Notas em Matemática Aplicada e-ISSN 2236-5915

Volume 52, 2011

Editores

Célia A. Zorzo Barcelos
Universidade Federal de Uberlândia - UFU
Uberlândia, MG, Brasil
Eliana X.L. de Andrade
Universidade Estadual Paulista - UNESP
São José do Rio Preto, SP, Brasil
Maurílio Boaventura
Universidade Estadual Paulista - UNESP
São José do Rio Preto, SP, Brasil

A Sociedade Brasileira de Matemática Aplicada e Computacional - SBMAC publica, desde as primeiras edições do evento, monografias dos cursos que são ministrados nos CNMAC.

Para a comemoração dos 25 anos da SBMAC, que ocorreu durante o XXVI CNMAC, foi criada a série **Notas em Matemática Aplicada** para publicar as monografias dos minicursos ministrados nos CNMAC, o que permaneceu até o XXXIII CNMAC em 2010.

A partir de 2011, a série passa, também, a publicar livros nas áreas de interesse da SBMAC. Os autores que submeterem textos à série Notas em Matemática Aplicada devem estar cientes de que poderão ser convidados a ministrarem minicursos nos eventos patrocinados pela SBMAC, em especial nos CNMAC, sobre assunto a que se refere o texto.

O livro deve ser preparado em **Latex (compatível com o Miktex versão 2.7), as figuras em eps** e deve ter entre **80 e 150 páginas**. O texto deve ser redigido de forma clara, acompanhado de uma excelente revisão bibliográfica e de **exercícios de verificação de aprendizagem** ao final de cada capítulo.

Veja outros títulos publicados em formato e-book na página
http://www.sbmac.org.br/notas.php

Sociedade Brasileira de Matemática Aplicada e Computacional

2011

CRIPTOGRAFIA

Antonio Cândido Faleiros

antonio.faleiros@ufabc.edu.br

Centro de Matemática, Computação e Cognição
Universidade Federal do ABC

Sociedade Brasileira de Matemática Aplicada e Computacional

São Carlos - SP, Brasil
2011

Coordenação Editorial: Elbert Einstein Nehrer Macau

Coordenação Editorial da Série: Eliana Xavier Linhares de Andrade

Editora: SBMAC

Capa: Matheus Botossi Trindade

Patrocínio: SBMAC

Catalogação elaborada pela Biblioteca do IBILCE/UNESP
Bibiotecária: Maria Luiza Fernandes Jardim Froner

Faleiros, Antonio Cândido
Criptografia - São Carlos, SP : SBMAC, 2011, 138 p., (Notas em Matemática Aplicada; v. 52)

e-ISBN 978-85-86883-54-5

1. Criptografia 2. Criptografia simétrica 3. Criptografia de chave pública I. Faleiros, Antonio Cândido II. Título. III. Série.

CDD - 51

Conteúdo

Prefácio

Este livro foi escrito ao longo de anos e em diversas etapas. Uma versão inicial do primeiro capítulo tomou forma a partir de um pequeno esboço datilografado que foi oferecido em seminário no Departamento de Física do ITA. A partir de muitas notas esparsas, foi remodelado em pouco mais de uma semana para ser apresentado no 52^o Seminário Brasileiro de Análise, que aconteceu no ITA em comemoração aos seus 50 anos.

Àquela época, durante os dias e madrugadas que se seguiram, dei asas à minha imaginação e deixei meu espírito voar sobre o oceano do saber, pousando, aqui e acolá, nas ilhas da fantasia. Segui de perto Isadora Duncan, famosa bailarina que, em uma passagem célebre de sua vida afirmou: "Nunca pude compreender que alguém pudesse ter vontade de fazer alguma coisa e deixasse de fazê-la. Tive a satisfação de abrir o meu caminho, mesmo quando fazer o que desejava resultasse em trágicas conseqüências. Fazer o que se quer é a melhor solução contra a angústia e a frustração. Minha arte é um esforço para expressar a verdade do meu ser, sob a forma de gesto e movimento. A dança que eu criei é a expressão da minha própria liberdade."Não recomendo que a sigam até as últimas conseqüências, mas ser livre é uma das aspirações de todo ser humano. Escrevi o que sentia e senti-me livre para fazê-lo, objetivando uma leitura amena, sem me esquecer da proposta de oferecer um curso introdutório.

Quando estiverem lendo este livro e duvidarem da seriedade de minhas palavras, podem ter certeza, nem sempre estive falando sério. Por vezes, eu estive brincando. Acredito seriamente que o aprender não deve se desprender do prazer, pois senão... Permitam-me citar a quem dispensa apresentações: "É de fato um milagre que os métodos modernos de instrução não tenham estrangulado inteiramente a santa curiosidade da pesquisa pois, esta delicada planta precisa, além de estímulo, principalmente de liberdade; sem isso, ela se arruinará infalivelmente. É um erro muito grave pensar que o prazer de examinar e indagar possa ser promovido por meio da coerção e do senso do dever." – Albert Einstein.

Desde então o material apresentado foi ampliado e oferecido em cursos de Criptografia que ministrei no ITA e na UFABC, indo muito além daquilo que se

pode aprender em algumas poucas horas. Minha pretensão foi a de oferecer a você, leitor, material para prosseguir nesta jornada, na eventualidade bem plausível de se apaixonar por esta bela página da ciência.

Um livro não se faz sozinho. Ele surge acompanhado pelo estímulo de muitos e, neste momento, não há como esquecer as pessoas que estiveram ao meu lado. Meus filhos sempre foram fontes de constante inspiração. No caso deste livro em particular, o Júnior sempre me encorajou e, sob suas palavras de ânimo, ele tomou forma. Minha esposa Maria da Graça aceitou com paciência e me amparou nos períodos de trabalho mais intenso. Muito aprendi com os alunos e colegas do ITA. Eles apresentaram diversas sugestões e foram responsáveis pela evolução e aprimoramento das notas iniciais. Ao ITA cheguei, recém formado em Matemática, bolsista da FAPESP, para fazer mestrado em Mecânica Celeste e lá permaneci por muitos e saudosos anos, entre aluno e professor. Meus alunos e colegas do Centro de Matemática, Computação e Cognição da UFABC, local onde estas notas adquiriram o formato atual, forneceram o ambiente ideal para completar este trabalho. Em 2010 uma versão levemente modificada deste livro foi apresentada no I Encontro em Teoria dos Códigos e Criptografia, ocorrido na UFABC. Dentre os novos colegas, faço menção especial ao professor Ercílio Carvalho da Silva, que desde o primeiro momento acreditou na publicação das notas em forma de livro, e ao professor Igor Leite Freire, que sugeriu a sua submissão ao CMAC - SUDESTE 2011.

Antonio Cândido Faleiros
UFABC, junho de 2011

Capítulo 1

Criptografia primitiva

1.1 Alerta

Os métodos que serão apresentados neste capítulo fazem parte da era dos cifrários com lápis e papel. Todos são inseguros contra um ataque via computador e podem ser decodificados por uma análise paciente feita apenas com lápis e papel, sem o uso de qualquer dispositivo mecânico ou eletrônico. Nosso objetivo é o de divulgar os primórdios desta empolgante área do conhecimento humano e que, nesta era das comunicações através de ondas eletromagnéticas, seja pela televisão, rádio, telefone ou internet, é uma ferramenta indispensável para manter a segurança e privacidade das comunicações.

O leitor irá se deparar com alguns programas de computador desenvolvidos para implementar no computador os métodos apresentados. Neles, não houve compromisso com a eficiência mas com a compreensão e legibilidade. Quem os escreveu não foi o programador mas o professor.

Na realização dos programas usei o Mathematica, da Wolfram Research, e a linguagem C. Quem pretende iniciar seus estudos no Mathematica, poderá ler o livro de Faleiros [1], citado na bibliografia. Posso assegurar que esta recomendação nada tem a ver com o fato de que gosto muito do autor.

1.2 Introdução

Desde os tempos remotos o homem vem desenvolvendo e utilizando técnicas que possibilitam a troca de mensagens secretas que podem ser facilmente interpretadas e lidas pelo receptor autorizado mas dificilmente por aqueles que a interceptam sem autorização do remetente ou do destinatário. Seja pelo seu teor passional, diplomático, comercial ou bélico, muitas mensagens precisavam ser mantidas em

segredo. Muitos amantes e não poucos revolucionários perderam a vida por usarem métodos que se mostraram frágeis, permitindo a fácil decodificação das mensagens por pessoas não autorizadas. Em tempos de guerra, a troca de mensagens secretas é fundamental para o sucesso de uma empreitada. A Criptografia nem sempre esteve ligada a tragédias. Nossas compras pela internet e as comunicações que efetuamos com nossa conta bancária são protegidas pela Criptografia.

Todos nos sentimos fascinados pelo mistério que cerca uma mensagem secreta. Diversos autores exploraram este fascínio para dar maior brilho e sabor às suas obras. Piratas que escondiam seus tesouros, amantes com seus amores impossíveis, idealistas querendo derrubar o poder vigente.

A **Criptografia** (do grego, kryptos = secreto) é a ciência - nos primórdios da humanidade era uma arte - que se dedica ao desenvolvimento dos **cifrários** nomes que recebem as técnicas destinadas a tornar secreta uma mensagem. Antes de terem sido criptografadas, as mensagens são denominadas de **mensagem**, **texto claro**, **texto original**. Aquelas tornadas secretas pela criptografia são chamadas **textos cifrados** ou **criptografados** e ainda, **mensagens cifradas** ou **criptografadas**. Um cifrário é considerado **seguro** ou **forte** quando as mensagens por ele codificadas não puderem ser decodificadas em um tempo razoável por pessoas ou equipes não autorizadas. O conceito de tempo razoável é um pouco volátil, pois depende daquilo que se pretende manter secreto. Alguns segredos se tornam inócuos em pouco tempo ao passo que outros devem ser mantidos por longo período. O conceito de forte evolui com o tempo. O advento do computador tornou frágeis cifrários que, no período pré-computacional, eram considerados seguros. Quando se consegue, de maneira não autorizada, decodificar sistematicamente as mensagens cifradas por um código criptográfico, diz-se que ele foi **quebrado**.

Ao desenvolver um novo cifrário, outra equipe, diferente daquela que o desenvolveu, deve testar sua segurança, tentando quebrá-lo mediante o uso de todos os recursos disponíveis. É neste ponto que entra a **Criptoanálise**, ciência que busca na Matemática, ferramentas que permitem verificar a segurança de um método criptográfico. Nesta área existe criatividade mas, principalmente, competência e profissionalismo. Um método será considerado seguro apenas depois de ter sido testado por um longo período de tempo, durante o qual ele se apresenta inquebrável ao ataque dos mais experientes criptoanalistas. Dá-se o nome de **Criptologia** à ciência que engloba a Criptografia e a Criptoanálise.

Hoje, neste mundo globalizado, a Criptografia retoma seu papel de destaque entre as ciências. Quando nos comunicamos com nossa conta bancária através da internet, a manutenção do sigilo é fundamental para a nossa saúde financeira. A própria segurança nacional pode ser afetada se o sigilo de comunicações governamentais for comprometido. Por volta de um trilhão de reais circula todos os dias pela compensação interbancária, controlada pelo Banco Central do Brasil e protegida pela Criptografia. Numa sociedade em que um jovem solitário, à frente de

seu notebook, pode invadir computadores de corporações que pretensamente são seguros, que garantias teremos contra um estado opositor, tendo à sua disposição grande poder computacional e equipes de especialistas? No simples acionar de um programa, pode-se desorganizar todo o sistema bancário do país.

A história da Criptologia se divide em três fases distintas. Na primeira, a dos cifrários que usavam apenas lápis e papel. Na segunda, a do telégrafo, com o código Morse e as máquinas eletro-mecânicas (o Enigma da Alemanha e a Púrpura do Japão). Na terceira, a dos cifrários da época computacional. Está sugindo uma outra fase, a que envolve técnicas resistentes ao ataque de um computador quântico, tecnologia que se encontra em sua infância. Neste primeiro capítulo iremos nos ater principalmente aos métodos da primeira fase. Passemos à descrição de algumas técnicas primitivas

1.3 Princípios básicos

Os textos são formados por sequências de caracteres que não precisam, necessariamente, pertencer ao alfabeto romano. Designaremos os textos por letras em negrito, $\mathbf{x}$, $\mathbf{y}$, ..., e seus caracteres individuais por letras indexadas sem negrito, x_i, y_i, ..., ou apenas por letras sem negrito x, y, ... Com esta convenção um texto será indicado por

$$\mathbf{x} = x_1 x_2 x_3 \dots x_n$$

onde $\mathbf{x}$ representa o texto todo (ou um bloco do texto) e x_1, x_2, ..., x_n são os seus caracteres, que podem ser letras, dígitos, símbolos, espaços e pontuações.

Para criptografar um texto, é preciso transformá-lo em outro. Para tanto, aplica-se a cada caractere do texto original x_i uma transformação c que o cifra obtendo o caractere

$$y_i = c(x_i)$$

Esta transformação c, que depende de uma chave k que será explicitada em cada técnica apresentada, recebe o nome de **cifrador** ou **criptografador**. Sendo

$$\mathbf{y} = y_1 y_2 y_3 \dots y_n.$$

o texto cifrado, escrevemos $\mathbf{y} = c(\mathbf{x})$. Para ser possível decifrar a mensagem $\mathbf{y}$, é preciso inverter a tranformação c. Sua inversa d leva os caracteres y_i texto cifrado de volta aos caracteres x_i do texto original e escrevemos $x_i = d(y_i)$ ou

$$\mathbf{x} = d(\mathbf{y}).$$

A função d recebe o nome de **decifrador** ou **decriptografador**.

Nota 1.1. *Os símbolos x_i do texto claro e y_i do texto cifrado não precisam pertencer ao mesmo alfabeto.*

Nos bons métodos criptográficos, $c(x_i)$ deve ser fácil de calcular mas, conhecido y_i, deve ser muito difícil calcular $x_i = d(y_i)$ sem o conhecimento da chave apropriada. Isto impede que pessoas indevidas se apossem do texto cifrado **y** e recuperem a mensagem **x**. Apenas aqueles que possuem a chave adequada é que conseguem calcular facilmente o **x** estando em posse de **y**. **Chave** é a informação adicional necessária para cifrar e tornar fácil a decifração. Nas boas técnicas criptográficas, mesmo conhecendo o texto cifrado **y** e o mecanismo de ação do decifrador d não há como recuperar facilmente o **x** sem o conhecimento da chave. Este fato impede ou dificulta que uma pessoa não autorizada, que não conhece a chave, consiga recuperar o conteúdo do texto original **x** mesmo possuindo **y**.

1.4 Cifrário de César

Conta-nos o historiador Suetônio, em sua obra A Vida dos Césares, que Julio César usava um cifrário para se comunicar com seus generais. Era bastante simples, mas adequado para uma época em que poucos eram alfabetizados. Seu cifrário consistia em um disco dividido em 21 setores iguais. Na borda do disco, distribuídas pelos 21 setores, as letras do alfabeto (na época, as letras J, U, W, Y, Z ainda não haviam sido incluídas no alfabeto latino) eram dispostas em ordem alfabética no sentido horário, formando um anel de letras. Para criptografar uma mensagem, cada letra era substituída por outra que ficava três setores adiante, percorrendo o anel no sentido horário. Ao receber a mensagem, o general decodificava o texto, realizando a operação contrária, substituindo cada letra da mensagem cifrada pela que ficava a três setores dela, percorridos no sentido anti-horário.

A correspondência feita pelos escribas de César entre as letras do texto original (letras minúsculas) e do texto criptografado (letras maiúsculas) é descrita pela tabela abaixo

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

Neste quadro trabalhamos com o alfabeto atual, incluindo as letras J, U, W, Y, Z que não existiam na época. O cifrário de César também é conhecido como **cifrário por rotação**.

Um dia, assistindo a televisão, tive o imenso prazer de ver e ouvir Cora Coralina, pseudônimo de Ana Lins dos Guimarães Peixoto Bretas. Nascida na cidade de Goiás, em 20 de agosto de 1889, veio a faleceu em Goiânia no dia 10 de abril de

1985. Esta senhora, já avançada em anos à época, irradiava uma grande energia e transmitia àqueles que a ouviam um intenso otimismo e amor pela vida. Em sua homenagem, vamos transcrever um trecho de seu poema denominado Aos Moços:

Eu sou aquela mulher
a quem o tempo
muito ensinou.
Ensinou a amar a vida.
Não desistir da luta.
Recomeçar na derrota.
Renunciar a palavras e pensamentos negativos.
Acreditar nos valores humanos.
Ser otimista.

O trecho

Eu sou aquela mulher a quem o tempo muito ensinou

ao ser codificado pelo método de César se torna

HX VRX DTXHOD PXOKHU D TXHP R WHPSR PXLWR HQVLQRX

O segredo de César, além do desconhecimento da técnica utilizada, repousava no número de caracteres deslocados ao longo do disco que, neste caso, é 3. Tanto a técnica quanto o número de setores girados eram desconhecidos pelos inimigos. Conhecida a técnica, o segredo, frágil por sinal, se apoiava no número de setores usados para deslocar cada letra ao cifrar a mensagem. Este número de setores é a chave **k** que abre as portas da técnica de César à decifração.

Ainda nos relata o historiador Suetônio que o Imperador Augusto, sobrinho de César, usava o mesmo disco, deslocando os caracteres uma única posição no sentido horário, caso em que podemos identificar a chave **k** ao número 1. Percebe-se que é possível obter variações do cifrário de César, atribuindo valores diferentes à chave **k**, que pode ser um inteiro entre 1 e 25. Com o alfabeto atual, se as letras fossem transladadas 26 setores no disco de César, retornaríamos às letras originais.

Logo se nota que este método não é seguro pois possui apenas 25 chaves distintas. Sendo conhecida a técnica, o texto original pode ser recuperado testando todas as chaves possíveis até encontrar um texto que faça algum sentido. Esta é a chamada **busca exaustiva da chave**.

Numa técnica criptográfica qualquer, o segredo que permite decifrar as mensagens cifradas com facilidade denomina-se **chave**.

Possuir muitas chaves é uma condição necessária para garantir a segurança de uma técnica criptográfica mas, como veremos, não é suficiente. Existem técnicas de criptografar com um número enorme de chaves e que pode ser quebrada com lápis, papel, conhecimento e muita paciência.

1.5 O atbash hebraico

O atbash hebraico é conhecido desde o Antigo Testamento Bíblico, tendo sido usado no Livro de Jeremias para cifrar o nome da Babilônia. Não iremos usar o alfabeto hebraico mas sim o latino, com as atuais 26 letras. Neste cifrário, dobra-se o alfabeto ao meio e dispõe-se a segunda metade sob a primeira, como no esquema abaixo

$$\begin{array}{cccccccccccccc} a & b & c & d & e & f & g & h & i & j & k & l & m \\ \updownarrow & \updownarrow & \updownarrow & \updownarrow & \updownarrow & \updownarrow & \updownarrow & \updownarrow & \updownarrow & \updownarrow & \updownarrow & \updownarrow & \updownarrow \\ z & y & x & w & v & u & t & s & r & q & p & o & n \end{array}$$

Na primeira linha as letras estão dispostas em ordem alfabética crescente e, na segunda linha, em ordem alfabética decrescente.

Ao criptografar o texto claro cada letra é substituída pela outra que se encontra na mesma coluna da tabela. A decifragem se faz mediante a mesma regra. Cifrário onde o processo que cifra é idêntico ao que decifra é chamado **involutivo**. Neles, cifrar uma mensagem cifrada faz recuperar a mensagem original. Como veremos, o Atbash é um caso particular de cifrário por substituição e sua chave consiste na própria tabela de substituição.

O refrão

não há oh gente oh não luar como este do sertão

da bela toada Luar do Sertão, dos autores Catulo da Paixão Cearense e João Pernambuco, ao ter seus acentos gráficos retirados e sendo codificado com o Atbash, se transforma em

MZL SZ LS TVMGV LS MZL OFZI XLNL VHGV WL HVIGZL

Retiramos os acentos para dificultar o trabalho de um eventual receptador não autorizado que tenha se apossado do texto cifrado de forma indevida e queira conhecer seu conteúdo. Aliás, para tornar os métodos primitivos um pouco mais seguros, deve-se retirar da mensagem original os acentos, as pontuações, as separações silábicas, os erres e os esses duplos, além de escrever os números por extenso.

1.6 A cítala espartana

O historiador grego Plutarco, contemporâneo do historiador romano Suetônio, nos relata o uso da cítala espartana por volta do ano 400 antes de Cristo, através da qual os éforos (governantes) se comunicavam secretamente com os estrategos

(generais). Para maiores detalhes consulte o livro de Sgarro [20], uma pequena história da criptografia, escrita em grande estilo.

A cítala era formada por um cilindro de madeira, no qual se enrolava uma tira de pergaminho ou papiro, sem sobrepor uma parte sobre a outra, como num carretel de corda. O escriba quebrava a mensagem em blocos de caracteres, todos do mesmo tamanho, à exceção do último bloco, que poderia ter um número menor de caracteres. Neste bloco pode-se inserir **nulas**, que são letras escolhidas ao acaso e inseridas num texto para completá-lo, devendo ser escolhidas de modo a não alterar o sentido da mensagem quando ela for lida em seu destino. As letras de cada bloco eram escrita ao longo do eixo do cilindro, de cima para baixo, cada letra sendo escrita em um laço diferente da fita, na mesma vertical. Os diferentes blocos de texto eram distribuídos uniformemente em redor do cilindro. Ao desenrolar a fita, a mensagem nela impressa está criptografada. Quando o mensageiro entregava a fita ao destinatário, este a enrolava num bastão com o mesmo diâmetro que aquele usado para criptografar. Com este expediente, a mensagem podia ser lida coluna por coluna ao longo do eixo do cilindro.

O número de colunas usadas para transcrever o texto ao longo da superfície do cilindro é a chave deste método. Se forem usadas 6 colunas, a chave é $k = 6$. Poder-se-ia também mudar o diâmetro do bastão e torná-lo parte da chave.

Em síntese, a cítala é equivalente ao método que consiste em escrever o texto nas células de uma tabela, preenchendo uma coluna por vez e reescrevendo o texto linha por linha, partindo da primeira e prosseguindo até a última. A chave é o número de colunas da tabela.

Vamos criptografar com este método um trecho clássico da literatura brasileira

sou bravo sou forte sou filho do norte meu canto de morte guerreiros ouvi

do poema Juca Pirama, de (brasileiro, 1823–1864). Vamos supor que o bastão é suficientemente grosso para que se possa escrever nove colunas de texto ao longo do seu eixo. Dividimos o texto em seis partes que serão transcritos na fita enrolada no sentido vertical, isto é, do eixo do bastão sendo que completamos a coluna final com três nulas.

s	o	t	l	r	a	o	r	o
o	s	e	h	t	n	r	r	u
u	o	s	o	e	t	t	e	v
b	u	o	d	m	o	e	i	i
r	f	u	o	e	d	g	r	x
a	o	f	n	u	e	u	o	t
v	r	i	o	c	m	e	s	b

Retiramos os espaços entre as palavras e os acentos gráficos para dificultar a quebra da mensagem. Este expediente era usualmente usado pelos criptólogos da época pois ele dificultava o trabalhos dos criptoanalistas. Poderíamos ainda ter retirado o duplo erre da palavra guerreiro. Depois de desenrolar a fita, ficamos com o texto criptografado

SOTLRAORO OSEHTNRRU UOSOETTEV BUODMOEII
RFUOEDGRX AOFNUEUOT VRIOCMESB

que é suficientemente complicado para desanimar um leigo na tentativa de descobrir o seu sentido original. Todavia, ele não intimida um criptoanalista experiente. Os espaços inseridos no texto criptografado foram colocados a cada 9 letras apenas para facilitar a sua "leitura". Numa criptografia real eles não existiriam pois tais espaços iriam auxiliar um eventual criptoanalista a inferir o número de colunas usadas para cifar o texto.

1.7 Transposição colunar

A técnica de transposição colunar é de natureza semelhante à cítala espartana. Nela, o texto é dividido em blocos de n caracteres que vão sendo posicionados sucessivamente abaixo dos anteriores. Em seguida, transpõem-se as colunas de acordo com uma regra pré-definida entre o mensageiro e o destinatário. Finalmente o texto é copiado. A cópia pode ser feita linha por linha ou coluna por coluna, de cima para baixo e da esquerda para a direita.

Vamos a um exemplo. Deste belo trecho de Iracema, romance de José de Alencar, um dos nossos mais celebrados escritores brasileiros, que nasceu em Fortaleza, no dia 1 de maio de 1829, e faleceu no Rio de Janeiro, em 12 de dezembro de 1877:

Mais rápida que a ema selvagem,
a morena virgem corria o sertão pelas matas do Ipu,
onde campeava sua guerreira tribo,
da grande nação tabajara

vamos criptografar a frase

Mais rápida que a ema selvagem

Depois de retirar os espaços entre as palavras, os acentos, os sinais gráficos, substituir o c com cedilha pela letra c, eliminando os duplos erres e esses, quebramos o

texto original em linhas com 9 caracteres e completamos a última linha com duas nulas, b e k. Colocamos uma linha sobre a outra, de modo a termos blocos de texto com 9 colunas, numeradas de 1 a 9.

1	2	3	4	5	6	7	8	9
m	a	i	s	r	a	p	i	d
a	q	u	e	a	e	m	a	s
e	l	v	a	g	e	m	b	k

Em seguida, vamos transpor as colunas, escrevendo-as em outra ordem: primeiro a coluna 4, seguida pelas colunas 3, 8, 5, 1, 9, 6, 2 e 7. Nota-se que, neste método, a chave necessária para cifrar e decifrar é o número 438519627, que fornece a ordem na qual as colunas ficarão dispostas no texto codificado.

4	3	8	5	1	9	6	2	7
s	i	i	r	m	d	a	a	p
e	u	a	a	a	s	e	q	m
a	v	b	g	e	k	e	l	m

Transpostas as colunas, copia-se o texto. A cópia pode ser feita linha por linha ou coluna por coluna. Vamos copiá-lo linha por linha para obter

$$SIIRMDAAP\ EUAAASEQM\ AVBGEKELM$$

onde os espaços foram introduzidos apenas para facilitar a "leitura". Numa criptografia real, esta separação silábica auxiliar não ocorre. A pessoa que recebe a mensagem, estando em posse da chave, que informa a posição original das colunas é capaz de recuperar a mensagem original.

Informa-nos David Kahn em The Codebreakers[11], um dos maiores clássicos da história da Criptografia, que os japoneses usaram este método, aliado a uma tabela de transposição dos ideogramas japoneses para letras latinas, para criptografar suas mensagens diplomáticas de baixa segurança durante o período que antecedeu a Segunda Guerra Mundial. Estas mensagens foram sistematicamente interceptadas pelos americanos e decodificadas num período que variava de 6 horas a 6 dias, sendo 3 dias uma boa média. Este esforço de espionagem e a distribuição das mensagens às autoridades americanas recebeu o nome de MAGIC. Numa tentativa de impedir que os americanos decifrassem seus textos, os japoneses inseriam espaços em branco no meio da mensagem, em posições previamente combinadas entre o remetente e o destinatário. Este expediente retardava mas não impedia a decodificação da mensagem.

Os japoneses usavam também o **cifrário niilista**, chamado ainda de **cifrário de dois tempos**: depois de transpor as colunas, se transpõe as linhas. Neste cifrário a chave é formada por dois números, onde um é usado para transpor as colunas e o outro para transpor as linhas.

1.8 Substituição e transposição

No cifrário de César e no atbash hebraico, ao criptografar o texto claro, suas letras são substituídas por outras. Técnicas com esta característica são denominados **cifrários por substituição**.

Na cítala espartana e na transposição colunar, as letras do texto original e do texto criptografado são as mesmas. O expediente adotado é o de embaralhar as letras do texto original. Técnicas com esta característica são denominados de **cifrário por transposição**.

1.9 Cifrários por substituição

Para estabelecer uma **cifra por substituição** usando o nosso alfabeto, construa uma tabela com duas linhas e 26 colunas. Em cada linha, coloque as 26 letras do alfabeto, sendo que, na primeira linhas, coloque-as em orfem alfabética e na segunda linha, disponha as letras de forma arbitrária, à sua escolha.

Para exemplificar, consideremos a tabela abaixo. Para criptografar uma mensagem, as letras minúsculas do texto claro serão levadas nas letras maiúsculas que ficam na mesma coluna da linha de baixo.

a	b	c	d	e	f	g	h	i	j	k	l	m
↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕
N	K	G	F	P	I	J	B	T	M	V	D	Z

n	o	p	q	r	s	t	u	v	w	x	y	z
↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕
L	R	H	W	Q	A	Y	S	C	X	U	E	O

Vamos usar esta tabela para criptografar um trecho do poema Romantismo de Cecília Meireles, que nasceu no Rio de Janeiro, em 7 de novembro de 1901 e partiu para a morada eterna nesta mesma cidade em 9 de novembro de 1964

Nas pedras, à sombra, sentados,
respiraremos a frescura
dos verdes reinos encantados
das lianas e da fonte pura.

Ao cifrá-lo pelo método da substituição usando a tabela acima obtemos

LNA HPFQNA N ARZKQN APLYNFRA
QPAHTQNQPZRA N IQPAGSQN
FRA CPQFPA QPTLRA PLGNLYNFRA
FNA DTNLNA P FN IRLYP HSQN

Para construir uma tabela de susbstituição, fazemos uma permutação do alfabeto. Cada permutação fornece uma tabela diferente. Cada tabela é uma chave que abre as portas para a decifra. O número de chaves deste método é igual ao número de permutações das 26 letras do alfabeto, que é igual a

$$26! = 403\ 291\ 4\,61\ 126\ 605\ 635\ 584\ 000\ 000$$

ou seja, mais de 400 setilhões de chaves. Sem dúvida, um número consideravelmente grande. A despeito desta enormidade de chaves este método é inseguro.

A sua criptoanálise se faz com eficiência através da contagem do número de vezes (freqüência) com que cada caractere aparece no texto. Cada idioma tem um padrão de ocorrência das diversas letras do seu alfabeto. Pode-se obter este padrão contando o número de vezes que as letras ocorrem em jornais, revistas e romances. Hoje isto pode ser feito rapidamente usando um escaner, um programa para o reconhecimento ótico de caracteres e outro para contar a freqüência das letras. Como uma letra do texto em claro é levada sempre na mesma letra do alfabeto, esta frequência se mantém texto criptografado. Comparando a freqüência com que as letras aparecem no texto criptografado com a freqüência das letras nos textos usuais do idioma, pode-se ter uma indicação a respeito da correspondência entre as letras do texto claro e do cifrado. Quanto maior a for o texto cifrado, mais fácil será decodificá-lo, uma vez que as frequências das letras cifradas se aproximam daquelas que lhe correspondem nos textos da lingua usada.

Para dissimular a alta ocorrência de uma letra do alfabeto e aumentar a segurança da técnica de substituição, pode-se usar dois ou mais caracteres para substitui-la. A letra "e" ocorre com freqüência muito alta em textos da língua portuguesa. Poder-se ia usar duas letras como J e Q para substitui-la no texto cifrado e ainda poder-se ia levar duas letras distintas como j e q, de baixa ocorrência em português, numa mesma letra no texto cifrado. Para quem conhece a regra de substituição, este procedimento não dificulta o processo de decifra mas dificulta bastante a criptoanálise de frequência do texto cifrado.

Usando palavras chave

Uma das maiores dificuldades de se usar um cifrário por substituição genérico consiste em gravar a tabela de conversão entre as letras. Gravá-la na memória é difícil e registrá-la em papel é inseguro.

Uma técnica de substituição com uma chave simples de se lembrar é aquela na qual se estabelece uma palavra como chave. Por exemplo, se usarmos a palavra BRASIL como chave, podemos colocá-la no início da tabela de substituição e completar a tabela com as letras ainda não utilizadas do alfabeto, como mostra o exemplo abaixo. Numa criptografia real, a palavra BRASIL não é uma boa escolha. Deve-se escolher palavras ou chaves difíceis de serem obtidas num processo de busca através de um dicionário.

a	b	c	d	e	f	g	h	i	j	k	l	m
B	R	A	S	I	L	C	D	E	F	G	H	J

n	o	p	q	r	s	t	u	v	w	x	y	z
K	M	N	O	P	Q	T	U	V	W	X	Y	Z

Quando a palavra ou texto chave possuir letras repetidas, apenas sua primeira ocorrência é usada e as demais são omitidas. A vantagem deste método reside na facilidade de guardar a chave que agora se restringe à lembrança de uma palavra ou um texto fácil de decorar.

1.10 Cifrários por transposição

A cítala espartana e a transposição colunar são cifrários por transposição. As letras usadas no texto original são embaralhadas e mantidas no texto cifrado. Existem muitos outros com esta característica que criptografam a mensagem mantendo as letras do texto original e apenas embaralhando suas letras. A partir de agora descreveremos alguns.

1.10.1 Permutação

Uma **permutação** do conjunto $\{1, 2, \ldots, n\}$ é uma função bijetora π deste conjunto nele mesmo

$$\pi : \{1, 2, \ldots, n\} \to \{1, 2, \ldots, n\}\,.$$

Por exemplo, a função $\pi : \{1, 2, 3\} \to \{1, 2, 3\}$ definida por

$$\pi(1) = 3, \quad \pi(2) = 1, \quad \pi(3) = 2$$

é uma permutação do conjunto $\{1, 2, 3\}$. Identificamos a permutação π com o terno ordenado, lavado, passado e engomado $(3, 1, 2)$ (não consegui me conter e me dei a liberdade de fazer essa brincadeira pois, o que seria da Ciência, não fosse o prazer

de com ela brincar). Esta identificação se deve ao fato de que (3, 1, 2) fornece todos os dados necessários para reconstruir a função π pois ele lista $\pi(1)$, $\pi(2)$ e $\pi(3)$, nesta ordem. Por esta razão, se escreve

$$\pi = (3, 1, 2)\,.$$

Para esta técnica criptográfica, a chave é uma permutação do conjunto $\{1, 2, \ldots, n\}$. Seu mecanismo é o seguinte: Combinada uma permutação $\pi = (k_1, k_2, \ldots k_n)$ entre as pessoas que irão trocar mensagens secretas, para cifrar um texto, quebre-o em blocos de tamanho fixo, cada um contendo n caracteres $x_1x_2\ldots x_n$. Se o último bloco não possuir n caracteres, complete-o com nulas. Cada bloco é cifrado usando a regra

$$\begin{aligned} y_1 &= x_{k_1} \\ y_2 &= x_{k_2} \\ &\vdots \\ y_n &= x_{k_n} \end{aligned}$$

gerando um bloco cifrado $y_1y_2\ldots y_n$ contendo n caracteres. Para não fornecer a um eventual espião a informação do tamanho n da permutação pode-se inserir de 1 a $n-1$ nulas no final do texto cifrado.

A permutação inversa $\pi^{-1} = (s_1, s_2, \ldots, s_n)$ existe pois π é bijetora. Para decodificar, basta aplicar a mesma regra que criptografa usando π^{-1}

$$\begin{aligned} x_1 &= y_{s_1} \\ x_2 &= y_{s_2} \\ &\vdots \\ x_n &= y_{s_n} \end{aligned}$$

Exemplo 1.1. *A permutação* $\pi = (3, 1, 4, 2)$ *leva* $(1, 2, 3, 4)$ *em* $(3, 1, 4, 2)$. *Sua inversa* π^{-1} *leva* $(1, 2, 3, 4)$ *em* $(2, 4, 1, 3)$. *Esquematicamente,*

$$(1,2,3,4) \xrightarrow{\pi} (3,1,4,2) \qquad e \qquad (1,2,3,4) \xrightarrow{\pi^{-1}} (2,4,1,3).$$

ou

	π					π^{-1}			
1	2	3	4		1	2	3	4
↓	↓	↓	↓	*invertendo*	↓	↓	↓	↓
3	1	4	2		2	4	1	3

Queremos usar esta permutação para criptografar um fragmento da música Paz do Meu Amor

Você é isso, uma beleza imensa,
toda a recompensa de um amor sem fim.
Você é isso, uma nuvem calma,
no céu de minh'alma, é ternura em mim.

composta pelo querido mestre Luiz Vieira, sob as canções de quem, minha geração amou e sonhou. Aqueles que quizerem apreciar este e outros versos deste compositor e cantor poderão se dirigir ao seu site

www.luizvieira.com.br.

Depois de retirar os acentos, os espaços entre as palavras, os duplos erres e esses que eventualmente existirem, o texto deve ser dividido em grupos de quatro letras. Em cada grupo copie as letras seguindo a ordem $(3, 1, 4, 2)$

v	o	c	e	e	i	s	o	u	m	a	b	e	l	e	z
3	1	4	2	3	1	4	2	3	1	4	2	3	1	4	2
↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
C	V	E	O	S	E	O	I	A	U	B	M	E	E	Z	L

resultando no texto criptografado

CVEO SEOI AUBM EEZL

que, certamente, deve ser enviado sem os espaços entre os blocos e, possivelmente, com nulas no seu final. Este expediente deve ser considerado sempre que várias mensagens forem cifradas com a mesma permutação. Sem as nulas ao final do texto cifrado, o comprimento de todos eles será múltiplo de quatro. Isto deve ser evitado pois forneceria ao criptoanalista o tamanho da chave. Contorna-se este problema inserindo nulas contendo de 1 a 3 caracteres no final de algumas mensagens criptografadas.

À medida que aumentamos o número de elementos da permutação, o método vai ficando mais forte. Todavia, este cifrário, bem como todos os outros que serão citados neste capítulo podem ser quebrados facilmente, usando apenas lápis e papel. Um livro interessante sobre a criptoanálise de técnicas primitivas é o Cryptanalysis de Elen Fouché Gaines [8].

1.10.2 A máscara de Matias Sandorf

Muitos escritores usaram a criptografia para dar uma pitada de emoção aos seus contos. Dentre os mais famosos, citamos Edgard Allan Poe (norte americano, 1809 – 1849), com O Escaravelho de Ouro, e Júlio Verne (francês, 1828–1905), com os livros Viagem ao Centro da Terra, A Jangada e Matias Sandorf. Os dois autores conheciam algumas técnicas criptográficas da época e delas lançaram mãos para dar um tempero especial de mistério em seus livros. Além destes autores, muitos outros fizeram uso da criptografia, seja através de micropontos, onde se usam letras microscópicas, indistinguíveis a olho nu, seja através do uso de tinta invisível. Estas são técnicas conhecidas como **esteganografia** (steganós = coberto, secreto) que é a arte de ocultar a mensagem.

Edgard Allan Poe, em O Escaravelho de Ouro, editado em 1843, descreve um pergaminho que continha um texto indicando a posição de um tesouro escondido, escrito com tinta invisível e criptografado. A tinta havia se tornado visível por acaso, sob a ação do vapor de uma chaleira. O cifrário usado no pergaminho, assim conta o autor, era o da substituição monoalfabética sobre o alfabeto latino. O herói do romance o decifrou efetuando a contagem da freqüência das letras no texto criptografado.

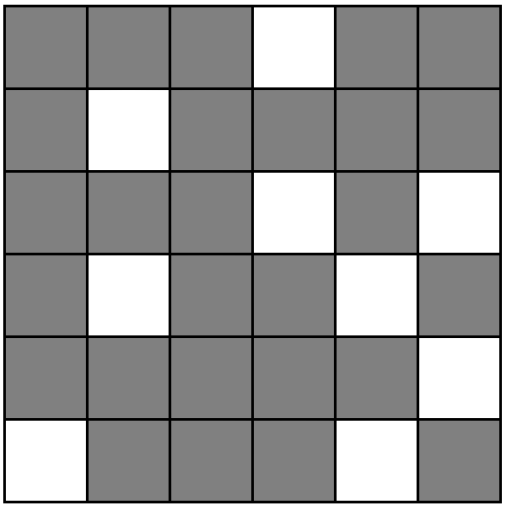

Figura 1.1: A máscara de Matias Sandorf.

Júlio Verne, no livro de sua autoria, denominado Matias Sandorf, descreve um cifrário bem interessante, que utiliza uma grade que serve de máscara. Conta-nos

o autor que Matias Sandorf, personagem central do romance, mantinha correspondência criptografada com os companheiros de ideal revolucionário. As cartas eram enviadas por intemédio de pombos correios, tendo um deles caído nas mão de um aventureiro, chamado Sarcany. Este, depois de copiar a mensagem, mesmo sem entendê-la, recolocou-a cuidadosamente no pombo e o soltou para observar onde ele pousava. Tendo visto o local de pouso, após algumas artimanhas, consegue emprego na casa de Matias. Um dia, estando ausente o dono da casa, Sarcany vasculha os móveis em busca de algo que pudesse fornecer uma indicação sobre o conteúdo da mensagem. Depois de muito procurar, encontrou um pedaço de couro, no qual estava desenhado uma grade como em um tabuleiro de xadrez com 6 linhas e 6 colunas, totalizando 36 casas. Nove casas do tabuleiro estavam recortadas, de modo que era possível ver através delas. A máscara apresentava o aspecto da figura 1.1 e era a chave que ele precisava para abrir as portas da codificação e decodificação das mensagens de Sandorf e seu companheiros.

Aparentemente, as grades criptográficas foram inventadas por Girolamo Cardano, italiano, no século 16. Para descrever o mecanismo usado para cifrar com a máscara, usaremos um trecho do poema Canção do Exílio de Gonçalves Dias (brasileiro, 1823 – 1864):

Minha terra tem palmeiras,
Onde canta o sabiá;
As aves que aqui gorjeiam,
Não gorjeiam como lá.

Inicialmente, eliminam-se os espaços entre as palavras e os acentos gráficos do texto, que então é escrito sobre uma grade com 6 linhas e 6 colunas que possuem as mesmas dimensões da máscara

m	i	n	h	a	t
e	r	r	a	t	e
m	p	a	l	m	e
i	r	a	o	n	d
e	c	a	n	t	a
o	s	a	b	i	a

de onde, sorrateiramente, retiramos o ”s” de palmeiras, os acentos e os sinais gráficos. Ficamos com um texto de exatas 36 letras. Se o texto tiver mais do que

36 letras, basta quebrá-lo em blocos de 36 letras cada. Se o último bloco possuir menos do que 36 letras, basta completá-lo com nulas.

Sobre o texto coloque a grade de Matias Sandorf que deixa visíveis apenas 9 caracteres. Copie, da esquerda para a direita e de cima para baixo, as letras visíveis através da máscara. Em seguida, gire a máscara de 90 graus, no sentido horário e repita o procedimento anterior. Gire uma segunda e uma terceira vez a máscara, sempre copiando os caracteres visíveis. A máscara é perfurada de tal modo que uma mesma letra da mensagem não é copiada duas vezes e, depois da terceira rotação, todas as letras do bloco foram copiadas. O texto criptografado assim obtido é

HRLERN AOIDTO BRASME ATAIMP ETIMCA NNAAAE

A pessoa que recebe a mensagem, tendo uma máscara idêntica à usada para criptografar, pode decodificar a mensagem. Divida o texto cifrado em blocos de 36 letras. Para decifrar cada bloco, coloque a máscara sobre o papel e copie as 9 primeiras letras do bloco cifrado nas janelas da máscara. Gire a máscara 90 graus no sentido anti-horário e copie as próximas 9 letras do texto cifrado nas janelas da máscara. Repita este procedimento outras duas vezes e o primeiro bloco de 36 letras estará recuperado. Este procedimento, realizado bloco a bloco, decifra a mensagem toda.

Para implementar este método criptográfico em computador, pode-se tratar a grade como sendo uma matriz 6×6, cujos índices das linhas e das colunas variam entre 0 e 5. Cada células está numa linha e numa coluna da grade e sua posição pode ser caracterizada por um par (i, j) de números, com i crescendo de 0 a 5 ao longo da vertical, percorrida de cima para baixo e j crescendo de 0 a 5 ao longo da horizontal, percorrida da esquerda para a direita. A célula que ocupa a posição (i, j) após uma rotação passa a ocupar a posição $(j,\, 6 - i)$.

1.11 Cifras diversas

Muitas outras cifras foram apresentadas ao longo da história e passaremos a descrever alguns.

1.11.1 O cifrário de Políbio

O historiador grego Políbio, que viveu dois séculos antes de Cristo, nos descreve uma técnica criptográfica interessante para alfabetos com 25 letras. Dado que o alfabeto que usamos contém 26 letras, vamos eliminar a letra Q e, sempre que dela necessitarmos, vamos substitui-la pela letra K. Para cifrar, dispomos as letras do alfabeto em uma grade, como num tabuleiro de xadrez com 5 linhas e 5 colunas, numerando-as de 1 a 5, como abaixo.

$\ddots$	1	2	3	4	5
1	a	b	c	d	e
2	f	g	h	i	j
3	k	l	m	n	o
4	p	r	s	t	u
5	v	w	x	y	z

Cada palavra do texto claro é substituída pelo número que indica sua linha e sua coluna, colocados nesta ordem. O a se transforma no 11, o r no 42 e assim por diante.

Vamos criptografar um trecho da poesia Meus Oito Anos de Casimiro de Abreu (brasileiro, 1839 – 1860), um clássico das antologias brasileiras

Oh! que saudades tenho
Da aurora da minha vida,
Da minha infância querida que os anos não trazem mais!
Que amor, que sonhos, que flores,
Naquelas tardes fagueiras
À sombra das bananeiras,
Debaixo dos laranjais!

Usando o tabuleiro de Políbio para cifrar o trecho

Oh que saudades tenho

sem o ponto de exclamação, obteríamos os seguintes números, correspondentes ao texto cifrado

35 23 31 45 15 43 11 45 14 11 14 11 14 15 43 44 15 34 23 35

Para decodificar, basta aplicar a substituição inversa, trocando $11 -> a$, $12 -> b$, etc.

Este método apresenta a desvantagem de duplicar o comprimento da mensagem. Sua segurança pode ser melhorada se as letras forem distribuídas ao acaso no tabuleiro. Neste caso, remetente e destinatário precisariam guardar cópias exatas desta tabela o que aumenta o risco de roubo e a quebra do método.

Para facilitar, pode-se escolher uma palavra chave que será colocada nas primeiras posições do tabuleiro. Se a palavra chave possuir letras repetidas, mantém-se

apenas a primeira ocorrência de cada letra, abandonando as que se repetem. Em seguida o tabuleiro é preenchido pelas letras que ainda não foram utilizadas, seguindo a ordem alfabética. Com a palavra ”interessante”como chave, obteríamos o seguinte tabuleiro

⋱	1	2	3	4	5
1	i	n	t	e	r
2	s	a	b	c	d
3	f	g	h	j	k
4	l	m	o	p	u
5	v	w	x	y	z

1.11.2 Cifrário de Playfair

O cifrário de Playfair foi inventado por Charles Wheatstone (inglês, 1802 – 1875), cujo nome está ligado ao dispositivo elétrico conhecido como ponte de Wheatstone, que consta ter sido inventada por Stephen Henry Christy (inglês, 1878 – 1914). Levou o nome de Palyfair pois consta ter sido Lorde Lyon Playfair (escocês, 1818 – 1898) quem o divulgou.

Este cifrário usa um tabuleiro exatamente igual ao de Políbio, sem a numeração nas bordas. Para criptografar uma mensagem, as letras do texto original são separadas e criptografadas aos pares. Se as letras do par pertencerem à mesma coluna do tabuleiro, cada uma delas será substituída pela letra que se encontra embaixo e na mesma coluna. A última letra da coluna é substituída pela primeira. Se as letras do par pertencerem à mesma linha do tabuleiro, cada uma delas será substituída pela letra que se encontra à sua direita e na mesma linha. A última letra da linha é substituída pela primeira. Se as letras do par não estiverem na mesma linha e nem na mesma coluna, suas posições definem os vértices de um retângulo no tabuleiro. Cada uma delas é substituída pela letra que se encontra no outro vértice do retângulo situado na mesma linha. O par AO, por exemplo, será substituído pelo par MB.

Quando houver um par formado pela mesma letra, tal como em EE, BB ou SS, elimina-se uma delas ou inclui-se uma nula entre as duas, refazendo-se a divisão do texto em pares de letras. Deve-se escolher o procedimento que menos prejudique o sentido exato do texto original quando se decifrar a mensagem. Em geral, duplos erres e esses devem ser automaticamente reduzidos a um único erre e um único esse.

O texto ”você e eu”, quando separado em pares de letras fornece

vo | ce | ee | u.

Como aparece um duplo e no penúltimo par, podemos inserir um x entre os dois e obter a separação silábica

vo | ce | ex | eu
que, ao ser criptografado pelo método de Playfair, resulta em
$KZDACZJZ$.

1.11.3 Cifrário bifendido de Delastelle

Este cifrário foi usado por Félix–Marie Delastelle (francês, 1840 – 1902). Nele, retire a letras Q do alfabeto e disponha as demais num tabuleiro de Políbio.

Para codificar uma mensagem, divida-a em linhas com n caracteres cada uma. Embaixo de cada letra coloque o número da sua linha e, embaixo do número da sua linha, coloque o número da sua coluna no tabuleiro. Numa segunda etapa, copie os números, linha por linha, colocando-os um ao lado do outro. Numa terceira etapa, estes números são agrupados dois a dois. Cada grupo de dois dígitos indica uma letra na tabela e este grupo de dígitos é substituído pela letra que ocupa a posição indicada por ele.

A chave deste método reside na escolha do tamanho n de caracteres de cada linha. Ainda, se a ordem em que se dispõe as letras no tabuleiro não for a ordem alfabética, a disposição das letras no tabuleiro passa a fazer parte da chave.

Para ilustrar o cifrário bifendido de Delastelle, peguemos o Hino da Independência

Brava gente brasileira!
Longe vá temor servil!
– Ou ficar a pátria livre,
– Ou morrer pelo Brasil.
...

composto por Evaristo da Veiga (1799 a 1837, poeta brasileiro). A título de curiosidade, a música associada a este hino foi composta por Dom Pedro I (1798 a 1834, imperador do Brasil de 1822 a 1831). Vamos criptografar sua primeira linha

brava gente brasileira

Depois de retirar os espaços entre as palavras, construímos a tabela

b	r	a	v	a	g	e	n	t	e	b	r	a	s	i	l	e	i	r	a
1	4	1	5	1	2	1	3	4	1	1	4	1	4	2	3	1	2	4	1
2	2	1	1	1	2	5	4	4	5	2	2	1	3	4	2	5	4	2	1

pois a letra b fica na primeira linha segunda coluna do tabuleiro de Políbio, a letra r fica na quarta linha segunda coluna e assim por diante. Copiando os dígitos linha por linha, obtemos

$$14151213411414231241\ 22111254452213425421$$

que devem ser separados aos pares

$$|14|15|12|13|41|14|14|23|12|41|22|11|12|54|45|22|13|42|54|21|$$

Usando agora o tabuleiro de Políbio, cada par nos fornece uma letra

$$\text{DEBCPDDHBPGABYUGCRYF}$$

É interessante observar que a decodificação da mensagem é feita do mesmo modo que a codificação. Cifrando novamente a mensagem cifrada, recupera-se a mensagem original.

1.12 A Matemática oculta da cifra de César

Vamos agora desvendar um segredo que nem Júlio César nem seus escribas suspeitavam. É possível associar este processo de cifra a uma operação aritmética que até hoje é aplicada nos cifrários mais modernos e utilizados na atualidade.

Para apresentar a Matemática oculta na cifra de César, vamos associar a cada letra do alfabeto um número, de acordo com a tabela abaixo,

$$\begin{array}{cccccccccc} a & b & c & d & e & \cdots & w & x & y & z \\ \updownarrow & \updownarrow & \updownarrow & \updownarrow & \updownarrow & \vdots & \updownarrow & \updownarrow & \updownarrow & \updownarrow \\ 0 & 1 & 2 & 3 & 4 & \cdots & 22 & 23 & 24 & 25 \end{array}$$

Mediante esta tabela, associamos o número 0 (zero) à letra a, o número 1 à letra b e assim por diante, associamos o número 25 à letra z. Não faremos distinção entre letras maiúsculas e minúsculas e escreveremos $a = A = 1$, $b = B = 2$, $\ldots$, $y = Y = 24$, $z = Z = 25$. Nesta seção **não faremos distinção entre a letra e o número que lhe corresponde**. Neste contexto é lícito dizer que a chave k da cifra de César é o número 3 ou a letra d e o alfabeto é identificado ao conjunto numérico

$$\mathbf{Z}_{26} = \{0, 1, 2, \ldots 25\}$$

Para codificar uma letra no cifrário de César, basta deslocá-la três posições no disco, percorrendo-o no sentido horário. Usando a correspondência entre letras e

números estabelecida a pouco, este espediente de deslocar três setores no sentido horário corresponde a uma adição de 3 unidades à letra a ser cifrada

$$\begin{aligned} c(a) = D \quad &\text{ou} \quad c(0) = 0 + 3 = 3 \\ c(b) = E \quad &\text{ou} \quad c(1) = 1 + 3 = 4 \end{aligned}$$

e assim por diante. Quando chegamos ao x, ao y e ao z somos surpreendidos pelos resultados obtidos

$$\begin{aligned} c(x) = c(23) = 23 + 3 = 26 \neq A \\ c(y) = c(24) = 24 + 3 = 27 \neq B \\ c(z) = c(25) = 25 + 3 = 28 \neq C \end{aligned}$$

Esta resultado nos faz concluir que a aritmética a ser usada não deve ser a adição usual dos inteiros pois

$$\begin{aligned} c(x) = A = 0 \\ c(y) = B = 1 \\ c(z) = C = 2 \end{aligned}$$

Depois de uma breve reflexão verificamos que, para cifrar corretamente o x, o y e o z é preciso subtrair 26 sempre que o resultado obtido for maior do que 25.Esta operação é denominada **adição modular**. Vamos defini-la.

Dado um número inteiro x, pode-se subtrair dele um múltiplo inteiro de 26, de modo a obter um número r entre 0 e 25. Em outras palavras, existe um inteiro q para o qual

$$r = x - 26q$$

é um número entre 0 e 25. Se $x \geq 0$ então $q \geq 0$ e, se $x < 0$ então $q < 0$. Este é teor do chamado **algoritmo da divisão** para números inteiros: dado um número inteiro x, existem dois inteiros q e r tais que $r = x- 26q$ está entre 0 e 25. O número q é denominado de **quociente** e o r é o **resto** da **divisão inteira** de x por 26, também denominado de x **módulo** 26 e escrevemos

$$r = x \bmod 26$$

Quando x for um número inteiro entre 0 e 25, então $x \bmod 26 = x$.

A **adição módulo** 26 é a operação que leva dois números inteiros x e y no número inteiro r entre 0 e 25 definido por

$$r = (x + y) \bmod 26$$

que recebe o nome de **soma módulo** 26 de x e y.

Exemplo 1.2. $(5+7) \bmod 26 = 12$ *pois* $5+7 = 26 \times 0 + 12$. *O* 0 *é o quociente e o* 12 *é o resto da divisão inteira de* $5+7=12$ *por* 26.

$(15+17) \bmod 26 = 6$ *pois* $15+17 = 26 \times 1 + 6$. *O* 1 *é o quociente e o* 6 *é o resto da divisão inteira de* $15+17 = 32$ *por* 26.

$(45+72) \bmod 26 = 13$ *pois* $45+72 = 26 \times 4 + 13$. *O* 4 *é o quociente e o* 13 *é o resto da divisão inteira de* $45+72 = 117$ *por* 26.

$(32-54) \bmod 26 = 4$ *pois* $32-54 = 26 \times (-1) + 4$. *O* -1 *é o quociente e o* 4 *é o resto da divisão inteira de* $32-54 = -22$ *por* 26.

Esta é a operação que cifra corretamente quando se usa o disco de César. Se x pertence a Z_{25} a função que cifra cada caractere x do texto original é

$$c(x) = (x+3) \bmod 26.$$

Observe:

$$\begin{aligned}
c(a) = c(0) &= (0+3) \bmod 26 = 3 = D \\
c(b) = c(1) &= (1+3) \bmod 26 = 4 = E \\
&\cdots \\
c(x) = c(23) &= (23+3) \bmod 26 = 0 = A \\
c(y) = c(24) &= (24+3) \bmod 26 = 1 = B \\
c(z) = c(25) &= (25+3) \bmod 26 = 2 = C
\end{aligned}$$

A **subtração módulo** 26 de dois números inteiros x e y é o número inteiro $(x-y) \bmod 26$. O resultado da subtração é chamado de **diferença módulo** 26 entre x e y. Para decifrar um caractere y com o disco de César, basta subtrair dele 3 unidades módulo 26. De fato, observe que

$$\begin{aligned}
d(A) = d(0) &= (0-3) \bmod 26 = 23 = x \\
d(B) = d(1) &= (1-3) \bmod 26 = 24 = y \\
d(C) = d(2) &= (2-3) \bmod 26 = 25 = z \\
&\cdots \\
d(Y) = d(24) &= (24-3) \bmod 26 = 21 = v \\
d(Z) = d(25) &= (25-3) \bmod 26 = 22 = w
\end{aligned}$$

Na cifra de César, a chave k é o número 3. Quem conhecer o método e souber a chave que está sendo usada, é capaz de decodificar a mensagem. Poderíamos usar como chave um outro número inteiro qualquer entre 1 e 25.

Pelo fato de existirem apenas 25 chaves para o método de César, esta técnica pode ser quebrada pela **busca exaustiva** da chave, que consiste em checar todas as chaves possíveis e tomar aquela que, ao ser aplicada ao texto cifrado, retorne um texto com sentido.

1.13 Adição e multiplicação modular

Não precisamos nos restringir à adição módulo 26. Isto aconteceu até o momento pelo mero fato de nosso alfabeto possuir 26 letras. Hoje em dia, para os métodos computacionais modernos, o alfabeto é formado por blocos de bits. Se as "letras"forem blocos de 64 bits, o "alfabeto"possuirá 2^{64} símbolos diferentes, um número da ordem de 18 quintilhões. Existem técnicas criptográficas onde cada "letra"é um bloco de 1024 bits, que fornece um alfabeto com 2^{1024} símbolos diferentes, um número impronunciável, da ordem de 10^{308}. Por ser fundamental nos métodos criptográficos modernos, faremos uma descrição da aritmética modular em

$$\mathbf{Z}_n = \{\ 0,\ 1,\ 2,\ \ldots,\ n-1\ \}$$

onde n é um número inteiro maior ou igual a 2.

Como é usual, o sinal $+$ indicará a adição usual de inteiros e o ponto $\cdot$ indicará a multiplicação usual de números inteiros. Se x e y forem dois números inteiros, $x + y$ é a soma usual enquanto $x \cdot y$ é o produto usual dos inteiros x e y. O ponto pode ser omitido de modo que xy também indica o produto usual de x e y.

O primeiro fato a ser enunciado é o **algoritmo da divisão**: Seja n um número inteiro diferente do zero. Dado outro número inteiro x, pode-se subtrair dele um múltiplo inteiro de n de modo a obter um número r entre 0 e $n-1$. Em outras palavras, dados dois números inteiros n e x, com $n \neq 0$, existe um outro número inteiro q tal que

$$r = x - nq$$

é um número entre 0 e $n-1$. Como

$$x = nq + r,$$

o número q é o **quociente** e r é o **resto** da **divisão inteira** de x por n. O resto r também é chamado de x **módulo** n e é denotado por $x \bmod n$. Quando x é um inteiro de $\mathbf{Z}_n$ então $x \bmod n = x$.

Exemplo 1.3. $197 \bmod 8 = 5$ *pois ao dividir* 197 *por* 8 *o quociente é* 24 *e o resto é* 5. *Verificando:* $24 \times 8 + 5 = 197$.

$-653 \bmod 49 = 33$ *pois ao dividir* -653 *por* 49 *o quociente é* -14 *e o resto é* 33. *Verificando:* $-14 \times 49 + 33 = -653$.

Sendo n um número inteiro maior do que zero, dados dois número inteiros x e y definimos a **adição** e a **multiplicação módulo** n por como sendo aquelas operações que resultam, respectivamente, nos números inteiros $(x + y) \bmod n$ e $(xy) \bmod n$. Estes números são denominados, respectivamente, de **soma módulo** n e **produto módulo** n de x e y.

Exemplo 1.4. $(84 + 76) \bmod 41 = 37$ *pois* $84 + 76 = 160$ *que ao ser dividido por* 41 *fornece quociente* 3 *e no resto* 37;

$(65 \cdot 34) \bmod 72 = 50$ *pois* $65 \cdot 34 = 2210$ *que ao ser dividido por* 72 *resulta no quociente* 30 *e no resto* 50.

Propriedades da adição e da multiplicação modular

Seja n um número inteiro positivo. Se x e y forem números inteiros, vamos denotar a adição e a multiplicação módulo n de x e y por

$$x \oplus y = (x + y) \bmod n$$
$$x \odot y = (xy) \bmod n$$

Valem as propriedades:

1. **Comutatividade da adição**. Sendo x e y números inteiros, então
   $x \oplus y = y \oplus x$.

2. **Associatividade da adição**. Sendo x, y e z números inteiros, então
   $$(x \oplus y) \oplus z = x \oplus (y \oplus z).$$

3. **Elemento neutro da adição**. O zero é o elemento neutro da adição. Com isto queremos dizer que, para todo x inteiro,
   $$x \oplus 0 = 0 \oplus x = x \bmod n.$$

4. **Simétrico aditivo ou oposto**. Dado um número inteiro x existe um inteiro y tal que
   $$x \oplus y = y \oplus x = 0.$$
   Para cada x inteiro, existe um único inteiro y em $\mathbf{Z}_n$ com esta propriedade.

5. **Comutatividade da multiplicação**. Sendo x e y números inteiros, então
   $x \odot y = y \odot x$.

6. **Associatividade da multiplicação**. Sendo x, y e z números inteiros, então
   $$(x \odot y) \odot z = x \odot (y \odot z).$$

7. **Elemento neutro da multiplicação**. A unidade é o elemento neutro da multiplicação. Com isto queremos dizer que, para todo x inteiro,
   $$x \odot 1 = 1 \odot x.$$

8. **Distributividade do produto em relação à adição**. Sendo x, y e z números inteiros, então

$$x \odot (y \oplus z) = (x \odot y) \oplus (x \odot z)$$

e

$$(x \oplus y) \odot z = (x \odot z) \oplus (x \odot z).$$

Estas propriedades asseguram que o conjunto $\mathbf{Z}_n = \{0, 1, 2, \ldots, n-1\}$, com a operação de adição módulo n, é uma estrutura matemática denominada **grupo comutativo** e $\mathbf{Z}_n$, com as operações de adição e multiplicação módulo n, é um **anel comutativo**.

Para calcular o valor de uma expressão aritmética contendo vários operadores, ela deve ser percorrida da esquerda para a direita. Quando um operador é encontrado, ele é executado se sua hierarquia for superior à do operador seguinte. Em caso contrário, este operador fica em estado de espera e passa-se ao próximo operador, repetindo o procedimento descrito. Quando um abrir parêntesis é encontrado, a operação à esquerda permanece em estado de espera até que todas as operações desde o abrir até o fechar parêntesis sejam realizadas. O processo termina quando todos os operadores forem executados.

Vamos convencionar que a redução módulo n possui hierarquia superior à multiplicação que, por seu turno, possui hierarquia superior à adição. Se x e y forem números inteiros, valem as propriedades

$$(x + y) \bmod n = (x \bmod n + y \bmod n) \bmod n,$$
$$(x\, y) \bmod n = (x \bmod n \cdot y \bmod n) \bmod n.$$

Elas destacam que, antes de adicionar ou multiplicar dois números módulo n pode-se, de antemão, reduzir cada um deles módulo n. Isto evita, por exemplo, a necessidade de trabalhar com números muito grandes, bem maiores que n. As propriedades acima valem para um número finito de parcelas ou fatores. Se x_1, x_2, $\ldots$, x_r forem números inteiros,

$$(x_1 + \cdots + x_r) \bmod n = (x_1 \bmod n + \cdots + x_r \bmod n) \bmod n,$$
$$(x_1 \times \cdots \times x_r) \bmod n = (x_1 \bmod n \times \cdots \times x_r \bmod n) \bmod n.$$

Estas igualdades destacam que, para adicionar dois ou mais números módulo n, pode-se reduzir cada parcela módulo n. Em seguida, adicionam-se os dois primeiros e efetue a redução módulo n. Adicione este resultado ao terceiro e efetue a redução módulo n e assim por diante, até realizar todas as adições. Esta propriedade também se aplica, *mutatis mutantis*, à multiplicação.

Estas propriedades são importantes pois garantem que, ao trabalhar com operações módulo n sempre estaremos restritos a números menores do que $2n$ no caso

da adição e n^2 no caso da multiplicação. Este fato é muito importante nos métodos criptográficos atuais onde o n pode ser um número muito grande com 1024 bits ou mais.

1.14 Inverso modular

Seja $n > 1$ um número inteiro. Um número inteiro x é o **simétrico multiplicativo módulo** n ou **inverso módulo** n do número inteiro y se $x\,y \bmod n = 1$. Desta forma, x é o inverso de y se e só se y for o inverso de x. Nem todo inteiro possui inverso módulo n. O zero, por exemplo, não possui inverso módulo n pois $0x \bmod n = 0$ para todo x inteiro. Se x possuir inverso módulo n, diremos que ele é **invertível módulo** n. Quando x possui inverso módulo n, ele não é único mas dois deles sempre diferem por um múltiplo inteiro de n e haverá um único y em $\mathbf{Z}_n$ tal que $x\,y \bmod n = 1$. Este y será denotado por $x^{-1} \bmod n$.

Exemplo 1.5. *O número* 9 *é um inverso de* 3 *módulo* 26 *pois* $9 \times 3 \bmod 26 = 1$ *e* $3^{-1} \bmod 26 = 9$ *e* $9^{-1} \bmod 26 = 3$.

Exemplo 1.6. *O número* 2 *não possui inverso módulo* 26. *De fato, se existisse um inteiro* x *tal que* $2x \bmod 26 = 1$, *então, para algum inteiros* q *teríamos* $2x - 26q = 1$. *Como o lado esquerdo é par e o direito é ímpar, esta igualdade é impossível.*

1.15 Máximo divisor comum

Vamos provar nesta seção que um número x possui inverso módulo n se e só se x e n forem primos entre si. Vamos começar pelo algoritmo da divisão que, pela sua relevância, repeti-lo nunca é demais.

Sejam x e n números inteiros, com $n > 1$. Pelo algoritmo da divisão, e existe um único número inteiro q e um único número inteiro r, com r entre 0 e $n - 1$, tais que

$$x = qn + r.$$

Quando $r = 0$, $x = qn$ e se diz que n **divide** x e se escreve $n \mid x$. Também se diz que n é um **divisor** de x, que x é **divisível** por n, ou que x é **múltiplo** de n, ou que n é **submúltiplo** de x. Não se define a divisibilidade pelo zero. O zero é divisível por todo inteiro n não nulo pois $0 = 0n$. O 1 divide todo inteiro x pois $x = 1x$.

Número primo é aquele número inteiro maior do que 1, que possui exatamente dois divisores positivos distintos, ele e a unidade. Com esta definição, o número 1 não é primo e o número 2 é o único número primo par. São primos o 2, 3, 5, 7, 11, 13, ... O conjunto de números primos é infinito.

Número composto é aquele número inteiro positivo maior do que 1 que não é primo. Ele possui outros divisores positivos além de si próprios e a unidade. São números compostos o 4, 6, 8, 9, 10, ...

Todo número composto é igual a um produto de números primos. Por exemplo, $4 = 2 \times 2$, $6 = 2 \times 3$, $8 = 2^3$, $10 = 2 \times 5$, $12 = 2^2 \times 3$, ... Esta **decomposição** ou **fatoração** de um número composto num produto de primos é única, a menos de uma permutação dos fatores primos.

Um número inteiro positivo m é o **máximo divisor comum** de dois números inteiros x e y se:

1. o m divide x e y;
2. todo divisor positivo de x e y divide m.

Vamos provar que o máximo divisor comum de dois números inteiros x e y existe quando pelo menos um deles não é nulo e nós o indicaremos por $\mathrm{mdc}(x, y)$.

Evidentemente, $\mathrm{mdc}(x, y) = \mathrm{mdc}(y, x)$ e $\mathrm{mdc}(x, y) = \mathrm{mdc}(|x|, |y|)$ e, por este motivo, ao calcular o máximo divisor comum de dois números podemos calcular os máximos divisores comuns de seus valores absolutos. Observe que, quando x for um número inteiro positivo,

$$\mathrm{mdc}(x, 0) = x.$$

Se o $\mathrm{mdc}(x, y)$ existir, ele é o maior divisor positivo de x e y. De fato, sendo d outro divisor positivo de x e y, então será um divisor do $\mathrm{mdc}(x, y)$. Como o d e o $\mathrm{mdc}(x, y)$ são ambos positivos, $d \leq \mathrm{mdc}(x, y)$.

Se o $\mathrm{mdc}(x, y)$ existir, ele é único. De fato, se m_1 e m_2 forem dois máximos divisores comuns de x e y, então m_1 divide m_2 e m_2 divide m_1. Como ambos são positivos, $m_1 = m_2$.

Teorema 1.1. *Sejam x e y dois números inteiros com $y \neq 0$. Seja r o resto da divisão inteira de x por y. Um número inteiro divide x e y se e só se dividir y e r.*

Demonstração: Como r é o resto da divisão de x por y, existe um inteiro q para o qual $x = qy + r$.

Se d for um divisor de x e y, então existem inteiros q_1 e q_2 para os quais $x = q_1 d$ e $y = q_2 d$. Logo, $r = x - qy = q_1 d - q q_2 d = (q_1 - q q_2) d$, mostrando que r é divisível por d. Logo d divide y e r.

Se d for um divisor de y e r então existem inteiros q_3 e q_4 tais que $y = q_3 d$ e $r = q_4 d$, o que acarreta em $x = qy + r = q q_3 d + q_4 d = (q q_3 + q_4) d$, mostrando que x é divisível por d. Logo d divide x e y. ■

Corolário 1.1. *Seja $y \neq 0$ e r o resto da divisão inteira de x por y. O* $\mathrm{mdc}(x, y) = \mathrm{mdc}(y, r)$, *se um deles existir.*

Demonstração: Se $m = \text{mdc}(x, y)$, então m divide y e r. Se d for um divisor positivo de y e r, também divide x e y, sendo assim um divisor de m. Logo, $m = \text{mdc}(y, r)$.

Se $m = \text{mdc}(y, r)$, então m divide x e y. Se d for um divisor positivo de x e y, também divide y e r, sendo assim um divisor de m. Logo, $m = \text{mdc}(x, y)$. ■

Sejam x e y dois números inteiros com $y > 0$ e defina $r_0 = x$ e $r_1 = y$. Considere a seqüência de divisões

$$\begin{aligned} r_0 &= q_2 r_1 + r_2 \\ r_1 &= q_3 r_2 + r_3 \\ &\cdots \\ r_{j-2} &= q_j r_{j-1} + r_j \\ r_{j-1} &= q_{j+1} r_j + r_{j+1} \end{aligned}$$

onde r_i é o resto da divisão inteira de r_{i-2} por r_{i-1}. Como $0 \leq r_i < r_{i-1}$, a sequência r_2, r_3, ... é decrescente e positiva. Numa sequência não negativa e decrescente de números inteiros, chega-se a $r_{j+1} = 0$, para algum índice j. Logo,

$$\begin{aligned} \text{mdc}(x, y) &= \text{mdc}(r_0, r_1) = \text{mdc}(r_1, r_2) = \cdots \\ &= \text{mdc}(r_{j-1}, r_j) = \text{mdc}(r_j, 0) = 0. \end{aligned}$$

Isto mostra que o $\text{mdc}(x, y)$ existe e é igual a r_j.

Ao mesmo tempo que provamos a existência do máximo divisor comum de dois números inteiros, onde pelo menos um é diferente de zero, determinamos um algoritmo eficiente para determiná-lo, denominado **algoritmo de Euclides**. Sua convergência é rápida, mesmo quando x e y são grandes. Podemos enunciar

Teorema 1.2. *(Algoritmo de Euclides). Sejam r_0 e r_1 dois números inteiros com $r_1 \neq 0$. Construa recursivamente os quocientes q_i e os restos r_i dividindo sucessivamente r_{i-1} por r_i*

$$r_{i-1} = q_{i+1} r_i + r_{i+1}$$

para $i = 1, 2, \ldots, j$ até obter $r_{j+1} = 0$. Então o $\text{mdc}(r_0, r_1)$ existe e

$$\text{mdc}(r_0, r_1) = r_j.$$

Exemplo 1.7. *Se $r_0 = 232$ e $r_1 = 76$, então $r_2 = r_0 \bmod r_1 = 232 \bmod 76 = 4$ e $r_3 = r_1 \bmod r_2 = 76 \bmod 4 = 0$. Logo, $\text{mdc}(r_0, r_1) = r_2 = 4$.*

Quando o $\text{mdc}(x, y) = 1$, se diz que x e y são **primos entre si** ou **relativamente primos**. Explicitano $r_2, r_3, \ldots, r_j$ na seqüência que estabelece o Algoritmo

de Euxlides, obtemos

$$\begin{aligned} r_2 &= a_2 r_0 + b_2 r_1 \\ r_3 &= a_3 r_0 + b_3 r_1 \\ r_4 &= a_4 r_0 + b_4 r_1 \\ &\cdots \\ r_j &= a_j r_0 + b_j r_1 \end{aligned}$$

onde

$$\begin{array}{ll} a_2 = 1 & b_2 = -q_2 \\ a_3 = -q_3 & b_3 = 1 \\ a_4 = a_2 - q_4 a_3 & b_4 = b_2 - q_4 b_3 \\ \cdots & \cdots \\ a_j = a_{j-2} - q_j a_{j-1} & b_j = b_{j-2} - q_j b_{j-1} \end{array}$$

Se definirmos $a_0 = 1$, $a_1 = 0$, $b_0 = 0$ e $b_1 = 1$, obtemos as fórmulas recursivas

$$\begin{aligned} a_i &= a_{i-2} - q_i a_{i-1} \\ b_i &= b_{i-2} - q_i b_{i-1} \end{aligned}$$

válidas para $i = 2, 3, \ldots, j$, o que nos permite enunciar o algoritmo estendido de Euclides.

Teorema 1.3. *(Algoritmo estendido de Euclides) Sejam r_0 e r_1 dois números inteiros com $r_1 \neq 0$. Sendo $a_0 = 1$, $a_1 = 0$, $b_0 = 0$ e $b_1 = 1$, calcule recursivamente*

$$\begin{aligned} a_i &= a_{i-2} - q_i a_{i-1} \\ b_i &= b_{i-2} - q_i b_{i-1} \\ r_i &= a_i r_0 + b_i r_1 \end{aligned}$$

para $i = 2, 3, \ldots, j$, onde q_i é o quociente da divisão inteira de r_{i-2} por r_{i-1}, parando quando $r_{j+1} = 0$. Então $r_j = a_j r_0 + b_j r_1$ é o máximo divisor comum de r_0 e r_1.

Teorema 1.4. *Dados dois números inteiros x e y onde pelo menos um deles é diferente de zero, então existem inteiros a e b tais que*

$$a\,x + b\,y = \mathrm{mdc}(x, y).$$

Demonstração: Se $y = 0$, basta tomar $a = 1$ e o valor de b é de livre escolha pois $\mathrm{mdc}(x, 0) = x$. Sendo $y \neq 0$, basta tomar no teorema anterior $r_0 = x$, $r_1 =$

y, $a = a_j$ e $b = b_j$. ■

Sejam a, b, x e y números inteiros onde x ou y é diferente de zero. Então o $\text{mdc}(x, y)$ divide $ax+ by$. Isto significa que o $\text{mdc}(x, y)$ divide todo elemento do conjunto

$$\{\ a\,x + b\,y : a, b \in \mathbf{Z}\ \text{ e }\ a\,x + b\,y > 0\ \}$$

e assim, o $\text{mdc}(x, y)$ é o menor elemento deste conjunto.

1.16 Inverso modular

Agora, com o algoritmo estendido de Euclides, mostraremos que, se x e $n > 0$ forem número inteiros, então x possuirá um inverso módulo n se e só se forem primos entre si.

Teorema 1.5. *Seja n um número inteiro positivo. Um número inteiro x possui inverso módulo n se e só se x e n forem primos entre si, isto é, quando o* $\text{mdc}(x, n) = 1$.

Demonstração: Se x e n forem primos entre si existem números inteiros a e b tais que $a\,x+ b\,n = 1$. Reduzindo esta igualdade módulo n, obtemos

$$1 = 1 \bmod n = (a\,x + b\,n) \bmod n = a\,x \bmod n + b\,n \bmod n = a\,x \bmod n$$

pois $b\,n \bmod n = 0$. Logo, a é o inverso módulo n de x.

Se y for o inverso módulo n de x então $x\,y \bmod n = 1$ e isto indica que existe um inteiro q para o qual $x\,y = -q\,n + 1$ ou $x\,y+ q\,n = 1 = \text{mdc}(x, n)$, mostrando que x e n são primos entre si. ■

Exemplo 1.8. *Como* $26 = 2 \times 13$, *em* $\mathbf{Z}_{26}$, *apenas os números ímpares, à exceção do* 13, *possuem inversos módulo* 26. *Ao calcular* $3x \bmod 26$, *fazendo o x igual a* 2, 3, ... *verificamos que* $3 \times 9 \bmod 26 = 1$. *Assim o inverso de* 3 *móculo* 26 *é* 9 *e que* 3 *é oinverso de* 9 *módulo* 26. *Do mesmo modo se pode verificar que* $5^{-1} \bmod 26 = 21$, *que* $7^{-1} \bmod 26 = 15$, *que* $11^{-1} \bmod 26 = 19$, *que* $17^{-1} \bmod 26 = 23$ *e que* $25^{-1} \bmod 26 = 25$.

Este processo de busca exaustiva do inverso apresentado no exemplo anterior é viável porque 26 é um número pequeno mesmo quando dispomos apenas de lápis e papel à nossa disposição. Quando n é grande, algo da ordem de 10^{50} ou maior, fica inviável determinar $x^{-1} \bmod n$ por uma busca exaustiva, mesmo quando temos um computador pessoal à nossa disposição. Felizmente, existe um algoritmo, denominado Algoritmo Estendido de Euclides que é rápido e eficiente

mesmo quando n é um número de 2048 bits, que é da ordem de 2^{2048} um número da ordem de 3×10^{616}. Números dessa grandeza são utilizados na Criptografia de Chave Pública, assunto que trataremos em capítulo posterior.

Como consequência do teorema anterior concluímos que, se p for primo, então todo elemento não nulo de $\mathbf{Z}_p$ possui inverso módulo p uma vez que todo inteiro entre 1 e $n-1$ é primo de p. O $\mathbf{Z}_p$, com as operações de adição e multiplicação módulo p passa a ser um **corpo** pois é um anel comutativo onde todo elemento não nulo possui inverso.

Exemplo 1.9. *O conjunto* $\mathbf{Z}_7 = \{0, 1, 2, 3, 4, 5, 6\}$ *com as operações de adição e multiplicação módulo* 7 *é um corpo pois* $1 \times 1 \bmod 26 = 1$, $2 \times 4 \bmod 26 = 1$, $3 \times 5 \bmod 26 = 1$ *e* $6 \times 6 \bmod 26 = 1$.

1.17 Equações modulares

Sejam n, x, y e z forem números inteiros tais que $n > 0$, x e y pertencem a $\mathbf{Z}_n$ e z possui inverso módulo n, que denotaremos por z^{-1}. Valem as implicações

$$y = (x + z) \bmod n \Longrightarrow x = (y - z) \bmod n$$

e

$$y = (x \cdot z) \bmod n \Longrightarrow x = (y \cdot z^{-1}) \bmod n.$$

Se x e y forem inteiros quaisquer, então

$$y \bmod n = (x + z) \bmod n \Longrightarrow x \bmod n = (y - z) \bmod n$$

e

$$y \bmod n = (x\,z) \bmod n \Longrightarrow x \bmod n = (y\,z^{-1}) \bmod n.$$

1.18 Quantidade de números primos entre si

Alguns métodos criptográficos modernos de grande importância comercial se baseiam no conceito de número primo. Dentre eles citamos o RSA (iniciais de Rivest, Shamir e Adleman que o desenvolveram). Se $n \geq 1$, quantos primos ele possui entre 1 e n? O conhecimento desta quantidade é importante e merece nossa atenção.

Cada número inteiro maior do que 1, ou é primo ou possui uma decomposição em fatores primos. A decomposição de 360 em fatores primos é $2^3 3^2 5$. A menos de uma reordenação dos fatores, esta decomposição é única. Todo número inteiro n maior do que 1 possui uma decomposição em fatores primos da forma

$$n = p_1^{e_1} p_2^{e_2} \cdots p_k^{e_k}$$

onde p_1, p_2, ..., p_k são números primos e e_1, e_2, ..., e_k são seus expoentes. Se n for primo, a sua decomposição em fatores primos só terá um fator. Seja $\phi(n)$ o número de inteiros entre 1 e n que são primos de n. Certamente, $\phi(1) = 1$ e, a partir de $n > 1$, $\phi(n) < n$. Quando n é primo, $\phi(n) = n - 1$ pois todos os inteiros positivos menores do que n são seus primos. Quando n é o produto de dois números primos p e q, então

$$\phi(n) = (p-1)(q-1) = n(1-1/p)(1-1/q).$$

No caso geral, quando $n = p_1^{e_1} p_2^{e_2} \cdots p_k^{e_k}$ e p_1, p_2, ..., p_k são primos distintos, então

$$\begin{aligned}\phi(n) &= \left(p_1^{e_1} - p_1^{e_1-1}\right)\left(p_2^{e_2} - p_2^{e_2-1}\right)\cdots\left(p_k^{e_k} - p_k^{e_k-1}\right) \\ &= n\left(1 - \frac{1}{p_1}\right)\left(1 - \frac{1}{p_2}\right)\cdots\left(1 - \frac{1}{p_k}\right).\end{aligned}$$

Sendo $360 = 2^3 3^2 5$, há

$$\phi(360) = 360\,(1-1/2)\,(1-1/3)\,(1-1/5) = 96$$

primos de 360 no intervalo de inteiros entre 1 e 359.

Para o nosso alfabeto, $n = 26 = 2 \times 13$, de modo que existem em $\mathbf{Z}_{26}$

$$\phi(26) = (2^1 - 2^0)(13^1 - 13^0) = (2-1)(13-1) = 12$$

números primos em relação a 26.

O conjunto

$$\mathbf{Z}_n^* = \{x \in \mathbf{Z}_n : \text{mdc}(x, n) = 1\}$$

formado pelos números de $\mathbf{Z}_n$ que possuem inverso módulo n, com a multiplicação módulo n, formam um grupo denominado **grupo multiplicativo módulo** n.

Exemplo 1.10. *O* $\phi(2) = 1$ *pois o* 1 *é primo de* 2, $\phi(3) = 2$ *pois* 1 *e* 2 *são primos de* 3, $\phi(4) = 2$ *pois* 1 *e* 3 *são primos de* 4, $\phi(5) = 4$ *pois o* 1, 2, 3, 4 *são primos de* 5, $\phi(6) = 2$ *pois* 1 *e* 5 *são primos de* 6. *Assim,* $\mathbf{Z}_2^* = \{1\}$, $\mathbf{Z}_3^* = \{1, 2\}$, $\mathbf{Z}_4^* = \{1, 3\}$, $\mathbf{Z}_5^* = \{1, 2, 3, 4\}$ *e* $\mathbf{Z}_6^* = \{1, 5\}$.

Quando p é primo, todo elementos não nulo de $\mathbf{Z}_p$ possui inverso módulo p. Como consequência, este conjunto, com as operações de adição e multiplicação módulo p é um corpo.

1.19 Origem do nome anel

Os Matemáticos gostam de atribuir nomes às estruturas matemáticas relevantes. Quando o $\mathbf{Z}_n$ e as operações de adição e multiplicação módulo n nos são apresentados, somos informados de que este conjunto com estas operações formam uma estrutura denominada anel. Este nome ficou intrigando minha mente por muito tempo. Hoje, acredito ter redescoberto a razão para ele.

Pensemos no conjunto $\mathbf{Z}$ dos números inteiros. Peguemos um barbante imaginário "infinito"e façamos nele infinitos nós, todos eles igualmente espaçados, um para cada número inteiro

$$\ldots,\ -3,\ -2,\ -1,\ 0,\ 1,\ 2,\ 3,\ \ldots$$

Alguém poderia objetar que não é possível dar nós num barbante de comprimento infinito e que, afinal, não teria paciência nem tempo de vida suficiente para fazer todos estes nós. Não importa, todo matemático gosta de brincar com o infinito. Pegue um barbante finito, faça alguns nós e, tudo bem, imagine os demais – este "tudo bem" foi colocado para imitar a maneira jovem de escrever.

Segure o nó correspondente ao zero e enrole este barbante no anel de César, de forma que os nós correspondentes a 0, 26, 52, ..., fiquem enrolados no sentido horário e mantidos sobre a letra a. Os nós correspondentes a 0, -26, -52, ..., enrolados no sentido anti-horário devem estar sobre a letra a. Os demais nós, correspondentes ao 1, 2, ..., 25 devem se sobrepor às letras a, b, ..., z, o mesmo acontecendo com os números inteiros que diferem deles por um múltiplo de 26. Teríamos a tabela de sobreposição

...	x	y	z	a	b	c	d	e	...
...	-29	-28	-27	-26	-25	-24	-23	-22	...
...	-3	-2	-1	0	1	2	3	4	...
...	23	24	25	26	27	28	29	30	...

Como as letras do alfabeto estão em correspondência biunívoca com o conjunto $\mathbf{Z}_{26}$, podemos pensar que ele é o conjunto $\mathbf{Z}$ dos inteiros no qual se identificam os números cuja diferença seja igual a 26. A soma modular é a soma usual em $\mathbf{Z}$, com a ressalva que se deve somar ou subtrair 26 tantas vezes quanto forem necessárias para resultar em um número entre 0 e 25. Eis que o $\mathbf{Z}_{26}$ é o $\mathbf{Z}$ enrolado num anel, resultando no nome anel para a estrutura constituída pelo $\mathbf{Z}_{26}$ com as operações modulares.

1.20 Cifrário por um deslocamento afim

Na cifra de César, identificamos as letras minúsculas às maiúsculas e associamos a cada uma delas um número inteiro entre zero e 25, seguindo a tabela

a	b	c	d	e	f	g	h	i	j	k	l	m
$\updownarrow$	$\updownarrow$	$\updownarrow$	$\updownarrow$	$\updownarrow$	$\updownarrow$	$\updownarrow$	$\updownarrow$	$\updownarrow$	$\updownarrow$	$\updownarrow$	$\updownarrow$	$\updownarrow$
00	01	02	03	04	05	06	07	08	09	10	11	12
n	o	p	q	r	s	t	u	v	w	x	y	z
$\updownarrow$	$\updownarrow$	$\updownarrow$	$\updownarrow$	$\updownarrow$	$\updownarrow$	$\updownarrow$	$\updownarrow$	$\updownarrow$	$\updownarrow$	$\updownarrow$	$\updownarrow$	$\updownarrow$
13	14	15	16	17	18	19	20	21	22	23	24	25

A função usada para cifrar com aquele método é

$$y_i = (x_i + 3) \bmod 26$$

e a usada para decifrar é

$$x_i = (y_i + 3) \bmod 26$$

onde x_i é a i-ésima letra do texto claro e y_i é a i-ésima letra do texto cifrado.

Uma possível generalização desta técnica, denominada de deslocamento afim, consiste em tomar a seguinte função de criptografar

$$y_i = c(x_i) = (a\, x_i + b) \bmod 26.$$

onde a e b são números em $\mathbf{Z}_{26}$. Se o número a não possuir inverso módulo 26, a função que cifra não é injetora. O codificador

$$c(x_i) = (13x_i + 3) \bmod 26$$

não serve uma vez que leva os números pares 0, 2, 4, 6, ... em $\mathbf{Z}_{26}$ no 3 e os números ímpares em $\mathbf{Z}_{26}$ no 16. O texto cifrado só possuirá dois caracteres, o D e o Q que correspondem aos números 3 e 16 e não há como recuperar o texto claro a partir de um texto criptografado com apenas duas letras distintas.

O a possui inverso módulo 26 se e só se a e 26 forem primos entre si. Como $26 = 2 \times 13$, o a não pode ser par nem o 13. Quando este for o caso, a função que decifra é

$$x_i = d(y_i) = a^{-1}(x_i - b) \bmod n\,,$$

onde a^{-1} indica o inverso módulo 26 de a.

Exemplo 1.11. *Considere o codificador*

$$y_i = c(x_i) = (7x_i + 2) \bmod 26$$

O inverso de 7 *módulo* 26 *é* 15 *pois* $15 \times 7 \bmod 26 = 1$. *Para decodificar, explicitamos o* x_i *na equação modular acima*

$$x_i = 7^{-1}(y_i - 2) = 15(y_i - 2) = 15y_i - 30 = 15y_i + 22$$

onde as operações são realizadas módulo 26. *Lembre-se que subtrair* 30 *na operação módulo* 26 *corresponde à adição por* 22. *O decodificador correto é*

$$x_i = d(y_i) = (15y_i + 22) \bmod 26.$$

A chave do deslocamento afim é formada pelo par

$$K = (a, b)\,.$$

Quantas chaves permanecem à nossa disposição? O b pode ser qualquer inteiro entre 0 e 25. O a e o 26 devem ser primos entre si. Existem 12 números em $\mathbf{Z}_{26}$ satisfazendo a esta condição que são 1, 3, 5, 7, 9, 11, 15, 17, 19, 21, 23, 25 onde notamos a ausência dos pares e do 13. Teremos assim até

$$26 \times 12 = 312$$

chaves distintas para este método. Algumas chaves não podem ser usadas. A chave $(a,\, b) = (1,\, 0)$ não modifica o texto original e não deve ser usada. Concluímos que o método é fraco pois um computador pessoal é capaz de testar todas as chaves em frações de segundos.

Alguém poderia levantar a objeção de que mesmo com poucas chaves o método continua forte quando o criptoanalista não conhece o método pois teria que testar todos os cifrários possíveis e imagináveis. A experiência – e neste ramo há uma experiência acumulada de séculos – nos ensinou que não se deve confiar o segredo de uma mensagem na suposição de que os possíveis invasores da privacidade não conhecem o método. É preciso que o método seja suficientemente forte para que o segredo da mensagem seja totalmente confiada ao segredo da chave.

1.21 Cifrário de Vigenère

Consideremos as técnicas chamadas **monoalfabéticas**, em que a criptografia de um texto é realizada caractere por caractere e, escolhida uma chave k, cada caractere do alfabeto dos textos em claro é levado sempre num mesmo caractere do alfabetos dos textos criptografados. São deste tipo o disco de César e o Atbash hebraico.

Existem ainda as técnicas denominadas **polialfabéticas**, onde um determinado caractere pode ser criptografado ora em um ora em outro símbolo, na dependência de sua posição na mensagem ou mesmo dos caracteres que estão ao seu redor.

Como tivemos a oportunidade de observar, pode-se aumentar a segurança do cifrário de César tomando como chave um par de números, que são aplicados alternadamente nos deslocamentos circulares das letras. Por exemplo, tomando-se $k = (3,\, 8)$, cifra-se deslocando as letras ora três, ora oito setores no disco de César, percorrendo-os no sentido horário. Esta percepção demorou 15 séculos para vir à tona, quando se percebeu que a chave poderia ser uma sequência qualquer de números e, quanto mais longa e menos previzivel for a chave, mais seguro será o método.

Um cifrário com esta característica é o de Blaise Vigenère, (francês, 1523 – 1596). Para falar sobre esta técnica, vamos imaginar que o alfabeto seja o latino com vinte e seis letras, onde se estabelece uma correspondência entre as letras do alfabeto e os inteiros em $\mathbf{Z}_{26}$, de modo que identificaremos letras a números,

$$a \leftrightarrow 0, \quad b \leftrightarrow 1, \quad c \leftrightarrow 2, \quad \ldots, \quad y \leftrightarrow 24, \quad z \leftrightarrow 25$$

sendo lícito falar em número b ou em letra 1.

A chave do cifrário de Vigenère é uma seqüência ordenada de números em $\mathbf{Z}_{26}$ como, por exemplo,

$$k = (3, 21, 7, 15) = (d, v, h, p)\,.$$

Pode-se escrever tanto $k = (d, v, h, p)$ quanto $k = dvhp$. A chave pode ser uma palavra ou um texto. Neste exemplo a chave possui 4 caracteres. Quanto maior a chave, mais difícil é a criptoanálise do método. Chaves de comprimento infinito, fornecem o chamado one-time-pad, considerado o cifrário indecifrável, o cifrário perfeito. Falaremos sobre ele posteriormente.

A técnica de Vigenère é considerada polialfabética pois, originalmente, para cifrar com este método eram usados 26 alfabetos romanos, dispostos em uma matriz. A primeira linha continha as letras dispostas em ordem alfabética. A segunda linha continha o alfabeto deslocado circularmente um caractere para a esquerda e começava com o b e terminava com o a. A terceira linha continha o alfabeto deslocado circularmente dois caracteres para a esquerda e começava com o c e terminava com o b. Este processo de preenchimento das linhas continuava de modo que a última linha da tabela começava com o z e terminava com o y. Para cifrar uma letra, buscava-se na primeira linha a coluna que ela ocupava. Procurava-se na primeira coluna a linha ocupada pela letra da vez na chave. A letra do texto era substituída por aquela que se encontrava no encontro da coluna com a linha localizadas nos passos anteriores.

Podemos pensar ainda que cada símbolo no alfabeto da cifra de Vigenère é uma sequência de letras cujo tamanho é igual ao tamanho da chave. Se a chave for ckn, por exemplo, então os símbolos que compõe o alfabeto da cifra de Vigenère são os anagramas com três letras. Este alfabeto possuiria 26^3 letras.

Usando a aritmética modular e a correspondência estabelecida entre letras e números, para criptografar um texto com a chave $\mathbf{k} = k_1k_2\ldots k_n$, quebra-se a mensagem em blocos de n caracteres $x_1x_2\ldots x_n$ e criptografa-se cada bloco mediante a operação de adição em $\mathbf{Z}_{26}$

$$\begin{aligned} y_1 &= x_1 + k_1 \\ y_2 &= x_2 + k_2 \\ &\vdots \\ y_n &= x_n + k_n \end{aligned}$$

Se o último bloco da mensagem não possuir n caracteres ele pode ser completado com nulas. Para não fornecer nenhuma pista sobre o tamanho da chave a um possível espião que se apodera do texto cifrado, pode-se incluir no seu final de uma a $n-1$ nulas.

A pessoa autorizada que recebe o texto cifrado e conhece a chave, sabe descartar as nulas inseridas no final, eliminando o último bloco com menos de n caracteres ao final do texto cifrada. Para decodificar a mensagem criptografada basta, a cada grupo de n caracteres, realizar as subtrações módulo 26 abaixo

$$\begin{aligned} x_1 &= (y_1 - k_1) \bmod 26 \\ x_2 &= (y_2 - k_2) \bmod 26 \\ &\vdots \\ x_n &= (y_n - k_n) \bmod 26 \end{aligned}$$

Pode-se construir chaves usando palavras fáceis de guardar como CONSTITUICAO, BRASIL. Todavia, cabe aqui um alerta a respeito de chaves: Se a palavra usada como chave for muito comum, um adversário, desejando quebrar o sigilo da mensagem pode testar diversas chaves diferentes usando palavras usuais, do nosso dia a dia. Quando se usa uma palavra chave que pode ser reproduzida facilmente por um processo de tentativa e busca num dicionário, diz-se que a chave escolhida é fraca. As palavras acima são exemplos de chaves fracas. Não me atrevo a dar exemplos de chaves fortes, para evitar risinhos maliciosos dos criptoanalistas: Hi, hi, hi, como ele é ingênuo!

Eventualmente se usa um bom método criptográfico mas a carne é fraca – oops, desculpem-me por este ato falho – digo, a chave é fraca.

A troca frequente de chaves foi um expediente muito usado na cifra de Vigenère. Para tanto, usava-se um livro de conhecimento público e muito lido como um dicionário de chaves. A mensagem levava embutida em pontos específicos, previamente combinados, a localização da chave no livro. A descoberta do livro e dos pontos do texto cifrado que continham a localização da chave quebrava a sua segurança.

Um dos primeiros a apresentar um método de ataque aos cifrários polialfabéticos foi Friedrich Kasiski (1805-1881, oficial prussiano) em 1863. Posteriormente, William Friedman (russo, naturalizado americano, 1891 – 1969), considerado um dos maiores criptoanalistas deste século, aprimorou a criptoanálise (nome cunhado por ele) dos cifrários polialfabéticos usando técnicas estatísticas. Este cientista trabalhou para o Exército americano no esforço de quebrar as mensagens criptografadas japonesas, durante o período que antecedeu o ataque a Pearl Harbor, durante a Segunda Guerra mundial.

Vamos citar um pequeno trecho da Bíblia, onde o Mestre dos mestres, Jesus Cristo, em Mateus, 11:28-30, nos exorta a encontrar nele a paz que não obtemos

no mundo

Vinde a mim vós todos que estais
cansados e oprimidos, e eu vos aliviarei.
Tomai sobre vós o meu jugo, e aprendei de mim,
que sou manso e humilde de coração
e encontrareis descanso para as vossas almas,
pois o meu jugo é suave e o meu fardo é leve.

Vamos criptografar a primeira linha

Vinde a mim vós todos

com o software Mathematica, usando a chave $k = dvhp$. Uma opção, que não é a única, pois existem implementações mais elegantes, consiste em usar os comandos do Mathematica abaixo. Estabeleça a tabela de conversão de letras para números digitando

alfanum = { "a"→ 0, "b"→ 1, "c"→ 2, "d"→ 3, "e"→ 4,
"f"→ 5, "g"→ 6, "h"→ 7, "i"→ 8, "j"→ 9,
"k"→ 10, "l"→ 11, "m"→ 12, "n"→ 13, "o"→ 14,
"p"→ 15, "q"→ 16, "r"→ 17, "s"→ 18, "t"→ 19,
"u"→ 20, "v"→ 21, "w"→ 22, "x"→ 23, "y"→ 24, "z"→ 25 }

e a de números para letras

numalfa = { 0 –> "A", 1 –> "B", 2 –> "C", 3 –> "D", 4 –> "E",
5 –> "F", 6 –> "G", 7 –> "H", 8 –> "I", 9 –> "J",
10 –> "K", 11 –> "L", 12 –> "M", 13 –> "N", 14 –> "O",
15 –> "P", 16 –> "Q", 17 –> "R", 18 –> "S", 19 –> "T",
20 –> "U", 21 –> "V", 22 –> "W", 23 –> "X", 24 –> "Y", 25 –> "Z"}

Para definir a chave e o texto na forma de uma lista de números, de acordo com a correspondência **alfanum** estabelecida acima, use

chave = Characters["dvhp"] /. alfanum]

e

texto = Characters["vindeamimvostodos"] /. alfanum]

Particione o **texto** em blocos do tamanho da chave com

blocos = Partition[texto, Length[chave]]]

Neste texto, o último caractere foi truncado pois seu comprimento não é múltiplo de 4. Não desejando perder nenhum caractere, pode-se inserir nulas no final. Criptografe os blocos do texto que estão na forma numérica

cifrada = Mod[Map[(chave + #)&, blocos] , 26]]

Recupere a mensagem cifrada na forma de texto com o comando

StringJoin[cifrada /. numalfa]]

que retorna o texto cifrado

YDUS HVTX PQVH WJKD

1.22 Variantes do cifrário de Vigenère

Foram propostas diversas variantes do cifrário de Vigenère ou mesmo casos particulares que são mencionados devido à sua importância histórica.

1.22.1 O cifrário de Tritêmio

O alemão Johanes Trithemius, por volta de 1460, propõe um cifrário de Vigenère, onde a chave é uma palavra com 26 letras, formada por uma transposição das letras do alfabeto. Uma chave deste cifrário poderia iniciada por uma palavra qualquer, fácil de guardar e completada pelas letras ainda não usadas do alfabeto, ordenadas em ordem crescente. Se a palavra chave tiver letras repetidas, usa-se apenas a sua primeira ocorrência, descartando as demais.

1.22.2 O cifrário de Gronsfelde

É um cifrário de Vigenère no qual as palavras chaves são formadas com as 10 primeiras letras do alfabeto,

A B C D E F G H I J

e, como a cada uma destas letras se associam os dígitos

0, 1, 2, 3, 4, 5, 6, 7, 8, 9

a cada palavra chave pode-se associar um número. À palavra CAFE, se associa o número 2054.

1.22.3 O cifrário de Beaufort

No cifrário de Vigenère, se soma a palavra chave ao texto. Beaufort sugeriu duas variações: numa primeira, que se subtraísse do texto a palavra chave e, na segunda, que se subtraísse da palavra chave o texto. Se a palavra chave for

$$\mathbf{k} = k_1 k_2 k_3 \dots k_n$$

temos o seguinte esquema

Vigenère	$y_i = x_i + k_i$	
Primeiro de Beaufort	$y_i = x_i - k_i$	$i = 1, 2, 3, \dots, n$
Segundo de Beaufort	$y_i = k_i - x_i$	

onde as operações são realizadas módulo 26. A primeira variante de Beaufort é o próprio cifrário de Vigenère pois cada número k_i em $\mathbf{Z}_{26}$ tem oposto $-k_i$ em $\mathbf{Z}_{26}$

1.23 One-time-pad

A única cifra absolutamente segura é o one-time-pad. É a cifra de Vigenère com uma chave infinita. Na prática não se consegue fazer uma chave infinita mas se pode simular uma gerando uma chave muito grande e, para cifrar um texto, usa-se um segmento da chave do tamanho da mensagem que se pretende cifrar e joga-se fora o segmento de chave usado. A chave é do tamanho da mensagem, sendo usada uma única vez e jogada fora.

Foi Claude Shannon que, em 1949, num artigo denominado Communication Theory of Secrecy Systems provou a segurança absoluta do one-time-pad sob a suposição de que a chave é criada de forma aleatória. Infelizmente, o one-time-pad só pode ser utilizado em situações muito especiais pois a distribuição de chaves para este método é complicada. Ela foi usada pelos espiões da União Soviética durante a guerra fria e no telefone vermelho que conectava Casa Branca em Washington ao Kremlim em Moscou. Neste caso, as chaves eram entregues por equipes de agentes do governo que as transportavam de uma ponta a outra.

A distribuição das chaves entre os agentes secretos e a central de inteligência deu origem ao nome do método. Traduzindo livremente, "one-time-pad" significa "bloco de notas usado uma vez". A central de inteligência possuía uma equipe de datilógrafos que se encarregavam de pressionar, ao acaso, o teclado de uma máquina de escrever, gerando sequências de letras, divididas em pequenas notas, que continham uma identificação única, e que eram copiadas mediante o uso de papel carbono. Imaginem uma folha de papel A4, dividida em oito partes iguais onde, em cada uma delas, se datilografava uma sequência aleatória de letras. Estes notas eram montadas em pequenos blocos, que eram entregues aos espiões que as usavam como chave de uma cifra de Vigenère. Uma cópia do bloco com as respectivas identificações ficavam retidas na agência central. Para cifrar uma mensagem, o espião usava uma letra da chave para cifrar uma letra do texto, descartando-a em seguida. A letra que ocupava a posição n da chave era usada para cifrar a n-ésima letra texto. Cada letra da chave era usada para cifrar uma letra do texto. Se a mensagem a ser enviada fosse maior do que a chave existente em uma folha, usava-se a chave existente na outra folha do bloco para continuar a criptografia. O espião inseria a identificação da chave usada no meio do texto cifrado de uma forma previamente combinada com a central de inteligência. O mesmo procedimento de cifra era usado pela agência central para enviar mensagens cifradas para os agentes. Quando a mensagem cifrada era entregue ao destinatário ele podia decifrar a mensagem pois conhecia a convenção usada na inserção da identidade da chave no meio do texto cifrado. Identificada a chave, ela podia ser usada para decifrar a mensagem.

Muitos desses blocos de notas foram apreendidos pela contra espionagem norte americana. Como os datilógrafos russos não eram capazes de gerar uma sequência

verdadeiramente aletória – uns utilizavam com maior frequência as letras que ficavam no meio do teclado, outros tinham maior destreza com uma das mãos – cada um deles possuia um padrão que passou a ser reconhecido pela equipe de criptoanálise. Com isto começaram a lhes atribuir nomes fictícios como Mary, Daisy, Frida, etc. Sabiam, inclusive, identificar os blocos de notas provenientes de novos funcionários, que logo recebiam um nome imaginário.

O espião e o receptor das mensagens possuiam bloquinhos de notas idênticos, onde cada página possuia uma identificação e um texto formado por letras datilografadas ao acaso. Façamos um exemplo: A central de inteligência precisava ordenar a retirada de um espião que se sabia ter sido descoberto pela contra espionagem. Ela quer cifrar a mensagem "fuja imediatamente, sua atividade foi descoberta". A agência sabe que uma das folhas do bloco de chaves que o espião possui é aquela identificada por "mnrs" que contém a chave

$$\text{ajsuemcioejfhasmcowphafcbplvsfdrcambwiufkasj}$$

A agência retirava os acentos e os espaços entre as palavras do texto obtendo

$$\mathbf{x} = \text{fujaimediatamentesuaatividadefoidescoberta}$$

e usava o segmento

$$\mathbf{k} = \text{ajsuemcioejfhasmcowphafcbplvsfdrcambwiufka}$$

da chave para cifrar o texto, usando Vigenère.

Observe que as letras finais "sj" da chave foram eliminadas uma vez que o texto claro contém 42 letras, e a chave, 44. O texto cifrado resultante é

$$\mathbf{y} = \text{FDBUMYGLWECFTEFFGGQPHTNXJSLYWKRZFEEDKJYWDA}$$

A identificação "mnrs" da folha que continha a chave era inserida de maneira dissimulada no interior da mensagem. Ao perceberem que seriam capturados, muitos espiões comeram seu bloco criptográfico para evitar que ele caísse nas mãos do inimigo.

Mesmo sendo absolutamente seguro, este método tem aplicabilidade reduzida, graças à dificuldade de se gerar uma chave aleatória de comprimento pelo menos tão grande quanto o total de letras contida na correspondência e na dificuldade de distribui-la de forma segura entre as partes.

Se fosse possível compartilhar de forma segura uma chave tão grande quanto o texto total trocado entre as partes, a criptografia dos textos seria desnecessária. Bastaria enviá-lo de forma segura da mesma forma que ser envia a chave. Há, todavia uma vantagem no one-time-pad: a chave pode ser enviada previamente, quando é possível combiná-la com segurança entre as partes.

1.24 Cifrário de Hill

Lester Hill, americano, deu uma contribuição valiosa à tarefa de tornar mais algébrica a criptografia e, no final da década de 1920, propõe a técnica criptográfica que passamos a descrever. Para aplicar a técnica de Hill, escolhe-se uma matriz quadrada

$$K = \begin{pmatrix} k_{11} & k_{12} & \cdots & k_{1n} \\ k_{21} & k_{22} & \cdots & k_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ k_{n1} & k_{n2} & \cdots & k_{nn} \end{pmatrix}$$

cujas entradas k_{ij} são números inteiros em $\mathbf{Z}_{26}$. Esta matriz é a chave do método. Dado um texto $\mathbf{x}$ a ser criptografado, deve-se quebrá-lo em segmentos de n caracteres

$$x_1 x_2 x_3 \ldots x_n$$

e efetuar o produto matricial

$$\begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} = \begin{pmatrix} k_{11} & k_{12} & \cdots & k_{1n} \\ k_{21} & k_{22} & \cdots & k_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ k_{n1} & k_{n2} & \cdots & k_{nn} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}$$

onde as operações são efetuadas módulo 26, para obter o bloco criptografado

$$y_1 y_2 y_3 \ldots y_n.$$

Se o último bloco de letras do texto original não possuir exatamente n letras, ele pode ser completado com **nulas**, isto é, com letras escolhidas ao acaso. Para evitar que o criptoanalista obtenha informações a respeito do número de linhas e colunas da matriz K, pode-se acrescentar no final da mensagem criptografada entre 1 e $n-1$ caracteres escolhidos a esmo. Este expediente certamente irá aumentar a segurança do cifrário.

Para recuperar a mensagem original, basta inverter a matriz K em $\mathbf{Z}_{26}$ e efetuar o produto matricial

$$\mathbf{x} = K^{-1}\mathbf{y}$$

onde $\mathbf{x} = [x_1\ x_2\ \cdots\ x_n]^T$ e $\mathbf{y} = [y_1\ y_2\ \cdots\ y_n]^T$ onde o T sobre as matrizes indica a sua transposta. A matriz K deve ser inversível em $\mathbf{Z}_{26}$.

Uma matriz quadrada A com elementos em $\mathbf{Z}_{26}$ é inversível em $\mathbf{Z}_{26}$ se existir outra matriz B com elementos em $\mathbf{Z}_{26}$ tal que $AB = I$, onde I é a matriz unidade com n linhas e n colunas.

Exemplo 1.12. *A inversa da matriz*

$$K = \begin{pmatrix} 11 & 8 \\ 3 & 7 \end{pmatrix}$$

em $\mathbf{Z}_{26}$ *é*

$$K^{-1} = \begin{pmatrix} 7 & 18 \\ 23 & 11 \end{pmatrix}.$$

Vamos criptografar a palavra $algo = (0, 11, 6, 14)$ *com a técnica de Hill, usando a matriz* K *como chave e efetuando as operações módulo* 26

$$\begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \begin{pmatrix} 11 & 8 \\ 3 & 7 \end{pmatrix} \begin{pmatrix} 0 \\ 11 \end{pmatrix}$$
$$= \begin{pmatrix} 88 \\ 77 \end{pmatrix} = \begin{pmatrix} 10 \\ 25 \end{pmatrix} = \begin{pmatrix} k \\ z \end{pmatrix}$$

$$\begin{pmatrix} y_3 \\ y_4 \end{pmatrix} = \begin{pmatrix} 11 & 8 \\ 3 & 7 \end{pmatrix} \begin{pmatrix} 6 \\ 14 \end{pmatrix}$$
$$= \begin{pmatrix} 178 \\ 116 \end{pmatrix} = \begin{pmatrix} 22 \\ 12 \end{pmatrix} = \begin{pmatrix} w \\ m \end{pmatrix}$$

resultando no anagrama $kzwm$.

A posição de um caractere no bloco interfere no texto criptografado. A palavra *lago*, que é um anagrama de *algo*, é criptografado com o algoritmo anterior e com a mesma chave K em $rhwm$. A simples transposição do a com o l, modificou as duas primeiras letras do texto criptografado.

Para inverter uma matriz K com as operações módulo 26, vale o método da eliminação de Gauss e a fórmula usual de inversão de matrizes

$$K^{-1} = [\det(K)]^{-1} \operatorname{cof}\left(K^T\right)$$

onde $\det(K)$ é o determinante de K e $\operatorname{cof}(K^T)$ é a matriz cofatora da transposta de K. Para que a matriz K seja inversível em $\mathbf{Z}_{26}$, é preciso que seu determinante tenha inverso em $\mathbf{Z}_{26}$. Com o auxílio do Mathematica, usando os comandos

$$K = \{\{\, k_{11},\ k_{12}\,\},\ \{\, k_{21},\ k_{22}\,\}\}$$
$$invDetK = \text{PowerMod}[\,\text{Det}[K]\,,\ -1,\ 26]$$
$$cofKT = \text{Mod}[\,\text{Det}[K]\ \text{Inverse}[K]\,,\ 26]$$
$$invK = \text{Mod}[\,invDetK\ cofKT\,,\ 26]$$

é possível, atribuindo valores inteiros entre 0 e 25 a k_{11}, k_{12}, k_{21}, k_{22}, testar a inversibilidade de uma matriz em $\mathbf{Z}_{26}$. Acima, $invDetK$ é o inverso do determinante de K, $cofKT$ é a cofatora de K transposta e $invK$ é a matriz inversa de K, ambas módulo 26. Para testar que $invK$ é a matriz inversa de K, basta emitir o comando

$$\text{Mod}[\,\text{K}\cdot\,\text{invK}\,,\ 26\,]$$

e verificar se o resultado é a matriz identidade.

Apresentamos algumas matrizes e suas inversas em $\mathbf{Z}_{26}$

$$K=\begin{pmatrix}7&5\\3&8\end{pmatrix}\qquad K^{-1}=\begin{pmatrix}4&17\\5&23\end{pmatrix}$$

$$K=\begin{pmatrix}2&5\\9&7\end{pmatrix}\qquad K^{-1}=\begin{pmatrix}9&1\\7&10\end{pmatrix}$$

$$K=\begin{pmatrix}2&9\\11&3\end{pmatrix}\qquad K^{-1}=\begin{pmatrix}5&11\\25&12\end{pmatrix}$$

$$K=\begin{pmatrix}11&9\\17&4\end{pmatrix}\qquad K^{-1}=\begin{pmatrix}20&7\\19&3\end{pmatrix}$$

A cifra por permutação é um caso particular da cifra de Hill. Se a matriz $K=(k_{ij})$ for definida por

$$k_{ij}=\begin{cases}1 & \text{se } j=\pi(i)\\ 0 & \text{se } j\neq\pi(i)\end{cases}$$

isto é, o elemento da linha i coluna $\pi(i)$ é igual a 1. Os demais elementos são nulos.

No caso do exemplo $\pi=(3,1,4,2)$, a matriz de Hill K que fornece o mesmo criptograma teria k_{13}, k_{21}, k_{34}, k_{42} iguais a 1 e os demais elementos nulos. Assim,

$$K=\begin{pmatrix}0&0&1&0\\1&0&0&0\\0&0&0&1\\0&1&0&0\end{pmatrix}$$

1.25 Implementação em computador

Vamos implementar algumas técnicas clássicas usando o computador.

1.25.1 Cifra de César

Abaixo apresentamos uma implementação da cifra de César usando a linguagem C.

```
//----------------------------------------

 #include<windows.h>
 #include <conio.h>
 #include<stdio.h>
 #include <ctype.h>

int main()
{
  int texto[100], decifrado[100], cifrado[100];
  int i, letra = 0, ind = 0, op = 0;

  // opte entre cifrar (c ou C) ou decifrar (d ou D).
  while( op != 67 && op != 68 )
  {
    system("cls");
    printf("Disco de Cesar\n\n\n");
    printf("Pressione C para cifrar e D para decifrar:  ");
    op = toupper(getche());
    printf("\n\n");
  }

  // Lendo a frase.
  printf("Digite a frase com ateh 100 caracteres,\n\n");
  printf("sem os acentos e o cedilha.\n\n");
  printf("Pressione ENTER para encerrar.\n\n");
  printf("=> ");
  while( letra != 13 && ind < 100)
  // A frase se encerra ao pressionar
  // a tecla ENTER (ASCII = 13)
  {
    letra = getche();
    // Correspondencia ASCII
    // A : Z = 65 : 90
    // a : z = 97 : 122
    // Se o caractere pressionado for uma letra
    if( (letra >= 65 && letra <= 90) ||
```

```
      (letra >= 97 && letra <= 122) )
  {
    texto[ind] = letra;
    ind++;
  }
}

// Apresenta a frase a ser cifrada ou decifrada
if( op = 'C' )
{
  printf("\n\n
      Frase a ser cifrada sem pontuacao e espacos:
      \n\n => ");
  for(i = 0; i < ind ; i++)
  {
    // Frase exibida sem espacos, com letras minusculas
    printf("%c", tolower(texto[i]));
  }
}
else
{
  printf("\n\n
      Frase a ser decifrada sem pontuacao e espacos:
      \n => ");
  for(i = 0; i < ind ; i++)
  {
    // Exibimos a frase sem espacos e com letras maiúsculas
    printf("%c", toupper(texto[i]));
  }
}

// Apresenta a frase cifrada ou decifrada
if(op = 'C')
{
  printf("\n\n Texto cifrado:\n\n => ");
  for(i = 0; i < ind ; i++)
  {
    decifrado[i] = tolower(texto[i]);
    cifrado[i] = ( decifrado[i]-97+3 ) % 26 + 65;
    printf( "%c", cifrado[i] );
  }
```

```
  }
  else
  {
    printf("\n\n Texto decifrado:\n\n => ");
    for(i = 0; i < ind ; i++)
    {
      cifrado[i] = toupper( texto[i] );
      decifrado[i] = (( cifrado[i]-65+3) % 26) + 97;
      printf( "%c", decifrado[i] );
    }
  }
  printf("\n\n");
  getch();
  return 0;
}
//----------------------------------------
```

1.25.2 A máscara de Matias Sandorf

Abaixo programamos a máscara de Matias Sandorf usando o Mathematica. As linhas numeradas são linhas de comentários. Cada comando leva um número que não deve ser digitado. O In[i] e o Out[i] são colocados automaticamente pelo programa e não devem ser digitados.

Codificação

```
1. Definindo o texto claro

In[1]:=
mensagem = Characters["minhaterratempalmeiraondecantaosabia"]

Out[1]=
{m, i, n, h, a, t, e, r, r, a, t, e, m, p, a, l, m, e, i, r, a,
o, n, d, e, c, a, n, t, a, o, s, a, b, i, a}

2. Definindo a chave e a sequencia em que as letras
   serao lidas, girando o tabuleiro tres vezes.
   Vamos usar uma mascara 6 por 6 com buracos nas posicoes
   14, 22, 34, 36, 42, 45, 56, 61, 65.

In[2]:=
```

```
chave1 = IntegerDigits[{14, 22, 34, 36, 42, 45, 56, 61, 65}];
chave2 = ({7 - #[[2]], #[[1]]}) & /@ chave1;
chave3 = ({7 - #[[2]], #[[1]]}) & /@ chave2;
chave4 = ({7 - #[[2]], #[[1]]}) & /@ chave3;
Flatten[{chave1, chave2, chave3, chave4}, 1];
chave = (6 #[[1]] + #[[2]] - 6) & /@ %

Out[7]=
{4, 8, 16, 18, 20, 23, 30, 31, 35, 13, 26, 15, 3, 28,
10, 5, 36, 12, 33, 29, 21, 19, 17, 14, 7, 6, 2, 24,
11, 22, 34, 9, 27, 32, 1, 25}

3. Cifrando o texto claro com a chave

In[8]:=
cifrado = StringJoin[ToUpperCase[mensagem[[ chave ]]]]

Out[8]=
HRLERNAOIMCANNAAAEATAIMPETIDTOBRASME
```

Decodificação

```
1. Para decifrar, basta repetir a operacao que
   prepara a chave e comandar

In[9]:=
mensagem[[chave]] = ToLowerCase[Characters[cifrado]];
StringJoin[mensagem]

Out[10]=
minhaterratempalmeiraondecantaosabia
FIM DO PROGRAMA
```

e se tem o texto em claro.

1.26 Alertas de segurança

Como primeiro ponto a destacar em relação à segurança, é que a tentativa de usar alfabetos exóticos ou secretos para o texto cifrado acrescenta pouquíssima dificuldade adicional às técnicas primitivas. Independentemente dos símbolos usados, o que importa é o alfabeto da mensagem original.

Nos processos primitivos de criptografia, alguns cuidados eram tomados para dificultar a tarefa de um eventual espião. Como as letras com acentos e o c com o cedilha ocorrem com uma freqüência bem menor do que os outros caracteres, eles devem ser evitados na mensagem a ser codificada. Deve-se escrever moca em lugar de moça, cafe em lugar de café, e assim por diante. O mesmo comentário se aplica às vírgulas, pontos finais, pontos e vírgulas pois todos possui baixo índice de ocorrência. Os duplos esses e erres também devem ser evitados na mensagem a ser criptografada, pois eles fornecem informações adicionais ao criptoanalista.

O dígitos 0, 1, 2, 3, 4, 5, 6, 7, 8, e 9 devem ser evitados no texto a ser criptografado pois ocorrem com uma freqüência bem menor que as letras. Para contornar este problema, escreva os números por extenso.

Os espaços, se mantidos como tal na mensagem cifrada, irá fornecer ao criptoanalista pistas sobre o comprimento das palavras da mensagem original, que o auxilia na quebra do sigilo. Por outro lado, se forem criptografados, sua alta incidência no texto iria denunciá-los como se os houvéssemos mantido nos seus lugares de origem. Os textos a serem cifrados devem ser curtos. Quanto maior o texto, mais informações terá o criptoanalista que se apoderar do texto cifrado.

Um sério erro de segurança consiste em codificar o mesmo texto com **dois métodos diferentes**. Os japoneses cometeram este erro ao enviar mensagens às suas diversas embaixadas usando métodos com diferentes níveis de segurança. Estas mensagens, interceptadas e comparadas, permitiram a quebra das mensagens cifradas. Eles acreditavam na segurança de suas mensagens pois contavam com a conjunção de dois fatores: o **segredo acidental** existente pelo uso de uma língua pouco difundida e de seu alfabeto com ideogramas, e com o **segredo intencional**, obtido com a utilização de um código secreto. Aliás, como as mensagens eram enviadas em Código Morse por rádio ou telégrafo, os japoneses dispunham de tabelas matriciais. Nelas, a primeira linha e a primeira coluna possuiam caracteres latinos e nas demais entradas, os ideogramas japoneses. Cada ideograma era substituído pelo par de letras latinas que se encontravam na mesma linha e mesma coluna. Havia ainda uma lista de palavras com quatro caracteres latinos para substituir os ideogramas e saudações de uso freqüente pelos japoneses.

Os americanos, durante a II Guerra Mundial, usaram um sistema de criptofonia (criptografia + telefonia) com índios da tribo Navajo, cuja lingua é pouco conhecida e cuja estrutura é bem diferente das linguas existentes na Europa. Foi uma das mais bem sucedidas técnicas criptográficas da época.

Como recomendou Auguste Kerckhoffs von Nieuwenhof (1835 – 1903) em seu La Cryptographie Militaire, escrito em 1883, os cifrários devem trabalhar com chaves de fácil manuseio e substituição, devem ser fáceis de operar de forma que basta uma pessoa para codificar as mensagens e outra para decodificá-las. Recomendou ainda que, quanto maior for o valor e maior for o tempo durante o qual se deseja guardar um segredo, mais seguros deve ser a técnica para cifrá-los. A criptografia

que cuida dos segredos de longo prazo é chamada de **estratégica** e a que cuida dos segredos que podem ser revelados dentro de poucas horas ou poucos dias é chamada de **tática**. Os sistemas estratégicos deverão ser projetados de forma que sua segurança seja preservada pela chave, sem a qual, ninguém deverá ser capaz de decodificar as mensagens. Certamente, se o oponente não conhece o método, sua segurança estará reforçada. Todavia, o sistema deve ser projetado de forma tal a continuar seguro mesmo depois de cair em mãos alheias, desde que estas mãos não possuam a chave.

Os conselhos de Kerckhoffs continuam em vigor. Tanto é que os pesquisadores e firmas especializadas em criptografia, colocam seus programas na internet e os apresentam em congressos para serem testados. Sua inviolabilidade, se mantida por um longo período de tempo e após muitos ataques, fará com que os eventuais usuários passem a acreditar em sua segurança.

1.27 Cifrários compostos

Para aumentar a segurança, pode-se **compor cifrários**, sobrecifrando a mensagem com duas ou mais técnicas. Todavia, deve-se observar que nem sempre a dupla cifra nos leva a um método mais seguro. A composição de dois cifrários de César, com chaves B (= 1) e E (= 4) nos leva a um cifrário de César cuja chave é F (= 5 = 1 + 4). A composição de dois cifrários de Vigenère é equivalente a outro cifrário de Vigenère. Se a chave do primeiro possuir comprimento n e a do segundo comprimento q, a chave do composto será uma palavra cujo comprimento será, no máximo, igual ao mínimo múltiplo comum entre p e q. Por exemplo, se a chave do primeiro for o anagrama BC (= 1, 2) e a chave do segundo for DEF (= 3, 4, 5) então a chave do cifrário de Vigenère composto será $EGGFFH$ (= 4, 6, 6, 5, 5, 7) pois (4, 6, 6, 5, 5, 7) é a soma termo a termo de (1, 2, 1, 2, 1, 2) com (3, 4, 5, 3, 4, 5). Observe que a sena (1, 2, 1, 2, 1, 2) corresponde à aplicação da chave (1, 2) a três blocos de duas letras e a sena (3, 4, 5, 3, 4, 5) corresponde à aplicação da chave (3, 4, 5) a dois blocos de três letras. A primeira chave tem comprimento 2, a segunda tem comprimento 3 e o $\text{mmc}(2, 3) = 6$. Hoje se sabe que compor cifrários totalmente substitutivos ou totalmente transpositivos é arriscado, pode levar a cifrários mais frágeis do que cada um deles em particular. É melhor alternar transposições com substituições. Um dos primeiros a usar tal composição foi o bifendido de Delastelle.

1.28 Criptografia mecânica

A revolução industrial, iniciada na segunda metade do século 18 trouxe à luz uma série de dispositivos que influenciaram a Criptografia. Um grande passo na direção

da automação dos processos industriais foi dado pela invenção do tear mecânico, cujas operações eram comandadas por fitas perfuradas, desenvolvido na última década daquele século, por Joseph – Marie Jacquard (francês, 1752 – 1834). Em 1820 Charles Babbage apresentou seu projeto de calculadora mecânica operada por cartões e fitas perfuradas. Este projeto, que não foi adiante, era o prenuncio dos computadores modernos. A patente do telefone em 1871, feita por Antonio Meucci (italiano naturalizado norte americano, 1808 – 1889) e a apresentação de telégrafo elétrico na Universidade de Columbia em 1835, feita por Samuel Morse (norte americano, 1791 – 1872) abriram as portas para a comunicação a distância e o governo, o comércio e a indústria passaram a usar de modo mais corriqueiro as mensagens secretas que podiam ser enviadas de forma segura através destes novos meios de comunicação.

Ainda na última década do século 18, Thomas Jefferson (norte americano, 1743 – 1826), autor da Declaração da independência dos Estados Unidos e seu presidente em 1801, inventou uma máquina de criptografar que foi esquecida e reinventada por Étienne Bazeries (francês, 1846 – 1931) por volta de 1890. **A máquina de Jefferson e Bazeries** possuía um eixo no qual se encaixavam 38 discos. Os dois discos da extremidade eram fixos e os demais podiam girar em redor do eixo. Na superfície lateral de cada disco eram impressas todas as 26 letras do alfabeto, igualmente espaçadas e dispostas ao acaso ao longo da lateral. A ordem em que as letras eram impressas variavam de um disco para outro. Nos discos laterais fixos, eram impressos os números de zero a 25 em ordem crescente. O conjunto montado apresentava forma cilíndrica com uns 15 cm de altura e 5 cm de diâmetro. Para codificar uma mensagem, ela devia ser quebrada em fragmentos de 36 letras. Cada fragmento era formado na linha zero e os textos das outras linhas eram cópias codificadas da linha zero. A chave se resume na escolha da linha a ser enviada pelo remetente. Quando se escolhia, por exemplo, a linha de número 11, o receptor formava cada bloco de texto na décima primeira linha e lia a mensagem decodificada na linha zero. Como existem 25! permutações circulares de 26 elementos e a máquina de Jefferson possui 36 cilindros, pode-se construir $(25!)^{36} \simeq 7 \times 10^{906}$ máquinas diferentes. Acredite se quizer. Embora possam ser construídas muitas máquinas distintas, este cifrário é fraco por possuir apenas 25 chaves. Qualquer pessoa com um mecanismo semelhante, testando as 25 chaves, poderia decifrar a mensagem. O dispositivo de Bazeries tinha 20 discos e, ao ser adotada pelos americanos por volta de 1920, foi construida com 25 discos.

Esta máquina simples com rotores foi a precursora de máquinas elétricas mais sofisticadas, que usavam cilindros rotativos para criptografar mensagens. Duas máquinas que fizeram muito sucesso durante a Segunda Guerra Mundial foram o Enigma, do alemães, e a Púrpura, dos japoneses. Os americanos, mesmo sem conhecer a Púrpura, foram capazes de construir uma máquina que reproduzia os textos criptografados japoneses. Ironicamente, mesmo sendo mais complexa dos

que os códigos LA, PA–K2 e os da série J, que usavam apenas lápis e papel, depois de quebrada a máquina Púrpura, os textos criptografados com ela eram mais fáceis de serem decodificados. Dos códigos com lápis e papel, 10% das chaves não foram quebradas e apenas 3% das chaves usadas na Púrpura não foram desvendadas. Trabalhou no serviço de criptoanálise da Púrpura o pesquisador americano William Friedman, considerado um dos maiores criptoanalistas deste século, sendo na época o chefe do grupo de criptoanalistas do Signal Corps, que era uma seção da inteligência do exército americano. Esta e outras histórias são contadas com detalhes minuciosos no capítulo One Day of Magic do livro de Khan [2]. Magic foi o nome dado pelo Contra Almirante Walter S. Anderson, da inteligência da marinha norte americana, ao esforço daquele país de interceptar e decodificar as mensagens criptografadas dos japoneses.

Um exemplar do Enigma alemão foi roubado pelos ingleses. Mesmo assim, não foi fácil quebrar a técnica usada por esta máquina pois, como todo método eficiente, mesmo em posse da máquina, há que se descobrir a chave. O matemático inglês Alan Turing (1912 – 1954), considerado um dos pais da informática, participou do esforço de guerra para desvendar o Enigma. Nesta época Turing era um membro da equipe de criptoanalistas da espionagem britânica, com sede em Bletchley Park em Buckinghamshire. Para se fazer justiça, os ingleses foram auxiliados por poloneses que haviam se exilado na Inglaterra e já haviam iniciado o trabalho de decifrar o Enigma antes de deixar seu país natal. Em Bletchley Park, por volta de 1943, foram construidos diversos dispositivos eletro-eletrônicos gigantescos, denominados Colossos, que eram usados para decodificar as mensagens alemãs. Estes dispositivos foram os precursores dos computadores modernos.

Em síntese, o Enigma possuía um teclado semelhante ao de uma máquina de escrever e o mecanismo interno era formado por 5 discos montados em um eixo. O primeiro era mantido fixo e os quatro outros podiam girar. As duas faces de cada disco eram divididas em 26 setores iguais, correspondendo cada um a uma letra. Todos os setores das duas faces dos discos possuiam contatos elétricos. Cada setor em uma das faces era ligado eletricamente a um e somente um setor da outra face. Os discos no eixo se tocavam de modo que um sinal elétrico gerado na face anterior do primeiro disco percorria um caminho tortuoso pelos cinco discos até atingir a face posterior do último, retornando ao primeiro. Quando se apertava uma tecla, um sinal elétrico era introduzido no primeiro disco, naquele setor correspondente à letra teclada. Este sinal percorria todos os discos e chegava a um determinado setor do último e retornava. Este sinal elétrico acendia uma lâmpada em um painel luminoso. Cada lâmpada deste painel correspondia a uma letra, que era copiada para formar a mensagem criptografada. A cada tecla apertada, o primeiro rotor girava um setor. Quando o primeiro rotor completava uma volta, o segundo girava um setor. Quando o segundo rotor completava uma volta, o terceiro girava um setor e assim por diante, como no hodômetro de um

automóvel. Mesmo em posse da máquina era difícil decodificar a mensagem pois era preciso saber a posição inicial dos rotores e com quatro rotores existem $26^4 = 456.976$ posições iniciais diferentes. Para dificultar o trabalho dos criptoanalistas, o Exército alemão possuía 9 tipos diferentes de rotores que eram substituídos com freqüência. Mesmo possuindo uma destas máquinas, era impossível quebrar as mensagens criptografadas apenas com lápis e papel. Neste ponto entrava o Colossos. Até hoje não se divulgou o trabalho realizado na violação do Enigma. Esta história nos ensina que a segurança em criptografia não deve se basear na manutenção do método em segredo. Ele pode cair – e fatalmente cai – em mão inimigas. O bom método é aquele em que a mensagem só pode ser decodificada em presença da chave.

1.29 De olho no presente sem perder o futuro

E viveram felizes para sempre. É assim que terminam os contos infantis. Neste momento, encerramos nosso capítulo sobre as técnicas primitivas, a infância da Criptografia, e começaremos sua juventude no próximo. Por aqui encerramos, onde o mundo real deste sistema globalizado tem início. Todavia, antes de fechar o capítulo, vamos lançar os olhos um pouco à frente.

Com o advento do computador, os métodos apresentados neste capítulo passaram a ter apenas valor histórico. Muitos outros foram desenvolvidos. Alguns resistiram aos ataques sofridos ao longo do tempo e outros sucumbiram. Podemos dizer que a década de 70 foi um marco na história da Criptografia.

O **DES** (Data Encription Standard), proposto em 1977 pela IBM (International Business Machines), se baseia numa composição de cifrários mais simples e foi amplamente divulgado, a ponto de que qualquer pessoa com experiência em programação de computadores é capaz de programá-lo. A própria IBM pagou uma equipe de especialistas para testar a segurança do método. Depois de um trabalho em equipe equivalente a 17 anos de trabalho de um homem, ele foi dado por encerrado e, em seu relatório final, que não foi divulgado, a equipe concluiu que o DES é seguro. Um bom método é aquele para o qual **a inviolabilidade da mensagem é garantida pela chave**. O DES comporta 2^{56} chaves diferentes.

O **RSA**, nome derivado das iniciais dos seus inventores, Rivest, Shamir e Adleman, revolucionou a Criptografia, sendo a primeira técnica de chave pública a ser apresentada na literatura aberta. Na criptografia de chave pública, cada um constrói uma chave pública usada para cifrar a mensagem e uma chave privada, usada para decifrar e para assinaturas digitais. O RSA se mantém seguro até hoje e esta segurança se apóia na dificuldade de recuperar dois números primos, dado o seu produto. Claro, estamos falando em números da ordem de 10^{100} ou maiores (isto mesmo, números com 100 algarismos na base decimal).

Para o setor de telecomunicações são de particular importância os **cifrários sequenciais** (stream ciphers), que usam registradores com deslocamento (shift registers) não lineares. Nestes cifrários, a chave evolui no tempo. O stream cipher ideal, conhecido como **one-time-pad**, ou cifrário de Vernam, é um cifrário de Vigenère cuja palavra chave tem comprimento infinito. Na verdade, cada letra da chave é usada uma única vez e depois é descartada obrigando aquele que a usa a ter uma chave cujo tamanho é igual ou superior à totalidade das mensagens que envia e recebe. O telefone vermelho entre Washington e o Kremlim usava este cifrário nos tempos da guerra fria. Como é muito difícil trabalhar com chaves longas, o que se faz é criar algoritmos computacionais determinísticos que gerem uma seqüência de números aleatórios. Sabe-se que os métodos determinísticos existentes não geram seqüências aleatórias mas sequências que se parecem aleatórias e que tais sequências possuem um período depois do qual se repetem. Quando o período desta seqüência é muito grande, nos aproximamos do cifrário ideal.

E o futuro, o que nos reserva? Aparentemente, os métodos de cifra em bloco baseados em **curvas elípticas**, prometem mais segurança com chaves menores. A **criptografia quântica**, quem diria, baseada na Mecânica Quântica, prometem ser a salvação do sistema bancário no próximo milênio. Hoje já se fala em processos capazes de burlar a segurança quântica e os usuários estão mais céticos em relação à sua efetividade.

Finalizando o capítulo, afirmamos que a Criptografia moderna é um novo mundo, aberto aos espíritos destemidos que desejam explorá-lo. Nele não há mais lugar para o amadorismo. Sua importância tem assumido um papel tão estratégico, que alguns governos têm colocado barreiras para a sua divulgação externa. Hoje, quando a soma de dinheiro que corre pela Internet é imensa, nunca se fez tão presente a necessidade de criptografar as informações que percorrem esta infovia. Pela rede de compensação interbancária, controlada pelo Banco Central do Brasil, percorre, em média, um PIB brasileiro a cada três dias.

A Criptografia além de ser fundamental para a segurança do país, é um inesgotável manancial de pesquisa para o matemático e representa um desafio a ser vencido e conquistado por aquelas nações independentes e que assim pretendem permanecer.

1.30 Exercícios

1. Criptografe a primeira estrofe da canção "Quando eu estou aqui" composta por Erasmo Carlos e Roberto Carlos, usando: A cifra de César, a cítala espartana e o atbash hebráico. Para as duas primeiras use uma chave $k = 6$. Como referência, a primeira estrofe desta canção é: "Quando eu estou aqui / Eu vivo esse momento lindo / Olhando pra você / E as mesmas emoções / Sentindo..."

2. Use uma transposição colunar e uma cifra por substituição para cifrar um trecho da canção Carinhoso, de João de Barro e Pixinguinha. Pode escolher as chaves que desejar e, como referência, criptografe o trecho "Meu coração / Não sei porque / Bate feliz, quando te vê / E os meus olhos ficam sorrindo / E pelas ruas vão te seguindo / Mas mesmo assim, foges de mim."

3. Use a permutação $\pi = (7, 6, 5, 4, 3, 2, 1)$ e a máscara de Matias Sandorf para cifrar "Aqueles olhos verdes / Translúcidos serenos / Parecem dois amenos / Pedaços do luar / Mas têm a miragem / Profunda do oceano / E trazem todo o engano / Das procelas do mar". Este é um trecho da canção "Aqueles olhos verdes" cuja letra foi composta por Adolfo Utrera (Cubano) e cuja melodia é de Nilo Menéndez.

4. Criptografe a primeira estrofe "Caminhando contra o vento / Sem lenço, sem documento / No sol de quase dezembro / Eu vou" da canção "Alegria, alegria" de Caetano Veloso, usando as cifras e Políbio, Playfair e o bifendido de Delastelle.

5. Use o cifrário por deslocamento afim $c(x_i) = (15\,x_i + 8) \bmod 26$ para cifrar a estrofe "Ver na vida algum motivo / Prá sonhar / Ter um sonho todo azul / Azul da cor do mar..." da canção Azul da cor do mar de Tim Maia. Retire os acentos e os espaços entre as palavras. Determine a função $d(y_i)$ que decifra, usando apenas números em $\mathbf{Z}_{26}$.

6. Criptografe, usando a cifra de Vigenère, "Y con el embrujo de tus canciones / Iba renasciendo tu amor em mi / Y en la noche hermosa de plenilunio / De tus blancas manos sentí el calor / Que con sus caricias me dio el amor" de "Recuerdos de Ypacarai" cuja autoria é de Zulema de Mirkin e Demetrio Ortiz. Use a chave $k = (17, 2, 9, 21, 11, 4, 3)$. Para cifrar adiciona-se módulo 26 e para decifrar se subtrai módulo 26. Portanto, a função que decifra é uma cifra de Vigenère cuja chave é $\sigma = (-17, -2, -9, -21, -11, -4, -3)$. Expresse esta chave em termos de números em $\mathbf{Z}_{26}$. Em outras palavras, determine os opostos de 17, 2, 9, 21, 11, 4 e 3 em $\mathbf{Z}_{26}$.

7. Criptografe, usando a cifra de Hill, a primeira estrofe da canção "Garota de Ipanema" de Tom Jobim: "Olha que coisa mais linda / Mais cheia de graça / É ela menina / Que vem e que passa / Num doce balanço / A caminho do mar". Use como chave a matriz $K = \{\ \{7, 24, 7\}, \{14, 21, 19\}, \{24, 3, 12\}\ \}$. Aqui, 7, 24, 7 é a primeira linha, 14, 21 19 a segunda e 24, 3, 12 a terceira. Complete o último bloco com nulas. Calcule o determinante de K módulo 26 e a inversa de K módulo 26. No Mathematica você pode executar estas tarefas com comandos Mod[Det[k], 26] e Inverse[k, Modulus -> 26]. No Mathematica é preferível iniciar o nome de uma variável com

letra minúscula, uma vez que ele reserva as letras maiúsculas para iniciar o nome de variáveis internas. Este não é o caso do K, que não está reservado, mas é o da letra E, por exemplo, que representa o número de Napier em homenagem a John Napier (escocês, 1550 – 1617).

8. Use o algoritmo de Euclides para determinar o máximo divisor comum entre (a) 1703 e 1053, (b) 1885 e 1382, (c) 1241 e 4768, (d) 14843 e 49009, (e) 15387 e 47919.

9. Calcule, usando o algoritmo de Euclides, o máximo divisor comum entre: (a) 9378 e 126466; (b) 34508737 e 2363067; (c) 337725495 e 423685.

10. Use o algoritmo estendido de Euclides para determinar os inversos multiplicativos modulares, se eles existirem, dos números inteiros (a) 4824 módulo 73921, (b) 5124 módulo 63694, (c) 264589 módulo 3292134. Com o Mathematica você pode calcular a^b mod n com o comando PowerMod[a, b, n]. O inverso multiplicativo de a módulo n pode ser calculado com PowerMod[a, -1, n].

11. Verifiquem se os inversos multiplicativos existem. Se existirem calcule-os usando o algoritmo estendido de Euclides.

    (a) $265^{-1} \bmod 193$ (Resposta: 126)

    (b) $2385635^{-1} \bmod 723$ (Resposta: 251)

    (c) $62745^{-1} \bmod 325$ (Resposta: não existe)

    (d) $843869^{-1} \bmod 421$ (Resposta: 66)

12. Calcule os produtos abaixo.

    (a) $38 \times 47 \bmod 96$ (Resposta 58)

    (b) $91 \times 52 \bmod 27$ (Resposta 7)

13. Determine x tal que

    (a) $21x \equiv 9 \pmod{26}$ (Resposta: 19)

    (b) $827x \equiv 38 \pmod{927}$ (Resposta: 871)

    (c) $478x \equiv 38 \pmod{1024}$ (Resposta: 285)

    (d) $976x \equiv 38 \pmod{512}$ (Resposta: não tem solução)

14. Se você resolveu os problemas anteriores à mão, meus parabéns, você tem muita paciência. Agora escreva os algoritmos que os resolvem em alguma linguagem computacional de seu conhecimento.

Capítulo 2

Criptografia simétrica

A criptografia simétrica é aquela em que as chaves para cifrar e decifrar são iguais ou uma é facilmente dedutível a partir da outra. Para duas pessoas se comunicarem usando-a é preciso combinarem antecipadamente a chave a ser usada, que deve ser mantida em segredo. Esta troca de chaves entre as partes é um grande problema. Enquanto é fácil trocar de forma segura uma chave quando as pessoas estão próximas fisicamente, esta tarefa não é muito simples quando as pessoas que pretendem se comunicar estão longe uma da outra.

O computador grava os arquivos em disco na forma de bits, que são as unidades básicas manipuladas no computador, traduzidas em termos de zeros e uns. Para facilitar esta tarefa, os bits são agrupados em número de oito, formando um byte. Em um byte podem ser gravadas $2^8 = 256$ informações diferentes. A tabela ASCII, por exemplo, associa a cada byte uma letra, um dígito, um sinal gráfico ou um comando para o sistema operacional executar.

A criptografia simétrica, tal como as técnicas primitivas, cifra um caractere do texto de cada vez. A ressalva é que, agora, os caracteres são blocos de bits e, por este motivo, são denominadas cifras de blocos.

Iremos descrever algumas cifras de bloco neste capítulo. Dentre elas, o Data Encryption Standard (DES) e o Advanced Encryption Standard (AES). O DES cifra blocos de 64 bits de cada vez e o AES blocos de 128 bits de cada vez. O "alfabeto"do DES possui 2^{64} "letras"e o do AES possui 2^{128} "letras". Estes números são significamente altos o que confere a ambos um alto grau de confiabilidade.

A chave do DES possui 56 bits e do AES aceita chaves de 128, 192 ou 256 bits. Este último é o novo padrão criptográfico, que substituirá o primeiro. Um espaço de chaves com $2^{56} = 7 \times 10^{16}$ elementos diferentes é pequeno para o poder computacional da atualidade.

2.1 O DES, Data Encryption Standard

Em 15 de maio de 1973, o então National Bureau of Standards (NBS), que atualmente é o National Institute of Standards and Technology (NIST), solicitou um criptosistema padrão. O escolhido, originalmente denominado LUCIFER, foi desenvolvido na IBM e adotado como padrão no dia 15 de janeiro de 1977, passando a ser conhecido por Data Encryption Standard (DES). A arquitetura utilizada é denominada de Feistel em homenagem Horst Feistel que inventou esta arquitetura.

O DES recebe 64 bits e devolve outros 64 bits cifrados. Para tanto ele utiliza uma chave de 56 bits. Na época em que foi desenvolvido, a eletrônica dos computadores não era tão confiável como hoje. A chave fornecida possuia 8 bytes mas um bit de cada byte era usado para verificar a paridade detectando possíveis erros computacionais.

Esquema geral

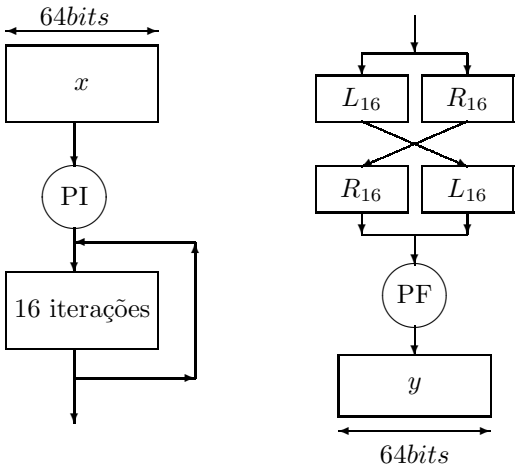

Figura 2.1: Visão geral do funcionamento do DES. Nesta figura os números entre parêntesis indicam o número de bits que entram e saem. PI é uma permutação inicial e PII é a permutação inicial inversa.

O DES recebe um bloco x de 64 bits e realiza uma permutação inicial PI. Na saída, os 32 bits da esquerda são denotados por L_0 e os outros 32 bits da direita são denotados por R_0. Estas metades passam 16 vezes pelo mesmo processo de cifra. Na saída, os 32 bits da esquerda, denotados L_{16}, trocam de posição com os 32 da direita, denotados R_{16} que passam por uma permutação final PF, igual à

inversa da permutação inicial PI, fornecendo o bloco cifrado y. Este processo está esquematizado na figura 2.1. As duas permutações estão nas tabelas Permutação PI e Permutação PF.

Permutação PI							
58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

Permutação PF							
40	8	48	16	56	24	64	32
39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30
37	5	45	13	53	21	61	29
36	4	44	12	52	20	60	28
35	3	43	11	51	19	59	27
34	2	42	10	50	18	58	26
33	1	41	9	49	17	57	25

De acordo com estas tabelas, se os bits de x forem $x_1x_2\cdots x_{64}$, então os bits de $PI(x)$ serão $x_{58}x_{50}\cdots x_{15}x_7$. Os índices dos bits acompanham a sequência dos números da tabela, percorrendo-a da esquerda para a direita e de cima para baixo. Do mesmo modo, os bits de $PF(x)$ serão $x_{40}x_8\cdots x_{57}x_{25}$.

Esquema de cada rodada

Vamos abrir o processo que cifra, apresentado na figura 2.2. O bloco x de 64 bits passa pela permutação inicial PI é dividido pela metade que denotaremos por L_i e R_i. O L vem de left, o R de right e o i é o índice da rodada, que percorre dos valores inteiros de 1 a 16. As metades das entradas iniciais são designadas por L_0 e R_0. Em cada rodada, entram L_{i-1} e R_{i-1}, saem L_i e R_i, de acordo com a fórmula

$$\begin{aligned} L_i &= R_{i-1} \\ R_i &= L_{i-1} \oplus f\left(K_i, R_{i-1}\right) \end{aligned}$$

onde $\oplus$ é o XOR bit a bit ou, se preferirem, a adição módulo 2 bit a bit. Lembrando que o XOR bit a bit é comutativo e que $0 \oplus 0 = 0$, $0 \oplus 1 = 1$, $1 \oplus 1 = 1$, então o

XOR bit a bit dos bytes 10011100 e 10101010 é 00110110. A função f, que será descrita em seguida, recebe como argumento R_{i-1}, o lado direito da entrada da rodada anterior e K_i, um número de 48 bits, proveniente da chave e que é usada na rodada de número i, cuja obtenção será descrita adiante.

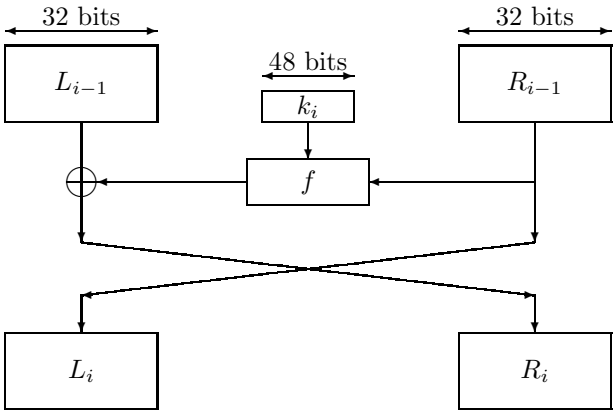

Figura 2.2: Esta figura apresenta o esquema de uma rodada do DES.

Como $a \oplus b = c$ implica em $a = b \oplus c$, podemos inverter a operação que cifra, explicitando R_{i-1} e L_{i-1}

$$\begin{aligned} R_{i-1} &= L_i \\ L_{i-1} &= R_i \oplus f\left(K_i, R_{i-1}\right) \end{aligned}$$

Lembrando que $R_{i-1} = L_i$, obtemos L_{i-1} e R_{i-1} em função de L_i e R_i

$$\begin{aligned} R_{i-1} &= L_i \\ L_{i-1} &= R_i \oplus f\left(K_i, L_i\right) \end{aligned}$$

e esta é a fórmula que decifra. Impressionante! É idêntica à fórmula que cifra com os papéis de L e R intercambiados. Podemos usar o mesmo programa que cifra para decifrar a mensagem, bastando trocar as posições de L e R. Tal fato é de grande importância, principalmente quando se quer implementar o DES em processadores com pequena capacidade de memória.

Cifrar e decifrar

A fórmula que decifra é idêntica à que cifra, com os papéis de L e R invertidos. Esta é a razão pela qual se permuta a posição de L_{16} com a de R_{16} na última

rodada do DES. Com este expediente, o programa que cifra será usado no processo de decifragem. A única diferença reside no índice i das rodadas que, para cifrar percorre os valores $1, 2, \ldots, 16$, nesta ordem e, para decifrar, percorre estes mesmos valores em ordem inversa.

Descrição da função $f(K, R)$

A função f recebe dois argumentos K e R, sendo K a chave da rodada, que possui 48 bits e R, o segmento direito do bloco que entra no processo de cifra e que tem 32 bits. Numa primeira etapa, o R sofre uma expansão, duplicando os bits 1, 4, 5, 8, 9, 12, 13, 16, 17, 20, 21, 24, 25, 28, 29 e 32, de acordo com a tabela de expansão E

Expansão E					
32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1

Isto significa que, se os bits de R forem $R_1R_2 \cdots R_{31}R_{32}$, então os bits de $E(R)$ serão $R_{32}R_1R_2 \cdots R_{31}R_{32}R_1$. Os índices acompanham as entradas da tabela de expansão da esquerda para a direita e de cima para baixo.

Descrição das caixas S

As oito caixas S são definidas pelos elementos de oito matrizes, cada uma contendo quatro linhas e dezesseis colunas. Cada linha contém todos os números inteiros de 0 a 15 e a disposição deles na linha foi selecionada pelos projetistas, possivelmente para evitar os ataques conhecidos. As tabelas abaixo, denominadas Caixa S_1, Caixa S_2, ..., Caixa S_8 mostram as caixas S do DES na base hexadecimal.

Caixa S_1

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	E	4	D	1	2	F	B	8	3	A	6	C	5	9	0	7
1	0	F	7	4	E	2	D	1	A	6	C	B	9	5	3	8
2	4	1	E	8	D	6	2	B	F	C	9	7	3	A	5	0
3	F	C	8	2	4	9	1	7	5	B	3	E	A	0	6	3

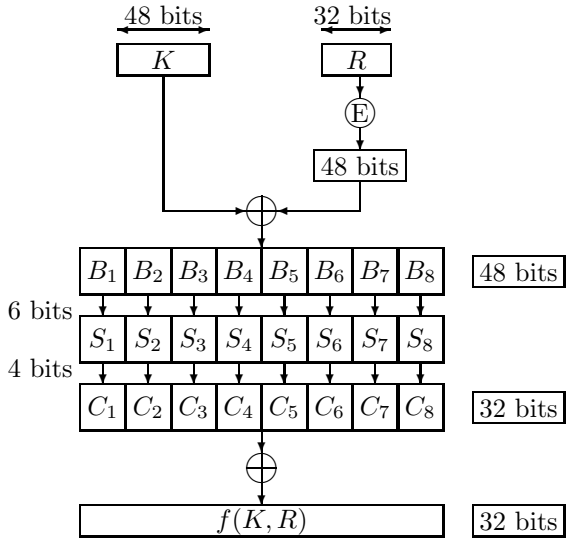

Figura 2.3: Esta figura apresenta o funcionamento da função f do DES.

Caixa S_2

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	F	1	8	E	6	B	3	4	9	7	2	D	C	0	5	A
1	3	D	4	7	F	2	8	E	C	0	1	A	6	9	B	5
2	0	E	7	B	A	4	D	1	5	8	C	6	9	3	2	F
3	D	8	A	1	3	F	4	2	B	6	7	C	0	5	E	9

Caixa S_3

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	A	0	9	E	6	3	F	5	1	D	C	7	B	4	2	8
1	D	7	0	9	3	4	6	A	2	8	5	E	C	B	F	1
2	D	6	4	9	8	F	3	0	B	1	2	C	5	A	E	7
3	1	A	D	0	6	9	8	7	4	F	E	3	B	5	2	C

Caixa S_4

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	7	D	E	3	0	6	9	A	1	2	8	5	B	C	4	F
1	D	8	B	5	6	F	0	3	4	7	2	C	1	A	E	9
2	A	6	9	0	C	B	7	D	F	1	3	E	5	2	8	4
3	3	F	0	6	A	1	D	8	9	4	5	B	C	7	2	E

Caixa S_5

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	2	C	4	1	7	A	B	6	8	5	3	F	D	0	E	9
1	E	B	2	C	4	7	D	1	5	0	F	A	3	9	8	6
2	4	2	1	B	A	D	7	8	F	9	C	5	6	3	0	E
3	B	8	C	7	1	E	2	D	6	F	0	9	A	4	5	3

Caixa S_6

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	C	1	A	F	9	2	6	8	0	D	3	4	E	7	5	B
1	A	F	4	2	7	C	9	5	6	1	D	E	0	B	3	8
2	9	4	F	5	2	8	C	3	7	0	4	A	1	D	B	6
3	4	3	2	C	9	5	F	A	B	E	1	7	6	0	8	D

Caixa S_7

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	4	B	2	E	F	0	8	D	3	C	9	7	5	A	6	1
1	D	0	B	7	4	9	1	A	E	3	5	C	2	F	8	6
2	1	4	B	D	C	3	7	E	A	F	6	8	0	5	9	2
3	6	B	D	8	1	4	A	7	9	5	0	F	E	2	3	C

Caixa S_8

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	D	2	8	4	6	F	B	1	A	9	3	E	5	0	C	7
1	1	F	D	8	A	3	7	4	C	5	6	B	0	E	9	2
2	7	B	4	1	9	C	E	2	0	6	A	D	F	3	5	8
3	2	1	E	7	4	A	8	D	F	C	9	0	3	5	6	B

Quando um bloco B_i de 6 bits entra na caixa S_k, sai um bloco C_k com 4 bits. Esta ação pode ser descrita pela função

$$C_k = S_k(B_k) = s^k_{i,\,j}$$

onde $s^k_{i,\,j}$ é o elemento da linha i coluna j da caixa k. A seleção da linha i e da coluna j são efetuadas do seguinte modo: o bloco B_k possui 6 bits, aqui designados por

$$b_5b_4b_3b_2b_1b_0$$

A linha i é fornecida pelo número binário

$$b_5b_0$$

formado pelos bits b_5 e b_0. A coluna j é fornecida pelo número binário

$$b_4b_3b_2b_1$$

formado com os bits restantes do bloco B_k e obtemos assim

$$C_k = S_k(B_k) = s^k_{i,\,j} = s^k_{b_5b_0\,,\,b_4b_3b_2b_1}.$$

Exemplo 2.1. *Sendo* B_1 *igual a* 101011_2, *a linha selecionada será* $11_2 = 3$ *e a coluna selecionada será* $0101_2 = 5$. *Consultando a tabela* S_1, *vemos que o elemento que está na linha* 3 *e na coluna* 5 *é o* $9 = 10012$. *Assim,* $S_1(B_1) = 9$.

Os blocos C_k que saem das caixas S_k formam um bloco de 32 bits que passam por uma permutação definida pela tabela Permutação P

Permutação P							
16	7	20	21	29	12	28	17
1	15	23	26	5	18	31	10
2	8	24	14	32	27	3	9
19	13	30	6	22	11	4	25

Escalonamento da chave nas 16 rodadas

A chave é uma sequência de 64 bits. Os bits 8, 16, ..., 64 são alocados para garantir a paridade dos 8 bytes da chave e, portanto, não fazem parte da chave. Descartados, permanecem apenas 56 bits que designaremos por K. Na época em que o DES foi projetado, o uso de bits de paridade era muito comum. Os circuitos ainda não eram muito confiáveis de modo que, dos 8 bits de um byte, apenas 7 eram aproveitados. O oitavo era usado para garantir que o byte contivesse sempre um número par de zeros e um número par de uns.

Os 56 bits da chave passam por uma permutação PC_1 apresentada na tabela Permutação PC_1. Se $x = x_1\ x_2\ \cdots\ x_{55}\ x_{56}$, então os bits de $y = PC_1(x)$ serão y_{57} $y_{49}\ y_{41}\ \cdots\ y_{20}\ y_{12}\ y_4$. Os índices acompanham as entradas da tabela Permutação PC_1 da esquerda para a direita e de cima para baixo.

Permutação PC_1							
57	49	41	33	25	17	9	1
58	50	42	34	26	18	10	2
59	51	43	35	27	19	11	3
60	52	44	36	63	55	47	39
31	23	15	7	62	54	46	38
30	22	14	6	61	53	45	37
29	21	13	5	28	20	12	4

Na saída de PC_1, os 56 bits são divididos ao meio, sendo C_0 a metade da esquerda e D_0 a da direita de modo que $C_0D_0 = PC_1(K)$. A partir deste momentos cada metade passa por 16 etapas semelhantes, fornecendo as chaves de cada rodada. Para $r = 1, 2, \ldots, 16$,

$$C_r = LS_r(C_{r-1})$$
$$D_r = LS_r(D_{r-1}),$$

onde LS_r é um deslocamento circular para a esquerda de um ou dois bits, que dependem da rodada r : um bit quando $r = 1, 2, 9, 16$, e dois bits nas demais rodadas. Para cada r, os 56 bits que formam C_rD_r perdem os bits que ocupam as posições 9, 18, 22, 25, 35, 38, 43, 54 e os demais são permutados, de acordo com uma tabela de permutação com redução PC_2 que fornece a chave da rodada r,

$$K_r = PC_2(C_rD_r),$$

que possui 48 bits. A tabela que aparece logo à frente apresenta a permutação com redução PC_2. Se $x_1\ x_2\ \cdots\ x_{55}\ x_{56}$ forem os bits de um bloco x, então os bits de $y = PC_2(x)$ serão $y_{14}\ y_{47}\ y_{11}\ \cdots\ y_{36}\ y_{29}\ y_{32}$. Os índices acompanham a tabela da esquerda para a direita e de cima para baixo.

Permutação com redução PC_2							
14	17	11	24	1	5	3	28
15	6	21	10	23	19	12	4
26	8	16	7	27	20	13	2
41	52	31	37	47	55	30	40
51	45	33	48	44	49	39	56
34	53	46	42	50	36	29	32

A figura 2.4 apresenta o esquema de escalonamento da chave.

2.2 O AES, Advanced Encryption Standard

Em 1997, o antigo National Bureau of Standards (NBS), agora com o nome de National Institute of Standards and Technology (NIST), fez uma chamada pública

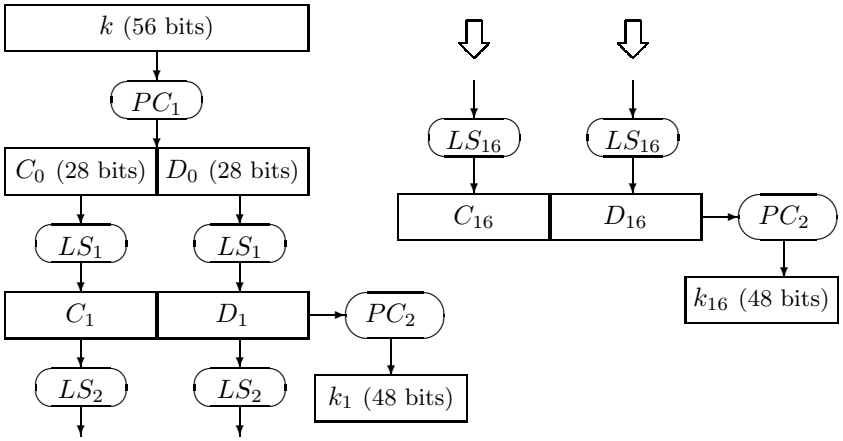

Figura 2.4: Esta figura apresenta o modo como a chave do DES é manipulada para fornecer os segmentos usados em cada rodada.

para que fossem apresentados candidatos para substituir o DES. O novo padrão seria denominado Advanced Encryption Standard (AES). Os requisitos básicos eram: o substituto do DES deveria aceitar chaves de 128, 192 e 256 bits e cifrar blocos de 128 bits. Deveria ser implementável eficientemente em arquiteturas de 8, 16 ou 32 bits. A velocidade seria um dos itens a ser considerado na avaliação. Em 1998, dentre 15 finalistas, 5 foram selecionados: o **MARS**, projetado pela IBM, o **RC6**, projetado pelo laboratório RSA, o **Rijndael**, dos belgas Joan Daemen e Vincente Rijmen, o **Twofish**, de Bruce Schneier, John Kelsey, Doug Whiting, David Wagner, Chris Hall e Niels Ferguson e o **Serpent**, de Ross Anderson, Eli Biham e Lars Knudsen. O escolhido para ser o AES foi o Rijndael. Todos os cinco algoritmos são muito bons e poderão ser usados futuramente.

Esta descrição segue de perto o livro dos seus idealizadores Daemen e Rijmen [3]. Embora o algoritmo original permita a cifração de blocos com 128, 192 e 256 bits, vamos descrever apenas a versão que cifra blocos com 128 bits e que foi adotada pelo AES. A chave pode conter 128, 192 e 256 bits.

2.2.1 Descrição do Rijndael, o novo AES

A melhor ilustração que vi sobre o AES foi realizada em flash por Enrique Zabala, enquanto aluno da Universidad ORT, Montevideo, Uruguai. Basta digitar Rijndael_Anim.exe em algum site de busca, tal como o Google, para encontrar um site que possui esta apresentação..

O algoritmo se baseia em quatro operações

1. SubByte: Transformação não linear, que age sobre um byte e é resistente à criptoanálise diferencial e linear.
2. ShiftRow: Transformação que realiza um deslocamento circular de 4 bytes e que produz difusão.
3. MixColumn: Transformação que trata um conjunto de 4 bytes como elementos de um corpo finito e realiza uma operação que produz difusão.
4. AddRoundKey: Realiza um XOR dos bits do bloco que está sendo cifrado com a chave da rodada.

Disposição matricial dos bytes de um bloco

O AES cifra blocos de 16 bytes, produzindo um bloco cifrado com 16 bytes. Os bytes de entrada são dispostos em uma matriz de tamanho 4×4 onde cada entrada é um byte.

Os 16 bytes

$$p_0 p_1 \cdots p_{15}$$

do bloco são dispostos na matriz, preenchendo-a coluna por coluna, de cima para baixo e da esquerda para a direita

$$A = \begin{bmatrix} p_0 & p_4 & p_8 & p_{12} \\ p_1 & p_5 & p_9 & p_{13} \\ p_2 & p_6 & p_{10} & p_{14} \\ p_3 & p_7 & p_{11} & p_{15} \end{bmatrix}.$$

A matriz A é chamada de **estado do sistema**. Se a entrada da linha i coluna j de A for designada por $a_{i\,,\,j}$, então, para $0 \leq i \leq 3$ e $0 \leq j \leq 3$

$$a_{i\,,\,j} = p_{i+4j}$$

e, para $0 \leq k \leq 15$.

$$p_k = a_{k \bmod 4\,,\; k/4}$$

onde $k/4$ é o quociente da divisão inteira de k por 4 (é a parte inteira de $k/4$).

Ao cifrar o AES realiza diversas transformações sobre a matriz A e se chega a uma matriz

$$B = \begin{bmatrix} b_{00} & b_{01} & b_{02} & b_{03} \\ b_{10} & b_{11} & b_{12} & b_{13} \\ b_{20} & b_{21} & b_{22} & b_{23} \\ b_{30} & b_{31} & b_{32} & b_{33} \end{bmatrix} = \begin{bmatrix} c_0 & c_4 & c_8 & c_{12} \\ c_1 & c_5 & c_9 & c_{13} \\ c_2 & c_6 & c_{10} & c_{14} \\ c_3 & c_7 & c_{11} & c_{15} \end{bmatrix},$$

partir da qual se obtém os 16 bytes do bloco cifrado

$$c_0 c_1 \ldots c_{15}.$$

A relação entre c_k e $b_{i\,,\,j}$ é

$$c_k = b_{k \bmod 4\,,\,k/4}$$

para $0 \leq k < 15$, onde $k/4$ é o quociente da divisão inteira de k por 4. A igualdade que fornece $b_{i\,,\,j}$ a partir de c_k é

$$b_{i,j} = c_{i+4j}$$

para $0 \leq i \leq 3$ e $0 \leq j \leq 3$.

Para decifrar, o AES realiza uma sequência de transformações sobre a matriz B até recuperar a matriz A.

A transformação AddRoundKey (ARK)

O AES aceita chaves com 16, 24 ou 32 bytes. Independentemente do tamanho da chave, ela é escalonada de forma que a chave da rodada K^r é sempre uma matriz 4×4, cujas entradas são bytes. A chave da rodada r é uma matriz 4×4

$$K^r = \begin{bmatrix} k^r_{00} & k^r_{01} & k^r_{02} & k^r_{03} \\ k^r_{10} & k^r_{11} & k^r_{12} & k^r_{13} \\ k^r_{20} & k^r_{21} & k^r_{22} & k^r_{23} \\ k^r_{30} & k^r_{31} & k^r_{32} & k^r_{33} \end{bmatrix}$$

cujas entradas $k^r_{i\,,\,j}$ são bytes. A técnica usada para obter K^r a partir da chave original K será descrita posteriormente.

A operação *AddRoundKey*, por brevidade indicada por *ARK*, recebe na entrada A e K^r e faz a operação XOR bit a bit entre os bytes de A e de K^r

$$ARK(A, K^r) = \left[a_{i\,,\,j} \oplus k^r_{i\,,\,j}\right]$$

onde a operação $\oplus$ é o XOR bit a bit dos bytes envolvidos.

Observa-se que a operação XOR bit a bit é comutativa, associativa e posui elemento neutro, o bit zero. Sendo x um bit (0 ou 1), então $x \oplus x = 0$. Isto significa que, na operação de adição módulo 2, o oposto de x é o próprio x. Daí, se x, y e z forem três bits tais que $x \oplus y = z$, então $x = y \oplus z$.

A operação *AddRoundKey* é invertível e sua inversa é ela mesma. Se

$$b_{i\,,\,j} = a_{i\,,\,j} \oplus k^r_{i\,,\,j} \qquad \text{então} \qquad a_{i\,,\,j} = b_{i\,,\,j} \oplus k^r_{i\,,\,j}.$$

A transformação SubBytes (SB)

A transformação *SubBytes*, indicada por *SB* para ser breve, age individualmente sobre cada byte da matriz A, sendo a única tranformação não linear do Rijndael.

Já tivemos a oportunidade de verificar que o conjunto $\mathbf{Z}_2 = \{0, 1\}$ com as operações de adição e multiplicação módulo 2 é um corpo finito. Observe que $\mathbf{Z}_2$ possui dois elementos e 2 é um número primo.

Prova-se que, se $\mathbf{K}$ for um corpo finito, então ele possui p^n elementos, onde p é primo e n é um inteiro maior do que zero. Existem corpos finitos com 2, 2^2, 2^3, ..., 3, 3^2, 3^3, ... 5, 5^2, 5^3, ... elementos mas não existe corpo finito com 6 elementos. Além do mais, dado um conjunto com p^n elementos, sempre é possível definir sobre ele uma adição e uma multiplicação que o torna um corpo finito.

Dois corpos $(\mathbf{K}_1, +, \times)$ e $(\mathbf{K}_2, \oplus, \otimes)$ são **isomorfos** se existir uma função bijetora $f : \mathbf{K}_1 \to \mathbf{K}_2$ tal que $f(x + y) = f(x) \oplus f(y)$ e $f(x \times y) = f(x) \otimes f(y)$. Tal função f é denominada de **isomorfismo** entre os corpos $\mathbf{K}_1$ e $\mathbf{K}_2$. O isomorfismo, por ser bijetor, é invertível e sua inversa é um isomorfismo. Prova-se que dois corpos finitos com o mesmo número de elementos são isomorfos. Tal fato torna indistinguíveis dois corpos finitos com o mesmo número de elementos pois o isomorfismo transfere todas as propriedades de um para o outro. Ao apresentar um corpo finito com p^n elementos, caracterizamos todos os demais corpos finitos com p^n elementos.

Os corpos finitos recebem o nome de Corpos de Galois, em homenagem a Évariste Galois (1811 – 1832), matemático francês que morreu na flor da idade em um duelo na disputa pelo amor de uma mulher. Há uma narrativa bastante interessante na Wikipedia sobre a vida e os acontecimentos que resultaram na morte prematura deste grande matemático. Designa-se por $GF(p^n)$ o corpo finito com p^n elementos, onde GF são as iniciais de Galois Field. Em particular, o $\mathbf{Z}_2$ com as operações de adição e multiplicação módulo 2 é um corpo. Sendo p um número primo, os corpos com p elementos são os $\mathbf{Z}_p$ com as operações de adição e multiplicação módulo p.

Existe um corpo com 2^8 elementos e vamos apresentá-lo. Lembramos que qualquer outro corpo com o mesmo número de elementos será isomorfo a ele e, portanto, um e outro serão indistinguíveis.

Consideremos o conjunto dos polinômios na variável X de grau menor do que 8 com coeficientes em $\mathbf{Z}_2$ que possui 2^8 elementos e que designaremos por $GF(2^8)$. Sendo

$$p(X) = p_0 + p_1X + \cdots + p_6X^6 + p_7X^7$$
$$q(X) = q_0 + q_1X + \cdots + q_6X^6 + q_7X^7$$

dois elementos de $GF(2^8)$, definimos a operação de adição

$$p(X) \oplus q(X) = (p_0 \oplus q_0) + (p_1 \oplus q_1)X + \cdots + (p_6 \oplus q_6)X^6 + (p_7 \oplus q_7)X^7$$

onde $p_i \oplus q_i$ é a adição módulo 2.

Exemplo 2.2. *Sejam* $p(X) = 1 + X + X^2 + X^3 + X^4 + X^5$ *e* $q(X) = 1 + X^2 + X^4 + X^7$ *dois polinômios em* $GF(2^8)$,

$$\begin{aligned} p(X) \oplus q(X) &= (1 \oplus 1) + (1 \oplus 0)X + (1 \oplus 1)X^2 + (1 \oplus 0)X^3 \\ &\quad + (1 \oplus 1)X^4 + (1 \oplus 0)X^5 + (0 \oplus 0)X^6 + (0 \oplus 1)X^7 \\ &= X + X^3 + X^5 + X^7 \end{aligned}$$

pois $0 \oplus 0 = 0$, $1 \oplus 0 = 1$, $0 \oplus 1 = 1$ *e* $1 \oplus 1 = 0$.

A multiplicação "usual" de dois polinômios em $GF(2^8)$, aquela em que usamos as regras usuais de multiplicação de polinômios,

$$p(X) \times q(X) = (p_0 \otimes q_0) + (p_0 \otimes q_1) \oplus (p_1 \otimes q_0)X + \cdots$$

onde $\oplus$ e $\otimes$ são, respectivamente, a adição e a multiplicação módulo 2, não resulta em um polinômio em $GF(2^8)$. Por exemplo,

$$(1 + X^2 + X^5 + X^7) \cdot (X^3 + X^6) = X^3 + X^6 + X^5 + X^{11} + X^{10} + X^{13}$$

não é um elemento de $GF(2^8)$. Para definir um produto em $GF(2^8)$ que resulte num elemento dele, precisamos de uma redução modular como aquela que é feita em $\mathbf{Z}_p$.

O número primo é aquele que não é decomponível no produto de outros dois. Precisamos um conceito análogo no reino dos polinômios que é o de polinômio irredutível.

Um polinômio $n(X)$ com coeficientes em $GF(2)$ é redutível em $GF(2)$ se $n(X) = p(X)\ q(X)$, onde $p(X)$ e $q(X)$ são polinômios com coeficientes em $GF(2)$ cujos graus são maiores que zero e menores do que o grau de $n(X)$. Um polinômio que não é redutível é chamado irredutível. Um exemplo de polinômio irredutível em $GF(2)$ é

$$n(X) = X^8 + X^4 + X^3 + X + 1.$$

Dado um polinômio $k(X)$ com coeficientes em $GF(2)$, podemos dividi-lo por $n(X)$ obtendo um quociente e um resto $r(X)$, cujo grau é menor do que 8 e, consequentemente, um elemento de $GF(2^8)$. Denotamos o resto $r(X)$ por $k(X) \bmod n(X)$, onde se lê "$k(X)$ módulo $n(X)$". Sendo $p(X)$ e $q(X)$ dois polinômios em $GF(2^8)$ definimos agora a multiplicação

$$p(X) \otimes q(X) = p(X) \cdot q(X) \bmod n(X)$$

que é um elemento de $GF(2^8)$. O $GF(2^8)$ com as operações $\oplus$ e $\otimes$ é um corpo e foi usado pelos idealizadores do AES. Como $n(X)$ é irredutível em $GF(2)$, ele não

é divisível por nenhum outro polinômio com coeficientes em $GF(2)$. Dado $p(X)$ em $GF(2^8)$, ele não possui fator comum com $n(X)$ e assim os dois polinômios são primos entre si e pode-se obter o inverso de $p(X)$ módulo $n(X)$ usando o algoritmo de Euclides estendido para polinômios, que é idêntico ao algoritmo desenvolvido para números inteiros.

Cada bit pode assumir o valor 0 ou 1 e, como cada byte possui 8 bits, existem 2^8 bytes diferentes, que podem ser considerados números binários que vão de 0000 0000 a 1111 1111. Na base 10, os valores dos bytes vão do zero ao 255. Sendo $b_7 b_6 \dots b_1 b_0$ os bits de um byte, onde b_7 é o bit mais significativo e b_0 o menos significativo, podemos identificá-lo ao polinômio

$$b_7 X^7 + b_6 X^6 + \cdots + b_1 X + b_0.$$

que pertence a $GF(2^8)$. Logo, se $\mathbf{b}$ for um byte, mediante a identificação feita, podemos calcular seu inverso multiplicativo $\mathbf{b}^{-1}$ em $GF(2^8)$ usando o algoritmo de Euclides estendido fornece um meio eficiente de calcular este inverso. Definimos a função

$$g : GF(2^8) \to GF(2^8)$$

por $g(0) = 0$ e, quando $\mathbf{b}$ não é o zero, $g(\mathbf{b}) = \mathbf{b}^{-1}$. Esta função é bijetora e sua inversa é ela mesma. Em seguida, definimos a transformação

$$f : GF(2^8) \to GF(2^8)$$

que leva o byte $\mathbf{b} = b_7\ b_6\ b_5\ b_4\ b_3\ b_2\ b_1\ b_0$ no byte $\mathbf{c} = f(\mathbf{b}) = c_7\ c_6\ c_5\ c_4\ c_3\ c_2\ c_1\ c_0$ mediante a fórmula

$$\begin{pmatrix} c_7 \\ c_6 \\ c_5 \\ c_4 \\ c_3 \\ c_2 \\ c_1 \\ c_0 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \end{pmatrix} \otimes \begin{pmatrix} b_7 \\ b_6 \\ b_5 \\ b_4 \\ b_3 \\ b_2 \\ b_1 \\ b_0 \end{pmatrix} \oplus \begin{pmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \end{pmatrix}$$

onde o produto matricial é aquele usual com a ressalva de que as operações de adição e multiplicação são aquelas módulo 2. A função f é invertível e sua inversa $\mathbf{b} = f^{-1}(\mathbf{c})$ é

$$\begin{pmatrix} b_7 \\ b_6 \\ b_5 \\ b_4 \\ b_3 \\ b_2 \\ b_1 \\ b_0 \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \end{pmatrix} \otimes \begin{pmatrix} c_7 \\ c_6 \\ c_5 \\ c_4 \\ c_3 \\ c_2 \\ c_1 \\ c_0 \end{pmatrix} \oplus \begin{pmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \end{pmatrix}$$

A transformação *SubBytes*, por brevidade designada SB, é aquela que leva a matriz $A = [a_{i\,,\,j}]$ de tamanho 4×4 cujas entradas são bytes, na matriz $B = SB(A) = [b_{i\,,\,j}]$ que possui mesmo tamanho que A. As entradas de A e B estão relacionados por

$$b_{i\,,\,j} = f \circ g(a_{i\,,\,j}) = f\left(g(a_{i\,,\,j})\right).$$

Como f e g são invertíveis, a inversa da *SubBytes*, denominada *InvSubBytes*, abreviada para ISB, é aquela que leva a matriz $B = [b_{i\,,\,j}]$ na matriz $A = [a_{i\,,\,j}]$ onde $a_{i\,,\,j} = g^{-1} \circ f^{-1}(b_{i\,,\,j})$.

Como $GF(2^8)$ possui 256 elementos, as funções f, g, $f \circ g$ e suas inversas, podem ser implementadas de modo eficientes através de tabelas, cada uma possuindo 256 bytes.

De acordo com Daemen e Rijmen, a escolha desta transformação foi inspirada pelo artigo "Differentially uniform mappings for cryptography´´, publicado no Advances in Cryptography, Proc. Eurocript'93, LNCS 765, T. Helleseth, Ed., Springer-Verlag, 1994, pp. 55-64, de Kaisa Tellervo Nyberg, atualmente Research Fellow em Wireless Security da Nokia. Neste trabalho, a autora sugere diversos métodos para a construção de uma caixa S não linear possuindo boas qualidades criptográficas.

Para todo byte $\mathbf{b}$, vale ressaltar que

$$f \circ g(\mathbf{b}) \oplus \mathbf{b} \neq 00_{16},$$
$$f \circ g(\mathbf{b}) \oplus \mathbf{b} \neq FF_{16},$$

onde o subíndice 16 indica que os bytes estão apresentados na base hexadecimal.

A transformação ShiftRow (SR)

O passo $ShiftRow$, denotada SR por brevidade, age sobre uma matriz de tamanho 4×4, cujas entradas são bytes. Ela produz difusão ótima, resistente ao ataque diferencial truncado e os ataques de saturação precisam ser maximizados para serem efetivos. Tal transformação produz um deslocamento horizontal circular nos bytes das diversas linhas. A primeira linha permanece como está. A segunda

linha sofre um deslocamento cíclico para a esquerda de um byte, a terceira linha de dois bytes e a quarta linha de três bytes.

$$SR\begin{bmatrix} a_{0,0} & a_{0,1} & a_{0,2} & a_{0,3} \\ a_{1,0} & a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,0} & a_{2,1} & a_{2,2} & a_{2,3} \\ a_{3,0} & a_{3,1} & a_{3,2} & a_{3,3} \end{bmatrix} = \begin{bmatrix} a_{0,0} & a_{0,1} & a_{0,2} & a_{0,3} \\ a_{1,1} & a_{1,2} & a_{1,3} & a_{1,0} \\ a_{2,2} & a_{2,3} & a_{2,0} & a_{2,1} \\ a_{3,3} & a_{3,0} & a_{3,1} & a_{3,2} \end{bmatrix}.$$

ou sejam, sendo $SR([a_{i\,,\,j}]) = [b_{i\,,\,j}]$, então

$$b_{i\,,\,j} = a_{i\,,\,(j+i) \bmod 4}.$$

Esta operação é invertível e sua inversa é chamada $InvShiftRow$ ou, abreviadamente, ISR. Se $ISR([b_{i\,,\,j}]) = [a_{i\,,\,j}]$ então

$$a_{i\,,\,j} = b_{i\,,\,(j-i) \bmod 4}$$

A transformação MixColumns (MC)

A transformação $MixColumns$ do AES, denotada MC por brevidade, age sobre as colunas das matrizes 4×4 cujas entradas são bytes. Cada coluna da matriz $A = [a_{i\,,\,j}]$ de tamanho 4×4, cujas entradas são bytes, é encarada como sendo um elemento de $GF\left(256^4\right) = GF\left(\left(2^8\right)^4\right)$. Os elementos deste corpo podem ser identificados aos polinômios de grau menor do que quatro com coeficientes em $GF\left(2^8\right)$. Sendo $[a_0\ a_1\ a_2\ a_3]^T$ uma matriz coluna cujas entradas são bytes, nós a identificamos ao polinômio

$$a(X) = a_0 + a_1X + a_2X^2 + a_3X^3.$$

A transformação $MixColumn$ leva $a(X)$ em

$$b(X) = a(X) \cdot c(X) \bmod m(X)$$

onde $m(X) = X^4 + 1$ e $c(X) = 03X^3 + 01X^2 + 01X + 02$. Neste polinômio, 01, 02 e 03 são os valores dos bytes escritos na base 16.

Sendo $a(X) = a_0 + a_1X + a_2X^2 + a_3X^3$ e $b(X) = b_0 + b_1X + b_2X^2 + b_3X^3$ podemos apresentar a relação entre a_i e b_i na forma matricial

$$\begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{pmatrix} = MC \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix} = \begin{pmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix}$$

onde as entradas 01, 02 e 03 da matriz são bytes expressos na base hexadecimal e as operações são realizadas em $GF(2^8)$.

O polinômio $m(X) = X^4 + 1$ não é irredutível pois $m(X) = (X + 1)^4$. Entretanto, os polinômios $c(X)$ e $m(X)$ são primos entre si o que garante a existência de um polinômio $d(X)$ tal que

$$d(X) \cdot c(X) \bmod m(X) = 1.$$

O polinômio $d(X)$ é o inverso de $c(X)$ módulo $m(X)$ e ele é

$$d(X) = (0B)X^3 + (0D)X^2 + (09)X + (0E)$$

onde os coeficientes, colocados entre parêntesis para destaque, são bytes expressos na base hexadecimal.

Sendo

$$b(X) = a(X) \cdot c(X) \bmod m(X),$$

então

$$a(X) = b(X) \cdot d(X) \bmod m(X).$$

Tal fato garante que a operação $MixColumn$ é invertível. Sua inversa, denominada $InvMixColumns$ será denotada abreviadamente por IMC.

Sendo $a(X) = a_0 + a_1X + a_2X^2 + a_3X^3$ e $b(X) = b_0 + b_1X + b_2X^2 + b_3X^3$ podemos apresentar a relação entre a_i e b_i na forma matricial

$$\begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix} = IMC \begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{pmatrix} = \begin{pmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{pmatrix} \begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{pmatrix}$$

onde as entradas da matriz são bytes expressos na base hexadecimal e as operações são realizadas em $GF(2^8)$.

Nota 2.1. *Paulo Sérgio Licciardi Messeder Barreto, Professor da Escola Politécnica da USP, observou que*

$$d(X) = (04X^2 + 05) \cdot c(X) \bmod (X^4 + 1)$$

o que permite implementar a operação $InvMixColumns$ através de um passo de pré-processamento seguido por um passo de $MixColumns$.

A chave

A chave pode ter 16, 24 ou 32 bytes. Vamos descrever o AES com chave de 16 bytes, o que corresponde a 128 bits. Quando este é o caso, os bytes $z_0, z_1, \ldots, z_{15}$

da chave são armazenados numa matriz K de tamanho 4×4, preenchida coluna por coluna, de cima para baixo e da esquerda para a direita

$$K = \begin{bmatrix} k_{00} & k_{01} & k_{02} & k_{03} \\ k_{10} & k_{11} & k_{12} & k_{13} \\ k_{20} & k_{21} & k_{22} & k_{23} \\ k_{30} & k_{31} & k_{32} & k_{33} \end{bmatrix} = \begin{bmatrix} z_0 & z_4 & z_8 & z_{12} \\ z_1 & z_5 & z_9 & z_{13} \\ z_2 & z_6 & z_{10} & z_{14} \\ z_3 & z_7 & z_{11} & z_{15} \end{bmatrix}.$$

A relação entre $k_{i\,,\,j}$ e z_k, para $0 \leq i \leq 3$ e $0 \leq j \leq 3$, é

$$k_{i\,,\,j} = z_{i+4j}$$

e a relação inversa, para $0 \leq k < 15$, é

$$c_k = a_{k \bmod 4\,,\,k/4}$$

onde $k/4$ é o quociente da divisão inteira de k por 4. A partir de transformações efetuadas na chave K, vamos obter as chaves K^r de cada rodada r.

Escalonamento da chave para as diversas rodadas

Denotemos por $W(0)$, $W(1)$, $W(2)$, $W(3)$ as quatro colunas de K. A elas serão anexadas outras quarenta colunas, geradas recursivamente do seguinte modo:

Para $j = 4, 5, \ldots, 43$,

1. Se j não for divisível por 4, então

$$W(j) = W(j-4) \oplus W(j-1).$$

2. Quando j for divisível por 4,

$$W(j) = W(j-4) \oplus T_j(W(j-1))$$

onde

$$T_j \begin{pmatrix} w_0 \\ w_1 \\ w_2 \\ w_3 \end{pmatrix} = \begin{bmatrix} w_1^{-1} \\ w_2^{-1} \\ w_3^{-1} \\ w_0^{-1} \end{bmatrix} \oplus \begin{bmatrix} 2^{(j-1)/4} \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

e $(j-1)\,/\,4$ é o quociente da divisão inteira de $j-1$ por 4.

As constantes $2^{(j-1)/4}$ adicionadas na primeira linha de W, sempre que j é um múltiplo de 4, foram usadas pelos projetistas do AES para eliminar simetrias no processo de cifra e tornando as colunas de cada rodada diferentes das anteriores.

3. A chave da rodada r é a matriz formada pelas colunas

$$W(4r), \quad W(4r+1), \quad W(4r+2), \quad W(4r+3).$$

Por este processo se constrói uma matriz W com 4 linhas e 44 colunas, cujas quatro primeiras colunas é a chave original. Vamos descrever byte a byte como se constrói a matriz $W = [w_{i\,,\,j}]$, de tamanho 4×44, cujas colunas são $W(j)$ com j assumindo os valores 0, 1, 2, ..., 43.

1. Para $i = 0, 1, 2, 3$ e $j = 0, 1, 2, 3$,

$$w_{i\,,\,j} = k_{i\,,\,j},$$

2. Para $i = 0, 1, 2, 3$ e $j \geq 4$,

   (a) quando j não for divisível por 4, faça

$$w_{i\,,\,j} = w_{i\,,\,j-4} \oplus w_{i\,,\,j-1}$$

   (b) quando j for divisível por 4, faça

$$w_{0\,,\,j} = w_{0\,,\,j-4} \oplus w^{-1}_{i\,,\,j-1} \oplus 2^{(j-1)/4}$$
$$w_{i\,,\,j} = w_{i\,,\,j-4} \oplus w^{-1}_{(i+1) \bmod 4\,,\,j-1}$$

Cifrar com o AES

Para cifrar, o AES realiza transformações sobre uma matriz de tamanho 4×4, cujas entradas são bytes. Vamos indicar por ARK a operação $AddRoundKey$, por SB a operação $SubByte$, por SR a operação $ShiftRow$ e por MC a operação $MixColumns$.

Para cifrar um bloco matricial A, realize as operações:

1. $B^1 \leftarrow ARK(A, K)$
2. Para r variando de 1 a 9 :
   (a) $C^r \leftarrow SB(B^r)$
   (b) $D^r \leftarrow SR(C^r)$
   (c) $E^r \leftarrow MC(D^r)$
   (d) $B^{r+1} \leftarrow ARK(E^r, K^r)$
3. Na décima rodada execute as operações:

(a) $C^{10} \leftarrow SB(B^{10})$

(b) $D^{10} \leftarrow SR(C^{10})$

(c) $B \leftarrow ARK(D^{10}, K^{10})$

4. O B é o bloco cifrado na forma matricial. Para obter o bloco cifrado, basta ler as entradas desta matriz coluna por coluna, percorrendo-as de cima para baixo e da esquerda para a direita.

Escrevendo de outra forma, para cifrar A, calcule

$B^1 \leftarrow$ ARK(A, K)
$C^1 \leftarrow$ SB(B^1), $D^1 \leftarrow$ SR(C^1), $E^1 \leftarrow$ MC(D^1), $B^2 \leftarrow$ ARK(E^1, K^1)
...
$C^9 \leftarrow$ SB(B^9), $D^9 \leftarrow$ SR(C^9), $E^9 \leftarrow$ MC(D^9), $B^{10} \leftarrow$ ARK(E^9, K^9)
$C^{10} \leftarrow$ SB(B^{10}), $D^{10} \leftarrow$ SR(C^{10}), B $\leftarrow$ ARK(D^9, K^9)

A figura 2.5 apresenta as nove rodadas do AES. A décima não realiza o MixColumn.

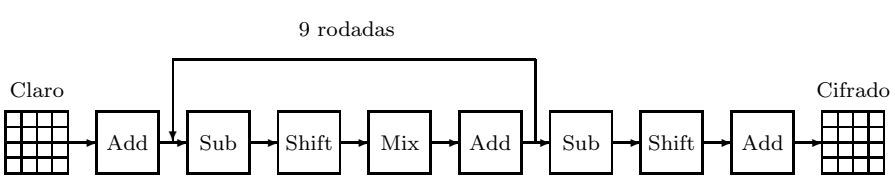

Figura 2.5: Esquema dos blocos principais de um round do AES.

Decifrar com o AES

Para decifrar basta seguir o caminho inverso àquele seguido no processo de cifra, iniciando na décima rodada. Vamos designar por ARK a operação $AddRoundKey$, por ISR a operação $ShiftRow$ inversa, por ISB a operação $SubByte$ inversa e por IMC a operação $MixColumns$ inversa.

Para decifrar o bloco B, basta executar as operações:

1. Na primeira rodada execute:

   (a) $D^{10} \leftarrow ARK(B, K^{10})$

   (b) $C^{10} \leftarrow ISR(D^{10})$

(c) $B^{10} \leftarrow ISB(C^{10})$

2. Para r variando de 9 a 1 calcule:

   (a) $E^r \leftarrow ARK(B^{r+1}, K^r)$
   (b) $D^r \leftarrow IMC(E^r)$
   (c) $C^r \leftarrow ISR(D^r)$
   (d) $B^r \leftarrow ISB(C^r)$

3. Na décima rodada calcule:

$A \leftarrow ARK(B^1, K)$

O bloco decifrado na forma matricial é A. Agora, basta ler as entradas desta matriz coluna por coluna, percorrendo-as de cima para baixo e da esquerda para a direita, obtendo o bloco de bytes decifrado.

Escrevendo de outra forma o processo que decifra B,

B $\leftarrow$ ARK(B, K_{10})
B $\leftarrow$ ISR(B), B $\leftarrow$ ISB(B), B $\leftarrow$ ARK(B, K_9)
B $\leftarrow$ IMC(B), B $\leftarrow$ ISR(B), B $\leftarrow$ ISB(B), B $\leftarrow$ ARK(B, K_8)
...
B $\leftarrow$ IMC(B) , B $\leftarrow$ ISR(B), B $\leftarrow$ ISB(B), B $\leftarrow$ ARK(B, K_0)

onde o B final é o bloco decifrado.

Método alternativo para decifrar com o AES

É interessante observar que, como o *SubByte* opera sobre os bytes individualmente e o $ShiftRow$ desloca circularmente as linhas da matriz sem modificar os bytes individuais, estas duas operações comutam

$$SR(SB(A)) = SB(SR(A)).$$

Além do mais, verificamos que a operação $MixColumns$, que denotamos por MC, pode ser colocada na forma matricial

$$MC \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix} = \begin{pmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{pmatrix} \otimes \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix}.$$

Tal como matrizes reais, a multiplicação de matrizes cujas entradas pertencem a $GF(2^8)$ é distributiva em relação à adição. Se A e B forem matrizes 4×4 cujas

entradas são bytes, que podem ser considerados elementos de $GF(2^8)$, e P for a matriz da operação MC, então

$$MC(A \oplus B) = P \otimes (A \oplus B) = (P \otimes A) \oplus (P \otimes B) = MC(A) \oplus MC(B).$$

Considerando que $\oplus$ é a adição módulo 2 usada na operação ARK, podemos escrever

$$MC(ARK(A, B)) = ARK(MC(A), MC(B))$$

Do mesmo modo, denotando a operação $InvMixColumns$ por IMC, temos

$$IMC(A \oplus B) = IMC(A) \oplus IMC(B)$$

ou, usando o fato de $\oplus$ ser a operação usada na ARK, segue

$$IMC(ARK(A, B)) = ARK(IMC(A), IMC(B))$$

Esta propriedade, aliada à comutatividade entre as operações $SubByte$ e $ShiftRow$, nos oferece uma técnica alternativa para decifrar que fica mais próxima do processo que cifra e pode ser expressa esquematicamente por

B = ARK(B, K_{10})
B = ISB(B), B = ISR(B), B = IMC(B), B = ARK(B, IMC(K_9))
...
B = ISB(B), B = ISR(B), B = IMC(B), B = ARK(B, IMC(K_1))
B = ISB(B), B = ISR(B), B = ARK(B, IMC(K_0))

Considerações do projeto

Os autores do Rijndael em [3] citam ainda que

1. Duas rodadas produzem difusão completa. Cada um dos 128 bits da saída depende de cada um dos 128 bits da entrada.
2. O S–box é altamente não linear, resistente à criptoanálise diferencial e linear, bem com ao ataque por interpolação.
3. ShiftRows foi introduzido para resistir aos ataques do quadrado e do diferencial truncado.
4. O MixColumns produz difusão entre os bytes.
5. O escalonamento da chave envolve o embaralhamento não linear dos bits da chave, pois ele usa o S–box.

6. A evolução da chave para as diversas rodadas foi projetada para resistir a ataques em que o criptoanalista conhece um pedaço da chave e tenta deduzir os outros bits.

7. Existem ataques mais eficientes que a força bruta até 6 rodadas. Não se conhecem ataques mais eficientes que a força bruta para 7 rodadas ou mais.

Pictogramas

Usando os **pictogramas** da figura 2.6, em 2.7 se apresenta o AES com duas rodadas. Em 2.8 se mostra como se decifra com o AES de duas rodadas e na figura 2.9 se pode apreciar o funcionamento do processo alternativo para decifrar.

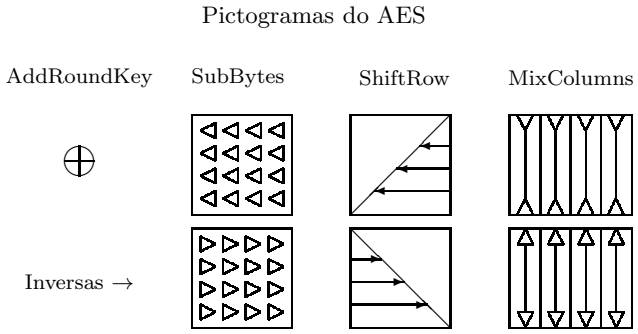

Figura 2.6: Esta figura descreve de forma pictográfica os componentes do AES.

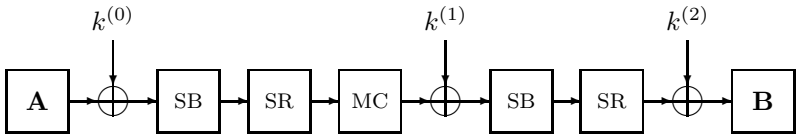

Figura 2.7: Esta figura mostra como cifra o AES com apenas duas rodadas.

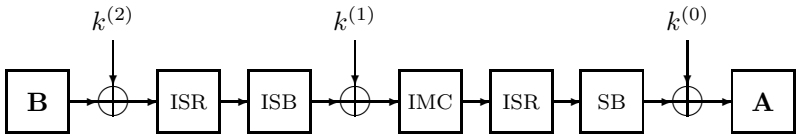

Figura 2.8: Esta figura mostra como decifra o AES com apenas duas rodadas.

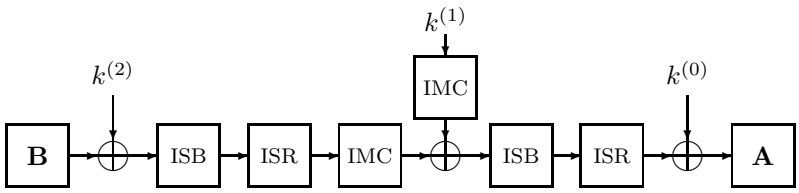

Figura 2.9: Esta figura apresenta um método alternativo para decifrar com o AES de duas rodadas.

2.3 Cifrando diversos blocos

Até o momento descrevemos como se cifra e decifra um único bloco de texto. De modo geral, os documentos a serem cifrados precisam ser quebrados em dois ou mais blocos para serem criptografados. Existem alguns procedimentos utilizados para cifrar os diversos blocos de um texto claro, gerando assim um texto criptografado. Tais procedimentos serão descritos nesta seção e se aplicam a às cifras de bloco.

Ao dividir um documento em blocos, nem sempre o último segmento formará um bloco completo para ser cifrado. Neste caso, acrescenta-se a ele alguns bits adicionais para que o último bloco fique completo e com o mesmo número de bits que os demais. Este processo é chamado de padding e será descrito ainda neste capítulo.

Vamos supor que desejamos cifrar uma mensagem

$$\mathbf{x} = x_1 x_2 \dots x_i \dots$$

onde x_i é um bloco genérico do texto, com tamanho apropriado para a cifra que será utilizada. A mensagem cifrada

$$\mathbf{y} = y_1 y_2 \dots y_i \dots$$

será obtida concatenando os blocos y_i calculados pelos modos de funcionamento que serão descritos a seguir.

Vamos denotar por E_k a função que cifra um bloco usando uma chave k e denotar por D_k a função que decifra com a mesma chave.

O modo de funcionamento mais simples é o **Electronic Codebook**, designado resumidamente por **ECB**. Para cifrar o texto $\mathbf{x}$, realizamos a operação

$$y_i = E_k(x_i)$$

para $i = 1, 2, \ldots$. A concatenação dos diversos blocos y_i resulta no texto cifrado $\mathbf{y}$.

Quem recebe o texto cifrado $\mathbf{y}$, possuindo a chave k pode decifrá-lo mediante a operação

$$x_i = D_k(y_i)$$

efetuada para $i = 1, 2, \ldots$. A concatenação dos blocos x_i recupera o texto claro $\mathbf{x}$.

A figura 2.10 apresenta esquematicamente o ECB e a figura 2.11 mostra o esquema de como se decifra usando este modo de funcionamento.

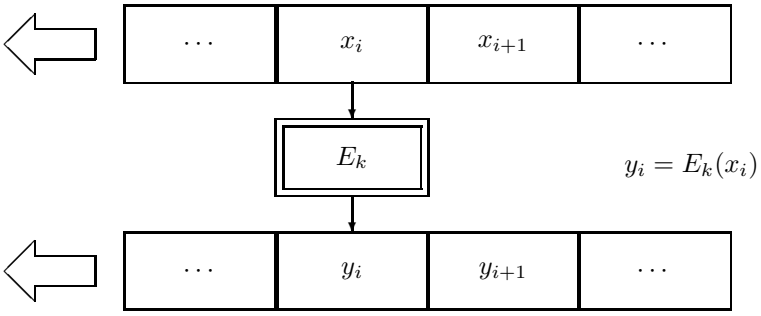

Figura 2.10: Esta figura apresenta o modo ECB usado para cifrar um texto partido em blocos.

Outro modo de usar uma cifra de bloco é o **Cipher Block Chaining** indicado abreviadamente por **CBC** Para cifrar o texto $\mathbf{x}$, as partes que irão se comunicar combinam previamente um bloco $y_0 = VI$ onde VI significa vetor de inicialização. Em seguida, os blocos são cifrados por

$$y_i = E_k(y_{i-1} \oplus x_i)$$

fazendo $i = 1, 2, \ldots$. Nesta operação, $\oplus$ é a operação XOR (ou exclusivo, adição módulo 2) realizada bit a bit. Graças às propriedades do XOR, o texto cifrado poderá ser decifrado usando

$$x_i = y_{i-1} \oplus D_k(y_i)$$

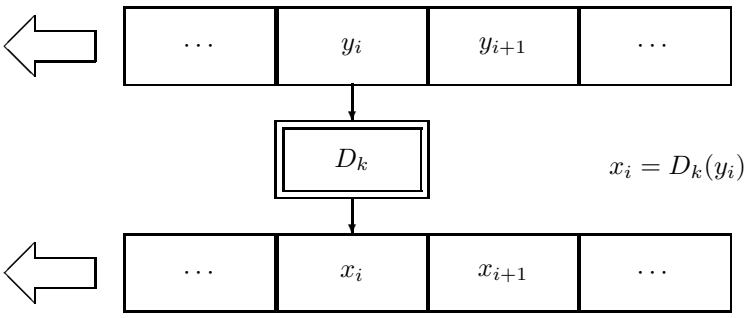

Figura 2.11: Esta figura apresenta o modo ECB usado para decifrar um texto partido em blocos.

efetuada para $i = 1, 2, \ldots$. Certamente, quem decifra precisa da chave k e do vetor de inicialização y_0. As figuras 2.12 e 2.13 apresentam os esquemas desse modo de funcionamento.

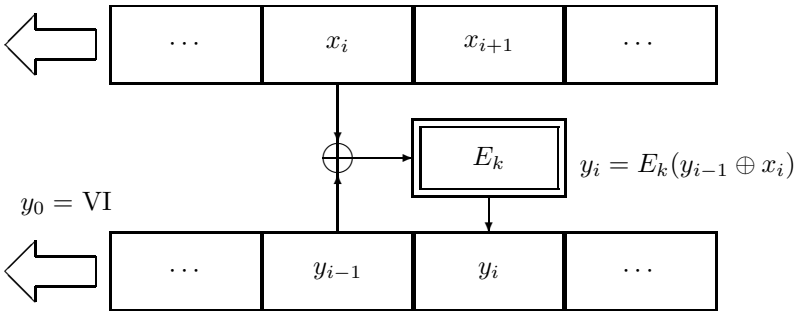

Figura 2.12: Esta figura apresenta o modo CBC usado para cifrar um texto partido em blocos. O bloco cifrado y_i é obtido mediante a fórmula $y_i = E_k(y_{i-1} \oplus x_i)$

Existe também o modo **Cipher Feedback** ou, resumidamente, **CFB**. Como no modo **CBC**, combina-se um vetor de inicialização $y_0 = VI$ e os blocos são cifrados mediante a operação

$$y_i = x_i \oplus E_k(y_{i-1})$$

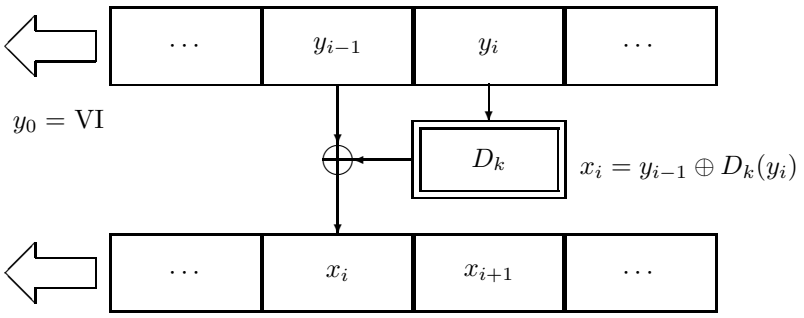

Figura 2.13: Esta figura apresenta o modo CBC usado para decifrar um texto partido em blocos. O bloco decifrado x_i é obtido mediante a fórmula $x_i = y_{i-1} \oplus D_k(y_i)$

e decifrados por

$$x_i = y_i \oplus E_k(y_{i-1})$$

fazendo $i = 1, 2, \ldots$. É interessante observar que aqui se usa apenas a operação E_k que cifra. Para dispositivos com pouca memória ele pode ser útil uma vez que basta programar a função E_k. Os esquemas para cifrar e decifrar deste modo de operação são apresentados nas figuras 2.14 e 2.15.

Existe o **Cipher Feedback** na versão de 8 bits,, denotado abreviadamente por **CFB8** que vamos descrever para cifras sobre blocos de 64 bits e opera do seguinte modo: As pessoas que desejam manter uma correspondência criptografada, combinam um vetor de inicialização $z_0 = VI$ e cifram o $\mathbf{x}$ usando

$$\begin{aligned} y_i &= x_i \oplus L_8(E_k(z_{i-1})) \\ z_i &= R_56(z_{i-1}) \| y_i \end{aligned}$$

fazendo $i = 1, 2, \ldots$ e onde $\|$ é a operação de concatenação de bits. Para decifrar basta realizar as operações

$$\begin{aligned} x_i &= y_i \oplus L_8(E_k(z_{i-1})) \\ z_i &= R_56(z_{i-1}) \| y_i \end{aligned}$$

com $i = 1, 2, \ldots$. Nas fórmulas acima, L_8 indicam os 8 bits mais significativos de um bloco (aqueles mais à esquerda) de 64 bits e R_56 os 56 bits menos significativos do bloco (aqueles mais à direita). As figuras 2.16 e 2.17 apresentam os esquemas deste modo de funcionamento.

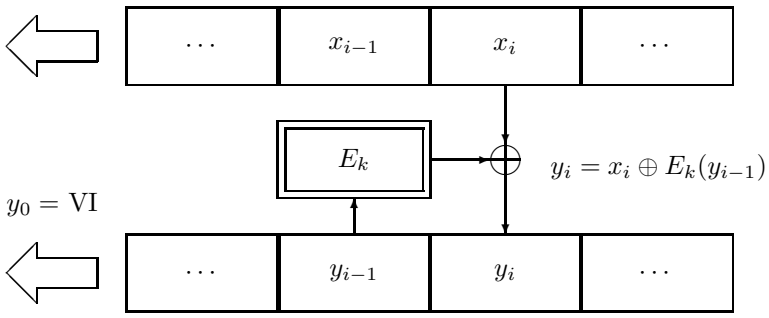

Figura 2.14: Esta figura apresenta o modo CFB usado para cifrar um texto partido em blocos. O bloco cifrado y_i é obtido pela operação $y_i = x_i \oplus E_k(y_{i-1})$

Embora existam outros modos de operação, vamos descrever o modo **Output Feedback**, designado abreviadamente por **OFB**. Este modo de operação está esquematizado nas figuras 2.18 e 2.19. Para cifrar a mensagem **x**, combina-se um vetor de inicialização $z_0 = VI$ e realizam-se as operações que seguem, fazendo $i = 1, 2, \ldots$.

$$z_i = E_k(z_{i-1})$$
$$y_i = x_i \oplus z_i$$

Para decifrar, calcula-se

$$z_i = E_k(z_{i-1})$$
$$x_i = y_i \oplus z_i$$

fazendo $i = 1, 2, \ldots$.

Existem versões do CFB e do OFB com feedback de 1 bit e de 8 bits usados para criptografar dados bit a bit ou byte a byte. O OFB é usado freqüentemente para criptografar transmissões via satélite. No CBC e no CFB, se o valor de um bloco x_i for modificado, todos os blocos cifrados subseqüentes serão afetados.

2.4 Construir um MAC usando CBC

O MAC (das iniciais de "Message Autentication Code") é um resumo da mensagem, obtida com o DES ou uma outra cifra de bloco qualquer utilizando uma

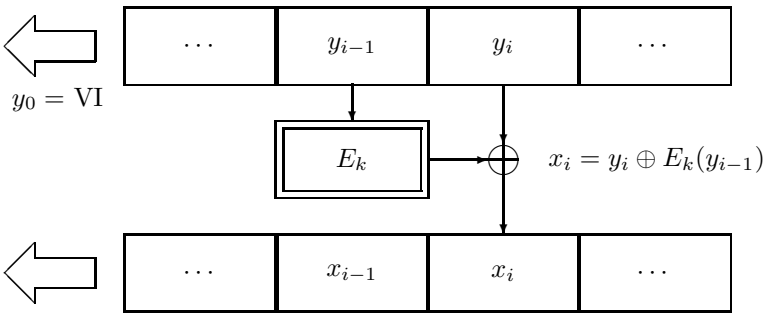

Figura 2.15: Esta figura apresenta o modo CFB usado para decifrar um texto partido em blocos. O bloco decifrado x_i é obtido pela operação $x_i = y_i \oplus E_k(y_{i-1})$

chave secreta k compartilhada entre as pessoas que estabelecerão uma correspondência secreta. O MAC serve para verificar se a mensagem enviada sofreu ou não alterações no meio do caminho. A pessoa que recebe a mensagem junto com o MAC pode, ela própria, recalcular o MAC da mensagem. Se o MAC original e o calculado forem iguais, este é um indício de que a mensagem não foi adulterada no transporte. Certamente que, se um terceiro possuir a chave secreta k poderá interceptar a mensagem, escrever outra, calcular o seu MAC com a chave e reenviar a nova mensagem para o destinatário que, neste caso, não terá como duvidar da autenticidade do MAC. Logo, o não comprometimento da chave é fundamental para a segurança da correspondência.

Os modos CBC e CFB são úteis para autenticação e podem ser usados para produzir um MAC. Com o CBC, por exemplo, se a mensagem for constituída pelos blocos x_1, x_2, ..., x_n, então as operações

$$\begin{aligned} y_0 &= VI = 0 \\ y_i &= E_k(x_i \oplus y_{i-1}) \end{aligned}$$

para $i = 1, 2, \ldots, n$, geram o bloco y_n que pode ser tomado como um código de autenticação da mensagem. Apenas aqueles que possuem a chave k podem reproduzi-lo. Desta forma,

$$y_n = MAC_k(x_1 x_2 \cdots x_n)$$

é um resumo autenticado da mensagem $x_1 x_2 \cdots x_n$. A mensagem com sua autenticação é constituída pela seqüência de blocos

$$x_1 x_2 \cdots x_n y_n$$

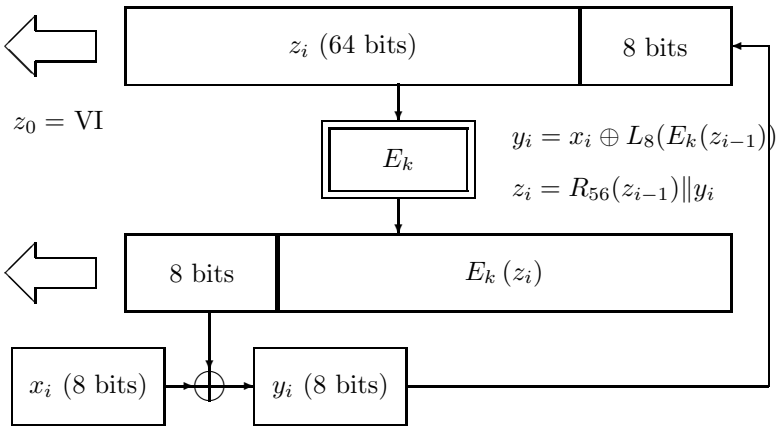

Figura 2.16: Esta figura apresenta o modo CFB, na versão de 8 bits, usado para cifrar um texto usando uma cifra de blocos.

Autenticação e sigilo

Para combinar autenticação com sigilo, calcula-se o MAC com uma chave k_1

$$x_{n+1} = MAC_{k_1}(x_1 x_2 \cdots x_n)$$

e, em seguida, criptografa-se a mensagem original concatenada ao MAC com uma chave k_2

$$y_1 y_2 \cdots y_n y_{n+1} = E_{k_2}(x_1 x_2 \cdots x_n x_{n+1})$$

que é a mensagem autenticada criptografada.

Alternativa

Há uma alternativa. Pode-se inicialmente cifrar a mensagem

$$y_1 y_2 \cdots y_n = E_{k_2}(x_1 x_2 \cdots x_n)$$

para depois autenticá-la

$$y_{n+1} = MAC_{k_1}(y_1 y_2 \cdots y_n)$$

sendo a mensagem cifrada autenticada fornecida por

$$y_1 y_2 \cdots y_n y_{n+1}.$$

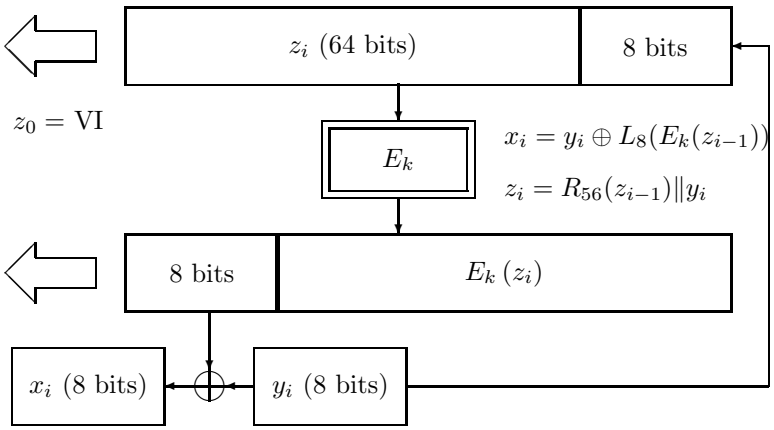

Figura 2.17: Esta figura apresenta o modo CFB, na versão de 8 bits, usado para decifrar um texto usando uma cifra de blocos.

2.5 Padding - completando o último bloco

As cifras de bloco dividem a mensagem em segmentos com um número fixo de bytes. Nem sempre o último segmento da mensagem forma um bloco completo e precisa ser completado, acrescentando-se ao seu final alguns bytes. Há várias formas de se completar o último bloco. Pode-se adicionar zeros até completar o último bloco, que é o mais comum nas implementações atuais. Quando é necessário fazer padding, a FIPS PUB 81 sugere dois modos:

1. Para arquivos binários, complete o último bloco com 0 se o último bit da mensagem for 1 ou complete-o com 1 se o último bit da mensagem for 0.

2. Para arquivos de texto, gravados de acordo com a tabela de conversão ASCII, preencha o último bloco com bits aleatórios e, no último byte, coloque o número de bytes adicionados ao bloco final. Como alternativa, pode-se adicionar bits aleatórios ao último byte, reservando seus últimos 3 ou 4 bits (se o bloco usado pela cifra for de 8 ou 16 bytes) para armazenar o número de bytes acrescentados ao bloco final.

Há uma frase, atribuída a Derek Bok, presidente da Harvard University entre 1971 e 1991, em resposta às críticas recebidas relativas ao alto custo daquela universidade norte americana: “Se você acha que a educação é cara, experimente a ignorância.” Existem outras versões desta frase como “Você acha que a educação é cara porque não experimentou o preço da ignorância”. Há quem afirme ter sido Benjamim Franklin (Norte Americano, 1706 – 1790) um dos pais da independência

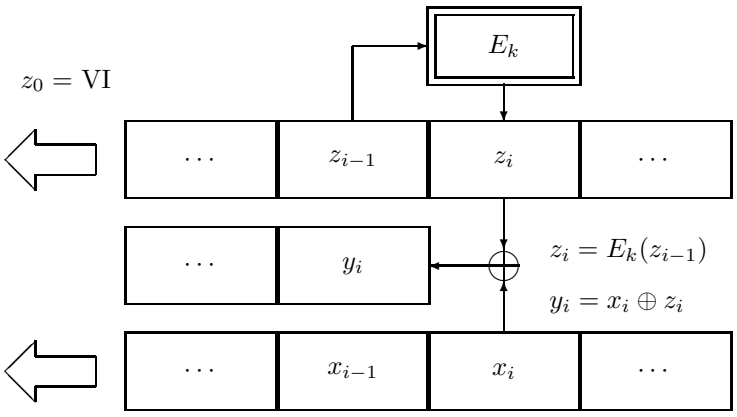

Figura 2.18: Esta figura apresenta o modo OFB usado para cifrar um texto partido em blocos.

norte americana, ou mesmo Thomas Jefferson (Norte Americano, 1743 – 1826), terceiro presidente dos Estados Unidos e um dos autores da declaração de independência quem teria dito "A única coisa mais cara do que a educação é a ignorância". Entretanto, as fontes são menos confiáveis.

Pois bem, a frase "Se você acha que a educação é cara, experimente a ignorância", convertida de acordo com a tabela ASCII escrita na base hexadecimal é: "53 65 20 76 6F 63 EA 20 / 61 63 68 61 20 71 75 65 / 20 61 20 65 64 75 63 61 / E7 E3 6F 20 E9 20 63 61 / 72 61 2C 20 65 78 70 65 / 72 69 6D 65 6E 74 65 20 / 61 20 69 67 6E 6F 72 E2 / 6E 63 69 61 2E 0D 0A". Os espaços foram introduzidos para separar os bytes um a um e as barras foram inseridas para separar os blocos de 64 bits, onde 0D (Carriage Return - Retorno) e 0A (Line Feed - Nova linha), comandos necessários para as impressoras, são colocados automaticamente pelo computador para indicar o fim do texto.

O último bloco contém apenas 7 bytes e, quando o completamos com dois zeros hexadecimais, que correspondem a um byte, teremos um total de 8 blocos de 64 bits: "53 65 20 76 6F 63 EA 20 / 61 63 68 61 20 71 75 65 / 20 61 20 65 64 75 63 61 / E7 E3 6F 20 E9 20 63 61 / 72 61 2C 20 65 78 70 65 / 72 69 6D 65 6E 74 65 20 / 61 20 69 67 6E 6F 72 E2 / 6E 63 69 61 2E 0D 0A 00"

Para efetuar a cifração usando o DES, fui buscar o programa que o simula no site

http:// www. numaboa. com/ downloads/ criptologia/ 6-criptografia

e a consulta foi realizada em 8/10/2010.

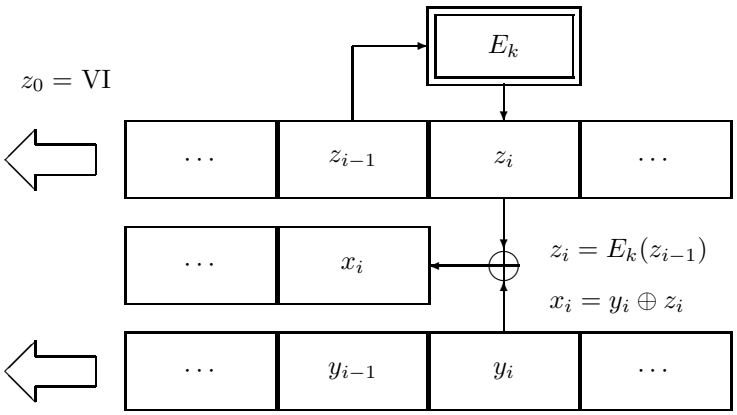

Figura 2.19: Esta figura apresenta o modo OFB usado para cifrar um texto partido em blocos.

Se criptografarmos essa mensagem com a chave hexadecimal "49 4F 45 7B B4 2C F2 E2" de 64 bits, teremos o seguinte texto criptografado "08 A8 A9 98 0C 01 AF 88 / E0 94 B0 02 BB A6 FE 40 / 2F 5B 39 42 42 AF 58 B1 / FE E6 DD 71 DF 79 DA C8 / 7B 6F F6 80 3E 7A 25 B2 / 17 3C DE 5D A6 01 54 FD / DA 50 1A CB 62 EC F8 EF / FF C7 16 07 74 E9 B4 5B /" escrito ainda na forma hexadecimal e onde as barras foram introduzidas apenas para separar os blocos de 64 bits.

2.6 Exercícios

1. No estudo das técnicas criptográficas é importante saber como se escreve números inteiros nas diversas bases, principalmente nas bases 2, 10 e 16. Portanto,

   (a) Passe para as bases 2 e 16 os números decimais (base 10): 1237, 5826, 603, 9571.

   (b) Passe para as bases 2 e 10 os números hexadecimais (base 16): $B7$, $A29$, 23, $7D$, $4F5$.

   (c) Passe para as bases 10 e 16 os números binários (base 2): 101100101, 100010, 10101010, 111001110011100.

   Os exercícios de número 2 a 9 se referem ao DES.

2. Faça um programa que recebe como entrada um bloco de 8 bytes e fornece como saída $PI(x)$. Forneça como entrada o número hexadecimal $1F$ $A3$ 59 BC $5E$ 94 $6D$ 74 e escreva a saída na base hexadecimal.

3. O argumento R que entra na função f possui 32 bits e, inicialmente sofre uma expansão E. Escreva um algoritmo numa linguagem computacional que você conhece que na entrada recebe R na base hexadecimal e devolve na saída $E(R)$ na base hexadecimal.

4. Cada caixa S_i é uma matriz (array bidimensional) de substituição, cuja regra foi descrita no texto. Cada S_i recebe uma entrada de 6 bits e devolve uma saída de 4 bits. Entretanto, ao implementar as caixas S numa linguagem computacional, é mais eficiente gravar a regra de substituição num vetor (array unidimensional). Use uma linguagem de seu conhecimento para gravar a caixa S_1 numa matriz X_i de modo que $X_i(n) = S_i(r, c)$ onde r e c indicam a linha e a coluna fornecida por n. Na matriz S_i, a primeira linha corresponde a $r = 0$ e a primeira coluna a $c = 0$. No vetor X_i, a primeira entrada corresponde a $n = 0$. Lembre-se que para obter r, tome dois bits de n, o mais e o menos significativo e para obter c tome os demais bits de n. Por exemplo, trabalhando com a base 2, se $n = 110010$, então $r = 10$ e $c = 1001$. Traduzindo para a base 10, se $n = 50$, então $r = 2$ e $c = 9$.

5. De cada caixa S saem 32 bits que passam por uma permutação P. Programe em uma linguagem de sua escolha esta permutação P que recebe n e fornece $P(n)$.

6. Escreva um programa que recebe uma entrada de 64 bits e devolve na saída dois segmentos de 32 bits formados, respectivamente, pela parte esquerda e direita da entrada.

7. Escreva um programa que recebe uma entrada de 64 e fornece na saída um bloco de 64 bits onde os 32 bits da esquerda são iguais aos 32 bits da direita da entrada e os 32 bits da direita na saída são os 32 bits da esquerda da entrada.

8. Programe a função $f(K, R)$.

9. Programe a arquitetura de Feistel

$$L_i = R_{i-1}$$
$$R_i = L_{i-1} \oplus f(K_i, R_{i-1})$$

10. Programe LS_r do escalonamento da chave trabalhando com os 56 bits da entrada, obtendo C_r e D_r simultâneamente.

11. Programe a permutação PC_1 e permutação com redução PC_2, ambas usadas no escalonamento da chave.

    Os exercícios que seguem são relativos a corpos finitos.

12. Considere o conjunto dos polinômios $\mathbf{Z}_2[X]$ com as operações usuais de adição e multiplicação de polinômios com coeficientes em $\mathbf{Z}_2$. Mostre que:

    (a) $(1+X)^2 = 1+X^2$.

    (b) $(1+X)^4 = 1+X^4$.

13. Considere o conjunto dos polinômios $\mathbf{Z}_3[X]$ com as operações usuais de adição e multiplicação de polinômios com coeficientes em $\mathbf{Z}_3$. Mostre que:

    (a) $(1+X)^3 = 1+X^3$.

    (b) $(1+X)^9 = 1+X^9$.

14. Considere o corpo finito $GF(2^3) = \mathbf{Z}_2[X]/(1+X^2+X^3)$ cujos elementos são polinômios de grau menor ou igual a 2 com coeficientes em $\mathbf{Z}_2$. Para adicionar dois polinômios neste corpo, nós os adicionamos normalmente em $\mathbf{Z}_2[X]$. Para multiplicar dois polinômios neste corpo, nós os multiplicamos normalmente em $\mathbf{Z}_2[X]$, dividimos o resultado por $1+X^2+X^3$ e o resto desta divisão é o produto dos dois polinômios em $GF(2^3)$. Calcule em $GF(2^3)$:

    (a) $\left(X^2+X+1\right)X^n$, para $n = 1, 2, 3$.

    (b) $\left(X^2+X\right)^{-1}$ usando o algoritmo de Euclides estendido.

15. Considere o corpo finito $GF(3^4) = \mathbf{Z}_3[X]/(2+X+X^4)$ cujos elementos são polinômios de grau menor ou igual a 3 com coeficientes em $\mathbf{Z}_3$. Para adicionar dois polinômios neste corpo, nós os adicionamos normalmente em $\mathbf{Z}_3[X]$. Para multiplicar dois polinômios neste corpo, nós os multiplicamos normalmente em $\mathbf{Z}_3[X]$, dividimos o resultado por $2+X+X^4$ e o resto desta divisão é o produto dos dois polinômios em $GF(3^4)$. Calcule em $GF(3^4)$:

    (a) $\left(X^3+X+1\right)X^n$, para $n = 1, 2, 3$.

    (b) $\left(X^2+X+2\right)^{-1}$ usando o algoritmo de Euclides estendido.

    Agora vamos programar o AES

16. Escreva um programa que recebe um número de 16 bytes $p_0p_1\cdots p_{15}$ e os dispõem numa matriz $A = [a_{i,j}]$, onde $a_{i,j} = p_{i+4j}$.

17. Escreva um programa que recebe na entrada uma matriz $A = [a_{i,j}]$ de tamanho 4×4 e devolve na saída um bloco de 16 bytes $p_0\ p_1 \cdots p_{15}$, onde

$$p_k = a_{k \bmod 4,\, k/4}$$

sendo $k/4$ o quociente da divisão inteira de k por 4.

18. Programe a transformação AddRoundKey (ARK).

19. Programe a função g usada na transformação SubBytes (SB) e calcule a tabela de substituição que pode ser usada para implementá-la.

20. Programe a função f usada na transformação SubBytes (SB) e calcule a tabela de substituição que pode ser usada para implementá-la.

21. Programe a função $f \circ g$ usada na transformação SubBytes (SB) e calcule a tabela de substituição que pode ser usada para implementá-la.

22. Programe a transformação ShiftRow (SR).

23. Programe a transformação MixColumns (MC).

24. Programe a matriz W de tamanho 4×44 tal como é descrita na subseção "Escalonamento da chave para as diversas rodadas".

25. Junte as partes já programadas para implementar o AES.

26. Procure na internet o AES já programado e identifique as funções ARK, SB, SR e MC. Compare com suas implementações.

Agora algumas questões sobre o modo de operação das cifras de bloco.

27. Programe o modo ECB para o DES ou o AES.

28. Programe o modo CBC para o DES ou o AES.

29. Programe o modo CFB para o DES ou o AES.

30. Programe o modo OFB para o DES ou o AES.

31. Use um dos modos acima para cifrar um arquivo de texto que você possui.

32. Construa um MAC a partir do modo CFB.

Capítulo 3

Criptografia de chave pública

3.1 Diffie-Hellman

Em 1976, Bailey Whitfield Diffie e Martin Edward Hellman publicaram um artigo denominado New Directions in Cryptography no volume 22 da revista IEEE Transactions on Information Theory. Neste artigo descreveram o primeiro método para trocar uma chave secreta entre dois agentes usando um canal público.

Existem ataques a este processo que vamos supor tenham sido contornados. O principal deles consiste em um espião que consegue impedir a comunicação direta entre os agentes autorizados e que possui tecnologia para se fazer passar pelos agentes legítmos, comunicando-se com eles e trocando chaves e mensagens falsas, enquanto um agente pensa que está se comunicando com o outro. Vamos ao princípio da troca de chaves idealizada por Diffie-Hellman.

Se p for um número primo, existem números g do grupo multiplicativo $\mathbf{Z}_p^*$ dos inteiros módulo p, para os quais o subgrupo

$$[g] = \{g^i \bmod p : i = \ 1, \ \ldots, \ p-1\},$$

formado pelas potências de g é igual a $\mathbf{Z}_p^*$. Estes números são denominados **geradores** de $\mathbf{Z}_p^*$. Sendo n um número inteiro positivo e $\phi(n)$ a quantidade números primos de n menores ou iguais a ele, provaremos adiante que existem $\phi(p-1)$ geradores de $\mathbf{Z}_p^*$ e ofereceremos um algoritmo para determinar um deles.

Vamos supor que Alice e Beto pretendem trocar uma chave secreta usando um canal público. Inicialmente ambos combinam usar um número primo p e um gerador g do grupo multiplicativo $\mathbf{Z}_p^* = \{1, \ 2, \ \ldots, \ p-1\}$. Estes números não precisam ser mantidos em segredo e podem ser de conhecimento público.

Agora Alice efetua o seguinte procedimento:

1. Escolhe aleatoriamente um número inteiro x_A no intervalo $1 < x_A < p-1$ e o mantém em segredo.
2. Calcula
$$y_A = g^{x_A} \bmod p$$
3. Envia y_A para Beto usando um canal público.

Beto realiza o mesmo procedimento:

1. Escolhe aleatoriamente um número inteiro x_B no intervalo $1 < x_B < p-1$ e o mantém em segredo.
2. Calcula
$$y_B = g^{x_B} \bmod p$$
3. Envia y_B para Alice usando um canal público.

Tendo Alice recebido y_B, ela calcula

$$(y_B)^{x_A} \bmod p = (g^{x_B})^{x_A} \bmod p = g^{x_A x_B} \bmod p = k$$

Tendo Beto recebido y_A, ele calcula

$$(y_A)^{x_B} \bmod p = (g^{x_A})^{x_B} \bmod p = g^{x_A x_B} \bmod p = k$$

e ambos passam a compartilhar da mesma chave.

Como a correspondência entre ambos foi efetuada por um canal público, vamos supor que existe um espião passivo, capaz de observar e anotar os dados transmitidos. Seria ele capaz de reproduzir a chave k tendo capturado p, g, y_A e y_B? Com eles é possível calcular o produto $y_A y_B \bmod p = \alpha^{x_A + x_B} \bmod p$, que não lhe fornece a chave. Conhecendo p, g, e y_A, ele poderia tentar calcular o inteiro x_A entre 1 e $p-1$, para o qual $y_A = g^{x_A} \bmod p$. O número x_A é chamado de **logaritmo discreto** de y_A **módulo** p **na base** g. Conhecendo x_A e y_B, o espião poderia efetuar as contas de Alice $y_B^{x_B} \bmod p = k$ e obter a chave.

Se p e g forem adequadamente escolhidos, o cálculo do logaritmo discreto x_A é um problema computacional de difícil solução. Sobre o tamanho e a natureza do p, recomendamos que aqueles que pretendem usar criptografia seriamente se inspirem na literatura especializada, notadamente às normas baixadas pelo NIST - National Institute of Standards and Technology, disponibilizadas no site www.nist.gov. Estas normas estão distribuídas entre FIPS - Federal Information Processing Standard, RFC - Request for Comments e ANS - American

National Standards. Recomenda-se que o p deve ser um número com pelo menos 1024 bits sendo que os expoentes x_A e x_B devem possuir mais de 300 bits. Uma discussão sobre a segurança do problema do logaritmo discreto e a obtenção de geradores de $\mathbf{Z}_p$, consulte, por exemplo, Menezes, Oorschot e Vanstone [14] ou Schneier [17].

O método de Diffie-Hellman resolve o problema de efetuar a troca de chave entre as partes para serem usadas na Criptografia Simétrica. Hoje, a utilização da técnica de Diffie-Hellman vai muito além daquilo que se propunha logo depois de sua invenção.

3.2 Algoritmos utilizados por Diffie-Hellman

Os algoritmos utilizados no procedimento desenvolvido por Diffie e Hellman para a troca de chave secreta em canal público são comuns às demais técnicas de Criptografia de Chave Pública.

3.2.1 Potência modular

Uma questão que se coloca de início na utilização do Diffie-Hellman é a de se calcular eficientemente a potência modular. Quando n e x forem número muito grandes, a potência

$$y = a^x \bmod n$$

pode ser calculada de modo eficiente usando as propriedades da operação modular. Duas delas são

$$(a + b) \bmod n = (a \bmod n + b \bmod n) \bmod n$$
$$(a \times b) \bmod n = (a \bmod n \times b \bmod n) \bmod n$$

De acordo com elas, ao adicionar ou multiplicar números módulo n, podemos reduzir os operandos módulo n de modo a realizar operações com números menores do que n. A adição de duas parcelas módulo n resulta num número menor do que $2n$ e o produto de dois fatores módulo n resulta num número menor do que n^2. Se n possui k bits, $2n$ possui $k + 1$ bits e n^2 possui $2k$ bits.

Com relação à potência, temos a propriedade

$$a^{x+y} \bmod n = (a^x \bmod n \times a^y \bmod n) \bmod n$$

onde tanto $a^x \bmod n$ quanto $a^y \bmod n$ são menores do que n. O produto destes dois fatores é menor do que n^2 e, no final, é reduzido módulo n. Tal propriedade garante que, na potência modular, os produtos intermediários são menores do que n^2. Se n possui k bytes, n^2 possui $2k$ bytes.

Ainda nos resta discorrer sobre o modo eficiente de calcular a exponenciação modular $y = a^x \bmod n$. Iniciemos, por exemplo, com o cálculo de $y = a^{15} \bmod n$. Como 15 na base 2 é 1111, podemos escrever $15_{10} = 1111_2$ onde o subíndice indica a base usada. Sendo $1111_2 = 1 + 2 + 4 + 8$, obtemos $a^{1111_2} = a^1 a^2 a^4 a^8$ e o produto pode ser calculado de forma eficiente se lembrarmos que a^8 é o quadrado de a^4, que é o quadrado de a^2, que é o quadrado de a. Deste modo,

$$\begin{aligned} a^2 \bmod n &= (a \bmod n) \times (a \bmod n) \bmod n \\ a^4 \bmod n &= (a^2 \bmod n) \times (a^2 \bmod n) \bmod n \\ a^8 \bmod n &= (a^4 \bmod n) \times (a^4 \bmod n) \bmod n \end{aligned}$$

e

$$\begin{aligned} y = a^{15_{10}} \bmod n &= a^{1111_2} \bmod n \\ &= (a^1 \bmod n)(a^2 \bmod n)(a^4 \bmod n)(a^8 \bmod n) \bmod n, \end{aligned}$$

Trabalhando com uma linguagem computacional como o C, temos o seguinte esquema:

$y = 1$; $A = a \bmod n$;
$y = (y * A) \bmod n$; // agora y é igual a $a \bmod n$
$A = (A * A) \bmod n$; // agora A é igual a $a^2 \bmod n$
$y = (y * A) \bmod n$; // agora y é igual a $a^1 a^2 \bmod n$
$A = (A * A) \bmod n$; // agora A é igual a $a^4 \bmod n$
$y = (y * A) \bmod n$; // agora y é igual a $a^1 a^2 a^4 \bmod n$
$A = (A * A) \bmod n$; // agora A é igual a $a^8 \bmod n$
$y = (y * A) \bmod n$; // agora y é igual a $a^1 a^2 a^4 a^8 \bmod n$ que é a potência desejada
Fim.

Para evitar a repetição de várias linhas iguais num programa (imagine o tamanho do programa se o expoente x tivesse quinhentos ou mil dígitos binários) podemos usar um loop while que ainda possui a virtude de não restringir o tamanho do expoente. Neste loop o expoente x vai perdendo um a um seus bits menos significativos, efetuando deslocamentos para a direita ($x >> 1$)

```
y = 1; A = a;
while( x != 0 )
{
  y = y * A % n;
  A = A * A % n;
  x = x >> 1; // elimina o bit menos significativo de x
```

```
    // o loop while se encerra quando o expoente x
    // perder todos os seus bits e ficar igual a zero
  }
```

Resta examinar o que ocorre quando o expoente x tem um bit nulo como em 1011_2, uma vez que $a^{1011_2} = a^1 a^2 a^8$. Neste caso se observa que o fator correspondente ao bit nulo, o a^4, não entra na multiplicação. Isto pode ser realizado usando um if que não executa a multiplicação $y = y*A \bmod n$ quando o bit correspondente for nulo.

```
y = 1; A = a;
while( x != 0 )
{
  if( x & 1 )
  // verifica se o bit menos significativo eh = a 1
  {
    y = y * A % n;
  }
  A = A * A % n;
  x = x >> 1;
}
```

Se não for preciso manter o valor de a, podemos eliminar a variável A, jogando fora a atribuição $A = a$ e usando a no lugar de A em todo o programa.

O programa completo em linguagem C++ para calcular a potência modular $y = a^x \bmod n$ pode assim ser escrito

```
 #include <conio.h>
 #include <stdio.h>
 #include<iostream>

 using std::cout;
 using std::cin;

main()
{
  int a, x, n;
  int y;
  printf("Potencia modular\n\n\n");
  printf("Calculo de y = a^x mod n \n\n\n");
  printf("Digite o valor da base     a: ");
  cin >> a;
  printf("Digite o valor do expoente x: ");
  cin >> b;
  printf("Digite o valor do modulo   n: ");
  cin >> n;
  printf("\n\n");
  y = 1;

  while( x )
  {
    if( x & 1 )
    {
      y = y * a % n;
    }
    a = a * a % n;
    x = x >> 1;
  }

  printf("O valor de y eh: ");
  cout << y;
 printf("\n\n");
}
```

Infelizmente, sem uma biblioteca de precisão múltipla, o programa acima não pode trabalhar com números do tamanho exigido pela Criptografia. O GNU Multi

Precision (GMP) é uma biblioteca de precisão múltipla, cuja distribuição é gratuita, que pode ser obtida em

http://gmplib.org

e incorporada ao seu compilador C ou C++. Com esta biblioteca, é possível trabalhar com números inteiros muito grandes, do tamanho exigido pela Criptografia moderna. O MinGW (Minimalist GNU for Windows) contém uma coleção de compiladores (GNU Compiler Collection - GCC), que inclui os compiladores C, C++, ADA e Fortran e pode ser obtido gratuitamente em

http://www.mingw.org

Se possuirmos o Mathematica da Wolfram Research, podemos usá-lo para fazer as contas para nós, com a vantagem de que ele comporta números inteiros grandes, facilidade que já vem embutida no software. Vamos supor que queiramos calcular $y = a^x \bmod n$, com $a = 58277462012345$, $x = 638456560684$ e $n = 524592348741345458$. No Mathematica basta digitar

```
a = 58277462012345;
x = 638456560684;
n = 524592348741345458;
y = PowerMod[a, x, n]
```

pressionando o Enter ao final de cada linha. Depois, mantenha a tecla Shift pressionada enquanto aperta e solta o Enter, soltando finalmente o Shift. Irá obter instantaneamente a resposta

342 339 415 255 370 017.

Desejando, pode-se programar no Mathematica tal como se faz em C ou C++

```
a = 58277462012345;
x = 638456560684;
n = 524592348741345458;
y = 1;
While[x != 0,
 If[ BitAnd[x, 1] > 0, y = Mod[y * a, n] ];
 a = Mod[ a * a , n];
 x = Floor[ x/2 ];
 ]
Print["y = ", y]
```

para obter a resposta

y = 342 339 415 255 370 017

3.2.2 A busca por números primos

A segunda questão que se coloca é a determinação de um número primo p grande o suficiente para resistir aos ataques ao problema do logaritmo discreto. A seleção do número primo é realizada por um processo de busca aleatória. São necessários dois algoritmos. Um para gerar aleatóriamente um número inteiro e outro para testar se p é primo.

O sucesso da busca aleatória por um número primo reside na sua abundância no conjunto dos números inteiros positivos. Dentre os números de 512 bits, aproximadamente 1 a cada 200 é primo. Esta densidade pode ser estimada usando o Teorema dos Números Primos onde se prova que

$$\pi(n) \sim \frac{n}{\ln n} \qquad (n \to \infty)$$

sendo $\pi(n)$ a quantidade de números inteiros x, no intervalo $1 \leq x \leq n$, com x e n primos entre si, e onde $\ln x$ é o logaritmo natural de x. O til indica que o lado esquerdo é assintótico ao lado direito quando n tende para o infinito. Duas funções na variável n são assintóticas no infinito se a razão entre elas tende a 1 quando o argumento tende ao infinito. Os interessados poderão encontrar um apanhado histórico sobre este famoso teorema no artigo de Paul Trevier Bateman e Harold G. Diamond denominado "A hundred years of prime numbers", publicado no The American Mathematical Monthly, Vol. 103, No. 9, 1996, p 729 - 741. Uma versão simplificada da prova de Neuman foi realizada por D. Zagier, num artigo que ele denominou Neuman's Short Proof of the Prime Number Theorem publicado no The American Mathematical Monthly, Vol. 104, No. 8, 1997, pp. 705 - 708.

Pois bem, este teorema nos traz informações que impressionam. Pense na seguinte questão: quantos números primos existem com 100 dígitos decimais? Vamos responder a esta pergunta fazendo uma estimativa usando o teorema dos números primos: Números com 100 dígitos decimais são aqueles maiores ou iguais a 10^{99} e menores do que 10^{100}, de modo que o número aproximado de primos neste intervalo é dado por

$$\pi(10^{100}) - \pi(10^{99}) = \frac{10^{100}}{\ln 10^{100}} - \frac{10^{99}}{\ln 10^{99}} \approx 4 \times 10^{97}$$

que é uma quantidade gigantesca. Vamos compará-la com o número de prótons e neutrons no universo tal como estimado pela ciência atual, com um raio r de 15 bilhões de anos luz e três núcleons (prótons ou neutrons) por metro cúbico. Um ano possui $60\times\ 60\times\ 24\times\ 365 = 31\,536\,000$ segundos e 15 bilhões de anos possuem $15\times\ 10^9\times\ 31\,536\,000 = 47304 \times 10^{13}$ segundos. Em cada segundo, a luz percorre 3×10^8 metros. O volume de uma esfera com um raio de 15 bilhões de anos luz é

$$\frac{4}{3}\pi r^3 = \frac{4}{3}\pi(3 \times 10^8 \times 47304 \times 10^{13})^3 \approx 10^{79}$$

metros cúbicos. Havendo em média 3 núcleons por metro cúbico estimamos a existência de 3×10^{79} núcleons no universo, de acordo com os conhecimentos atuais. Desta forma, se fôssemos gravar cada número primo com 100 algarismos em um núcleon, precisaríamos de 10^{18} (um quintilhão) universos iguais ao nosso. Fantástico, não acham?

Para estimar a densidade de números primos no intervalo de inteiros que vai de 10^{99} a 10^{100}, basta efetuar a conta

$$\left(\frac{10^{100}}{\ln 10^{100}} - \frac{10^{99}}{\ln 10^{99}}\right) \div \left(10^{100} - 10^{99}\right) \approx \frac{4}{1000} = \frac{1}{250}$$

Retirando os números pares, isto significa que neste intervalo, 1 a cada 125 números ímpares, um é primo. Em média, precisamos gerar uns 60 números ímpares aleatoriamente até obter um que é primo. Esta busca é bem rápida num computador pessoal.

Para obter um número primo grande, é preciso gerar um número inteiro ímpar aleatoriamente e testar sua primalidade. O algoritmo mais utilizado é de Miller-Rabin, devido a Gary Miller e Michael Rabin. Este algoritmo recebe como entrada um número inteiro e responde se ele é primo ou composto. Quando ele diz que o número é composto, ele está dizendo a verdade. Quando ele responde que o número é primo, ele poderá estar dizendo a verdade ou não. A probabilidade de o algoritmo receber como entrada um número composto e afirmar que ele é primo é de 1/4. Por este motivo, quando o Miller-Rabin responde que o número é primo, ele deve ser chamado novamente para confirmar sua resposta. O algoritmo possui um parâmetro interno que é escolhido aleatoriamente a cada chamada. Se este algoritmo responder sempre que o número é primo, mesmo depois de ser chamado umas 20 vezes, a probabilidade de estar fornecendo uma resposta errada é de $(1/4)^{20} \approx 10^{-12}$ que é uma probabilidade bem pequena. Vamos lembrar que a probabilidade de ganhar na mega sena é de, aproximadamente, 10^{-7}. Se o Miller-Rabin afirmar que o número é primo mesmo depois de chamado diversas vezes com bases diferentes, pode-se confiar com uma grande margem de segurança que o número é primo. Este teste se fundamenta na aplicação inteligente de alguns teoremas simples provenientes da Teoria dos Números que serão descritos na sequência.

Existe um teste determinístico para determinar se um número é primo ou composto e que executa em tempo polinomial. Ele foi publicado por Manindra Agrawal, Neeraj Kayal, Nitin Saxena, sob o título "PRIMES is in P", na revista científica Annals of Mathematics 160 (2004), no. 2, pp. 781–793. O preprint deste artigo foi apresentado em agosto de 2002. Entretanto, mesmo hoje, ainda se usa largamente o Miller-Rabin por ser mais rápido.

3.2.3 Geradores de grupos cíclicos

Depois de obtido um número primo, a distribuição de chaves desenvolvida por Diffie e Hellman, cujas pesquisas se apoiaram nos trabalhos de Merkle, exige a determinação de um gerador g do grupo multiplicativo $\mathbf{Z}_p^* = \{1, 2, \ldots, p\}$ onde a multiplicação é aquela módulo p. Vamos definir o conceito de gerador de um grupo multiplicativo e apresentar uma algoritmo para determiná-lo.

Seja g um elemento de um grupo multiplicativo finito G com $n \geq 2$ elementos. O conjunto das potências positivas de g

$$[g] = \{g^i : i = 1, 2, \ldots\}$$

é um subgrupo multiplicativo de G denominado **subgrupo gerado** por g. Sendo G finito, $[g]$ também o será. Logo, devem existir i e j tais que $g^i = g^j$ e, considerando $i > j$, temos $g^{i-j} = 1$. O menor expoente $k \geq 1$ para o qual $g^k = 1$ é chamado de **ordem** de g. Neste caso, $g^i = g^j$ se e só se $i \equiv j \pmod{k}$ e

$$[g] = \{g^i : i = 1, 2, \ldots, k\}$$

possui exatamente k elementos.

Quando existir g tal que $G = [g]$, este g é chamado de **elemento primitivo** de G ou **gerador** de G. Grupos que possuem geradores são denominados grupos **cíclicos**.

Teorema 3.1. *Seja G um grupo cíclico de ordem n. O número de elementos primitivos de G é $\phi(n)$. Além disso, a ordem de todo elemento de G divide n.*

Demonstração: Se g for um gerador de G, para todo α em G, existe um inteiro r entre 1 e n para o qual $\alpha = g^r$. Sendo k a ordem de α, então k é o menor inteiro para o qual $\alpha^k = g^{rk} = 1$. O expoente rk deve ser divisível por n e, evidentemente, é divisível por r. O menor inteiro k para o qual isto ocorre é aquele que satisfaz à igualdade $kr = mmc(n, r)$. Como

$$mmc(n, r)\ \mathrm{mdc}(n, r) = nr,$$

segue

$$k = \frac{n}{\mathrm{mdc}(n, r)}$$

mostrando que a ordem k de α divide n e que α é um elemento primitivo de G se e só se $\mathrm{mdc}(n, r) = 1$. Como existem $\phi(n)$ inteiros r para os quais isto ocorre, concluimos pela existência de $\phi(n)$ elementos primitivos de G. ■

Se $n \geq 2$, o grupo multiplicativo módulo n

$$\mathbf{Z}_n^* = \{x \in \mathbf{Z}_n : \mathrm{mdc}(x, n) = 1\},$$

é cíclico quando $n = 2$, 4, p^k ou $2p^k$ onde p é um número primo ímpar e k é um inteiro positivo. Para estes valores de n, possuindo $\phi(n)$ elementos, o número de geradores de $\mathbf{Z}_n^*$ é $\phi(\phi(n))$.

Um grupo cíclico possui geradores em abundância. Se ele tiver n elementos possuirá $\phi(n)$ geradores. Este é o motivo pelo qual podemos descobrir um gerador escolhendo de forma aleatória um elemento de G, testando se ele é ou não um gerador. Depois de algumas alternativas, acabamos encontrando um.

O teorema anterior mostra que a ordem de todo elemento de G divide a ordem do grupo. Seja n a ordem de G e

$$n = p_1^{e_1} p_2^{e_2} \cdots p_k^{e_k}$$

sua decomposição em fatores primos. Se α não for um gerador de G, sua ordem será um divisor de n e menor do que n. Portanto a ordem de α será um divisor de

$$n_1 = \frac{n}{p_1} \text{ ou } n_2 = \frac{n}{p_2} \text{ ou } \cdots \text{ ou } n_k = \frac{n}{p_k}$$

e $\alpha^{n_i} = 1$ para $i = 1, 2, \ldots, k$. Desta forma, se pelo menos um dos elementos

$$\alpha^{n_1} \text{ ou } \alpha^{n_2} \text{ ou } \cdots \text{ ou } \alpha^{n_k}$$

for igual a 1, α não é um gerador, o que sugere o seguinte algoritmo probabilístico para calcular um elemento primitivo de um grupo multiplicativo G que está descrito na obra de Alfred Menezes, Paul van Oorschot and Scott Vanstone [14].

Algoritmo para calcular um gerador de um grupo cíclico.

Entrada: um grupo cíclico G de ordem n e a sua decomposição em fatores primos $n = p_1^{e_1} p_2^{e_2} \ldots p_k^{e_k}$.

Saída: um gerador α de G.

1. Escolha aleatoriamente um elemento g em G.
2. Para i variando de 1 a k faça o seguinte:
   (a) Se $g^{n/p_i} = 1$, volte ao passo 1.
3. Retorne o valor de g.

Uma das dificuldades de se aplicar este algoritmo reside na etapa que exige a fatoração da ordem n do grupo, que pode ser uma tarefa difícil. No caso do grupo multiplicativo $\mathbf{Z}_p$, com $n = p - 1$ elementos, onde p é primo, sugere-se determinar

inicialmente um número primo q e, em seguida, encontrar um número pequeno R de modo que $p = 2Rq+ 1$ seja o número procurado. Neste caso, para fatorar $n = p - 1$, basta fatorar R que é pequeno. O melhor caso é aquele em que $R = 1$, quando $p - 1 = 2q$.

3.2.4 Congruência

Definição 3.1. *Sejam a, b e $m > 1$ números inteiros. Diremos que a é* ***côngruo*** *a b módulo m quando $a - b$ for divisível por m e escrevemos*

$$a \equiv b \pmod{m}.$$

A congruência $a \equiv b \pmod{m}$ é verdadeira se e só se

$$a \bmod m = b \bmod m$$

ou seja, o resto da divisão inteira de a por m é igual ao resto da divisão inteira de b por m.

Exemplo 3.1. *O leitor poderá verificar que*

$$\begin{aligned} 32 &\equiv 7 \pmod{5} \\ -12 &\equiv 37 \pmod{7} \\ 17 &\equiv 17 \pmod{13} \end{aligned}$$

Teorema 3.2. *Sejam a, b, c e m números inteiros, com $m > 1$. Valem as propriedades:*

1. $a \equiv a \pmod{m}$
2. $a \equiv b \pmod{m}$ *implica em* $b \equiv a \pmod{m}$.
3. *Se* $a \equiv b \pmod{m}$ *e* $b \equiv c \pmod{m}$, *então* $a \equiv c \pmod{m}$.
4. *Se* $a \equiv b \pmod{m}$, *então*
   (a) $a + c \equiv b + c \pmod{m}$
   (b) $a \times c \equiv b \times c \pmod{m}$
   (c) $a - c \equiv b - c \pmod{m}$
   (d) $a \equiv b + c\,m \pmod{m}$
5. *Se* $a + c \equiv b \pmod{m}$, *então* $c \equiv b - a \pmod{m}$.
6. *Quando a for primo de m e* $ac \equiv b \pmod{m}$, *então* $c \equiv ba^{-1} \pmod{m}$.

7. *(Lei do cancelamento) Quando a for primo de m e* $ac \equiv ab \pmod{m}$, *então* $c \equiv b \pmod{m}$.

Exemplo 3.2. *Sendo* $x + 7 \equiv 3 \pmod{17}$, *então* $x \equiv 3 - 7 \pmod{17}$ *ou* $x \equiv -4 \pmod{17} \equiv 13 \pmod{17}$.

Dados os números inteiros a, b e $m > 1$, consideremos a equação de congruência na incógnita x :

$$a\,x \equiv b \pmod{m}.$$

Quando $\text{mdc}(a, m) = 1$, uma solução que pertence ao intervalo $0 \leq x_0 < m$ é

$$x_0 = a^{-1}b \bmod m,$$

onde a^{-1} é o inverso de a módulo m. A solução não é única pois $x_0 + qm$ também será solução, para qualquer número inteiro q.

Quando $\text{mdc}(a, m) = d > 1$, nem sempre existe x tal que

$$ax \equiv b \pmod{m}.$$

Esta congruência terá uma solução se e só se d dividir b. Neste caso,

$$(a/d)x \equiv (b/d) \pmod{m/d}$$

Como a/d e m/d são primos entre si, $x_0 = (a/d)^{-1} \bmod (m/d)$ é uma solução desta equação e

$$x_0,\ x_0 + \frac{m}{d},\ x_0 + 2\frac{m}{d},\ \ldots, x_0 + (d-1)\frac{m}{d}$$

são todas as soluções em $\mathbf{Z}_m$ da equação equação original $ax \equiv b \pmod{m}$.

Exemplo 3.3. *A equação* $4x \equiv 6 \pmod{26}$ *possui solução pois* $\text{mdc}(4, 26) = 2$ *também divide* 6.

Resolvendo a equação $2x \equiv 3 \pmod{13}$, *obtemos* $x \equiv 8 \pmod{13}$. *Logo,* 8 *e* 21 *são soluções da equação de partida.*

Exemplo 3.4. *Vamos resolver a equação* $2x+7 \equiv 3 \pmod{17}$, *usando os teoremas enunciados. Subtraindo* 7 *dos dois membros da equação e multiplicando os dois lados por* $2^{-1}(\text{mod}17) = 9$, *obtemos* $x \equiv (-4) \times 9 \pmod{17}$. *A solução em* $\mathbf{Z}_{17}$ *é* $x = 15$.

Exercício 3.1. *Resolva* $5x + 6 \equiv 13 \pmod{11}$.

Exercício 3.2. *Resolva* $11111x \equiv 4 \pmod{1234}$.

Exemplo 3.5. *A equação* $12x \equiv 21 \pmod{39}$ *possui solução pois o o máximo divisor comum de* 12 *e* 29 *é igual a* 3 *e divide* 21. *Resolvendo a equação* $4x \equiv 7 \pmod{13}$ *obtemos* $x \equiv 5 \pmod{13}$. *As soluções da equação original são* 5, 18 *e* 31.

3.2.5 O teorema chinês do resto

Se $x \equiv 16 \pmod{35}$ então

$$\begin{aligned} x &\equiv 16 \pmod 5 \\ x &\equiv 16 \pmod 7 \end{aligned}$$

pois 5 e 7 são os divisores primos de 35. Reduzindo 16 módulo 5 e módulo 7 nas congruências acima, obtemos

$$\begin{aligned} x &\equiv 1 \pmod 6 \\ x &\equiv 2 \pmod 7 \end{aligned}$$

O teorema chinês do resto diz que este processo pode ser invertido. O seu enunciado é o seguinte:

Teorema 3.3. *(Teorema Chinês do Resto) Sejam m_1, m_2, ..., m_k, números inteiros dois a dois primos entre si e $m = m_1 m_2 \cdots m_k$. Dados os números inteiros a_1, a_2, ..., a_k, o sistema de congruências na incógnita x*

$$\begin{aligned} x &\equiv a_1 \pmod{m_1}, \\ x &\equiv a_2 \pmod{m_2}, \\ &\cdots \\ x &\equiv a_k \pmod{m_k}, \end{aligned}$$

possui solução. Uma delas é

$$x_0 = a_1 M_1 N_1 + a_2 M_2 N_2 + \cdots + a_k M_k N_k$$

onde

$$M_i = m/m_i \qquad e \qquad N_i = M_i^{-1} \bmod m_i$$

para $i = 1, 2, \ldots, k$.

Esta solução não é única. Para qualquer inteiro q, o número $y = x_0 + q\,m$ também é solução do sistema de congruências e duas soluções quaisquer diferem por um múltiplo de m. O sistema de congruências possui uma e somente uma solução em $\mathbf{Z}_m$.

Para provar este teorema precisamos de alguns resultados preparatórios.

Lema 3.1. *Sejam a e b números inteiros primos entre si.*

Se a e b dividem c, então o produto ab divide c.

Demonstração: Como a e b dividem c, existem inteiros q_1 e q_2 tais que $c = aq_1 = bq_2$. Sendo a e b primos entre si, existem inteiros x e y tais que $ax+by = 1$. Multiplicando por c, vem $c = cax+cby = abq_2x+abq_1y = ab(q_2x+q_1y)$, provando que c é divisível por ab. ■

Se a, b e c forem dois a dois primos entre si, então ab e c são primos entre si. Por indução, se m_1, m_2, ..., m_k forem números inteiros, dois a dois primos entre si, então $m_1 \times m_2 \times \cdots \times m_{k-1}$ e m_k são primos entre si. Neste caso, se m_1, m_2, ..., m_k forem dois a dois primos entre si e cada um deles dividir c, então o produto $m_1m_2 \cdots m_k$ divide c.

Vamos à prova do teorema chinês do resto.

Demonstração: Inicialmente, observamos que M_i contém os fatores m_1, ..., m_k, com exceção do m_i. Sendo m_1, ..., m_k todos primos entre si, M_i e m_i são primos entre si, o que garante a existência de $N_i = M_i^{-1} \bmod m_i$. Quando $r \neq i$, $M_rN_r \bmod m_i = 0$, pois M_r contém o fator m_i. Como $M_iN_i \bmod m_i = 1$, segue

$$\begin{aligned} x_0 \bmod m_i &= (a_1M_1N_1 + a_2M_2N_2 + \cdots + a_kM_kN_k) \bmod m_i \\ &= a_iM_iN_i \bmod m_i = a_i \bmod m_i \end{aligned}$$

ou

$$x \equiv a_i \pmod{m_i}$$

para $i = 1, 2, \ldots, k$, provando que x_0 é solução do sistema de congruência.

Sendo q um número inteiro e $y = x_0 + mq$, então $y \bmod m_i = (x_0 + mq) \bmod m_i = x_0 \bmod m_i = a_i \bmod m_i$, provando que y também é solução do sistema de congruências.

Se x e y forem duas soluções do sistema de congruências, $y \bmod m_i = a_i \bmod m_i = x \bmod m_i$. Isto garante que $y - x$ é divisível por m_i, para todo i e, portanto, divisível por m, mostrando que duas soluções quaisquer diferem por um múltiplo de m. ■

Nota 3.1. *É interessante observar que a função f que leva x de $\mathbf{Z}_m$ em ($x \bmod m_1$, $x \bmod m_2$, ..., $x \bmod m_k$) de $\mathbf{Z}_{m_1} \times \mathbf{Z}_{m_2} \times \cdots \times \mathbf{Z}_{m_k}$, é bijetora. Este fato permite representar elementos de $\mathbf{Z}_m$ usando elementos $(a_1, a_2, \ldots, a_k)$ de $\mathbf{Z}_{m_1} \times \mathbf{Z}_{m_2} \times \cdots \times \mathbf{Z}_{m_k}$ o que agiliza algumas operações na Criptografia de chave pública.*

Exemplo 3.6. *Vamos determinar x tal que $x \equiv 5 \pmod{11}$ e $x \equiv 8 \pmod{17}$). Neste sistema de congruências, $m_1 = 11$, $m_2 = 17$, $a_1 = 5$ e $a_2 = 8$. Calculamos $m = m_1 \times m_2 = 11 \times 17 = 187$, $M_1 = m/m_1 = 17$ e $M_2 = m/m_2 = 11$. Precisamos dos inversos multiplicativos $N_1 = M_1^{-1} \bmod m_1 = 17^{-1} \bmod 11 = 2$ e $N_2 = M_2^{-1} \bmod m_2 = 11^{-1} \bmod 17 = 14$. Usando o teorema chinês do resto,*

$$x = 5 \times 17 \times 2 + 8 \times 11 \times 14 = 1402.$$

Para obter a solução no intervalo $0 \leq x < m$, *basta calcular* $x_0 = 1402 \bmod 187 = 93$. *Observe que* $93 \bmod 11 = 5$ *e* $3 \bmod 17 = 8$.

Exemplo 3.7. *Vamos determinar as soluções de*

$$x \equiv 1 \pmod{15}, \quad x \equiv -1 \pmod{8}, \quad x \equiv 2 \pmod{13}.$$

Neste sistema de congruências,

$$\begin{aligned} a_1 &= 1, \quad a_2 = -1, \quad a_3 = 2, \\ m_1 &= 15, \quad m_2 = 8, \quad m_3 = 13. \end{aligned}$$

Calculamos o produto

$$m = m_1 m_2 m_3 = 15 \times 8 \times 13 = 1560,$$

os números

$$\begin{aligned} M_1 &= m/m_1 = 1560/15 = 104 \\ M_2 &= m/m_2 = 1560/8 = 195 \\ M_3 &= m/m_3 = 1560/13 = 120 \end{aligned}$$

e seus inversos modulares

$$\begin{aligned} N_1 &= M_1^{-1} \bmod 15 = 104^{-1} \bmod 15 = 14 \\ N_2 &= M_2^{-1} \bmod 8 = 195^{-1} \bmod 8 = 3 \\ N_3 &= M_3^{-1} \bmod 13 = 120^{-1} \bmod 13 = 9 \end{aligned}$$

Calculamos todos os números necessários para a solução do problema:

$$\begin{aligned} x &= a_1 M_1 N_1 + a_2 M_2 N_2 + a_3 M_3 N_3 \\ &= 1 \times 104 \times 14 - 1 \times 195 \times 3 + 2 \times 120 \times 9 \\ &= 3031. \end{aligned}$$

Reduzindo módulo m, *segue* $x_0 = x \bmod m = 3031 \bmod 1560 = 1471$. *Note que* $1471 \bmod 15 = 1$, $1471 \bmod 8 = 7 = -1 \bmod 8$ *e* $1471 \bmod 13 = 2$.

Exercício 3.3. *Verifique que uma solução de*

$$x \equiv 56 \pmod{1234} \quad e \quad x \equiv 78 \pmod{4321}$$

é $x = 2\,618\,604$.

3.2.6 Raiz quadrada modular

Sejam x, y e $m > 0$, números inteiros tais que

$$x^2 \equiv y \pmod{m}.$$

O x é uma **raíz quadrada módulo** m de y e y é o **quadrado módulo** m de x e neste caso se diz também que y é um **resíduo quadrático** módulo m.

Exemplo 3.8. *As soluções da equação* $x^2 \equiv 12 \pmod{26}$ *em* $\mathbf{Z}_{26}$ *são* 8 *e* 18.

A equação $x^2 \equiv 7 \pmod{26}$ *não possui solução em* $\mathbf{Z}_{26}$, *fato que se pode verificar, fazendo* x *percorrer os inteiros entre* 0 *e* 25, *elevá-lo ao quadrado e reduzindo módulo* 26.

A equação $x^2 \equiv 1 \pmod{m}$ é particularmente importante. Quando $m > 2$, ela possui pelo menos as soluções em $\mathbf{Z}_m$:

$$x = 1 \bmod m = 1$$
e
$$x = -1 \bmod m = m - 1.$$

Outras soluções podem ou não existir.

Quando m for um número primo ímpar ($m > 2$) e $x^2 \equiv 1 \pmod{m}$, então m divide $x^2 - 1 = (x - 1)(x + 1)$. Sendo m primo, ou ele divide $x - 1$ ou ele divide $x + 1$, não podendo dividir a ambos, uma vez que a diferença deles é igual a 2. Se m divide $x - 1$, então $x \equiv 1 \pmod{m}$ e, se divide $x + 1$, então $x \equiv -1 \pmod{m}$. Neste caso, existem apenas duas soluções em $\mathbf{Z}_m$ que são 1 e $-1 \bmod m$.

Vamos considerar $m = pq$, onde p e q são números primos. Se x for uma solução de

$$x^2 \equiv 1 \pmod{m}$$

então valem as congruências

$$x^2 \equiv 1 \pmod{p} \Longrightarrow x \equiv \pm 1 \pmod{p}$$
$$x^2 \equiv 1 \pmod{q} \Longrightarrow x \equiv \pm 1 \pmod{q}$$

que resultam nos quatro pares de congruência

$$\begin{array}{llll} x \equiv +1 \pmod{p} & \text{e} & x \equiv +1 \pmod{q} \\ x \equiv -1 \pmod{p} & \text{e} & x \equiv -1 \pmod{q} \\ x \equiv +1 \pmod{p} & \text{e} & x \equiv -1 \pmod{q} \\ x \equiv -1 \pmod{p} & \text{e} & x \equiv +1 \pmod{q}. \end{array}$$

As soluções das duas primeiras congruências são $x = \pm 1 \bmod m$. As outras duas resultam em soluções diferentes das anteriores, num total de quatro soluções. Do terceiro par, p divide $x - 1$ e q divide $x + 1$. Do quarto par, p divide $x + 1$ e q divide $x - 1$. O m divide $x^2 - 1 = (x - 1)(x + 1)$ e, sendo o $m \geq 4$, ele não pode dividir ao mesmo tempo $x - 1$ e $x + 1$ cuja diferença é é igual a 2. Já vimos que p e q dividem $(x - 1)$ ou $(x + 1)$ mas não podem dividir ambos pois, neste caso, então m também os dividiria o que sabemos não ser possível. Se, por exemplo, m não divide o $x - 1$, sabemos que p ou q divide, de modo que $\text{mdc}(m, x - 1)$ é um dos fatores primos de m. Logo, obter uma solução diferente de $\pm 1 \bmod m$ para a equação $x^2 \equiv 1 \pmod{m}$ permite fatorar o m. Esta é uma indicação de que obter raizes quadradas de 1 módulo m distintas de $+1$ e -1 deve ser tão difícil quanto fatorar m.

Exemplo 3.9. *Vamos resolver* $x^2 \equiv 1 \pmod{143}$. *Esta equação é equivalente ao sistema*

$$\begin{aligned} x^2 &\equiv 1 \pmod{11}, \\ x^2 &\equiv 1 \pmod{13}, \end{aligned}$$

cujas soluções são

$$\begin{aligned} x &\equiv \pm 1 \pmod{11}, \\ x &\equiv \pm 1 \pmod{13}, \end{aligned}$$

que podem ser combinadas de quatro modos

$$\begin{aligned} x &\equiv +1 \pmod{11} \quad e \quad x \equiv +1 \pmod{13}, \\ x &\equiv -1 \pmod{11} \quad e \quad x \equiv -1 \pmod{13}, \\ x &\equiv +1 \pmod{11} \quad e \quad x \equiv -1 \pmod{13}, \\ x &\equiv -1 \pmod{11} \quad e \quad x \equiv +1 \pmod{13}. \end{aligned}$$

As soluções de cada um desses pares de congruência é

$$\begin{aligned} x &\equiv 1 \pmod{143} \\ x &\equiv -1 \pmod{143} \\ x &\equiv 12 \pmod{143} \\ x &\equiv 131 \pmod{143} \end{aligned}$$

num total de quatro soluções. Observe $\text{mdc}(12 - 1,\ 143) = 11$ *e* $\text{mdc}(131 - 1,\ 143) = 13$ *são os fatores primos de* 143.

Em geral, quando

$$m = p_1 p_2 \cdots p_k$$

for o produto de k primos ímpares distintos, então

$$x^2 \equiv 1 \pmod{m}$$

possui 2^k soluções distintas módulo m.

3.2.7 Teoremas de Fermat e Euler

Teorema 3.4 (Fermat)**.** *Seja p um número primo, r côngruo a s módulo $(p-1)$.*

1. *Se o número inteiro b não for divisível por p, então*

$$b^{p-1} \equiv 1 \pmod{p},$$
$$b^r \equiv b^s \pmod{p}.$$

2. *Se r e s forem números inteiros positivos então, para todo número inteiro b vale*

$$b^r \equiv b^s \pmod{p}.$$

Demonstração: 1. Considere o conjunto $\mathbf{Z}_p^* = \{1, 2, \ldots, p-1\}$ dos números em $\mathbf{Z}_p$ que são primos com p. Quando x pertence a $\mathbf{Z}_p^*$, então $bx \bmod p$ também pertence a $\mathbf{Z}_p^*$ e a função $f : \mathbf{Z}_p^* \to \mathbf{Z}_p^*$ definida por $f(x) = (bx) \bmod p$ é injetora.

Isto implica que o conjunto $\{f(1), f(2), \ldots, f(p-1)\}$ é um rearranjo de $\mathbf{Z}_p^*$ e $f(1) \times f(2) \times \cdots \times f(p-1) = 1 \times 2 \times \cdots \times (p-1)$ e

$$1b \times 2b \times \cdots \times (p-1)b \bmod p = 1 \times 2 \times \cdots \times (p-1) \bmod p.$$

Multiplicando os dois lados pelo inversos de $2, \ldots, p-1$ módulo p, obtemos

$$b^{p-1} \bmod p = 1.$$

o que prova a primeira congruência.

Agora, sendo r e s côngruos módulo $p-1$, existe um inteiro k tal que $r = k(p-1) + s$ e $b^r \equiv b^{k(p-1)+s} \equiv \left(b^{(p-1)}\right)^k b^s \equiv b^s \pmod{p}$ o que prova a segunda congruência.

2. Já provamos que $b^r \equiv b^s \pmod{p}$ quando b não é divisível por p. Quando b for divisível por p, as potências negativas de b módulo p não existem. Entretanto, quando r e s forem positivos, tanto $b^r \bmod p$ quanto $b^s \bmod p$ são iguais a zero, provando que $b^r \equiv b^s \pmod{p}$. ■

Teorema 3.5 (Euler)**.** *Seja $n \geq 2$ um número inteiro, r e s côngruos módulo $\phi(n)$.*

1. *Se b e n forem primos entre si,*

$$b^{\phi(n)} \equiv 1 \pmod{n}$$
$$b^r \equiv b^s \pmod{n}.$$

2. *Se n for o produto de primos distintos, se $r > 0$ e $s > 0$, então*

$$b^r \equiv b^s \pmod{n},$$

para todo inteiro b.

Demonstração: 1. A prova é idêntica à do Pequeno Teorema de Fermat. O conjunto dos elementos de $\mathbf{Z}_n$ que possuem inverso multiplicativo módulo n é denotado por $\mathbf{Z}_n^*$. A função $f : \mathbf{Z}_n^* \to \mathbf{Z}_n^*$, definida por $f(x) = (bx) \bmod n$ é injetora. Por este motivo, $\{ax \bmod n : x \in \mathbf{Z}_n^*\}$ é um rearranjo de $\mathbf{Z}_n^*$ o que acarreta nas igualdades

$$\left(\prod_{x \in \mathbf{Z}_n^*} ax\right) \bmod n = \left(\prod_{x \in \mathbf{Z}_n^*} x\right) \bmod n$$

ou

$$a^{\phi(n)} \left(\prod_{x \in \mathbf{Z}_n^*} x\right) \bmod n = \left(\prod_{x \in \mathbf{Z}_n^*} x\right) \bmod n.$$

Multiplicando os dois lados da igualdade por $x^{-1} \bmod n$, para todo x em $\mathbf{Z}_n^*$, obtemos

$$b^{\phi(n)} \bmod n = 1.$$

2. Se $n = p_1 p_2 \cdots p_k$, onde todos fatores primos são distintos, sabe-se que

$$\begin{aligned} \phi(n) &= \phi(p_1)\phi(p_2)\cdots\phi(p_k) \\ &= (p_1 - 1)(p_2 - 1)\cdots(p_k - 1) \end{aligned}$$

Sendo $r \equiv s \pmod{n}$, então $r \equiv s \pmod{p_i}$, para $i = 1, 2, \ldots, k$. Pelo teorema de Fermat, $b^r \equiv b^s \pmod{p_i}$ quando b não for divisível por p_i e vale a congruência quando b é divisível por p_i pois os dois lados são côngruos a zero. Por este motivo, quando b é divisível por p_i, os expoentes devem ser positivos pois, se r ou s for nulo, em lugar de zero obtemos 1 e, se r ou s for negativo, não existe o inverso de b módulo p_i. Assim, para $i = 1, 2, \ldots, k$. p_i divide $b^r - b^s$ que portanto é divisível por n, provando que $b^r \equiv b^s \pmod{n}$. ■

Exemplo 3.10. *Seja* $n = 35 = 5 \times 7$, *quando* $\phi(n) = 4 \times 6 = 24$. *Observe que* $27 \equiv 3 \pmod{24}$ *e* $10^{27} \bmod 35 = 10^3 \bmod 35 = 20$, *mesmo diante do fato de* 10 *e* 35 *não serem primos entre si.*

Para ver que a segunda parte deste teorema não vale quando n é um produto de primos repetidos, basta apresentar um contra exemplo.

Exemplo 3.11. *Seja* $n = 24 = 2^3 3$ *quando* $\phi(n) = 24\ (1 - 1/2)\ (1 - 1/3) = 8$. *Observe que* $9 \equiv 1 \pmod 8$) *mas* $14^9 \bmod 24 = 8$ *é diferente de* $14^1 \bmod 24 = 14$. *Este resultado negativo se deve ao fato de* 24 *ser o produto dois fatores primos sendo que o* 2 *aparece elevado ao cubo.*

Quando b não for primo de n, então r e s devem ser números inteiros positivos para garantir que $b^r \equiv b^s \pmod n$. O que acontece quando r é negativo é que $b^r \bmod n = \left(b^{-1}\right)^{-r} \bmod n$ não existe uma vez que o inverso multiplicativo de b módulo n não existe.

Exemplo 3.12. *Sendo* $n = 127 \times 79 = 10\,033$ *então* $\phi(n) = 126 \times 78 = 9828$. *Os números* 9513 *e* -315 *são côngruos módulo* $\phi(n)$ *mas* $79^{9513} \bmod 10\,033$ *não é igual a* $79^{-315} \bmod 10\,033$. *O primeiro é igual a* 6478 *enquanto o segundo não está definido, uma vez que o inverso multiplicativo de* 79 *módulo* 10 033 *não existe.*

3.2.8 Teste de primalidade

Vamos apresentar o teste de primalidade de Miller e Rabin que foi obtido em Menezes, Oorschot, Vanstone [14], onde o leitor interessado poderá encontrar informações adicionais sobre ele. Como já dissemos ele recebe um número inteiro positivo e responde se o número é primo ou não. A probabilidade deste algoritmo afirmar que o número é primo dado que a entrada é um número composto é de 1/4. Algoritmos probabilísticos que recebem uma entrada e respondem a uma pergunta com um "sim" ou um "não" onde uma delas pode estar errada, são denominados algoritmos de Monte Carlo.

Vimos que, se n for um número primo ímpar, e $x^2 \equiv 1 \pmod n$ então $x = \pm 1 \bmod n$. Se $x \neq \pm 1 \bmod n$, já podemos afirmar que n não é primo.

Vamos descrever a base matemática para o algoritmo.

De acordo com o teorema de Fermat, se n for primo e a for primo de n, então $a^{n-1} \bmod n = 1$. Existem números compostos n para os quais $a^{n-1} \bmod n = 1$ e diremos que n é um **pseudo primo** para a base a. Entretanto, isto não se verifica para todo a. Mudando a base a de forma aleatória, obteremos uma para a qual $a^{n-1} \bmod n \neq 1$. Vimos ainda que, se n for primo e $a^2 \equiv 1 \pmod n$, então $a \equiv \pm 1 \pmod n$. O único primo par é o 2 e não precisamos usar nenhum teste para examinar sua primalidade. Vamos supor então que n é um número ímpar

quando $n-1$ é par e existem r e s inteiros, com r ímpar, tais que $n-1 = r2^s$. Quando n for primo e calculamos as potências

$$a^r \bmod n,\ a^{2r} \bmod n,\ a^{4r} \bmod n,\ \ldots,\ a^{2^{s-1}r} \bmod n,$$

vamos obter em algum ponto desta sequência 1 ou $n-1 = -1 \bmod n$ para que $a^{2^s r} \bmod n = 1$. Existem números compostos para os quais isto se verifica para uma base a e chamaremos estes números compostos de **pseudo primos fortes** para a base a. Entretanto são raros os pseudo primos fortes e quando os encontramos, são poucas as bases que são "enganadas" por eles.

Temos então ao algoritmo de Miller-Rabin:

Algoritmo de Miller-Rabin para testar se um número ímpar é primo ou não.

Entrada: um inteiro ímpar n que se deseja saber se é primo ou não.
Saída: uma das respostas: "n é primo" ou "n é composto".

1. Escreva $n-1 = 2^s r$, onde r é ímpar.
2. Escolha aleatoriamente um inteiro a, $1 < a < n-1$.
3. Calcule $b = a^r \bmod n$.
4. Se $b \equiv 1 \pmod{n}$ então responda "n é primo" e encerre.
5. Para i variando de 0 a $s-1$ faça
   (a) Se $b \equiv -1 \pmod{n}$ então responda "n é primo" e encerre.
   (b) Caso contrário, faça $b = b^2 \bmod n$.
6. Responda "n é composto" e encerre.

Aplicando o teste de Miller-Rabin diversas vezes, com diferentes valores de a, se a resposta for sempre "n é primo" em j aplicações sucessivas do teste, a probabilidade de n ser composto cai para $(1/4)^j$.

3.3 RSA

O trabalho de Diffie e Hellman foi um marco na Criptografia e abriu as portas para a Criptografia de Chave Pública, aquela na qual cada pessoa constrói um par de chaves, sendo uma pública, do conhecimento de todos, e uma chave privada, que ele

deve manter no mais absoluto segredo. As mensagens cifradas com a chave pública só podem ser decifradas por aquele que possui a chave privada correspondente. As mensagens cifradas com a chave privada só poderão ser decifradas com a chave pública correspondente. Este fato permite que uma pessoa assine um documento digitalmente. Ele escreve o documento e o cifra usando sua chave privada. Envia ambos para as pessoas interessadas. Desejando provar a origem do documento, basta decifrar com a chave pública a parte cifrada e comparar com o documento que foi enviado em claro. Havendo a necessidade de manter o documento em sigilo, ele e sua parte cifrada podem ser cifrados com a chave pública de quem os vai receber.

A primeira técnica de chave pública foi inventada em 1977 por Ronald Rivest, Adi Shamir e Leonard Adleman e recebeu o nome de RSA, que são as iniciais dos sobrenomes dos seus inventores. A técnica foi divulgada em 1977 por um memorando do MIT e publicada em 1978 no Communications of the ACM, sob o título A Method for Obtaining Digital Signatures and Public-Key Cryptosystems.

Para o computador, toda mensagem é traduzida numa seqüência finita de bits. O bit é a unidade básica de memória, capaz de gravar duas informações distintas: zero ou um. Podemos interpretar que um conjunto de bits é a representação na base binária de um número inteiro. Quebrando a mensagem em blocos de k bits pode-se visualizar a mensagem como uma seqüência finita de números inteiros. Isto permite tratar os números como sendo as letras do alfabeto das mensagens e o processo de cifrar pode ser efetuado através das operações aritméticas em $\mathbf{Z}_n$ para algum n escolhido de modo adequado. O alfabeto sobre o qual atua o RSA é formado por blocos de bits e por este motivo se diz que ele é um sistema de cifra em bloco.

Preparando as chaves

Beto prepara suas chaves pública e privada para receber correspondência criptografada:

1. Escolhe dois números primos p e q.
2. Calcula $n = pq$ e $\phi(n) = (p-1)(q-1)$.
3. Escolhe e de modo aleatório entre 2 e $n-1$, que seja primo de $\phi(n)$, e calcula $d = e^{-1} \bmod \phi(n)$ usando o algoritmo estendido de Euclides.
4. O par de números $(n,\ e)$ é a **chave pública** de Beto e são enviados para Alice ou para uma lista pública de chaves que Alice possa acessar.
5. Beto mantém em segredo o terno de números $(p,\ q,\ d)$, podendo descartar p e q se assim o desejar. O número d é a **chave privada** do Beto.

Cifrando uma mensagem

Qualquer pessoa, em posse da chave privada de Beto pode enviar-lhe mensagens criptografadas.

No computador, uma mensagem é gravada como uma série de zeros e uns que pode ser encarado como um número binário. Quando a mensagem é grande, é conveniente quebrar esta mensagem em blocos com um número fixo de bits.

Para Alice enviar uma mensagem cifrada para Beto ela a quebra o conjunto de bits em blocos, todos do mesmo tamanho. Cada bloco m da mensagem deve possuir menos bits que n. Na prática, o bloco m deve ter um byte a menos do que n. Ela calcula

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

enviando-o para o Beto. Pode usar qualquer meio para transmitir o c. Não é preciso usar um canal secreto.

Se for cifrar uma mensagem com diversos blocos, ela deve proceder como no caso da criptografia simétrica, usando os processos Electronic Code Book (ECB), Cipher Block Chaining (CBC), ou um outro processo qualquer para cifrar múltiplos blocos.

Decifrando uma mensagem

Quando Beto recebe a mensagem cifrada de Alice, ele a decifra calculando

$$m = c^d \bmod n.$$

Verificação

Vamos verificar que Beto de fato recupera a mensagem original. Como $ed \bmod \phi(n) = 1$ e $n = pq$, segue para todo c em $\mathbf{Z}_n$ que

$$c^d \bmod n = (m^e)^d \bmod n = m^{ed} \bmod n = m \bmod n = m$$

pois m pertence a $\mathbf{Z}_n$.

Exemplo 3.13. *Veja o notebook RSA.nb.*

Segurança do RSA

A segurança do RSA reside na dificuldade de se obter os fatores primos p e q do módulo n. Há um ataque possível ao RSA que consiste em uma pessoa publicar uma chave fazendo-se passar por outro. Este problema pode ser contornado com a existência de autoridades certificadoras confiáveis e assinaturas digitais, assuntos tratados em capítulo posterior.

3.4 A criptografia de ElGamal

No Crypto'84, Taher ElGamal, americano com descendência árabe, apresentou um esquema de criptografia de chave pública cuja segurança é garantida pela dificuldade de resolver o problema do logaritmo discreto. Este trabalho foi publicado no Advances in Cryptology: Proceedings of Crypto'84, Lecture Notes in Computer Science 196, 1985.

Vamos supor que Chapeuzinho Vermelho queira enviar uma mensagem para a Vovozinha, sem que o Lobo Mau consiga etender o seu conteúdo, mesmo que consiga obter a mensagem pelo meio do caminho. Chapeuzinho e a Vovó combinam cifrar as mensagens com ElGamal que é um sistema de criptografia em bloco com chave pública.

Para que a Vovó possa receber uma mensagem cifrada do Chapeuzinho, ela escolhe um número primo p e um elemento primitivo α de $\mathbf{Z}_p^*$. Vovó deve ainda escolher um inteiro k no intervalo $1 < k < p-1$ e calcular $\beta = \alpha^k \bmod p$. Em seguida, envia para o Chapeuzinho os valores de p, α e β. Não existe a necessidade de manter estes números em segredo e o Lobo Mau pode conhecê-los. A Vovó deve manter k em segredo pela Vovó, guardando-o com todo o cuidado. O terno ordenado (p, α, β) é a **chave pública** da Vovó e k é a sua **chave privada**.

Para o Chapeuzinho Vermelho enviar uma mensagem cifrada para a Vovó, ela quebra sua mensagem em blocos $m \in \mathbf{Z}_p$. Para cada bloco, escolhe aleatoriamente um número inteiro s no intervalo $1 < s < p-1$ e, mantendo-o em segredo, calcula

$$(y, z) = e_k(m, s) = (\ \alpha^s \bmod p,\ m\beta^s \bmod p\ ),$$

enviando $(y, z) \in \mathbf{Z}_p^* \times \mathbf{Z}_p$ para a Vovó. Para cada bloco m da mensagem, deve-se escolher um s diferente.

Para decifrar o texto cifrado, Vovó usa a chave privada k que apenas ela conhece e calcula

$$\begin{aligned} d_k(y, z) &= z(y^k)^{-1} \bmod p \\ &= m\beta^s(\alpha^{ks})^{-1} \bmod p \\ &= m\beta^s(\beta^s)^{-1} \bmod p \\ &= m \bmod p = m\,, \end{aligned}$$

recuperando assim o bloco original da mensagem.

Para este método de criptografar, $P = \mathbf{Z}_p^*$ é o alfabeto das mensagens originais e $C = \mathbf{Z}_p^* \times \mathbf{Z}_p^*$ é o alfabeto das mensagens cifradas. Neste algoritmo, a mensagem cifrada depende não apenas de m mas do valor s que é escolhido de modo aleatório por Chapeuzinho Vermelho. Modificando o s, modifica-se o texto criptografado.

De modo informal, podemos dizer que no ElGamal o texto m a ser cifrado é mascarado ao ser multiplicado por β^s, resultando no texto cifrado z. Junto com

z segue o $y = \alpha^s \bmod p$. Quando o destinatário recebe a mensagem, como apenas ele conhece k, pode obter β^s a partir de α^s, removendo a "máscara" que ocultava a mensagem.

Para garantir a segurança do ElGamal, o número p deve ter pelo menos 300 dígitos e $p-1$ deve ter pelo menos um fator primo "grande". Pode-se obter um tal primo a partir de um número primo grande q para o qual $p = 2Rp_1 + 1$ é primo, para algum número R não muito grande. Este expediente evita alguns ataques ao problema do logaritmo discreto. As ordens de grandeza recomendadas podem ser obtidas nos livros de Menezes [14] e Schneier [17].

Exemplo 3.14. *Vovó escolhe* $p = 2579$, $\alpha = 2$, $k = 765$ *e calcula*

$$\beta = 2^{765} \bmod 2579 = 949\,.$$

A Vovó envia para Chapeuzinho ou para um local público sua chave privada (p, α, β), *mantendo em segredo* k *que é sua chave privada. Para criptografar* $m = 1299$, *Chapeuzinho escolhe* $s = 853$ *ao acaso e calcula*

$$y = 2^{853} \bmod 2579 = 435\ ,$$
$$z = 1299 \times 949^{853} \bmod 2579 = 2396\,.$$

Quando o Chapeuzinho recebe (y, z), *ela calcula*

$$y(z^k)^{-1} \bmod 2579 = 2396 \times (435^{765}) \bmod 2579 = 1299\,,$$

recuperando assim o texto original x.

3.5 Considerações finais

O sistema de criptografia desenvolvido por ElGamal [6] utiliza de modo interessante um número aleatório para executar a criptografia de um texto. Tal como outras técnicas de criptografia de chave pública, o ElGamal se presta à realização de assinatura digital de documentos eletrônicos.

Como leitura complementar sobre Criptografia, recomendo os excelentes livros de Menezes [14] e Schneier [17] que, cobrindo um espectro bastante amplo, são ótimas referências para consulta. O livro de Stinson [23] é uma excelente referência para um curso de Criptografia. Mais voltado para a segurança de redes de computadores há o livro de Stallings [22] que foi premiado, inclusive. O livro de Tilborg [24], bastante detalhado, apresenta os algoritmos e sua implementação com o software Mathematica. O livro de Washington e Trappe [26], é muito bem escrito, de fácil leitura, apresenta diversos exemplos ilustrativos, sendo recomendado para uma primeira leitura solo. Uma referência mais teórica, ao sabor dos

matemáticos, pode ser encontrada em Koblitz [12]. Ao final do Capítulo 8 de [14] os autores recomendam a leitura dos artigos de Gordon [9], McCurley [13], Rivest e Sherman [15] e Rueppel et all. [16] onde são discutidos aspectos da segurança oferecida pelo ElGamal.

3.6 Exercícios

1. Compile e execute o programa listado no texto em C++, usando o ambiente e o compilador de sua preferência, para calcular a potência modular $y = a^x \bmod n$, fornecendo como entrada $n = 26$, $a = 2$, $x = 8$. Efetue a conta à mão e compare os resultados. Forneça como entrada $n = 897$, $a = 528$, $x = 67$ e leia o resultado. Você faria estas contas usando apenas lápis e papel? Tente e medite.

2. Programe o algoritmo proposto no texto para obter um gerador de um grupo cíclico e obtenha um gerador para $\mathbf{Z}_{241}$.

3. Determine x em $\mathbf{Z}_{241}$ tal que $45x \equiv 78 \pmod{241}$

4. Determine o único x em $\mathbf{Z}_{421\,872}$ tal que $x \equiv 27 \pmod{752}$ e $x \equiv 93 \pmod{561}$.

5. Sabendo que 79920 é uma raiz quadrada de 1 módulo 81317, obtenha os fatores primos de 81317. Sugestão: Os fatores primos são o mdc$(79920 - 1,$ 81317) e o mdc$(79920 + 1,$ 81317).

6. O número 1523 é primo. Calcule $45^{1522} \bmod 1523$ e verifique que, de acordo com o teorema de Fermat, este número é igual a 1. Verifique ainda que $45^{1500} \bmod 1523$ é igual a $45^{-22} \bmod 1523$.

7. Sendo $n = 127 \times 79 = 10\,033$, então $\phi(n) = (127-1)\,(79-1) = 9828$. Dados $r = 9513$ e $s = -315$, que são côngruos módulo $\phi(n)$, calcule $3^{9513} \bmod 10\,033$, $3^{-315} \bmod 10\,033$ e verifique que são iguais, como prevê o teorema de Euler.

8. Programe o algoritmo de Miller-Rabin na linguagem de sua preferência e verifique se são primos: (a) 153 796 723 e (b) 65 406 647. Chame o algoritmo 10 vezes com bases diferentes.

9. Sendo $n = 421 \times 719 = 302\,699$, então $\phi(n) = 420 \times 718 = 301\,560$. Escolha um número inteiro e aleatoriamente no intervalo $1 < e < \phi(n)$ e calcule $d = e^{-1} \bmod \phi(n)$. O par de números (e, n) será sua a chave pública e o par (d, n) sua chave privada da técnica RSA. Cifre o número $m = 247\,851$ usando sua chave pública e depois decifre o resultado usando sua chave privada. Você deve recuperar m.

10. Considere o número primo $p = 49\,957$. Determine um elemento primitivo α de $\mathbf{Z}_p^*$ e escolha um número inteiro k no intervalo $1 < k < p - 1$. Calcule $\beta = \alpha^k \bmod p$. O número k será sua chave privada e os números p, α e β serão sua chave pública para a técnica de ElGamal. Escolha aleatoriamente um número intiro s no intervalo $1 < s < p - 1$ e cifre a mensagem $m = 32\,712$ calculando o par (y, z), onde $y = \alpha^s \bmod p$ e $z = m\beta^s \bmod p$. Em seguida, decifre o par (y, z) calculando $z\left(y^k\right)^{-1} \bmod p$. Verifique que realmente você recuperou a mensagem m.

Bibliografia

[1] J. A. Buchmann, "Introduction to Cryptography", Springer, 2nd edition, 2004.

[2] S. C. Coutinho, "Números Inteiros e Criptografia RSA", segunda edição, IMPA, 2005.

[3] J. Daemen e V. Rijmen, "The Design of RijndaeL: AES - The Advanced Encryption Standard", Springer, 2002.

[4] D. E. Eastlake, K. Niles e E. Donald, "Secure XML: The New Syntax for Signatures and Encryption", Addison Wesley Professional, 2002.

[5] U. Eco, "O pêndulo de Foucault", Record, 1997.

[6] T. ElGamal, A public key cryptosystem and a signature scheme based on discrete logarithms, *IEEE Transactions on Information Theory*, **31** (1985), 469-472.

[7] A. C. Faleiros, "Aritmética, Álgebra e Cálculo com o Mathematica", Editora Edgard Blücher, 1998.

[8] H. F. Gaines, "Cryptanalysis. A Study of Ciphers and Their Solutions", Dover, 1939

[9] D. M. Gordon, Designing and detecting trapdoors for discrete log cryptosystems, *Advances in Cryptology–CRYPTO '92* **(LNCS 740)**, 66-75, 1993.

[10] F. Higenbottam, "Codes and Ciphers, Teach Yourself Books", Hodder & Stoughton Ltd, 1973.

[11] D. Kahn, "The Codebreakers, The Story of Secret Writing", Scribner, 1967. Revised edition of 1996.

[12] N. I. Koblitz, "A Course in Number Theory and Cryptography", Graduate Texts in Mathematics, No 114, Springer Verlag, 2nd edition, 1994.

[13] K. S. McCurley, A key distributions system equivalent to factoring, *Journal of Cryptology*, **1** (1988), 95–105.

[14] A. J. Menezes, P. C. V. Oorschot e S. A. Vanstone, "Handbook of Applied Cryptography", CRC Press, 1997.

[15] R. L. Rivest e A. T. Sherman, Randomized encryption techniques, *Advances in Cryptology*, **Proceedings of Crypto 82**, 145–163, 1983.

[16] R. A. Rueppel, A. Lenstra, M. Smid, K. McCurley, Y. Desmedt, A. Odlyzko e P Landrock, The Eurocrypt '92 controversial issue: trapdoor primes and moduli, *Advances in Cryptology – EUROCRYPT '92* **(LNCS 658)**, 194–199, 1993.

[17] B. Schneier, Applied Cryptography. "Protocols, Algorithms and Source Code in C", 2nd Edition, John Wiley & Sons, 1996.

[18] B. Schneier, "Secrets and Lies: Digital Security in a Networked World", Wiley, 2004.

[19] B. Schneier, J. Kelsey, D. Whiting, D. Wagner, C. Hall e N. Ferguson, "The Twofish encryption algorithm: a 128-bit block cipher", Wiley, 1999.

[20] A. Sgarro, "Códigos Secretos", Editora Melhoramentos, 1994.

[21] S. Singh, "O Livro dos Códigos", Editora Record, 2001.

[22] W. Stallings, "Cryptography and Network Security, Principles and Practice", second edition, Prentice Hall, 1999.

[23] D. R. Stinson, "Cryptography, Theory and Practice", CRC Press, 1995.

[24] H. C. A. van Tilborg, "Fundamentals of Cryptology, A Professional Reference and Interactive Tutorial", Kluwer Academic Publishers, 2000.

[25] Julio Verne, "Romances: O escaravelho de ouro, Matias Sandorf, A Jangada".

[26] L. C. Washington e W. Trappe, "Introduction to Cryptography with Coding Theory", Prentice Hall, 2002.

Índice