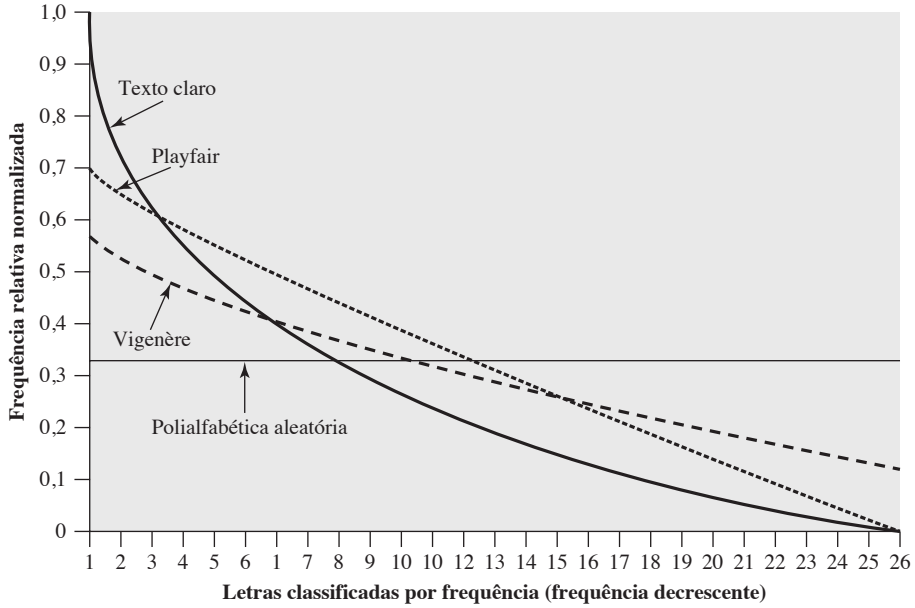
1. Letras de texto claro repetidas que estão no mesmo par são separadas por uma de preenchimento, como x, de modo que *balloon* seria tratado como ba lx lo on.
2. Duas letras de texto claro que estejam na mesma linha da matriz são substituídas pela letra à direita, com o primeiro elemento da linha vindo após o último, de forma rotativa. Por exemplo, ar é encriptado como RM.
3. Duas letras de texto claro que estejam na mesma coluna são substituídas pela letra abaixo, com o elemento de cima da coluna vindo após o último, de forma rotativa. Por exemplo, mu é encriptado como CM.
4. Caso contrário, cada letra de texto claro em um par é substituída por aquela que esteja em sua própria linha e na coluna ocupada pela outra letra de texto claro. Assim, hs torna-se BP, e ea torna-se IM (ou JM, a critério do cifrador).

A cifra Playfair é um grande avanço em relação às cifras monoalfabéticas simples. Primeiramente, enquanto existem apenas 26 letras, existem 26 × 26 = 676 digramas, de modo que a identificação de digramas individuais é mais difícil. Além do mais, as frequências relativas das letras individuais exibem um intervalo muito maior do que o dos digramas, tornando a análise de frequência muito mais difícil. Por esses motivos, a cifra Playfair foi, por muito tempo, considerada indecifrável. Ela foi usada como sistema de campo padrão pelo Exército britânico na Primeira Guerra Mundial, e ainda gozava de um uso considerável pelo Exército dos Estados Unidos e outras forças aliadas durante a Segunda Guerra Mundial.

Apesar desse nível de confiança em sua segurança, a cifra Playfair é relativamente fácil de ser quebrada, pois ainda deixa intacta grande parte da estrutura da linguagem de texto claro. Algumas centenas de letras de texto cifrado geralmente são suficientes para quebrá-la.

Um modo de revelar a eficácia da Playfair e outras cifras aparece na Figura 2.6. A linha rotulada com *texto claro* apresenta a distribuição de frequência dos mais de 70.000 caracteres alfabéticos presentes no artigo sobre criptologia da Enciclopédia Britânica. Essa também é a distribuição de frequência de qualquer cifra por substituição monoalfabética, pois os valores de frequência para letras individuais são os mesmos, apenas com diferentes letras substituídas pelas originais. O gráfico foi desenvolvido da seguinte maneira: o número de ocorrência de cada letra no texto foi contado e dividido pelo da letra mais utilizada. Usando os resultados da Figura 2.5, vemos que e é a letra empregada com mais frequência. Desse modo, e tem uma frequência relativa de 1, t, de 9,056/12,702 ≈ 0,72, e assim por diante. Os pontos no eixo horizontal correspondem às letras na ordem de frequência decrescente.

Figura 2.6 Frequência relativa de ocorrência das letras.



A Figura 2.6 também mostra a distribuição de frequência que resulta quando o texto é encriptado usando a cifra Playfair. Para normalizar o gráfico, o número de ocorrências de cada letra no texto cifrado novamente foi dividido pelo número de ocorrências de e no texto claro. O gráfico resultante, portanto, mostra a extensão à qual a distribuição de frequência das letras, que torna trivial solucionar as cifras de substituição, é mascarada pela encriptação. Se a informação de distribuição de frequência fosse totalmente escondida no processo de encriptação, o desenho de frequências do texto cifrado seria achatado, e a criptoanálise usando o texto cifrado se tornaria efetivamente impossível. Como mostra a figura, a cifra Playfair tem uma distribuição mais achatada do que o texto claro, mas, apesar disso, ela revela muita estrutura para um criptoanalista trabalhar.

O gráfico também mostra a cifra Vigenère, discutida mais adiante. As curvas Hill e Vigenère no gráfico são baseadas em resultados relatados em [SIMM93].

Cifra de Hill[5]

Outra cifra multiletras interessante é a de Hill, desenvolvida pelo matemático Lester Hill em 1929.

CONCEITOS DA ÁLGEBRA LINEAR Antes de descrevermos a cifra de Hill, vamos revisar rapidamente alguma terminologia da álgebra linear. Nesta discussão, estamos preocupados com a aritmética de matriz módulo 26. O leitor que precisa relembrar a multiplicação e inversão de matrizes deverá consultar o Apêndice E.

Definimos o inverso $\mathbf{M}^{-1}$ de uma matriz quadrada $\mathbf{M}$ pela equação $\mathbf{M}(\mathbf{M}^{-1}) = \mathbf{M}^{-1}\mathbf{M} = \mathbf{I}$, onde $\mathbf{I}$ é a matriz identidade. $\mathbf{I}$ é uma matriz quadrada que contém todos os elementos iguais a zero, exceto por 1's ao longo da diagonal principal do canto superior esquerdo ao inferior direito. O inverso de uma matriz nem sempre existe, mas, quando sim, satisfaz a equação anterior. Por exemplo,

$$\mathbf{A} = \begin{pmatrix} 5 & 8 \\ 17 & 3 \end{pmatrix} \qquad \mathbf{A}^{-1} \bmod 26 = \begin{pmatrix} 9 & 2 \\ 1 & 15 \end{pmatrix}$$

$$\mathbf{A}\mathbf{A}^{-1} = \begin{pmatrix} (5 \times 9) + (8 \times 1) & (5 \times 2) + (8 \times 15) \\ (17 \times 9) + (3 \times 1) & (17 \times 2) + (3 \times 15) \end{pmatrix}$$

$$= \begin{pmatrix} 53 & 130 \\ 156 & 79 \end{pmatrix} \bmod 26 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

Para explicar como é determinado o inverso de uma matriz, começamos com o conceito de determinante. Para qualquer matriz quadrada $(m \times m)$, o **determinante** é igual à soma de todos os produtos que podem ser formados apanhando-se exatamente um elemento de cada linha e um de cada coluna, com certos termos do produto precedidos por um sinal de menos. Para uma matriz 2×2,

$$\begin{pmatrix} k_{11} & k_{12} \\ k_{21} & k_{22} \end{pmatrix}$$

o determinante é $k_{11}k_{22} - k_{12}k_{21}$. Para uma matriz 3×3, o valor do determinante é $k_{11}k_{22}k_{33} + k_{21}k_{32}k_{13} + k_{31}k_{12}k_{23} - k_{31}k_{22}k_{13} - k_{21}k_{12}k_{33} - k_{11}k_{32}k_{23}$. Se uma matriz quadrada $\mathbf{A}$ tiver um determinante diferente de zero, então o inverso da matriz é calculado como $[\mathbf{A}^{-1}]_{ij} = (\det \mathbf{A})^{-1}(-1)^{i+j}(\mathrm{D}_{ji})$, onde (D_{ji}) é o subdeterminante formado pela exclusão da linha j e coluna i de $\mathbf{A}$, $\det(\mathbf{A})$ é o determinante de $\mathbf{A}$ e $(\det \mathbf{A})^{-1}$ é o inverso multiplicativo de $(\det \mathbf{A}) \bmod 26$.

Continuando nosso exemplo,

$$\det\begin{pmatrix} 5 & 8 \\ 17 & 3 \end{pmatrix} = (5 \times 3) - (8 \times 17) = -121 \bmod 26 = 9$$

Podemos mostrar que $9^{-1} \bmod 26 = 3$, pois $9 \times 3 = 27 \bmod 26 = 1$ (veja no Capítulo 4 ou no Apêndice E – este último em <sv.pearson.com.br>, em inglês). Portanto, calculamos o inverso de $\mathbf{A}$ como

[5] Essa cifra é um pouco mais difícil de entender do que as outras neste capítulo, mas ilustra um ponto importante sobre a criptoanálise, que será útil mais adiante. Esta subseção pode ser pulada em uma primeira leitura.

$$\mathbf{A} = \begin{pmatrix} 5 & 8 \\ 17 & 3 \end{pmatrix}$$
$$\mathbf{A}^{-1} \bmod 26 = 3\begin{pmatrix} 3 & -8 \\ -17 & 5 \end{pmatrix} = 3\begin{pmatrix} 3 & 18 \\ 9 & 5 \end{pmatrix} = \begin{pmatrix} 9 & 54 \\ 27 & 15 \end{pmatrix} = \begin{pmatrix} 9 & 2 \\ 1 & 15 \end{pmatrix}$$

O ALGORITMO DE HILL Esse algoritmo de encriptação utiliza m letras de texto claro sucessivas e as substitui por m letras de texto cifrado. A substituição é determinada por m equações lineares, em que cada caractere recebe um valor numérico (a = 0, b = 1, ..., z = 25). Para $m = 3$, o sistema pode ser descrito da seguinte forma:

$$c_1 = (k_{11}p_1 + k_{21}p_2 + k_{31}p_3) \bmod 26$$
$$c_2 = (k_{12}p_1 + k_{22}p_2 + k_{32}p_3) \bmod 26$$
$$c_3 = (k_{13}p_1 + k_{23}p_2 + k_{33}p_3) \bmod 26$$

Isso pode ser expresso em termos de vetores de linhas e matrizes:[6]

$$(c_1\ c_2\ c_3) = (p_1\ p_2\ p_3)\begin{pmatrix} k_{11} & k_{12} & k_{13} \\ k_{21} & k_{22} & k_{23} \\ k_{31} & k_{32} & k_{33} \end{pmatrix} \bmod 26$$

ou

$$\mathbf{C} = \mathbf{PK} \bmod 26$$

onde **C** e **P** são vetores de coluna de tamanho 3, representando o texto claro e o texto cifrado, e **K** é uma matriz 3 × 3, indicando a chave de encriptação. As operações são realizadas com mod 26.

Por exemplo, considere o texto claro "*paymoremoney*" e use a chave de encriptação

$$\mathbf{K} = \begin{pmatrix} 17 & 17 & 5 \\ 21 & 18 & 21 \\ 2 & 2 & 19 \end{pmatrix}$$

As três primeiras letras do texto claro são representadas pelo vetor (15 0 24). Então, (15 0 24)**K** = (303 303 531) mod 26 = (17 17 11) = RRL. Continuando dessa forma, o texto cifrado para o texto claro inteiro é RRLMWBKASPDH.

A decriptação exige o uso do inverso da matriz **K**. Podemos calcular det **K** = 23 e, portanto, (det **K**)$^{-1}$ mod 26 = 17. Nesse caso, o inverso é[7]

$$\mathbf{K}^{-1} = \begin{pmatrix} 4 & 9 & 15 \\ 15 & 17 & 6 \\ 24 & 0 & 17 \end{pmatrix}$$

Isso é demonstrado da seguinte maneira:

$$\begin{pmatrix} 17 & 17 & 5 \\ 21 & 18 & 21 \\ 2 & 2 & 19 \end{pmatrix}\begin{pmatrix} 4 & 9 & 15 \\ 15 & 17 & 6 \\ 24 & 0 & 17 \end{pmatrix} = \begin{pmatrix} 443 & 442 & 442 \\ 858 & 495 & 780 \\ 494 & 52 & 365 \end{pmatrix} \bmod 26 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

É fácil ver que, se a matriz $\mathbf{K}^{-1}$ for aplicada ao texto cifrado, então o texto claro é recuperado.

Em termos gerais, o sistema de Hill pode ser expresso da seguinte forma:

[6] Alguns livros de criptografia expressam o texto claro e o texto cifrado como vetores de colunas, de modo que o vetor de colunas é colocado após a matriz, em vez de o vetor de linhas ser posto antes da matriz. *Sage* usa vetores de linhas, de modo que adotamos essa convenção.

[7] Os cálculos neste exemplo podem ser vistos com detalhes no Apêndice E (em <sv.pearson.com.br>, em inglês).

$$\mathbf{C} = \mathrm{E}(\mathbf{K}, \mathbf{P}) = \mathbf{PK} \bmod 26$$
$$\mathbf{P} = \mathrm{D}(\mathbf{K}, \mathbf{C}) = \mathbf{CK}^{-1} \bmod 26 = \mathbf{PKK}^{-1} = \mathbf{P}$$

Assim como na cifra Playfair, o ponto forte da cifra de Hill é que ela oculta completamente as frequências de única letra. De fato, com Hill, o uso de uma matriz maior esconde mais informações de frequência. Assim, uma cifra de Hill de 3 × 3 encobre não apenas informações de frequência de única letra, mas também de duas letras.

Embora a cifra de Hill seja forte contra um ataque apenas de texto cifrado, ela é facilmente quebrada com um ataque de texto claro conhecido. Para uma cifra de Hill de $m \times m$, suponha que tenhamos m pares de texto claro/texto cifrado, cada um com tamanho m. Rotulamos os pares com $\mathbf{P}_j = (p_{1j}p_{1j} \ldots p_{mj})$ e $\mathbf{C}_j = (c_{1j}c_{1j} \ldots c_{mj})$, de modo que $\mathbf{C}_j = \mathbf{P}_j\mathbf{K}$ para $1 \le j \le m$ e para alguma matriz de chave desconhecida $\mathbf{K}$. Então, definimos duas matrizes $m \times m$, $\mathbf{X} = (p_{ij})$ e $\mathbf{Y} = (c_{ij})$. Depois, podemos formar a equação matricial $\mathbf{Y} = \mathbf{XK}$. Se $\mathbf{X}$ tiver um inverso, podemos determinar $\mathbf{K} = \mathbf{X}^{-1}\mathbf{Y}$. Se $\mathbf{X}$ não puder ser invertida, uma nova versão de $\mathbf{X}$ terá chances de ser formada com pares adicionais de texto claro/texto cifrado, até que se obtenha uma matriz $\mathbf{X}$ a ser invertida.

Considere este exemplo. Suponha que o texto claro "*hillcipher*" seja encriptado usando uma cifra de Hill 2 × 2 para gerar o texto cifrado HCRZSSXNSP. Assim, sabemos que (7 8)**K** mod 26 = (7 2); (11 11)**K** mod 26 = (17 25); e assim por diante. Usando os dois primeiros pares de texto claro/texto cifrado, temos

$$\begin{pmatrix} 7 & 2 \\ 17 & 25 \end{pmatrix} = \begin{pmatrix} 7 & 8 \\ 11 & 11 \end{pmatrix}\mathbf{K} \bmod 26$$

O inverso de $\mathbf{X}$ pode ser calculado:

$$\begin{pmatrix} 7 & 8 \\ 11 & 11 \end{pmatrix}^{-1} = \begin{pmatrix} 25 & 22 \\ 1 & 23 \end{pmatrix}$$

de modo que

$$\mathbf{K} = \begin{pmatrix} 25 & 22 \\ 1 & 23 \end{pmatrix}\begin{pmatrix} 7 & 2 \\ 17 & 25 \end{pmatrix} = \begin{pmatrix} 549 & 600 \\ 398 & 577 \end{pmatrix} \bmod 26 = \begin{pmatrix} 3 & 2 \\ 8 & 5 \end{pmatrix}$$

Esse resultado é verificado testando-se o par restante de texto claro-texto cifrado.

Cifras polialfabéticas

Outra forma de melhorar a técnica monoalfabética simples é usar diferentes substituições monoalfabéticas enquanto se prossegue pela mensagem de texto claro. O nome geral para essa técnica é **cifra por substituição polialfabética**. Essas técnicas têm as seguintes características em comum:

1. Um conjunto de regras de substituição monoalfabéticas é utilizado.
2. Uma chave define qual regra em particular é escolhida para determinada transformação.

CIFRA DE VIGENÈRE A mais conhecida – e uma das mais simples – cifras polialfabéticas é a de Vigenère. Nesse esquema, o conjunto de regras de substituição monoalfabéticas consiste nas 26 cifras de César, com deslocamentos de 0 a 25. Cada cifra é indicada por uma letra da chave, que é a letra do texto cifrado que substitui a letra do texto claro a. Assim, uma cifra de César com um deslocamento de 3 é indicada pelo valor de chave 3.[8]

Podemos expressar a cifra de Vigenère da seguinte maneira. Considere uma sequência de letras em texto claro $P = p_0, p_1, p_2, \ldots, p_{n-1}$ e uma chave consistindo na sequência de letras $K = k_0, k_1, k_2, \ldots, k_{m-1}$, onde normalmente $m < n$. A sequência de letras em texto cifrado $C = C_0, C_1, C_2, \ldots, C_{n-1}$ é calculada da seguinte forma:

$$\begin{aligned} C = C_0, C_1, C_2, \ldots, C_{n-1} &= \mathrm{E}(K, P) = \mathrm{E}[(k_0, k_1, k_2, \ldots, k_{m-1}), (p_0, p_1, p_2, \ldots, p_{n-1})] \\ &= (p_0 + k_0) \bmod 26, (p_1 + k_1) \bmod 26, \ldots, (p_{m-1} + k_{m-1}) \bmod 26, \\ &\quad (p_m + k_0) \bmod 26, (p_{m+1} + k_1) \bmod 26, \ldots, (p_{2m-1} + k_{m-1}) \bmod 26, \ldots \end{aligned}$$

[8] Para ajudar na compreensão do esquema e auxiliar em seu uso, é possível construir uma matriz conhecida como tabela de Vigenère.

Assim, a primeira letra da chave é somada à primeira letra do texto claro, mod 26, a segunda letra da chave é somada à segunda letra do texto claro, e assim por diante, pelas primeiras *m* letras do texto claro. Para as próximas *m* letras do texto claro, as da chave são repetidas. Esse processo continua até que toda a sequência de texto claro esteja encriptada. Uma equação geral do processo de encriptação é

$$C_i = (p_i + k_{i \bmod m}) \bmod 26 \quad \textbf{(2.3)}$$

Compare isso com a Equação 2.1 para a cifra de César. Basicamente, cada caractere de texto claro é encriptado com uma cifra de César diferente, dependendo do caractere de chave correspondente. De modo semelhante, a decriptação é uma generalização da Equação 2.2:

$$p_i = (C_i - k_{i \bmod m}) \bmod 26 \quad \textbf{(2.4)}$$

Para encriptar uma mensagem, é preciso que haja uma chave tão longa quanto ela. Normalmente, a chave é uma palavra-chave repetida. Por exemplo, se a palavra-chave for "*deceptive*" ["ilusório"], a mensagem "*we are discovered save yourself*" ["fomos descobertos, salve-se"] é encriptada desta forma:

```
chave:            deceptivedeceptivedeceptive
texto claro:      wearediscoveredsaveyourself
texto cifrado:    ZICVTWQNGRZGVTWAVZHCQYGLMGJ
```

Expresso numericamente, temos o seguinte resultado:

chave	3	4	2	4	15	19	8	21	4	3	4	2	4	15
texto claro	22	4	0	17	4	3	8	18	2	14	21	4	17	4
texto cifrado	25	8	2	21	19	22	16	13	6	17	25	6	21	19

chave	19	8	21	4	3	4	2	4	15	19	8	21	4
texto claro	3	18	0	21	4	24	14	20	17	18	4	11	5
texto cifrado	22	0	21	25	7	2	16	24	6	11	12	6	9

A força dessa cifra é que existem múltiplas letras de texto cifrado para cada uma do texto claro, uma para cada letra exclusiva da palavra-chave. Assim, informações referentes à frequência de letra são ocultadas. Porém, nem todo conhecimento da estrutura do texto claro é perdido. Por exemplo, a Figura 2.6 mostra a distribuição de frequência para uma cifra de Vigenère com uma palavra-chave de tamanho 9. Uma melhoria é obtida em relação à cifra Playfair, mas informações de frequência consideráveis ainda permanecem.

É instrutivo esboçar um método para quebrar essa cifra, pois ele revela alguns dos princípios matemáticos que se aplicam na criptoanálise.

Primeiro, suponha que o oponente acredite que o texto cifrado foi encriptado ou usando a substituição monoalfabética, ou uma cifra de Vigenère. Um teste simples pode ser feito para se distinguir esses dois cenários. Se uma substituição monoalfabética for empregada, então as propriedades estatísticas do texto cifrado deverão ser iguais às da linguagem do texto claro. Assim, referenciando a Figura 2.5, deverá haver uma letra de cifra com uma frequência relativa de ocorrência de aproximadamente 12,7%, uma com cerca de 9,06%, e assim por diante. Se apenas uma única mensagem estiver disponível para análise, não esperaríamos encontrar uma combinação exata do perfil estatístico da linguagem de texto claro. Entretanto, se a correspondência for próxima, podemos considerar uma substituição monoalfabética.

Se, por outro lado, uma cifra de Vigenère for utilizada, então o progresso da criptoanálise depende da determinação do tamanho da palavra-chave. Agora, nos concentraremos em como determinar o tamanho da palavra-chave. A ideia principal que leva a determinar o tamanho da palavra-chave é a seguinte: se duas sequências idênticas de letras de texto claro ocorrem a uma distância que seja um múltiplo inteiro do tamanho da palavra-chave, elas gerarão sequências de texto cifrado idênticas. No exemplo anterior, duas ocorrências da sequência "*red*" são separadas por nove posições de caractere. Consequentemente, em ambos os casos, r é codificado com a letra-chave *e*, e é codificado com a letra-chave *p*, e d é codificado com a letra-chave *t*. Assim, nos dois casos, a sequência é VTW. Indicamos isso anteriormente sublinhando as letras relevantes e sombreando os números relevantes do texto cifrado.

Um analista examinando apenas o texto cifrado detectaria as sequências repetidas VTW a uma distância de 9 e consideraria que a palavra-chave tem três ou nove letras de extensão. O surgimento de VTW duas vezes poderia ser por acaso, e não refletir letras de texto claro idênticas codificadas com letras-chave idênticas. Porém, se a mensagem for longa o suficiente, haverá diversas dessas sequências de texto cifrado repetidas. Procurando fatores comuns nos deslocamentos das diversas sequências, o analista deverá ser capaz de fazer uma boa escolha do tamanho da palavra-chave.

A solução da cifra agora depende de uma percepção importante. Se o tamanho da palavra-chave é *m*, então a cifra, por conseguinte, consiste em *m* cifras de substituição monoalfabéticas. Por exemplo, com a palavra-chave DECEPTIVE, as letras nas posições 1, 10, 19, e assim por diante, são todas encriptadas com a mesma cifra monoalfabética. Assim, podemos usar as características de frequência conhecidas da linguagem do texto claro para atacar cada uma das cifras monoalfabéticas separadamente.

A natureza periódica da palavra-chave pode ser eliminada usando-se uma palavra-chave não repetida que seja tão grande quanto a própria mensagem. Vigenère propôs o que é conhecido como um **sistema de auto-chave**, em que uma palavra-chave é concatenada ao próprio texto claro para oferecer uma chave corrente. Para o nosso exemplo,

```
chave:            deceptivewearediscoveredsav
texto claro:      wearediscoveredsaveyourself
texto cifrado:    ZICVTWQNGKZEIIGASXSTSLVVWLA
```

Até mesmo esse esquema é vulnerável à criptoanálise. Como a chave e o texto claro compartilham a mesma distribuição de frequência das letras, uma técnica estatística poderá ser aplicada. Por exemplo, pode-se esperar que e codificado por *e*, pela Figura 2.5, ocorra com uma frequência de $(0{,}127)^2 \approx 0{,}016$, enquanto t codificado por *t* ocorreria apenas com a metade dessa frequência. Essas regularidades podem ser exploradas para se conseguir sucesso na criptoanálise.[9]

CIFRA DE VERNAM A principal defesa contra a técnica criptoanalítica descrita é escolher uma palavra-chave que seja tão longa quanto o texto claro e que não possua relacionamento estatístico com ele. Esse sistema foi introduzido por um engenheiro da AT&T, chamado Gilbert Vernam, em 1918. Seu sistema funciona sobre dados binários (bits), em vez de letras. O sistema pode ser expresso de forma sucinta da seguinte forma (Figura 2.7):

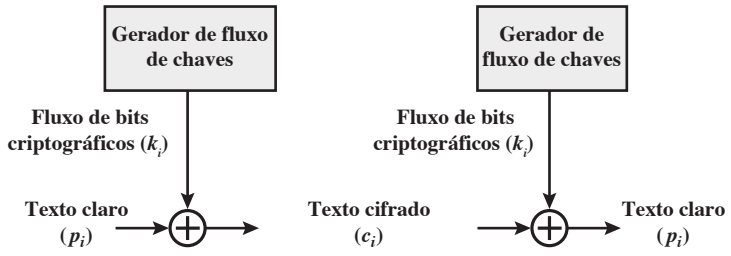
$$c_i = p_i \oplus k_i$$

onde

p_i = dígito binário na posição *i* do texto claro
k_i = dígito binário na posição *i* da chave
c_i = dígito binário na posição *i* do texto cifrado
$\oplus$ = operação ou – exclusivo (XOR)

Compare isso com a Equação 2.3, para a cifra de Vigenère.

Figura 2.7 Cifra de Vernam.



[9] Embora as técnicas para quebrar uma cifra de Vigenère não sejam complexas, uma edição de 1917 da *Scientific American* caracterizava esse sistema como "impossível de tradução". Esse é um ponto que merece ser lembrado quando afirmações semelhantes são feitas para algoritmos modernos.

Assim, o texto cifrado é gerado realizando-se o XOR (operação lógica ou-exclusivo) bit a bit entre texto claro e a chave. Por conta das propriedades do XOR, a decriptação simplesmente envolve a mesma operação bit a bit:

$$p_i = c_i \oplus k_i$$

que é comparada à Equação 2.4.

A essência dessa técnica é a forma de construção da chave. Vernam propôs o uso de uma palavra-chave muito longa, porém que eventualmente era repetida. Embora esse esquema com uma chave longa apresente dificuldades de criptoanálise formidáveis, ele pode ser quebrado com um número suficiente de texto cifrado, com o uso de sequências de texto claro conhecidas ou prováveis, ou ambos.

One-time pad

Um oficial do Exército, Joseph Mauborgne, propôs uma melhoria na cifra de Vernam, que gera o máximo em segurança. Mauborgne sugeriu o uso de uma chave aleatória que fosse tão grande quanto a mensagem, de modo que a chave não precisasse ser repetida. Além disso, a chave deve ser empregada para encriptar e decriptar uma única mensagem, e depois descartada. Cada nova mensagem exige uma nova chave com o mesmo tamanho. Esse esquema, conhecido como ***one-time pad***, é inquebrável. Ele produz saída aleatória que não possui qualquer relacionamento estatístico com o texto claro. Como o texto cifrado não contém qualquer informação sobre o texto claro, simplesmente não existe um meio de quebrar o código.

Um exemplo deverá ilustrar nosso argumento. Suponha que estejamos usando um esquema de Vigenère com 27 caracteres, no qual o vigésimo sétimo é o de espaço, mas com uma chave de uso único que é tão grande quanto a mensagem. Considere o texto cifrado

```
ANKYODKYUREPFJBYOJDSPLREYIUNOFDOIUERFPLUYTS
```

Agora mostramos duas decriptações diferentes usando duas chaves distintas:

```
texto cifrado:    ANKYODKYUREPFJBYOJDSPLREYIUNOFDOIUERFPLUYTS
chave:            pxlmvmsydofuyrvzwc tnlebnecvgdupahfzzlmnyih
texto claro:*     mr mustard with the candlestick in the hall

texto cifrado:    ANKYODKYUREPFJBYOJDSPLREYIUNOFDOIUERFPLUYTS
chave:            mfugpmiydgaxgoufhkllmhsqdqogtewbqfgyovuhwt
texto claro:**    miss scarlet with the knife in the library
```

Suponha que um criptoanalista conseguisse encontrar essas duas chaves. Dois textos claros plausíveis são produzidos. Como ele decide qual é a decriptação correta (ou seja, qual é a chave correta)? Se a chave real fosse produzida de uma forma verdadeiramente aleatória, então o criptoanalista não pode saber que uma dessas duas é mais provável que a outra. Assim, não existe um meio de decidir qual chave é a correta e, por consequência, qual texto claro é o correto.

De fato, dado qualquer texto claro de mesmo tamanho do texto cifrado, existe uma chave que o produz. Portanto, se você fizesse uma busca exaustiva em todas as chaves possíveis, acabaria com muitos textos claros legíveis, sem saber qual foi o intencionado. O código é inquebrável.

A segurança do *one-time pad* é inteiramente decorrente da aleatoriedade da chave. Se o fluxo de caracteres que constitui a chave for verdadeiramente aleatório, então o de caracteres que constitui o texto cifrado também o será. Assim, não existem padrões ou regularidades que um criptoanalista possa usar para atacar o texto cifrado.

Em teoria, não precisamos mais procurar uma cifra. O *one-time pad* oferece segurança completa, mas, na prática, tem dois empecilhos fundamentais:

1. Criar grandes quantidades de chaves aleatórias representa um problema prático. Qualquer sistema bastante utilizado poderia exigir milhões de caracteres aleatórios regularmente. O fornecimento de caracteres de fato aleatórios nesse volume é uma tarefa significativa.
2. Ainda mais assustador é o problema da distribuição e proteção da chave. A cada mensagem a ser enviada, uma chave de mesmo tamanho é necessária para uso do emissor e do receptor. Assim, existe um problema gigantesco de distribuição de chave.

---

* "Senhor Mustard com o candelabro no salão".
** "Senhorita Scarlet com a faca na biblioteca".

Por causa dessas dificuldades, o *one-time pad* tem utilidade limitada, e aplicação principalmente para canais de pouca largura de banda que exigem segurança muito alta.

O *one-time pad* é o único criptossistema que apresenta o que é conhecido como *segredo perfeito*. Esse conceito é explorado no Apêndice F (disponível na Sala Virtual, em <sv.pearson.com.br>, em inglês).

2.3 TÉCNICAS DE TRANSPOSIÇÃO

Todas as técnicas examinadas até aqui envolvem a substituição de um símbolo de texto cifrado por um de texto claro. Uma espécie bem diferente de mapeamento é obtida realizando-se algum tipo de permutação nas letras do texto claro. Essa técnica é referenciada como uma cifra de transposição.

A cifra mais simples desse tipo é a técnica de **cerca de trilho**, em que o texto claro é escrito como uma sequência de diagonais, e depois lido como uma sequência de linhas. Por exemplo, para cifrar a mensagem "*meet me after the toga party*" com uma cerca de trilho de profundidade 2, escrevemos o seguinte:

```
m e m a t r h t g p r y
 e t e f e t e o a a t
```

A mensagem encriptada é

MEMATRHTGPRYETEFETEOAAT

Esse tipo de coisa seria trivial de ser criptanalisada. Um esquema mais complexo é escrever a mensagem em um retângulo, linha por linha, e a ler coluna por coluna, mas permutar a ordem destas. A ordem das colunas, então, torna-se a chave para o algoritmo. Por exemplo,

```
Chave:          4 3 1 2 5 6 7
Texto claro:    a t t a c k p
                o s t p o n e
                d u n t i l t
                w o a m x y z
Texto cifrado:  TTNAAPTMTSUOAODWCOIXKNLYPETZ
```

Assim, neste exemplo, a chave é 4312567. Para encriptar, comece com a coluna rotulada com 1, neste caso, a coluna 3. Escreva todas as letras dessa coluna. Prossiga para a coluna 4, que é rotulada com 2, depois para a coluna 2, então para a coluna 1, por fim para as colunas 5, 6 e 7.

Uma cifra de pura transposição é facilmente reconhecida, pois tem as mesmas frequências de letra do texto claro original. Para o tipo de transposição de colunas mostrada, a criptoanálise é muito simples e envolve dispor o texto cifrado em uma matriz, além de mexer com as posições de coluna. As tabelas de frequência de digrama e trigrama podem ser úteis.

A cifra de transposição pode se tornar muito mais segura realizando-se mais de um estágio de transposição. O resultado é uma permutação mais complexa, que não é facilmente reconstruída. Assim, se a mensagem anterior for reencriptada usando o mesmo algoritmo,

```
Chave:    4 3 1 2 5 6 7
Entrada:  t t n a a p t
          m t s u o a o
          d w c o i x k
          n l y p e t z
Saída:    NSCYAUOPTTWLTMDNAOIEPAXTTOKZ
```

Para visualizar o resultado dessa dupla transposição, designe as letras na mensagem de texto claro original pelos números que indicam a sua posição. Assim, com 28 letras na mensagem, a sequência original das letras é

```
01 02 03 04 05 06 07 08 09 10 11 12 13 14
15 16 17 18 19 20 21 22 23 24 25 26 27 28
```

Depois da primeira transposição, temos

```
03 10 17 24 04 11 18 25 02 09 16 23 01 08
15 22 05 12 19 26 06 13 20 27 07 14 21 28
```

que tem uma estrutura um tanto regular. Mas, depois da segunda transposição, temos

```
17 09 05 27 24 16 12 07 10 02 22 20 03 25
15 13 04 23 19 14 11 01 26 21 18 08 06 28
```

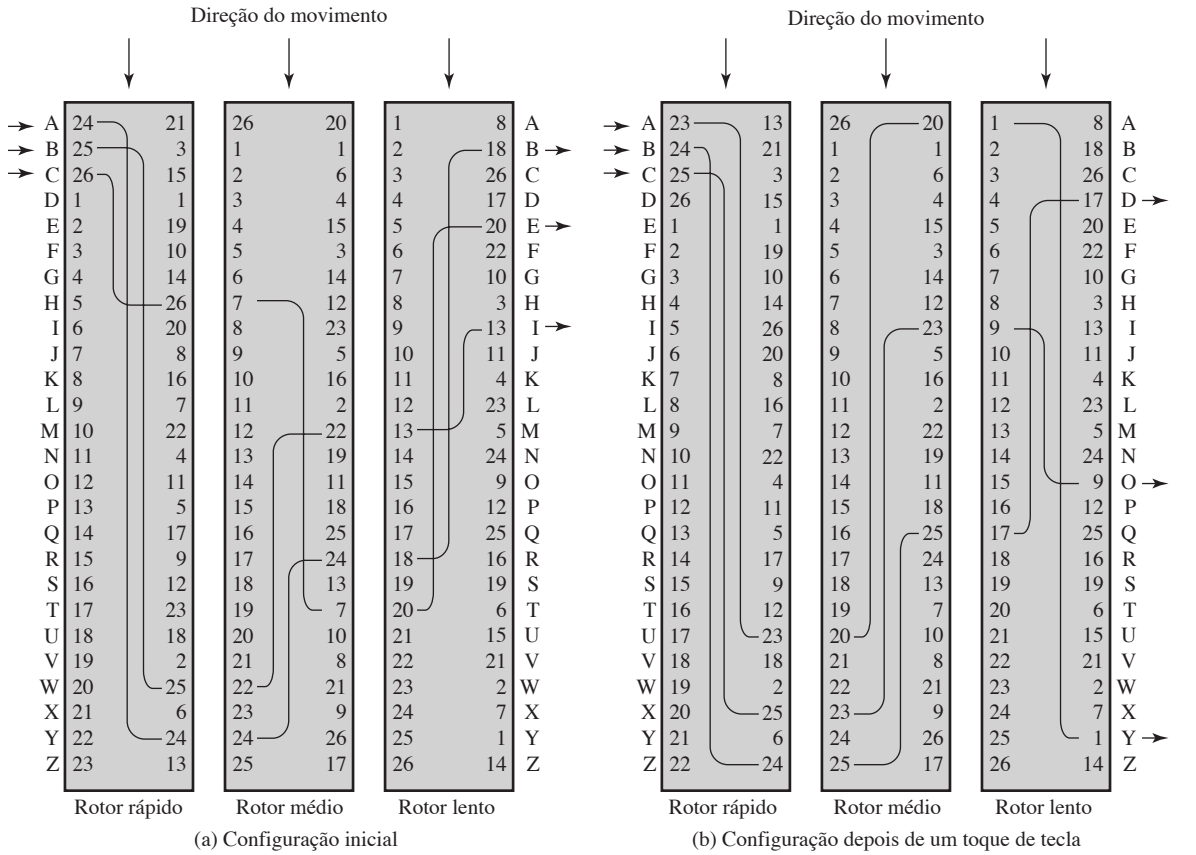
Essa é uma permutação muito menos estruturada e muito mais difícil de se criptoanalisar.

2.4 MÁQUINAS DE ROTOR

O exemplo que acabamos de dar sugere que várias etapas de encriptação podem produzir um algoritmo que é significativamente mais difícil para criptoanalisar. Isso vale tanto para cifras de substituição quanto para de transposição. Antes da introdução do DES, a aplicação mais importante do princípio de múltiplas etapas de encriptação era uma classe de sistemas conhecida como máquinas de rotor.[10]

O princípio básico da máquina de rotor é ilustrado na Figura 2.8. A máquina consiste em um conjunto de cilindros rotativos independentes, através dos quais pulsos elétricos podem fluir. Cada cilindro tem 26 pinos de entrada e 26 pinos de saída, com fiação interna que conecta cada pino de entrada a um único pino de saída. Para simplificar, mostramos somente três das conexões internas em cada cilindro.

Figura 2.8 Máquina de três rotores com fiação representada por contatos numerados.



[10] Máquinas baseadas no princípio de rotor foram usadas pela Alemanha (Enigma) e pelo Japão (Purple) na Segunda Guerra Mundial. A quebra desses dois códigos pelos Aliados foi um fator significativo para o resultado da guerra.

Se associarmos cada pino de entrada e saída a uma letra do alfabeto, então um único cilindro define uma substituição monoalfabética. Por exemplo, na Figura 2.8, se um operador pressionar uma tecla para a letra A, um sinal elétrico é aplicado ao primeiro pino do primeiro cilindro e flui pela conexão interna para o vigésimo quinto pino de saída.

Considere uma máquina com um único cilindro. Depois que cada tecla de entrada é pressionada, o cilindro gira uma posição, de modo que as conexões internas são deslocadas de acordo. Assim, é definida uma cifra de substituição monoalfabética diferente. Depois de 26 letras de texto claro, o cilindro estaria de volta à posição inicial. Assim, temos um algoritmo de substituição polialfabética com um período de 26.

Um sistema de único cilindro é trivial e não apresenta uma tarefa criptoanalítica formidável. O poder da máquina de rotor está no uso de múltiplos cilindros, em que os pinos de saída de um cilindro são conectados aos de entrada do seguinte. A Figura 2.8 mostra um sistema de três cilindros. A metade esquerda da figura mostra uma posição em que a entrada do operador para o primeiro pino (letra a em texto claro) é direcionada pelos três cilindros para aparecer na saída do segundo pino (letra B em texto cifrado).

Com múltiplos cilindros, aquele mais próximo da entrada do operador gira uma posição de pino a cada toque de tecla. A metade direita da Figura 2.8 mostra a configuração do sistema depois de um único toque de tecla. Para cada rotação completa do cilindro interno, o do meio gira uma posição de pino. Finalmente, para cada rotação completa do cilindro do meio, o externo gira uma posição de pino. Esse é o mesmo tipo de operação vista com os antigos marcadores de quilometragem (odômetros) de automóvel. O resultado é que existem $26 \times 26 \times 26 = 17.576$ alfabetos de substituição diferentes usados antes que o sistema repita. O acréscimo de quarto e quinto rotores resulta em períodos de 456.976 e 11.881.376 letras, respectivamente. Assim, determinada configuração de uma máquina de 5 rotores é equivalente a uma cifra de Vigenère com um tamanho de chave de 11.881.376.

Esse esquema apresenta um desafio criptoanalítico formidável. Por exemplo, se o criptoanalista tentar usar uma técnica de análise de frequência de letra, ele enfrentará o equivalente a mais de 11 milhões de cifras monoalfabéticas. Talvez fosse preciso cerca de 50 letras em cada cifra monoalfabética para uma solução, o que significa que o analista necessita estar em posse de um texto cifrado com um tamanho de mais de meio bilhão de letras.

O significado da máquina de rotor hoje é que ela aponta o caminho para a cifra mais utilizada de todos os tempos: Data Encryption Standard (DES), que será explicada no Capítulo 3.

2.5 ESTEGANOGRAFIA

Concluiremos com uma discussão sobre uma técnica que, estritamente falando, não é encriptação — a saber, a **esteganografia**.

Uma mensagem em texto claro pode estar oculta de duas maneiras. Os métodos de **esteganografia** escondem a existência da mensagem, enquanto os métodos de criptografia a tornam ininteligível a estranhos por meio de várias transformações do texto.[11]

Uma forma simples de esteganografia, mas demorada de se construir, é aquela em que um arranjo de palavras e letras dentro de um texto aparentemente inofensivo soletra a mensagem real. Por exemplo, a sequência de primeiras letras de cada palavra da mensagem geral soletra a mensagem escondida.

Diversas outras técnicas têm sido usadas historicamente, e alguns exemplos são [MYER91]:

- **Marcação de caractere:** letras selecionadas do texto impresso ou datilografado são escritas com lápis por cima. As marcas normalmente não são visíveis, a menos que o papel seja mantido contra uma fonte de luz clara.
- **Tinta invisível:** diversas substâncias podem ser usadas para a escrita sem deixar rastros visíveis, a menos que alguma química seja aplicada ao papel.
- **Perfurações:** pequenos furos em letras selecionadas normalmente não são visíveis, a menos que o papel tenha uma fonte de luz no fundo.
- **Fita corretiva de máquina de escrever:** usada entre as linhas digitadas com uma fita preta, os resultados de digitar com a fita corretiva são visíveis apenas sob uma luz forte.

[11] *Esteganografia* era uma palavra obsoleta que foi revivida por David Kahn e recebeu o significado que tem hoje [KAHN96].

Embora essas técnicas possam parecer arcaicas, elas possuem equivalentes contemporâneos. [WAYN09] propõe esconder uma mensagem usando os bits menos significativos dos frames em um CD. Por exemplo, a resolução máxima do formato Kodak Photo CD é de 3.096 por 6.144 pixels, com cada pixel contendo 24 bits de informações de cor RGB. O bit menos significativo de cada pixel de 24 bits pode ser alterado sem afetar muito a qualidade da imagem. O resultado é que você pode ocultar uma mensagem de 130 kB em uma única foto digital. Agora, existem diversos pacotes de software disponíveis que levam esse tipo de técnica à esteganografia.

A esteganografia tem diversas desvantagens quando comparada à encriptação. Ela exige muito *overhead* para esconder relativamente poucos bits de informação, embora algum esquema como o proposto no parágrafo anterior possa torná-la mais eficaz. Além disso, quando o sistema é descoberto, ele se torna praticamente inútil. Esse problema também pode ser contornado se o método de inserção depender de algum tipo de chave (por exemplo, veja o Problema 2.20). Como alternativa, uma mensagem pode ser primeiramente encriptada, e depois escondida, usando a esteganografia.

A vantagem da esteganografia é que ela pode ser empregada pelas partes que têm algo a perder se a sua comunicação secreta (não necessariamente o conteúdo) for descoberta. A encriptação sinaliza o tráfego como importante ou secreto, ou pode identificar o emissor ou o receptor como alguém com algo a esconder.

2.6 LEITURA RECOMENDADA

Para quem estiver interessado na história da criação e quebra de código, o livro a ser lido é [KAHN96]. Embora trate mais do impacto da criptologia do que de seu desenvolvimento técnico, essa é uma introdução excelente e compõe uma leitura interessante. Outro ótimo relato histórico é [SING99].

Uma abordagem curta incluindo as técnicas deste capítulo, além de outras, é [GARD72]. Existem muitos livros que abordam a criptografia clássica em um estilo mais técnico; um dos melhores é [SINK09]. [KORN96] é um livro agradável de se ler e contém uma extensa seção sobre técnicas clássicas. Dois livros de criptografia que contêm uma grande quantidade de material sobre técnicas clássicas são [GARR01] e [NICH99]. Para o leitor verdadeiramente interessado, os dois volumes de [NICH96] abrangem diversas cifras clássicas com detalhes, e oferecem muitos textos cifrados para serem criptoanalisados, com as soluções.

Um tratamento excelente das máquinas de rotor, incluindo uma discussão sobre sua criptoanálise, pode ser encontrado em [KUMA97].

[KATZ00] oferece um tratamento completo da esteganografia. Outra fonte interessante é [WAYN09].

GARD72 GARDNER, M. *Codes, Ciphers, and Secret Writing*. Nova York: Dover, 1972.

GARR01 GARRETT, P. *Making, Breaking Codes:* An Introduction to Cryptology. Upper Saddle River, NJ: Prentice Hall, 2001.

KAHN96 KAHN, D. *The Codebreakers:* The Story of Secret Writing. Nova York: Scribner, 1996.

KATZ00 KATZENBEISSER, S. (ed.). *Information Hiding Techniques for Steganography and Digital Watermarking*. Boston: Artech House, 2000.

KORN96 KORNER, T. *The Pleasures of Counting*. Cambridge, Inglaterra: Cambridge University Press, 1996.

KUMA97 KUMAR, I. *Cryptology*. Laguna Hills, CA: Aegean Park Press, 1997.

NICH96 NICHOLS, R. *Classical Cryptography Course*. Laguna Hills, CA: Aegean Park Press, 1996.

NICH99 NICHOLS, R. (ed.). *ICSA Guide to Cryptography*. Nova York: McGraw-Hill, 1999.

SING99 SINGH, S. *The Code Book:* The Science of Secrecy from Ancient Egypt to Quantum Cryptography. Nova York: Anchor Books, 1999.

SINK09 SINKOV, A. *Elementary Cryptanalysis:* A Mathematical Approach. Washington, DC: The Mathematical Association of America, 2009.

WAYN09 WAYNER, P. *Disappearing Cryptography*. Boston: AP Professional Books, 2009.

2.7 PRINCIPAIS TERMOS, PERGUNTAS PARA REVISÃO E PROBLEMAS

Principais termos

ataque de força bruta
cifra
cifra cerca de trilho
cifra de bloco
cifra de César
cifra de fluxo
cifra de Hill
cifra de transposição
cifra de Vigenère
cifra monoalfabética
cifra Playfair
cifra polialfabética
cifragem
computacionalmente seguro
criptoanálise
criptografia
criptologia
decifragem
decriptação
digrama
encriptação
encriptação convencional
encriptação de chave única
encriptação simétrica
esteganografia
incondicionalmente seguro
one-time pad
sistema criptográfico
texto cifrado
texto claro

Perguntas para revisão

2.1 Quais são os elementos essenciais de uma cifra simétrica?
2.2 Quais são as duas funções básicas usadas nos algoritmos de encriptação?
2.3 Quantas chaves são necessárias para duas pessoas se comunicarem por meio de uma cifra?
2.4 Qual é a diferença entre uma cifra de bloco e uma cifra de fluxo?
2.5 Quais são as duas técnicas gerais para atacar uma cifra?
2.6 Liste e defina rapidamente os tipos de ataque criptoanalítico com base naquilo que o atacante conhece.
2.7 Qual é a diferença entre uma cifra incondicionalmente segura e uma cifra computacionalmente segura?
2.8 Defina resumidamente a cifra de César.
2.9 Defina resumidamente a cifra monoalfabética.
2.10 Defina resumidamente a cifra Playfair.
2.11 Qual é a diferença entre uma cifra monoalfabética e uma polialfabética?
2.12 Quais são os dois problemas com o *one-time pad*?
2.13 O que é uma cifra de transposição?
2.14 O que é esteganografia?

Problemas

2.1 Uma generalização da cifra de César, conhecida como cifra de César afim, tem a seguinte forma: a cada letra de texto claro p, substitua-a pela letra de texto cifrado C:

$$C = \mathrm{E}([a, b], p) = (ap + b) \bmod 26$$

Um requisito básico de qualquer algoritmo de encriptação é que ele seja um para um. Ou seja, se $p \neq q$, então $\mathrm{E}(k, p) \neq \mathrm{E}(k, q)$. Caso contrário, a decriptação é impossível, pois mais de um caractere de texto claro é mapeado no mesmo caractere de texto cifrado. A cifra de César afim não é um-para-um para todos os valores de a. Por exemplo, para $a = 2$ e $b = 3$, então $\mathrm{E}([a, b], 0) = \mathrm{E}([a, b], 13) = 3$.

a. Existem limitações sobre o valor de b? Explique por que sim ou por que não.
b. Determine quais valores de a não são permitidos.
c. Ofereça uma afirmação geral sobre quais valores de a são e não são permitidos. Justifique-a.

2.2 Quantas cifras de César afins um para um existem?

2.3 Um texto cifrado foi gerado com uma cifra afim. A letra mais frequente do texto cifrado é B, e a segunda mais frequente é U. Quebre esse código.

2.4 O texto cifrado a seguir foi gerado usando um algoritmo de substituição simples:

```
53‡‡†305))6*;4826)4‡.)4‡);806*;48†8¶60))85;;]8*;:‡*8†83
(88)5*†;46(;88*96*?;8)*‡(;485);5*†2:*‡(;4956*2(5*-4)8¶8*
;4069285);)6†8)4‡‡;1(‡9;48081;8:8‡1;48†85;4)485†528806*81
(‡9;48;(88;4(‡?34;48)4‡;161;:188;‡?;
```

Decripte essa mensagem.
Dicas:

1. A letra que ocorre com mais frequência em inglês é e. Portanto, o primeiro ou segundo (ou talvez terceiro?) caractere mais comum na mensagem provavelmente signifique e. Além disso, e normalmente é visto em pares (por exemplo, *meet*, *fleet*, *speed*, *seen*, *been*, *agree* etc.). Tente encontrar um caractere no texto cifrado que seja decriptado para e.
2. A palavra mais comum em inglês é *the*. Use esse fato para adivinhar os caracteres que representam t e h.
3. Decifre o restante da mensagem deduzindo outras palavras.

Aviso: a mensagem resultante está em inglês, mas pode não fazer muito sentido em uma primeira leitura.

2.5 Um modo de solucionar o problema de distribuição de chave é usar uma linha de um livro que o emissor e o receptor possuem. Normalmente, pelo menos em romances de espionagem, a primeira sentença de um livro serve como chave. O esquema em particular discutido neste problema é de um dos melhores romances de suspense envolvendo códigos secretos, *Talking to Strange Men* (*Falando com Homens Estranhos*), de Ruth Rendell. Discuta este problema sem consultar esse livro!
Considere a mensagem a seguir:

SIDKHKDM AF HCRKIABIE SHIMC KD LFEAILA

Esse texto cifrado foi produzido usando-se a primeira sentença de *The Other Side of Silence* (*O Outro Lado do Silêncio* – um livro sobre o espião Kim Philby):

> The snow lay thick on the steps and the snowflakes driven by the wind looked black in the headlights of the cars.

Uma cifra de substituição simples foi utilizada.

- **a.** Qual é o algoritmo de encriptação?
- **b.** Qual a sua segurança?
- **c.** Para simplificar o problema de distribuição de chave, as duas partes podem combinar em usar a primeira ou última sentença de um livro como chave. Para mudá-la, eles simplesmente precisam escolher um novo livro. O uso da primeira sentença seria preferível ao da última. Por quê?

2.6 Em um de seus casos, Sherlock Holmes foi confrontado com a seguinte mensagem:

534 C2 13 127 36 31 4 17 21 41
DOUGLAS 109 293 5 37 BIRLSTONE
26 BIRLSTONE 9 127 171

Embora Watson estivesse confuso, Holmes foi imediatamente capaz de deduzir o tipo de cifra. Você consegue descobri-lo?

2.7 Este problema usa um exemplo do mundo real, de um antigo manual das Forças Especiais dos Estados Unidos (domínio público). O documento, com nome de arquivo *SpecialForces.pdf*, está disponível na Sala Virtual para este livro.

- **a.** Usando as duas chaves (palavras de memória) *cryptographic* e *network security*, encripte a mensagem a seguir:

> Be at the third pillar from the left outside the lyceum theatre tonight at seven. If you are distrustful bring two friends.

Faça suposições razoáveis sobre como tratar letras redundantes e letras em excesso nas palavras de memória e como tratar os espaços e a pontuação. Indique quais são as suas suposições. *Nota:* a mensagem é do romance de Sherlock Holmes *The Sign of Four* (*O Sinal dos Quatro*).

- **b.** Decripte o texto cifrado. Mostre o seu trabalho.
- **c.** Comente sobre quando seria apropriado usar esta técnica e quais são suas vantagens.

2.8 Uma desvantagem da cifra monoalfabética é que tanto o emissor quanto o receptor precisam confiar a sequência da cifra permutada à memória. Uma técnica comum para evitar isso é usar uma palavra-chave da qual a sequência da cifra possa ser gerada. Por exemplo, utilizando a palavra-chave *CIPHER*, escreva-a seguida por letras não usadas na ordem normal e combine com as do texto claro:

```
texto claro:    a b c d e f g h i j k l m n o p q r s t u v w x y z
texto cifrado:  C I P H E R A B D F G J K L M N O Q S T U V W X Y Z
```

Se esse processo não produzir uma mistura suficiente, escreva as letras restantes em linhas sucessivas e depois gere a sequência lendo as colunas, de cima para baixo:

```
C I P H E R
A B D F G J
K L M N O Q
S T U V W X
Y Z
```

Isso resulta na sequência

```
C A K S Y I B L T Z P D M U H F N V E G O W R J Q X
```

Esse sistema é usado no exemplo da Seção 2.2 (aquele que começa com "it was disclosed yesterday"). Determine a palavra-chave.

2.9 Quando o barco de patrulha norte-americano PT-109, sob o comando do tenente John F. Kennedy, foi afundado por um destróier japonês, uma mensagem foi recebida na estação sem fio australiana em código Playfair:

```
KXJEY UREBE ZWEHE WRYTU HEYFS
KREHE GOYFI WTTTU OLKSY CAJPO
BOTEI ZONTX BYBNT GONEY CUZWR
GDSON SXBOU YWRHE BAAHY USEDQ
```

A chave usada foi *royal new zealand navy*. Decripte a mensagem. Traduza TT para tt.

2.10 **a.** Construa uma matriz Playfair com a chave *largest*.

b. Construa uma matriz Playfair com a chave *occurrence*. Faça uma suposição razoável sobre como tratar letras redundantes na chave.

2.11 **a.** Usando esta matriz Playfair

M	F	H	I/J	K
U	N	O	P	Q
Z	V	W	X	Y
E	L	A	R	G
D	S	T	B	C

encripte esta mensagem:

Must see you over Cadogan West. Coming at once.

Nota: a mensagem é da história de Sherlock Holmes, *The Adventure of the Bruce-Partington Plans (A Aventura dos Planos de Bruce-Partington).*

b. Repita a parte (a) usando a matriz Playfair do Problema 2.10a.

c. Como você justifica os resultados desse problema? Você poderia generalizar sua conclusão?

2.12 **a.** Quantas chaves possíveis a cifra Playfair possui? Ignore o fato de que algumas chaves poderiam produzir resultados de encriptação idênticos. Expresse sua resposta como uma potência aproximada de 2.

b. Agora leve em conta o fato de que algumas chaves Playfair produzem os mesmos resultados de encriptação. Quantas chaves efetivamente exclusivas a cifra Playfair possui?

2.13 Que sistema de substituição resulta quando usamos uma matriz Playfair de 25 × 1?

2.14 **a.** Encripte a mensagem "meet me at the usual place at ten rather than eight oclock" usando a cifra de Hill com a chave $\begin{pmatrix} 9 & 4 \\ 5 & 7 \end{pmatrix}$. Mostre seus cálculos e o resultado.

b. Mostre os cálculos para a decriptação correspondente do texto cifrado a fim de recuperar o texto claro original.

2.15 Mostramos que a cifra de Hill sucumbe perante um ataque de texto claro conhecido se forem fornecidos suficientes pares de texto claro/texto cifrado. É ainda mais fácil de solucionar a cifra de Hill se um ataque de texto claro escolhido puder ser montado. Descreva esse ataque.

2.16 Pode-se mostrar que a cifra de Hill com a matriz $\begin{pmatrix} a & b \\ c & d \end{pmatrix}$ exige que $(ad - bc)$ seja relativamente primo de 26; ou seja, o único fator positivo comum entre $(ad - bc)$ e 26 é 1. Assim, se $(ad - bc) = 13$ ou se for par, a matriz não é permitida. Determine o número de chaves diferentes (boas) que existem para uma cifra de Hill 2 × 2 sem contá-las uma a uma, usando as seguintes etapas:

a. Descubra o número de matrizes cujo determinante seja par porque uma ou ambas as linhas são pares. (Uma linha é "par" se as duas entradas na linha forem pares.)

b. Descubra o número de matrizes cujo determinante é par porque uma ou ambas as colunas são pares. (Uma coluna é "par" se as duas entradas na coluna forem pares.)
c. Descubra o número de matrizes cujo determinante é par porque todas as entradas são ímpares.
d. Levando em conta as sobreposições, descubra o número total de matrizes cujo determinante é par.
e. Descubra o número de matrizes cujo determinante é um múltiplo de 13 porque a primeira coluna é um múltiplo de 13.
f. Descubra o número de matrizes cujo determinante é um múltiplo de 13, em que a primeira coluna não é um múltiplo de 13, mas a segunda é um múltiplo do primeiro módulo 13.
g. Descubra o número total de matrizes cujo determinante é um múltiplo de 13.
h. Descubra o número de matrizes cujo determinante é um múltiplo de 26 porque elas se ajustam aos casos (a) e (e), (b) e (e), (c) e (e), (a) e (f), e assim por diante.
i. Descubra o número total de matrizes cujo determinante não é um múltiplo de 2 nem um múltiplo de 13.

2.17 Calcule o determinante mod 26 de

a. $\begin{pmatrix} 20 & 2 \\ 5 & 4 \end{pmatrix}$ **b.** $\begin{pmatrix} 1 & 7 & 22 \\ 4 & 9 & 2 \\ 1 & 2 & 5 \end{pmatrix}$

2.18 Determine o inverso mod 26 de

a. $\begin{pmatrix} 2 & 3 \\ 1 & 22 \end{pmatrix}$ **b.** $\begin{pmatrix} 6 & 24 & 1 \\ 13 & 16 & 10 \\ 20 & 17 & 15 \end{pmatrix}$

2.19 Usando a cifra de Vigenère, encripte a palavra "explanation" usando a chave *leg*.

2.20 Este problema explora o uso de uma versão do *one-time pad* da cifra de Vigenère. Nesse esquema, a chave é um fluxo de números aleatórios entre 0 e 26. Por exemplo, se a chave for 3 19 5..., então a primeira letra do texto claro é encriptada com um deslocamento de três letras, a segunda com um deslocamento de 19 letras, a terceira com um deslocamento de cinco letras, e assim por diante.

a. Encripte o texto claro *sendmoremoney* com o fluxo de chaves

```
9 0 1 7 23 15 21 14 11 11 2 8 9
```

b. Usando o texto cifrado produzido na parte a, encontre uma chave, de modo que o texto cifrado seja decriptado para o texto claro *cashnotneeded*.

Problemas de programação

2.22 Elabore um programa que possa encriptar e decriptar usando a cifra de César geral, também conhecida como cifra aditiva.

2.23 Elabore um programa que possa encriptar e decriptar usando a cifra afim, descrita no Problema 2.1.

2.24 Elabore um programa que possa realizar um ataque de frequência de letra em uma cifra aditiva sem intervenção humana. Seu software deverá produzir textos claros possíveis em ordem aproximada de probabilidade. Seria bom se a sua interface com o usuário permitisse que ele especificasse "mostre os 10 textos claros mais prováveis".

2.25 Escreva um programa que possa realizar um ataque de frequência de letra em qualquer cifra de substituição monoalfabética sem intervenção humana. Seu software deverá produzir textos claros possíveis em ordem aproximada de probabilidade. Seria bom se a sua interface com o usuário permitisse que ele especificasse "mostre os 10 textos claros mais prováveis".

2.26 Crie um software que possa encriptar e decriptar usando uma cifra de Hill 2 × 2.

2.27 Crie um software que possa realizar um ataque rápido de texto claro conhecido sobre uma cifra de Hill, dada a dimensão m. Qual a velocidade dos seus algoritmos, como uma função de m?

Cifras de bloco e o data encryption standard

03

TÓPICOS ABORDADOS

OBJETIVOS DE APRENDIZAGEM

APÓS ESTUDAR ESTE CAPÍTULO, VOCÊ SERÁ CAPAZ DE:

- ☑ Entender a distinção entre cifras de fluxo e cifras de bloco.
- ☑ Apresentar uma visão geral da cifra de Feistel e explicar como a decriptação é o inverso da encriptação.
- ☑ Apresentar uma visão geral do data encryption standard (DES).
- ☑ Explicar o conceito do efeito avalanche.
- ☑ Discutir a força criptográfica do DES.
- ☑ Resumir os princípios mais importantes do projeto de uma cifra de bloco.

"Mas qual é a vantagem da mensagem cifrada sem a cifra?"

— *The Valley of Fear,* Sir Arthur Conan Doyle

O objetivo deste capítulo é ilustrar os princípios das cifras simétricas modernas. Para essa finalidade, focalizaremos a cifra simétrica mais utilizada: o data encryption standard (DES). Embora diversas cifras simétricas tenham sido desenvolvidas desde a introdução do DES e ele esteja destinado a ser substituído pelo *advanced encryption standard* (AES), DES continua sendo o algoritmo mais importante. Além disso, um estudo detalhado do DES oferece um conhecimento dos princípios usados em outras cifras simétricas.

Este capítulo começa com uma discussão sobre os princípios gerais das cifras de bloco simétricas, que são o tipo estudado neste livro (com exceção da cifra de fluxo RC4 no Capítulo 7). Em seguida, explicaremos o DES completo. Após essa abordagem focada em um algoritmo específico, retornaremos a uma discussão mais geral do projeto de cifras de bloco.

Em comparação com as cifras de chave pública, como RSA, a estrutura do DES, e da maioria das cifras simétricas, é muito complexa e não pode ser explicada tão facilmente quanto a citada e algoritmos semelhantes. Por conseguinte, o leitor pode querer começar com uma versão simplificada do DES, que é descrita no Apêndice G (<sv.pearson.com>, em inglês). Essa versão permite que o leitor simule a encriptação e decriptação à mão, e assim adquira um bom conhecimento de como funciona os detalhes do algoritmo. A experiência em sala de aula indica que um estudo dessa versão simplificada melhora a compreensão do DES.[1]

3.1 ESTRUTURA TRADICIONAL DE CIFRA DE BLOCO

Muitos algoritmos de encriptação de bloco simétricos em uso atual são baseados em uma estrutura conhecida como cifra de bloco de Feistel [FEIS73]. Por esse motivo, é importante examinar os princípios de projeto dela. Começaremos com uma comparação das cifras de fluxo e das cifras de bloco. Depois, discutiremos a motivação para a estrutura da cifra de bloco de Feistel. Finalmente, observaremos algumas de suas implicações.

Cifras de fluxo e cifras de bloco

Uma **cifra de fluxo** é aquela que encripta um fluxo de dados digital um bit ou um byte por vez. Alguns exemplos de cifras de fluxo clássicas são as autochaveadas Vigenère e Vernam. No caso ideal, uma versão *one-time pad* da cifra Vernam seria utilizada (Figura 2.7), na qual o fluxo de chaves (k_i) tem o tamanho do fluxo de bits do texto claro (p_i). Se o fluxo de chaves criptográfico for aleatório, então essa cifra é inquebrável por qualquer meio que não seja a aquisição dele. Porém, o fluxo de chaves precisa ser fornecido para os dois usuários com antecedência, por meio de algum canal independente e seguro. Isso gera problemas logísticos intransponíveis se o tráfego de dados intencionado for muito grande.

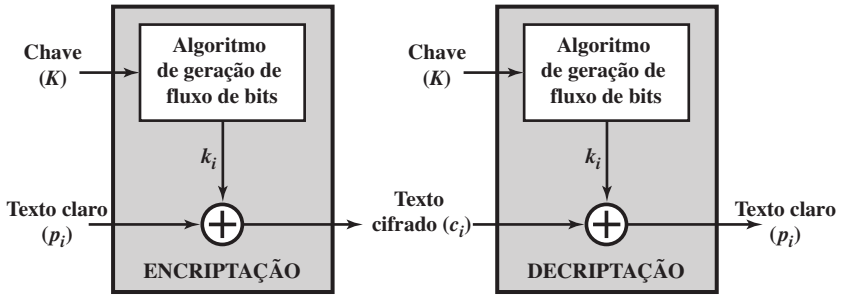
Por conseguinte, por motivos práticos, o gerador de fluxo de bits precisa ser implementado como um procedimento algorítmico, de modo que o fluxo de bits criptográfico seja produzido por ambos os usuários. Nessa técnica (Figura 3.1a), o gerador de fluxo de bits é um algoritmo controlado por chave e deverá produzir um fluxo de bits criptograficamente forte. Ou seja, deverá ser computacionalmente impraticável prever partes futuras do fluxo de bits com base em partes anteriores desse fluxo. Os dois usuários precisam apenas compartilhar a chave de geração, e cada um pode produzir o fluxo de chaves.

Uma **cifra de bloco** é aquela em que um bloco de texto claro é tratado como um todo e usado para produzir um de texto cifrado com o mesmo tamanho. Normalmente, um tamanho de bloco de 64 ou 128 bits é utilizado. Assim como a cifra de fluxo, os dois usuários compartilham uma chave de encriptação simétrica (Figura 3.1b). Usando alguns dos modos de operação explicados no Capítulo 6, uma cifra de bloco pode ser usada para conseguir o mesmo efeito de uma cifra de fluxo.

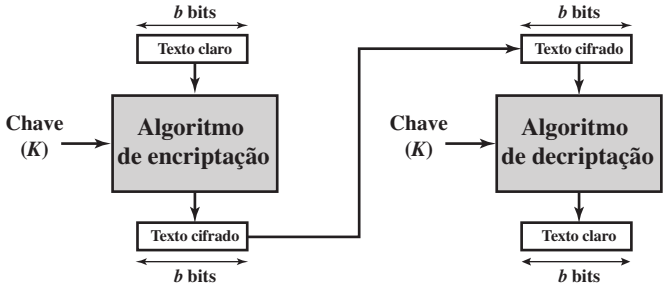
Tem sido destinado muito mais esforço para analisar as cifras de bloco, em geral, por elas serem adequadas a uma gama maior de aplicações do que as cifras de fluxo. A grande maioria das aplicações de criptografia simétrica baseadas em rede utiliza cifras de bloco. Assim, a preocupação neste capítulo, e nas nossas discussões no decorrer deste livro sobre encriptação simétrica, focalizará as cifras de bloco.

[1] Porém, você pode seguramente pular o Apêndice G, pelo menos em uma leitura inicial. Se você se perder ou ficar confuso com os detalhes do DES, então poderá voltar e começar com o DES simplificado.

Figura 3.1 Cifra de fluxo e cifra de bloco.



(a) Cifra de fluxo usando gerador algorítmico de fluxo de bits



(b) Cifra de bloco

Motivação para a estrutura de cifra de Feistel

Uma cifra de bloco opera sobre um bloco de texto claro de n bits para produzir um bloco de texto cifrado de n bits. Existem 2^n diferentes blocos de texto claro possíveis e, para a encriptação ser reversível (ou seja, para a decriptação ser possível), cada um produz um bloco de texto cifrado exclusivo. Essa transformação é chamada de reversível, ou não singular. Os exemplos a seguir ilustram uma transformação não singular e uma singular para $n = 2$.

Mapeamento reversível	
Texto claro	**Texto cifrado**
00	11
01	10
10	00
11	01

Mapeamento irreversível	
Texto claro	**Texto cifrado**
00	11
01	10
10	01
11	01

No segundo caso, um texto cifrado de 01 poderia ter sido produzido por um de dois blocos de texto claro. Assim, se nos limitarmos aos mapeamentos reversíveis, o número de transformações diferentes é $2^n!$.[2]

A Figura 3.2 ilustra a lógica de uma cifra de substituição geral para $n = 4$. Uma entrada de 4 bits produz um dos 16 estados de entrada possíveis, que é mapeado pela cifra de substituição para um dos 16 estados de saída possíveis, cada um representado por 4 bits de texto cifrado. Os mapeamentos de encriptação e decriptação são passíveis de ser definidos por uma tabulação, como mostra a Tabela 3.1. Essa é a forma mais geral de cifra de bloco e pode ser usada para definir qualquer mapeamento reversível entre texto claro e texto cifrado. Feistel se refere a isso como a *cifra de bloco ideal*, pois permite o número máximo de mapeamentos de encriptação a partir do bloco de texto claro [FEIS75].

[2] O raciocínio é o seguinte: para o primeiro texto claro, podemos escolher qualquer um dos 2^n blocos de texto cifrado. Para o segundo texto claro, selecionamos entre $2^n - 1$ blocos de texto cifrado restantes, e assim por diante.

Figura 3.2 Substituição de bloco geral de *n* bits para *n* bits (mostrados com *n* = 4).

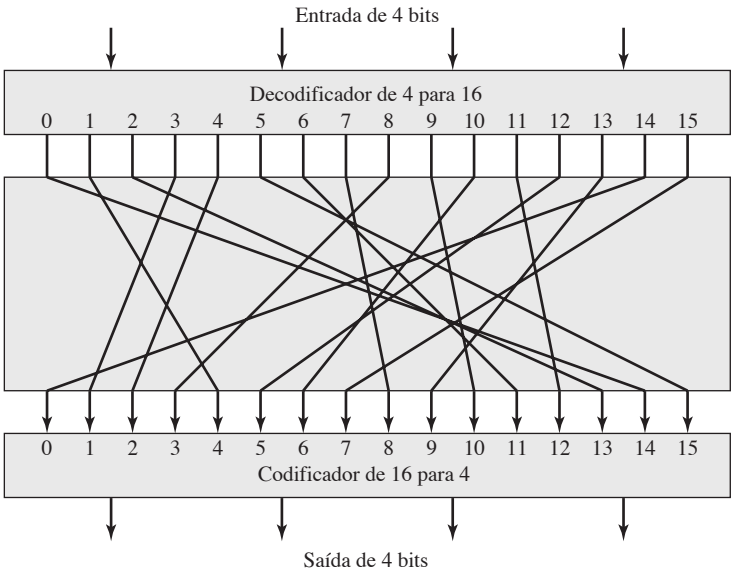


Tabela 3.1 Tabelas de encriptação e decriptação para a cifra de substituição da Figura 3.2.

Texto claro	Texto cifrado
0000	1110
0001	0100
0010	1101
0011	0001
0100	0010
0101	1111
0110	1011
0111	1000
1000	0011
1001	1010
1010	0110
1011	1100
1100	0101
1101	1001
1110	0000
1111	0111

Texto cifrado	Texto claro
0000	1110
0001	0011
0010	0100
0011	1000
0100	0001
0101	1100
0110	1010
0111	1111
1000	0111
1001	1101
1010	1001
1011	0110
1100	1011
1101	0010
1110	0000
1111	0101

Existe um problema prático com a cifra de bloco ideal. Se ela for usada em um tamanho de bloco pequeno, como $n = 4$, então o sistema é equivalente a uma cifra de substituição clássica. Esses sistemas, como já vimos, são vulneráveis a uma análise estatística do texto claro. Esse ponto fraco não é inerente ao uso de uma cifra de substituição, mas resulta do uso de um tamanho de bloco pequeno. Se n for suficientemente grande e uma substituição reversível qualquer entre texto claro e texto cifrado for permitida, então as características estatísticas do texto claro de origem são mascaradas a tal ponto que esse tipo de criptoanálise é inviável.

Entretanto, uma cifra de substituição reversível qualquer (a cifra de bloco ideal) para um grande tamanho de bloco não é prática, de um ponto de vista de implementação e de desempenho. Para tal transformação, o próprio mapeamento constitui a chave. Considere novamente a Tabela 3.1, que define um mapeamento reversível em particular do texto claro ao texto cifrado, para $n = 4$. O mapeamento pode ser definido pelas entradas na segunda coluna, que mostram o valor do texto cifrado para cada bloco de texto claro. Essa é basicamente a chave que determina o mapeamento específico entre todos os possíveis. Nesse caso, usando esse método simples de definição da chave, o tamanho de chave exigido é (4 bits) × (16 linhas) = 64 bits. Em geral, para uma cifra de bloco ideal de n bits, o tamanho da chave, definido nesses moldes, é $n \times 2^n$ bits. Para um bloco de 64 bits, que é uma dimensão desejável para afastar ataques estatísticos, o tamanho de chave exigido é $64 \times 2^{64} = 2^{70} \approx 10^{21}$ bits.

Considerando essas dificuldades, Feistel indica que é necessária uma aproximação do sistema de cifra de bloco ideal para um n grande, montado a partir de componentes que são facilmente observáveis [FEIS75]. No entanto, antes de passar para a técnica de Feistel, vamos fazer outra observação. Poderíamos usar a cifra de substituição de bloco geral mas, para tornar essa implementação tratável, confinamo-nos a um subconjunto dos $2^n!$ possíveis mapeamentos reversíveis. Por exemplo, suponha que definamos o mapeamento em termos de um conjunto de equações lineares. No caso de $n = 4$, temos

$$
\begin{aligned}
y_1 &= k_{11}x_1 + k_{12}x_2 + k_{13}x_3 + k_{14}x_4 \\
y_2 &= k_{21}x_1 + k_{22}x_2 + k_{23}x_3 + k_{24}x_4 \\
y_3 &= k_{31}x_1 + k_{32}x_2 + k_{33}x_3 + k_{34}x_4 \\
y_4 &= k_{41}x_1 + k_{42}x_2 + k_{43}x_3 + k_{44}x_4
\end{aligned}
$$

onde os x_i são os quatro dígitos binários do bloco de texto claro, os y_i são os quatro dígitos binários do bloco de texto cifrado, os k_{ij} são os coeficientes binários e a aritmética é mod 2. O tamanho de chave é apenas n^2, neste caso, 16 bits. O perigo com esse tipo de formulação é que pode ser vulnerável à criptoanálise por um atacante que saiba a estrutura do algoritmo. Neste exemplo, o que temos é basicamente a cifra de Hill discutida no Capítulo 2, aplicada a dados binários em vez de a caracteres. Como vimos no Capítulo 2, um sistema linear simples como esse é bastante vulnerável.

Cifra de Feistel

Feistel propôs [FEIS73] que podemos aproximar a cifra de bloco ideal utilizando o conceito de uma cifra de produto, que é a execução de duas ou mais cifras simples em sequência, de tal forma que o resultado ou produto final seja criptograficamente mais forte do que qualquer uma das cifras componentes. A essência da técnica é desenvolver uma cifra de bloco com um tamanho de chave de k bits e de bloco de n bits, permitindo um total de 2^k transformações possíveis, em vez de $2^n!$ transformações disponíveis com a cifra de bloco ideal.

Em particular, Feistel propôs o uso de uma cifra que alterna substituições e permutações, nas quais esses termos são definidos da seguinte forma:

- **Substituição:** cada elemento de texto claro ou grupo de elementos é substituído exclusivamente por um elemento ou grupo de elementos de texto cifrado correspondente.
- **Permutação:** uma sequência de elementos de texto claro é substituída por uma permutação dessa sequência. Ou seja, nenhum elemento é acrescentado, removido ou substituído na sequência, mas a ordem em que os elementos aparecem nela é mudada.

De fato, a cifra de Feistel é uma aplicação prática de uma proposta de Claude Shannon para desenvolver uma cifra de produto que alterne funções de *confusão* e *difusão* [SHAN49].[3] Examinaremos, em seguida, esses conceitos, e depois apresentaremos a cifra de Feistel. Mas, primeiro, vale a pena comentar sobre esse fato marcante: a estrutura da cifra de Feistel, que existe há mais de um quarto de século e que, por sua vez, é baseada na proposta de Shannon de 1945, é aquela utilizada por muitas cifras de bloco simétricas importantes atualmente.

[3] O artigo de 1949 apareceu originalmente como um relatório confidencial em 1945. Shannon goza de uma posição incrível e exclusiva na história dos computadores e da ciência da informação. Ele não apenas desenvolveu as primeiras ideias da criptografia moderna, mas também é responsável por inventar a disciplina da teoria da informação. Com base no seu trabalho sobre teoria da informação, ele elaborou uma fórmula para a capacidade de um canal de comunicações de dados, que ainda hoje é usado. Além disso, ele fundou outra disciplina, a aplicação da álgebra booleana ao estudo dos circuitos digitais; este último resultado ele conseguiu produzir rapidamente como uma dissertação de mestrado.

DIFUSÃO E CONFUSÃO Os termos *difusão* e *confusão* foram introduzidos por Claude Shannon para abranger os dois ingredientes básicos para a montagem de qualquer sistema criptográfico [SHAN49]. A preocupação de Shannon foi impedir a criptoanálise baseada em análise estatística. O raciocínio é o seguinte: considere que o atacante tem algum conhecimento das características estatísticas do texto claro. Por exemplo, em uma mensagem legível ao humano em alguma linguagem, a distribuição de frequência das várias letras pode ser conhecida. Ou, então, talvez haja palavras ou frases que provavelmente aparecem na mensagem (palavras prováveis). Se essas estatísticas estiverem de alguma forma refletidas no texto cifrado, o criptoanalista será capaz de deduzir a chave de codificação, parte dela ou pelo menos um conjunto de chaves que provavelmente contenha a correta. No que Shannon se refere como uma cifra fortemente ideal, todas as estatísticas do texto cifrado são independentes da chave utilizada em particular. A cifra de substituição arbitrária que discutimos anteriormente (Figura 3.2) é uma cifra assim, mas, como vimos, não é prática.[4]

Diferentemente de recorrer a sistemas ideais, Shannon sugere dois métodos para frustrar a criptoanálise estatística: difusão e confusão. Na **difusão**, a estrutura estatística do texto claro é dissipada em estatísticas de longa duração do texto cifrado. Isso é obtido fazendo-se que cada dígito do texto claro afete o valor de muitos do texto cifrado; em geral, isso é equivalente a fazer cada dígito do texto cifrado ser afetado por muitos do texto claro. Um exemplo da difusão é codificar uma mensagem $M = m_1, m_2, m_3, \ldots$ de caracteres com uma operação de média:

$$y_n = \left(\sum_{i=1}^{k} m_{n+i} \right) \bmod 26$$

acrescentando k letras sucessivas para obter uma de texto cifrado y_n. Pode-se mostrar que a estrutura estatística do texto claro foi dissipada. Assim, as frequências de letra no texto cifrado serão mais aproximadas do que no texto claro; as frequências de digrama também serão mais aproximadas, e assim por diante. Em uma cifra de bloco binária, a difusão pode ser alcançada realizando-se repetidamente alguma permutação dos dados, seguida pela aplicação de uma função a essa permutação; o efeito é que os bits de diferentes posições no texto claro original contribuem para um único bit de texto cifrado.[5]

Cada cifra de bloco envolve uma transformação de um bloco de texto claro para um de texto cifrado, no qual essa transformação depende da chave. O mecanismo de difusão busca tornar o relacionamento estatístico entre o texto claro e o texto cifrado o mais complexo possível, a fim de frustrar tentativas de deduzir a chave. Por outro lado, a **confusão** procura estabelecer o relacionamento entre as estatísticas do texto cifrado e o valor da chave de encriptação o mais complexo possível, novamente para frustrar tentativas de descobrir a chave. Assim, mesmo que o atacante possa ter alguma ideia das estatísticas do texto cifrado, o modo pelo qual a chave foi usada para produzir esse texto cifrado é tão complexo que torna difícil deduzir a chave. Isso é obtido com o uso de um algoritmo de substituição complexo. Contrastantemente, uma função de substituição linear simples resultaria em pouca confusão.

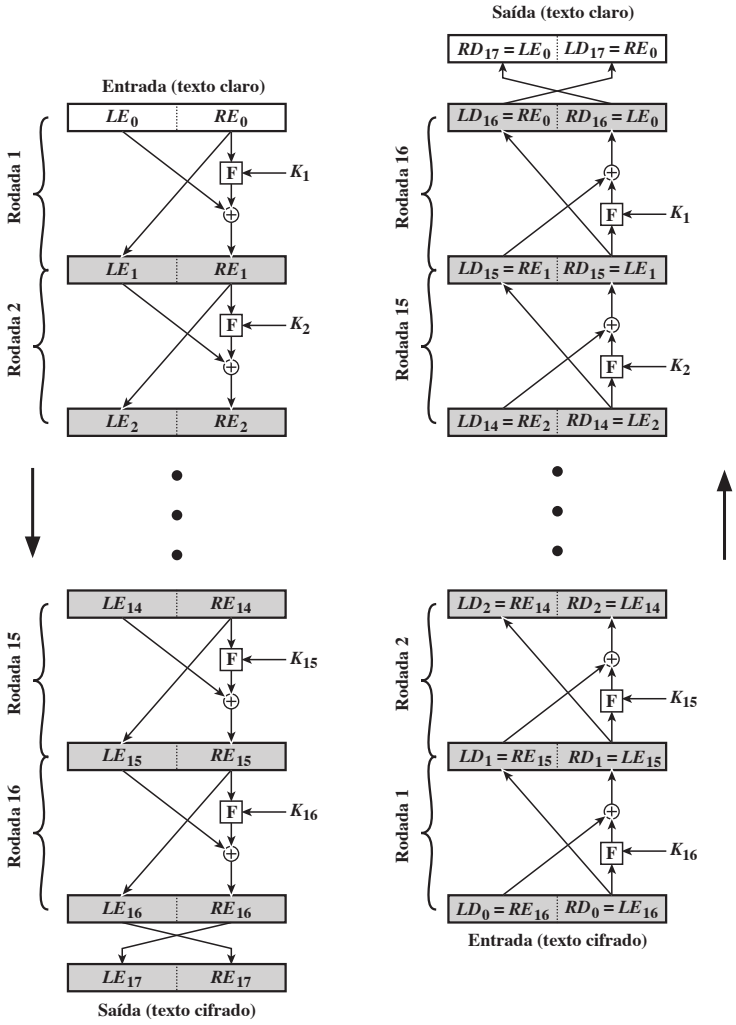
Conforme [ROBS95b] indica, tão bem-sucedidas são a difusão e a confusão na captura da essência dos atributos desejados de uma cifra de bloco que elas se tornaram a base do projeto moderno de cifras de bloco.

ESTRUTURA DA CIFRA DE FEISTEL O lado esquerdo da Figura 3.3 representa a estrutura proposta por Feistel. As entradas do algoritmo de encriptação são um bloco de texto claro de tamanho $2w$ bits e uma chave K. O bloco do texto claro é dividido em duas metades, L_0 e R_0. As duas metades dos dados passam por n rodadas de processamento, e depois se combinam para produzir o bloco do texto cifrado. Cada rodada i possui como entradas L_{i-1} e R_{i-1}, derivadas da rodada anterior, assim como uma subchave K_i derivada do K geral. Normalmente, as subchaves K_i são diferentes de K e umas das outras. Na Figura 3.3, 16 rodadas são utilizadas, embora qualquer número delas possa ser implementado.

[4] O Apêndice F (em <sv.pearson.com.br>, em inglês) expande os conceitos de Shannon referentes a medidas de sigilo e segurança dos algoritmos criptográficos.

[5] Alguns livros sobre criptografia igualam a permutação com a difusão. Isso é incorreto. A permutação, *por si só*, não muda as estatísticas do texto claro no nível de letras individuais ou blocos permutados. Por exemplo, no DES, a permutação inverte dois blocos de 32 bits, de modo que as estatísticas de *strings* de 32 bits ou menos são preservadas.

Figura 3.3 Encriptação e decriptação de Feistel (16 rodadas).



Todas as rodadas têm a mesma estrutura. Uma **substituição** é realizada na metade esquerda dos dados. Isso é feito aplicando-se uma *função* F à metade direita dos dados, e depois, a operação lógica de ou-exclusivo entre a saída dessa função e a metade esquerda dos dados. A função F tem a mesma estrutura geral para cada rodada, mas é parametrizada pela subchave da rodada K_i. Outra forma de expressar isso é dizer que F é uma função da metade direita de w bits do bloco e de uma subchave de y bits, que produz um valor de saída com comprimento de w bits: $F(RE_i, K_{i+1})$. Após essa substituição, é realizada uma **permutação** que consiste na troca das duas metades dos dados.[6] Essa estrutura é uma forma particular da rede de substituição-permutação (ou SPN, do acrônimo em inglês para *substitution-permutation network*) proposta por Shannon.

A execução exata de uma rede de Feistel depende da escolha dos seguintes parâmetros e recursos de projeto:

- **Tamanho de bloco:** tamanhos de bloco maiores significam maior segurança (mantendo as outras coisas iguais), mas velocidade de encriptação/decriptação reduzida para determinado algoritmo. Maior segurança é obtida por difusões maiores. Tradicionalmente, o tamanho de bloco de 64 bits foi considerado uma escolha razoável e quase universal no projeto de cifras de bloco. Porém, o novo AES usa um tamanho de bloco de 128 bits.

[6] A rodada final é seguida por uma troca que desfaz a troca que faz parte da rodada final. Alguém poderia simplesmente deixar as duas trocas fora do diagrama, sacrificando alguma consistência de apresentação. De qualquer forma, a falta efetiva de uma troca na rodada final é proporcionada para simplificar a implementação do processo de decriptação, conforme veremos.

- **Tamanho de chave:** tamanho de chave maior significa maior segurança, mas pode diminuir a velocidade de encriptação/decriptação. Maior segurança é obtida pela maior resistência a ataques de força bruta e maior confusão. Os tamanhos de chave de 64 bits ou menos agora são em grande parte considerados inadequados, e 128 bits tornou-se um padrão comum.
- **Número de rodadas:** a essência da cifra de Feistel é que uma única rodada oferece segurança inadequada, mas várias proporcionam maior segurança. Um tamanho típico é de 16 rodadas.
- **Algoritmo de geração de subchave:** maior complexidade nesse algoritmo deverá levar a maior dificuldade de criptoanálise.
- **Função F:** novamente, maior complexidade geralmente significa maior resistência à criptoanálise.

Existem duas outras considerações no projeto de uma cifra de Feistel:

- **Encriptação/decriptação rápidas em software:** em muitos casos, a encriptação é embutida nas aplicações ou funções utilitárias, de tal forma que impede uma implementação em hardware. Por conseguinte, a velocidade de execução do algoritmo torna-se uma preocupação.
- **Facilidade de análise:** embora quiséssemos tornar nosso algoritmo o mais difícil possível de criptoanalisar, existe um grande benefício em colocá-lo como fácil de ser analisado. Ou seja, se o algoritmo puder ser explicado de forma concisa e clara, é mais fácil analisá-lo em busca de vulnerabilidades criptoanalíticas e, portanto, desenvolver um nível mais alto de garantia quanto a sua força. DES, por exemplo, não tem uma funcionalidade facilmente analisável.

ALGORITMO DE DECRIPTAÇÃO DE FEISTEL O processo de decriptação com uma cifra de Feistel é basicamente o mesmo de encriptação. A regra é a seguinte: use o texto cifrado como entrada para o algoritmo, mas as subchaves K_i em ordem reversa. Ou seja, use K_n na primeira rodada, K_{n-1} na segunda, e assim por diante, até K_1 ser usada na última rodada. Essa é uma ótima característica, pois significa que não precisamos implementar dois algoritmos diferentes, um para encriptação e outro para decriptação.

Para ver se o mesmo algoritmo com uma ordem de chave invertida produz o resultado correto, considere a Figura 3.3, que mostra o processo de encriptação descendo pelo lado esquerdo, e o processo de decriptação subindo pelo lado direito, para um algoritmo de 16 rodadas. Por clareza, usamos a notação LE_i e RE_i para os dados que passam pelo algoritmo de encriptação, e LD_i e RD_i para os dados pelo algoritmo de decriptação. O diagrama indica que, a cada rodada, o valor intermediário do processo de decriptação é igual ao correspondente do processo de encriptação com as duas metades do valor trocadas. Colocando de outra forma, considere que a i-ésima rodada de encriptação seja $LE_i||RE_i$ (LE_i concatenado com RE_i). Então, a saída correspondente à rodada de decriptação $(16 - i)$ é $RE_i||LE_i$ ou, de modo equivalente, $LD_{16-i}||RD_{16-i}$.

Vamos percorrer a Figura 3.3 para demonstrar a validade das afirmações anteriores. Depois da última iteração do processo de encriptação, as duas metades da saída são trocadas, de modo que o texto cifrado é $RE_{16}||LE_{16}$. A saída dessa rodada é o texto cifrado. A seguir, apanhe esse texto cifrado e use-o como entrada para o mesmo algoritmo. A entrada para a primeira rodada é $RE_{16}||LE_{16}$, que é igual à troca de 32 bits da saída da décima sexta rodada do processo de encriptação.

Agora, gostaríamos de mostrar que a saída da primeira rodada do processo de decriptação é igual a uma troca de 32 bits da entrada com a décima sexta rodada do processo de encriptação. Primeiro, considere o processo de encriptação. Vemos que

$$\begin{aligned} LE_{16} &= RE_{15} \\ RE_{16} &= LE_{15} \oplus \mathrm{F}(RE_{15}, K_{16}) \end{aligned}$$

No lado da decriptação,

$$\begin{aligned} LD_1 &= RD_0 = LE_{16} = RE_{15} \\ RD_1 &= LD_0 \oplus \mathrm{F}(RD_0, K_{16}) \\ &= RE_{16} \oplus \mathrm{F}(RE_{15}, K_{16}) \\ &= [LE_{15} \oplus \mathrm{F}(RE_{15}, K_{16})] \oplus \mathrm{F}(RE_{15}, K_{16}) \end{aligned}$$

A operação lógica ou-exclusivo (ou, simplesmente, XOR, do acrônimo em inglês para *exclusive-or*) tem as seguintes propriedades:

$$[A \oplus B] \oplus C = A \oplus [B \oplus C]$$
$$D \oplus D = 0$$
$$E \oplus 0 = E$$

Assim, temos $LD_1 = RE_{15}$ e $RD_1 = LE_{15}$. A saída da primeira rodada do processo de decriptação é $RE_{15}||LE_{15}$, que é a troca de 32 bits da entrada para a décima sexta rodada da encriptação. Essa correspondência se mantém por todas as 16 iterações, como é facilmente mostrado. Podemos converter esse processo em termos gerais. Para a i-ésima iteração do algoritmo de encriptação,

$$LE_i = RE_{i-1}$$
$$RE_i = LE_{i-1} \oplus \mathrm{F}(RE_{i-1}, K_i)$$

Reorganizando os termos,

$$RE_{i-1} = LE_i$$
$$LE_{i-1} = RE_i \oplus \mathrm{F}(RE_{i-1}, K_i) = RE_i \oplus \mathrm{F}(LE_i, K_i)$$

Descrevemos as entradas para a i-ésima iteração como uma função das saídas, e essas equações confirmam as atribuições mostradas no lado direito da Figura 3.3.

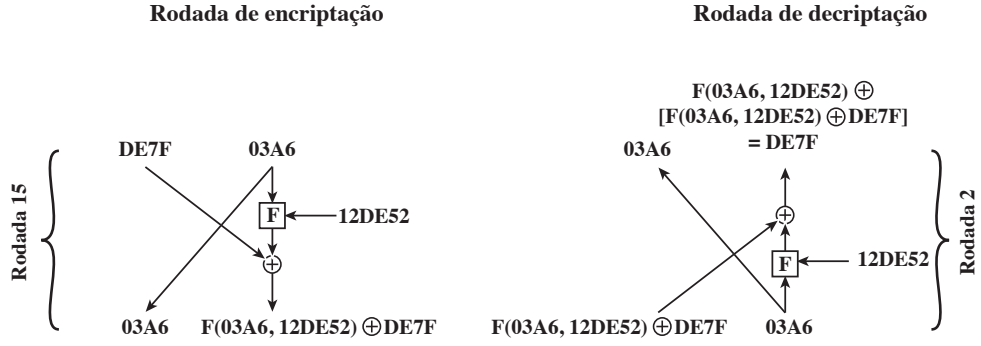
Finalmente, vemos que a saída da última rodada do processo de decriptação é $RE_0||LE_0$. Uma troca de 32 bits recupera o texto claro original, demonstrando a validade do processo de decriptação de Feistel.

Observe que a derivação não exige que F seja uma função reversível. Para ver isso, use um caso limitador em que F produz uma saída constante (por exemplo, tudo 1), independente dos valores de seus dois argumentos. As equações ainda são válidas.

Para ajudar a esclarecer os conceitos apresentados, vamos examinar um exemplo específico (Figura 3.4) e focar na décima quinta rodada de encriptação, correspondente à segunda rodada de decriptação. Suponha que os blocos em cada estágio tenham 32 bits (duas metades de 16 bits) e que o tamanho da chave seja de 24 bits. Imagine que, no final da rodada quatorze de encriptação, o valor do bloco intermediário (em hexadecimal) seja DE7F03A6. Então, LE_{14} = DE7F e RE_{14} = 03A6. Além disso, admita que o valor de K_{15} seja 12DE52. Após a rodada 15, temos LE_{15} = 03A6 e RE_{15} = F(03A6, 12DE52) $\oplus$ DE7F.

Agora, vamos examinar a decriptação. Vamos supor que $LD_1 = RE_{15}$ e $RD_1 = LE_{15}$, como mostra a Figura 3.3, e que iremos demonstrar que $LD_2 = RE_{14}$ e $RD_2 = LE_{14}$. Assim, começamos com LD_1 = F(03A6, 12DE52) $\oplus$ DE7F e RD_1 = 03A6. Depois, pela Figura 3.3, LD_2 = 03A6 = RE_{14} e RD_2 = F(03A6, 12DE52) $\oplus$ [F(03A6, 12DE52) $\oplus$ DE7F] = DE7F = LE14.

Figura 3.4 Exemplo Feistel.



3.2 DATA ENCRYPTION STANDARD

Até a introdução do advanced encryption standard (AES), em 2001, o data encryption standard (DES) era o esquema de encriptação mais utilizado. DES foi adotado em 1977 pelo National Bureau of Standards, agora National Institute of Standards and Technology (NIST), como Federal Information Processing Standard 46 (FIPS PUB 46). O algoritmo é conhecido como data encryption algorithm (DEA).[7] Para DEA, os dados são encriptados em blocos de 64 bits usando uma chave de 56 bits. O algoritmo transforma a entrada de 64 bits em uma série de etapas para uma saída de 64 bits. As mesmas etapas, com a mesma chave, são empregadas para reverter a encriptação.

Com o passar dos anos, DES tornou-se o algoritmo de encriptação simétrica dominante, especialmente em aplicações financeiras. Em 1994, o NIST reafirmou o DES para uso federal por outros cinco anos; o NIST recomendou o uso do DES para aplicações que não tenham informações de proteção ou confidenciais. Em 1999, o NIST emitiu uma nova versão do seu padrão (FIPS PUB 46-3), que indicava que o DES deveria ser utilizado apenas para sistemas legados, e que o triple DES (que basicamente envolve repetir o algoritmo DES três vezes sobre o texto claro com duas ou três chaves diferentes para produzir o texto cifrado) fosse empregado. Estudaremos o triple DES no Capítulo 6. Como os algoritmos básicos de encriptação e decriptação são os mesmos para DES e triple DES, continua sendo importante entender a cifra do DES. Esta seção oferece uma visão geral. Para o leitor interessado, o Apêndice S, na Sala Virtual (<sv.pearson.com.br>, em inglês), contém mais detalhes.

Encriptação DES

O esquema geral para a encriptação DES é ilustrado na Figura 3.5. Assim como em qualquer esquema de encriptação, existem duas entradas na função: o texto claro a ser encriptado e a chave. Nesse caso, o texto claro precisa ter 64 bits de extensão, e a chave tem 56 bits de extensão.[8]

Examinando o lado esquerdo da figura, podemos ver que o processamento do texto claro prossegue em três fases. Primeiro, o texto claro de 64 bits passa por uma permutação inicial (IP, do acrônimo em inglês para *initial permutation*), que reorganiza os bits a fim de produzir a *entrada permutada*. Isso é seguido por uma fase consistindo em 16 rodadas da mesma função, que envolve funções de permutação e substituição. A saída da última (décima sexta) rodada baseia-se em 64 bits que são uma função do texto claro de entrada e da chave. As metades esquerda e direita da saída são trocadas para produzir a **pré-saída**. Finalmente, a pré-saída é passada por uma permutação [IP^{-1}], que é o inverso da função de permutação inicial, a fim de produzir o texto cifrado de 64 bits. Com exceção das permutações inicial e final, DES tem a estrutura exata de uma cifra de Feistel, como mostra a Figura 3.3.

A parte da direita da Figura 3.5 apresenta o modo como a chave de 56 bits é usada. Inicialmente, a chave passa por uma função de permutação. Depois, para cada uma das 16 rodadas, uma *subchave* (K_i) é produzida pela combinação de um deslocamento circular à esquerda e uma permutação. A função de permutação é a mesma para cada rodada, mas uma subchave diferente é produzida, por conta dos deslocamentos repetidos dos bits da chave.

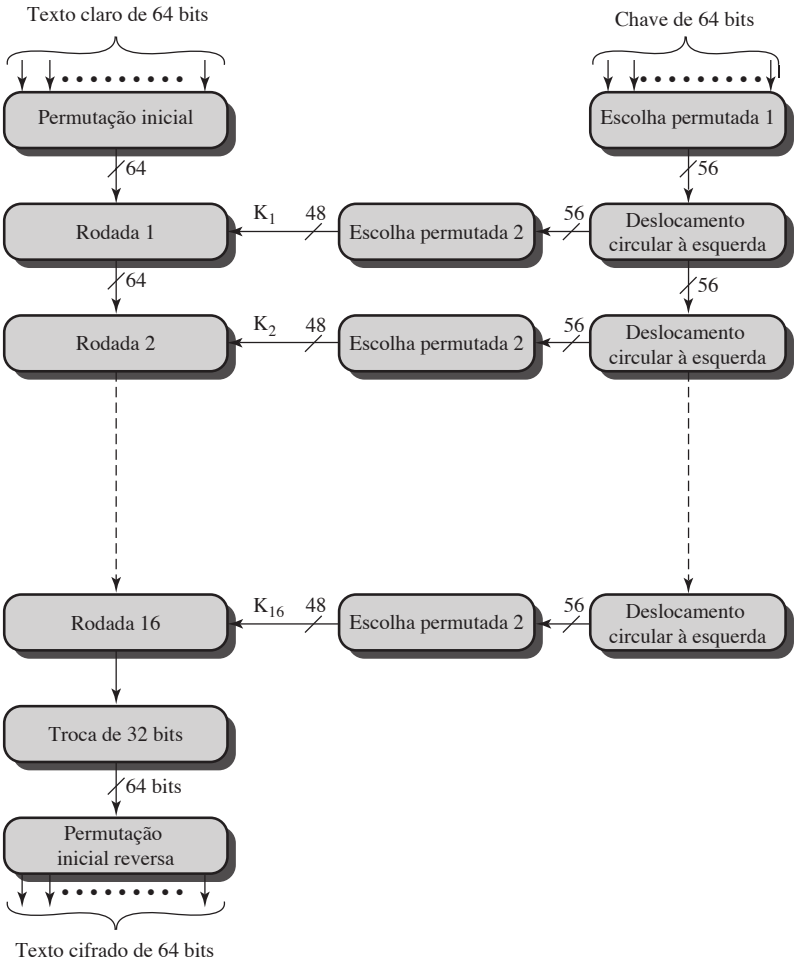
Decriptação DES

Assim como qualquer cifra de Feistel, a decriptação usa o mesmo algoritmo da encriptação, exceto que a aplicação das subchaves é invertida. Além disso, as permutações inicial e final são invertidas.

[7] A terminologia é um pouco confusa. Até pouco tempo, os termos *DES* e *DEA* poderiam ser usados equivalentemente. Porém, a edição mais recente do documento DES inclui uma especificação do DEA descrita aqui, mais o triple DEA (TDEA) citado no Capítulo 6. Tanto DEA quanto TDEA fazem parte do data encryption standard. Além disso, até a adoção recente do termo oficial *TDEA*, o algoritmo triple DEA era normalmente conhecido como triple *DES*, e apresentado como 3DES. Por questão de conveniência, usamos o termo *3DES*.

[8] Na realidade, a função espera uma chave de 64 bits como entrada. Porém, somente 56 desses bits são usados; outros 8 bits podem ser empregados como bits de paridade ou simplesmente definidos arbitrariamente.

Figura 3.5 Representação geral do algoritmo de encriptação DES.



3.3 UM EXEMPLO DO DES

Agora, trabalharemos um exemplo, considerando algumas de suas implicações. Embora você não tenha que replicar o exemplo manualmente, será esclarecedor estudar os padrões hexa que ocorrem de uma etapa para a seguinte.

Para este exemplo, o texto claro é um palíndromo hexadecimal. O texto claro, a chave e o texto cifrado resultante são os seguintes:

Texto claro:	**02468aceeca86420**
Chave:	**0f1571c947d9e859**
Texto cifrado:	**da02ce3a89ecac3b**

Resultados

A Tabela 3.2 mostra a progressão do algoritmo. A primeira linha apresenta os valores de 32 bits das metades esquerda e direita dos dados após a permutação inicial. As 16 linhas seguintes indicam os resultados após cada rodada. Também aparece o valor da subchave de 48 bits gerada para cada rodada. Observe que $L_i = R_{i-1}$. A última linha mostra os valores da esquerda e da direita após a permutação inicial inversa. Esses dois valores combinados formam o texto cifrado.

Tabela 3.2 Exemplo de DES.

Rodada	K_i	L_i	R_i
IP		5a005a00	3cf03c0f
1	1e030f03080d2930	3cf03c0f	bad22845
2	0a31293432242318	bad22845	99e9b723
3	23072318201d0c1d	99e9b723	0bae3b9e
4	05261d3824311a20	0bae3b9e	42415649
5	3325340136002c25	42415649	18b3fa41
6	123a2d0d04262a1c	18b3fa41	9616fe23
7	021f120b1c130611	9616fe23	67117cf2
8	1c10372a2832002b	67117cf2	c11bfc09
9	04292a380c341f03	c11bfc09	887fbc6c
10	2703212607280403	887fbc6c	600f7e8b
11	2826390c31261504	600f7e8b	f596506e
12	12071c241a0a0f08	f596506e	738538b8
13	300935393c0d100b	738538b8	c6a62c4e
14	311e09231321182a	c6a62c4e	56b0bd75
15	283d3e0227072528	56b0bd75	75e8fd8f
16	2921080b13143025	75e8fd8f	25896490
IP^{-1}		da02ce3a	89ecac3b

Nota: as subchaves DES são apresentadas como oito valores de 6 bits no padrão hexa.

Efeito avalanche

Uma propriedade desejável de qualquer algoritmo de encriptação é que uma pequena mudança no texto claro ou na chave produza uma alteração significativa no texto cifrado. Em particular, uma mudança em um bit do texto claro ou um bit da chave deverá produzir uma modificação em muitos bits do texto cifrado. Se a mudança fosse pequena, esta poderia oferecer um modo de reduzir o tamanho do espaço de texto claro ou de chave a ser pesquisado.

Usando o exemplo da Tabela 3.2, a Tabela 3.3 mostra o resultado quando o quarto bit do texto claro é mudado, de modo que o texto claro é **`12468aceeca86420`**. A segunda coluna da tabela apresenta os valores intermediários de 64 bits no final de cada rodada para os dois textos claros. A terceira coluna indica o número de bits que diferem entre os dois valores intermediários. A tabela demonstra que, após apenas três rodadas, os dois blocos já diferem em 18 bits. Ao terminar, os dois textos cifrados se diferenciam em 32 bits.

A Tabela 3.4 mostra um teste semelhante usando o texto claro original com duas chaves que diferem apenas na quarta posição de bit: a chave original, **`0f1571c947d9e859`**, e a chave alterada, **`1f1571c947d9e859`**. Novamente, os resultados apontam que cerca de metade dos bits no texto cifrado diferem e que o efeito avalanche já é notado após poucas rodadas.

Tabela 3.3 Efeito avalanche no DES: mudança no texto claro.

Rodada		δ
	02468aceeca86420 12468aceeca86420	**1**
1	3cf03c0fbad22845 3cf03c0fbad32845	**1**
2	bad2284599e9b723 bad3284539a9b7a3	**5**
3	99e9b7230bae3b9e 39a9b7a3171cb8b3	**18**
4	0bae3b9e42415649 171cb8b3ccaca55e	**34**
5	4241564918b3fa41 ccaca55ed16c3653	**37**
6	18b3fa419616fe23 d16c3653cf402c68	**33**
7	9616fe2367117cf2 cf402c682b2cefbc	**32**
8	67117cf2c11bfc09 2b2cefbc99f91153	**33**
9	c11bfc09887fbc6c 99f911532eed7d94	32
10	887fbc6c600f7e8b 2eed7d94d0f23094	34
11	600f7e8bf596506e d0f23094455da9c4	37
12	f596506e738538b8 455da9c47f6e3cf3	31
13	738538b8c6a62c4e 7f6e3cf34bc1a8d9	29
14	c6a62c4e56b0bd75 4bc1a8d91e07d409	33
15	56b0bd7575e8fd8f 1e07d4091ce2e6dc	31
16	75e8fd8f25896490 1ce2e6dc365e5f59	32
IP^{-1}	da02ce3a89ecac3b 057cde97d7683f2a	32

Tabela 3.4 Efeito avalanche no DES: mudança na chave.

Rodada		δ
	02468aceeca86420 02468aceeca86420	0
1	3cf03c0fbad22845 3cf03c0f9ad628c5	3
2	bad2284599e9b723 9ad628c59939136b	11
3	99e9b7230bae3b9e 9939136b768067b7	25
4	0bae3b9e42415649 768067b75a8807c5	29
5	4241564918b3fa41 5a8807c5488dbe94	26
6	18b3fa419616fe23 488dbe94aba7fe53	26
7	9616fe2367117cf2 aba7fe53177d21e4	27
8	67117cf2c11bfc09 177d21e4548f1de4	32
9	c11bfc09887fbc6c 548f1de471f64dfd	34
10	887fbc6c600f7e8b 71f64dfd4279876c	36
11	600f7e8bf596506e 4279876c399fdc0d	32
12	f596506e738538b8 399fdc0d6d208dbb	28
13	738538b8c6a62c4e 6d208dbbb9bdeeaa	33
14	c6a62c4e56b0bd75 b9bdeeaad2c3a56f	30
15	56b0bd7575e8fd8f d2c3a56f2765c1fb	33
16	75e8fd8f25896490 2765c1fb01263dc4	30
IP^{-1}	da02ce3a89ecac3b ee92b50606b62b0b	30

3.4 A FORÇA DO DES

Desde sua adoção como um padrão federal, tem havido preocupações prolongadas sobre o nível de segurança fornecido pelo DES. Essas preocupações, em grande parte, estão divididas em duas áreas: tamanho de chave e natureza do algoritmo.

Uso de chaves de 56 bits

Com um tamanho de chave de 56 bits, existem 2^{56} chaves possíveis, o que é aproximadamente $7{,}2 \times 10^{16}$ chaves. Assim, um ataque de força bruta parece ser impraticável. Supondo que, em média, metade do espaço de chave tenha que ser pesquisado, uma única máquina realizando uma encriptação DES por microssegundo levaria mais de mil anos para quebrar a cifra.

Porém, a suposição de uma encriptação por microssegundo é bastante conservadora. Desde 1977, Diffie e Hellman postularam que existia tecnologia para montar uma máquina paralela com 1 milhão de dispositivos de encriptação, cada um podendo realizar uma encriptação por microssegundo [DIFF77]. Isso traria o tempo médio de busca para cerca de 10 horas. Os autores estimaram que o custo seria de aproximadamente US$ 20 milhões em dólares de 1977.

Com a tecnologia atual, nem sequer é preciso usar hardware especial. Em vez disso, a velocidade dos processadores comerciais ameaça a segurança do DES. Um artigo recente da Seagate Technology [SEAG08] sugere que uma taxa de 1 bilhão (10^9) de combinações de chaves por segundo é razoável para os computadores multicore atuais. As ofertas recentes confirmam isso. Tanto a Intel quanto a AMD agora oferecem instruções baseadas em hardware para acelerar o uso do AES. Testes executados em uma máquina Intel multicore contemporânea resultaram em uma taxa de cerca de meio bilhão de encriptações por segundo [BASU12]. Outra análise recente sugere que, com a tecnologia de supercomputador contemporânea, uma taxa de 10^{13} encriptações por segundo é razoável [AROR12].

Com esses resultados em mente, a Tabela 3.5 mostra quanto tempo é necessário a um ataque de força bruta para diversos tamanhos de chave. Como podemos ver, um único PC pode quebrar o DES em cerca de um ano; se vários PCs trabalharem em paralelo, o tempo é reduzido drasticamente. E os supercomputadores de hoje devem ser capazes de descobrir uma chave em cerca de uma hora. Os tamanhos de chave de 128 bits ou mais são efetivamente inquebráveis usando apenas uma técnica de força bruta. Mesmo que conseguíssemos agilizar o sistema de ataque por um fator de 1 trilhão (10^{12}), ainda seriam necessários 100 mil anos para quebrar um código usando uma chave de 128 bits.

Felizmente, existem várias alternativas ao DES, e as mais importantes são AES e triple DES, discutidos nos capítulos 5 e 6, respectivamente.

Tabela 3.5 Tempo médio exigido para uma busca exaustiva no espaço de chaves.

Tamanho de chave (bits)	Cifra	Número de chaves alternativas	Tempo exigido a 10^9 decriptações/s	Tempo exigido a 10^{13} decriptações/s
56	DES	$2^{56} \approx 7{,}2 \times 10^{16}$	2^{55} ns = 1,125 ano	1 hora
128	AES	$2^{128} \approx 3{,}4 \times 10^{38}$	2^{127} ns = $5{,}3 \times 10^{21}$ anos	$5{,}3 \times 10^{17}$ anos
168	Triple DES	$2^{168} \approx 3{,}7 \times 10^{50}$	2^{167} ns = $5{,}8 \times 10^{33}$ anos	$5{,}8 \times 10^{29}$ anos
192	AES	$2^{192} \approx 6{,}3 \times 10^{57}$	2^{191} ns = $9{,}8 \times 10^{40}$ anos	$9{,}8 \times 10^{36}$ anos
256	AES	$2^{256} \approx 1{,}2 \times 10^{77}$	2^{255} ns = $1{,}8 \times 10^{60}$ anos	$1{,}8 \times 10^{56}$ ano
26 caracteres (permutação)	Monoalfabético	$2! = 4 \times 10^{26}$	2×10^{26} ns = $6{,}3 \times 10^{9}$ anos	$6{,}3 \times 10^{6}$ anos

Natureza do algoritmo DES

Outra preocupação é a possibilidade de que a criptoanálise seja possível explorando-se as características do algoritmo DES. O foco disso tem sido as oito tabelas de substituição, ou S-boxes, que são usadas em cada iteração (descritas no Apêndice S – <sv.pearson.com.br>, em inglês). Como os critérios de projeto para essas caixas, e, na realidade, para o algoritmo inteiro, não se tornaram públicos, existe uma suspeita de que elas foram construídas de modo que a criptoanálise seja possível para um oponente que conheça as fraquezas nelas. Essa afirmação é torturante, e, com o passar dos anos, diversas regularidades e comportamentos inesperados das S-boxes foram descobertos. Apesar disso, ninguém até aqui teve sucesso desvendando a suposta fraqueza fatal nas S-boxes.[9]

Ataques de temporização

Discutiremos os ataques de temporização com mais detalhes na Parte Dois, pois se relacionam aos algoritmos de criptografia de chave pública. Porém, a questão também pode ser relevante para cifras simétricas. Basicamente, um ataque de temporização é aquele em que a informação sobre a chave ou sobre o texto claro é obtida observando-se quanto tempo foi gasto para determinada implementação realizar decriptações em vários textos cifrados. Um ataque de temporização explora o fato de que um algoritmo de encriptação ou decriptação em geral exige quantidades ligeiramente diferentes de tempo para diversas entradas. [HEVI99] relata uma técnica que fornece o peso de Hamming (número de bits iguais a um) da chave secreta. Ela ainda está muito longe de obter a chave real, mas é um primeiro passo interessante. Os autores concluem que DES parece ser bastante resistente a um ataque de temporização bem-sucedido, mas sugerem alguns meios a serem explorados. Por mais que essa seja uma linha de ataque curiosa, até aqui parece improvável que essa técnica tenha sucesso contra DES, e menos ainda contra cifras simétricas mais poderosas, como triple DES e AES.

3.5 PRINCÍPIOS DE PROJETO DE CIFRA DE BLOCO

Embora tenha havido muito progresso no projeto de cifras de bloco criptograficamente fortes, os princípios básicos não mudaram tanto desde o trabalho de Feistel e da equipe de projeto do DES, no início da década de 1970. Nesta seção, examinaremos três aspectos críticos do projeto de cifras de bloco: o número de rodadas, o projeto da função F e o escalonamento de chave.

Número de rodadas

Quanto maior o número de rodadas, mais difícil é realizar a criptoanálise, mesmo para uma função F relativamente fraca. Em geral, o critério deverá ser de que o número de rodadas seja escolhido de modo que os esforços criptoanalíticos conhecidos exijam maior ação do que um ataque de busca de chave por força bruta. Esse critério certamente foi usado no projeto do DES. Schneier [SCHN96] observa que, para o DES com 16 rodadas, um ataque de criptoanálise diferencial é ligeiramente menos eficiente do que a força bruta: o ataque de criptoanálise diferencial exige $2^{55,1}$ operações,[10] enquanto o de força bruta, 2^{55}. Se o DES tivesse 15 ou menos rodadas, a criptoanálise diferencial exigiria menos esforço do que a busca de chave por força bruta.

Esse critério é atraente porque facilita julgar a força de um algoritmo e comparar diferentes algoritmos. Na ausência de uma descoberta revolucionária em criptoanálise, a força de qualquer algoritmo que satisfaça o critério acima pode ser julgada unicamente a partir do tamanho da chave.

Projeto da função F

O núcleo da cifra de bloco de Feistel é a função F, que oferece a propriedade de confusão em uma cifra de Feistel. Assim, é preciso que seja difícil "desembaralhar" a substituição realizada por F. Um critério óbvio é que F seja não linear. Quanto menos linear for F, mais difícil será qualquer tipo de criptoanálise. Existem várias medidas de não linearidade, que estão além do escopo deste livro. Em termos gerais, quanto mais difícil for aproximar F de um conjunto de equações lineares, mais não linear será F.

---

[9] Pelo menos, ninguém confirmou publicamente essa descoberta.

[10] A criptoanálise diferencial do DES exige 2^{47} textos claros *escolhidos*. Se tudo o que você tem para trabalhar é texto claro conhecido, então terá que percorrer uma grande quantidade de pares de texto claro/texto cifrado conhecido, procurando os úteis. Isso leva o nível de esforço para $2^{55,1}$.

Vários outros critérios deverão ser considerados no projeto de F. Gostaríamos que o algoritmo tivesse boas propriedades de avalanche. Lembre-se de que, em geral, isso significa que uma mudança em um bit da entrada deverá produzir uma alteração em muitos bits da saída. Uma versão mais rigorosa disso é o **critério de avalanche estrito** (**SAC**, do acrônimo em inglês para *strict avalanche criterion*) [WEBS86], que afirma que qualquer bit de saída *j* de uma S-boxes (veja, no Apêndice S – <sv.pearson.com.br>, em inglês –, uma discussão sobre S-boxes) deverá mudar com probabilidade 1/2 quando qualquer bit de entrada isolado *i* for invertido para todo *i, j*. Embora o SAC seja expresso em termos de S-boxes, um critério semelhante poderia ser aplicado a F como um todo. Isso é importante quando se considera projetos que não incluem S-boxes.

Outro critério proposto em [WEBS86] é o **critério de independência de bit** (BIC, do acrônimo em inglês para *bit independence criterion*), que afirma que os bits de saída *j* e *k* devem mudar independentemente quando qualquer bit de entrada isolado *i* for invertido, para todo *i*, *j* e *k*. Os critérios SAC e BIC parecem fortalecer a eficácia da função de confusão.

Algoritmo de escalonamento de chave

Com qualquer cifra de bloco de Feistel, a chave é usada a fim de gerar uma subchave para cada rodada. Em geral, gostaríamos de selecionar subchaves de forma a maximizar a dificuldade de deduzir subchaves individuais e de recuperar a chave principal. Nenhum princípio geral para isso foi promulgado até agora.

Adams sugere [ADAM94] que, no mínimo, o escalonamento de chave deve garantir o critério de avalanche estrita e o critério de independência de bit da chave/texto cifrado.

3.6 LEITURA RECOMENDADA

Há muitas informações sobre encriptação simétrica. Algumas das referências mais valiosas são listadas aqui. Um trabalho essencial é [SCHN96]. Esse material incrível contém descrições de praticamente todos os algoritmos criptográficos e protocolos publicados até a publicação deste livro. O autor reúne resultados de jornais, procedimentos de conferência, publicações do governo e documentos padrões, e os organiza em um estudo abrangente e compreensível. Outro estudo valioso e detalhado é [MENE97]. Ainda, um tratamento matemático rigoroso é [STIN06].

As referências anteriores cobrem encriptação de chave pública, e também simétrica.

Talvez a descrição mais detalhada do DES seja [SIMO95]; o livro também contém uma discussão extensa da criptoanálise diferencial e linear do DES. [BARK91] oferece uma análise legível e interessante da estrutura do DES e das potenciais técnicas criptoanalíticas contra este. [EFF98] esmiuça o ataque de força bruta mais eficiente sobre o DES. [COPP94] examina a força inerente do DES e sua capacidade de resistir à criptoanálise. O leitor também poderá achar utilidade no seguinte documento: "The DES Algorithm Illustrated", de J. Orlin Grabbe.

BARK91 Barker, W. *Introduction to the Analysis of the Data Encryption Standard (DES)*. Laguna Hills, CA: Aegean Park Press, 1991.

COPP94 Coppersmith, D. "The Data Encryption Standard (DES) and Its Strength Against Attacks". *IBM Journal of Research and Development*, maio 1994.

EFF98 Electronic Frontier Foundation. *Cracking DES: Secrets of Encryption Research, Wiretap Politics, and Chip Design*. Sebastopol, CA: O'Reilly, 1998.

MENE97 Menezes, A.; van Oorschot, P.; e Vanstone, S. *Handbook of Applied Cryptography*. Boca Raton, FL: CRC Press, 1997.

SCHN96 Schneier, B. *Applied Cryptography*. Nova York: Wiley, 1996.

SIMO95 Simovits, M. *The DES: An Extensive Documentation and Evaluation*. Laguna Hills, CA: Aegean Park Press, 1995.

STIN06 Stinson, D. *Cryptography*: *Theory and Practice*. Boca Raton, FL: Chapman & Hall, 2006.

3.7 PRINCIPAIS TERMOS, PERGUNTAS PARA REVISÃO E PROBLEMAS

Principais termos

chave
cifra de bloco
cifra de Feistel
cifra de produto
confusão
criptoanálise diferencial
Data encryption standard (DES)
difusão
efeito avalanche
função F
mapeamento irreversível
mapeamento reversível
permutação
rodada
subchave
substituição

Perguntas para revisão

3.1 Por que é importante estudar a cifra de Feistel?
3.2 Qual é a diferença entre uma cifra de bloco e uma cifra de fluxo?
3.3 Por que não é prático usar uma cifra de substituição reversível qualquer do tipo mostrado na Tabela 3.1?
3.4 O que é uma cifra de produto?
3.5 Qual é a diferença entre difusão e confusão?
3.6 Que parâmetros e escolhas de projeto determinam o algoritmo real de uma cifra de Feistel?
3.7 Explique o efeito avalanche.

Problemas

3.1 **a.** Na Seção 3.1, sob a subseção a respeito da motivação para a estrutura da cifra de Feistel, foi dito que, a um bloco de n bits, o número de mapeamentos reversíveis para a cifra de bloco ideal é $2^n!$. Justifique.

b. Nessa mesma discussão, afirmou-se que, para a cifra de bloco ideal, que permite todos os possíveis mapeamentos reversíveis, o tamanho da chave é $n \times 2^n$ bits. Mas, se houver $2^n!$ mapeamentos possíveis, serão necessários $\log_2 2^n!$ bits para discriminar entre os diferentes mapeamentos, por isso o tamanho da chave deverá ser $\log_2 2^n!$. Porém, $\log_2 2^n! < n \times 2^n$. Explique a discrepância.

3.2 Considere uma cifra de Feistel composta de 16 rodadas com tamanho de bloco de 128 bits e tamanho de chave de 128 bits. Suponha que, para determinado k, o algoritmo de escalonamento de chave defina valores às oito primeiras chaves de rodada, $k_1, k_2, \ldots k_8$, e depois estabeleça

$$k_9 = k_8, k_{10} = k_7, k_{11} = k_6, \ldots, k_{16} = k_1$$

Admita que você tenha um texto cifrado c. Explique como, com acesso a um oráculo de encriptação, você pode decriptar c e determinar m usando apenas uma única consulta a ele. Isso mostra que tal cifra é vulnerável a um ataque de texto claro escolhido. (Um oráculo de encriptação pode ser imaginado como um dispositivo que, dado um texto claro, retorna o texto cifrado correspondente. Os detalhes internos do dispositivo não são conhecidos, e você não pode abri-lo. Você só consegue obter informações do oráculo fazendo consultas a ele e observando suas respostas.)

3.3 Considere que π seja uma permutação dos inteiros 0, 1, 2, ..., $(2^n - 1)$, tais que $\pi(m)$ dê o valor permutado de m, $0 \leq m < 2^n$. Em outras palavras, π mapeia o conjunto de inteiros de n bits em si mesmo, e dois inteiros quaisquer não são mapeados no mesmo inteiro. DES faz essa permutação para inteiros de 64 bits. Dizemos que π tem um ponto fixo em m se $\pi(m) = m$. Ou seja, se π for um mapeamento de encriptação, então um ponto fixo corresponde a uma mensagem que se encripta para si mesma. Estamos interessados na probabilidade de que π não tenha pontos fixos. Mostre o resultado um tanto inesperado de que mais de 60% dos mapeamentos terão pelo menos um ponto fixo.

3.4 Tenha em conta um algoritmo que encripte blocos de tamanho n, e considere $N = 2^n$. Digamos que tenhamos t pares de texto claro/texto cifrado P_i, $C_i = E(K, P_i)$, onde consideramos que a chave K seleciona um dos $N!$ mapeamentos possíveis. Imagine que queremos encontrar K por busca exaustiva. Poderíamos gerar a chave K' e testar se $C_i = E(K, P_i)$ para $1 \leq i \leq t$. Se K' encriptar cada P_i ao seu C_i apropriado, então teremos evidência de que $K = K'$. Porém, pode ser que os mapeamentos $E(K, \cdot)$ e $E(K', \cdot)$ coincidam exatamente com os t pares de texto claro/texto cifrado P_i,C_i, e não com quaisquer outros pares.

a. Qual é a probabilidade de que $E(K, \cdot)$ e $E(K', \cdot)$ sejam, de fato, mapeamentos distintos?

b. Qual é a probabilidade de que $E(K, \cdot)$ e $E(K', \cdot)$ coincidam com outros pares de texto claro/texto cifrado t' onde $0 \leq t' \leq N - t$?

3.5 Para qualquer cifra de bloco, o fato de que ela é uma função não linear é fundamental à sua segurança. A fim de ver isso, suponha que tenhamos uma cifra de bloco linear EL que encripta blocos de 128 bits de texto claro em blocos de 128 bits de texto cifrado. Considere que EL(k, m) indique a encriptação de uma mensagem de 128 bits m sob uma chave k (o tamanho real em bits de k é irrelevante). Assim,

$$\text{EL}(k, [m_1 \oplus m_2]) = \text{EL}(k, m_1) \oplus \text{EL}(k, m_2) \text{ para todos os padrões de 128 bits } m_1, m_2$$

Descreva como, com 128 textos cifrados escolhidos, um adversário pode decriptar qualquer texto cifrado sem conhecimento da chave secreta k. (Um "texto cifrado escolhido" significa que um adversário tem a capacidade de selecionar um texto cifrado e depois obter sua decriptação. Aqui, você tem 128 pares de texto claro/texto cifrado para trabalhar e a capacidade de escolher o valor dos textos cifrados.)

3.6 Suponha que a função F do DES tenha mapeado cada entrada de 32 bits R, independente do valor da entrada K, para

a. uma *string* de 32 bits de uns,

b. complemento bit a bit de R.

Dica: use as seguintes propriedades da operação XOR:

1. Que função o DES calcularia, então?

2. Como ficaria a decriptação?

$$(A \oplus B) \oplus C = A \oplus (B \oplus C)$$
$$A \oplus A = \mathbf{0}$$
$$A \oplus 0 = A$$
$$A \oplus \mathbf{1} = \text{complemento bit a bit de } A$$

onde

A, B, C são *strings* de bits com n bits
0 é uma *string* de zeros com n bits
1 é uma *string* de uns com n bits

3.7 Mostre que a decriptação DES é realmente o inverso da encriptação DES.

3.8 A troca de 32 bits após a décima sexta iteração do algoritmo DES é necessária para que o processo de encriptação possa ser invertido simplesmente executando-se o texto cifrado de volta pelo algoritmo com a ordem de chaves invertida. Isso foi demonstrado no Problema 3.7. Porém, ainda pode não estar totalmente claro por que a troca de 32 bits é necessária. Para demonstrar a razão, solucione os exercícios a seguir. Primeiro, alguma notação:

$A||B$ = a concatenação das *strings* de bits A e B
$T_i(R||L)$ = a transformação definida pela i-ésima iteração do algoritmo de encriptação, para $1 \le I \le 16$
$TD_i(R||L)$ = a transformação definida pela i-ésima iteração do algoritmo de decriptação, para $1 \le I \le 16$
$T_{17}(R_7||L) = L||R$. Essa transformação ocorre após a décima sexta iteração do algoritmo de encriptação.

a. Mostre que a composição $TD_1(IP(IP^{-1}(T_{17}(T_{16}(L_{15}||R_{15})))))$ é equivalente à transformação que troca as metades de 32 bits, L_{15} e R_{15}. Ou seja, mostre que

$$TD_1(IP(IP^{-1}(T_{17}(T_{16}(L_{15}||R_{15}))))) = R_{15}||L_{15}$$

b. Agora, suponha que abolimos a troca final de 32 bits no algoritmo de encriptação. Então, desejaríamos que a seguinte igualdade se mantivesse:

$$TD_1(IP(IP^{-1}(T_{16}(L_{15}||R_{15})))) = L_{15}||R_{15}$$

Isso acontece?

***Nota*: os problemas a seguir referem-se aos detalhes do DES que estão descritos no Apêndice S (em <sv.pearson.com.br>, em inglês).**

3.9 Considere a substituição definida pela linha 1 da S-box S_1 na Tabela S.2. Mostre um diagrama de bloco semelhante à Figura 3.2, que corresponda a essa substituição.

3.10 Calcule os bits número 1, 16, 33 e 48 na saída da primeira rodada da decriptação DES, supondo que tanto o bloco de texto cifrado quanto a chave externa sejam compostos somente de uns.

3.11 Este problema oferece um exemplo de encriptação usando uma versão do DES com única rodada. Começamos com o mesmo padrão de bits para a chave K e o texto claro, a saber:

em notação hexadecimal: 0 1 2 3 4 5 6 7 8 9 A B C D E F
em notação binária: 0000 0001 0010 0011 0100 0101 0110 0111
1000 1001 1010 1011 1100 1101 1110 1111

a. Derive K_1, a subchave da primeira rodada.

b. Derive L_0, R_0.

c. Expanda R_0 para obter $E[R_0]$, onde $E[\cdot]$ é a função de expansão da Tabela S.1.

d. Calcule $A = E[R_0] \oplus K_1$.

e. Agrupe o resultado de 48 bits de (d) em conjuntos de 6 bits e avalie as substituições de S-box correspondentes.

f. Concatene os resultados de (e) para obter um resultado de 32 bits, B.

g. Aplique a permutação para obter $P(B)$.

h. Calcule $R_1 = P(B) \oplus L_0$.

i. Escreva o texto cifrado.

3.12 Compare a tabela de permutação inicial (Tabela S.1a) com a da escolha permutada um (Tabela S.3b). As estruturas são semelhantes? Se forem, descreva essas semelhanças. A que conclusões podemos chegar a partir dessa análise?

3.13 Ao usar o algoritmo DES para decriptação, as 16 chaves (K_1, K_2, ..., K_{16}) são usadas em ordem inversa. Portanto, o lado direito da Figura S.1 não é mais válido. Projete um esquema de geração de chave com o escalonamento de deslocamento apropriado (semelhante à Tabela S.3d) para o processo de decriptação.

3.14 **a.** Considere que X' seja o complemento bit a bit de X. Prove que, se o complemento do bloco de texto claro for apanhado e o de uma chave de encriptação for tomado, então o resultado da encriptação DES com esses valores é o complemento do texto cifrado original. Ou seja,

$$\text{Se } Y = E(K,X)$$
$$\text{Então } Y' = E(K',X')$$

Dica: comece mostrando que, para duas *strings* de bits quaisquer de mesmo tamanho, A e B, $(A \oplus B)' = A' \oplus B$.

b. Foi dito que um ataque de força bruta no DES exige a busca em um espaço de 2^{56} chaves. O resultado da parte (a) muda isso?

3.15 Mostre que, no DES, os 24 primeiros bits de cada subchave vêm do mesmo subconjunto de 28 bits da chave inicial, e que os próximos 24 bits de cada subchave vêm de um subconjunto disjunto de 28 bits da chave inicial.

***Nota*: os problemas a seguir referem-se ao DES simplificado, descrito no Apêndice G (em <sv.pearson.com.br>, em inglês).**

3.16 Verifique a Figura G.2, que representa a geração de chave para S-DES.

a. Qual é a importância da função de permutação P10 inicial?

b. Qual é a importância das duas funções de deslocamento LS-1?

3.17 As equações para as variáveis q e r para S-DES são definidas na seção sobre análise S-DES. Ofereça as equações para s e t.

3.18 Usando S-DES, decripte a *string* (`10100010`) manualmente com a chave (`0111111101`). Mostre os resultados intermediários depois de cada função (IP, F_K, S_W, F_K, IP^{-1}). Depois, decodifique os primeiros 4 bits da *string* de texto claro para uma letra, e os próximos 4 bits para outra, onde codificamos de A até P na base 2 (ou seja, A = 0000, B = 0001, ..., P = 1111). *Dica*: como uma verificação intermediária, depois da aplicação de SW, uma *string* deverá ser (`00010011`).

Problemas de programação

3.19 Crie um software que possa encriptar e decriptar usando uma cifra de bloco de substituição geral.

3.20 Crie um software que possa encriptar e decriptar usando S-DES. Dados de teste: use texto claro, texto cifrado e chave do Problema 3.18.

Conceitos básicos de teoria dos números e corpos finitos

04

TÓPICOS ABORDADOS

OBJETIVOS DE APRENDIZAGEM

APÓS ESTUDAR ESTE CAPÍTULO, VOCÊ SERÁ CAPAZ DE:

- ☑ Entender o conceito de divisibilidade e o algoritmo de divisão.
- ☑ Entender como usar o algoritmo de Euclides para achar o máximo divisor comum.
- ☑ Apresentar uma visão geral dos conceitos da aritmética modular.
- ☑ Explicar a operação do algoritmo de Euclides estendido.
- ☑ Distinguir entre grupos, anéis e corpos.
- ☑ Definir corpos finitos na forma GF(*p*).
- ☑ Explicar as diferenças entre aritmética polinomial comum, aritmética polinomial com coeficientes em Z_p e aritmética polinomial modular em GF(2^n).
- ☑ Definir corpos finitos na forma GF(2^n).
- ☑ Explicar os dois usos diferentes do operador mod.