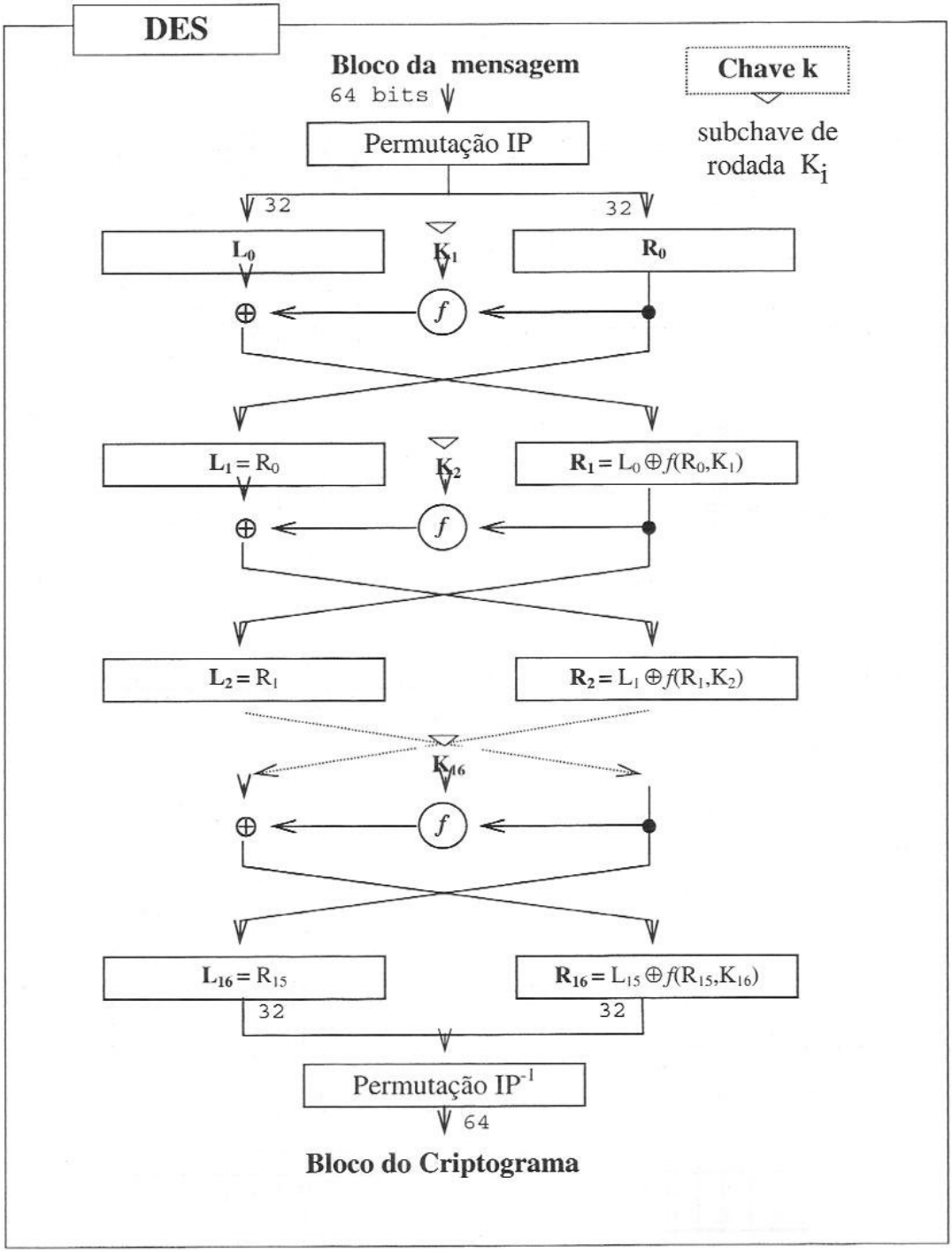
5: Algoritmos Criptográficos

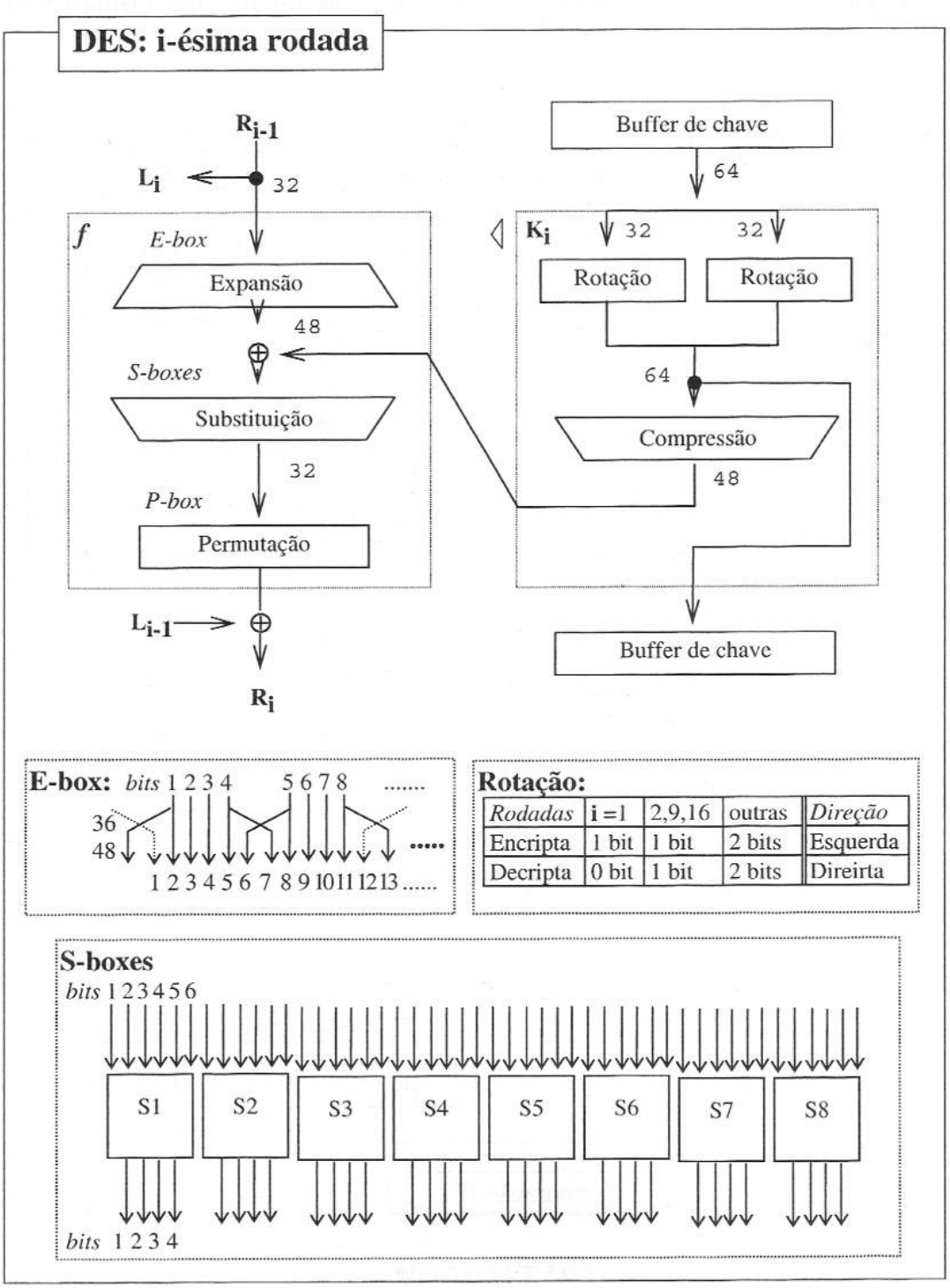
- **Cenário inicial da padronização em criptografia -**
  - **até 1970**:....Não existiam padrões, e praticamente nenhuma cultura ou pesquisa criptográfica fora da área militar. Pequenas companias vendiam hardware de criptografia restrita para governos.
  - **1972**.....*National Institute of Standards and Technology* iniciou um programa para proteção a computadores e comunicação de dados que previa o estabelecimento de um algoritmo criptográfico padrão que permitisse interoperabilidade nas comunicações.
  - **1973**.....NIST abriu concurso para escolha de um algoritmo padrão que obedecesse os critérios de segurança divulgados. Nenhum dos candidatos conseguiu alcançar esses critérios.
  - **1974**.....NIST relança o concurso. O algoritmo submetido pela IBM (baseado no algoritmo LUCIFER) passou nos critérios preliminares. Sua avaliação detalhada foi requisitada à National Security Agency, que o cosiderou seguro. Foi então escolhido.
  - **1975**.....NIST negocia com a IBM os direitos de uso irrestrito e livre do algoritmo, após alteração feita pela NSA que encurtou o tamanho das chaves e após debates públicos sobre sua segurança.
  - **1976**.....O algoritmo escolhido é adotado em 23/11/76 como padrão oficial nos EUA, renovável a cada 5 anos, com nome de *Data Encription Standard* (DES). O NIST publica várias documentos de especificação para implementação e uso do DES, com intenção de poder validar implementações em hardware.
  - **1981-92**.....ANSI, ISO acolhem e NIST renova o padrão DES.

Descrição do algoritmo padrão DES

- **Especificação geral - C**ifra de bloco de 64 bits: o bloco de entrada, chave e criptograma tem o mesmo tamanho. O algoritmo é uma *rede de Feistel*, constituida de 16 rodadas que alternam substituição *f*, XOR e permutação em subblocos L, R.



Cada oitavo bit da chave **k** (paridade do byte), é extraído e os outros permutados antes da geração das subchaves locais K_i. Estas subchaves são calculadas por permutação e compressão, repetidas a cada rodada, entrando na composição de ***f*** conforme o diagrama e tabelas abaixo



- **Especificação das substituições DES -**

As caixas de substituição (*S-boxes*) foram escolhidas de forma a minimizar a alcance da técnica estatística de ataque por texto pleno escolhido, conhecido a partir de 1990 como *criptoanálise diferencial*.

S1	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	C	E	2	D	1	A	6	C	B	9	5	3	8
2	4	1	D	8	C	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	D

↑ Bits (b1 b6) de entrada ←Bits (b2 b3 b4 b5) de entrada→ *valores hexadecimais*

S2	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

↓ S-box ...↑ Bits de saída ↑... *valores hexadecimais*

S3	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

S4	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

S5	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

S6	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	E	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

S7	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

S8	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

Análise do DES

- **Segurança -**

1. A IBM afirma ter o DES consumido 17 homens-ano de intenso trabalho de criptoanálise. A NSA supostamente induziu alterações no algoritmo original, no comprimento útil da chave (de 112 para 56 bits) e nas caixas S, talvez para eliminar possíveis *trapdoors* plantadas pela IBM.
2. Durante os debates para escolha do padrão pela NIST, alguns críticos suspeitaram terem as alterações sido impostas para que a NSA pudesse plantar *trapdoors* no algoritmo. Em 1978 uma comissão do senado americano com acesso a material secreto desautorizou essas suspeitas.

- **Chaves fracas -**

As quatro chaves abaixo, dentre as possíveis 72 057 594 037 927 936 chaves do DES, geram apenas 4 subchaves, usadas em 4 rodadas cada.

Valores de chaves fracas do DES (com bits de paridade)	
k= 0101 0101 0101 0101	*hexadecimal*
k= 0101 0101 FEFE FEFE	
k= FEFE FEFE 0101 0101	
k= FEFE FEFE FEFE FEFE	

- **Número de rodadas -**

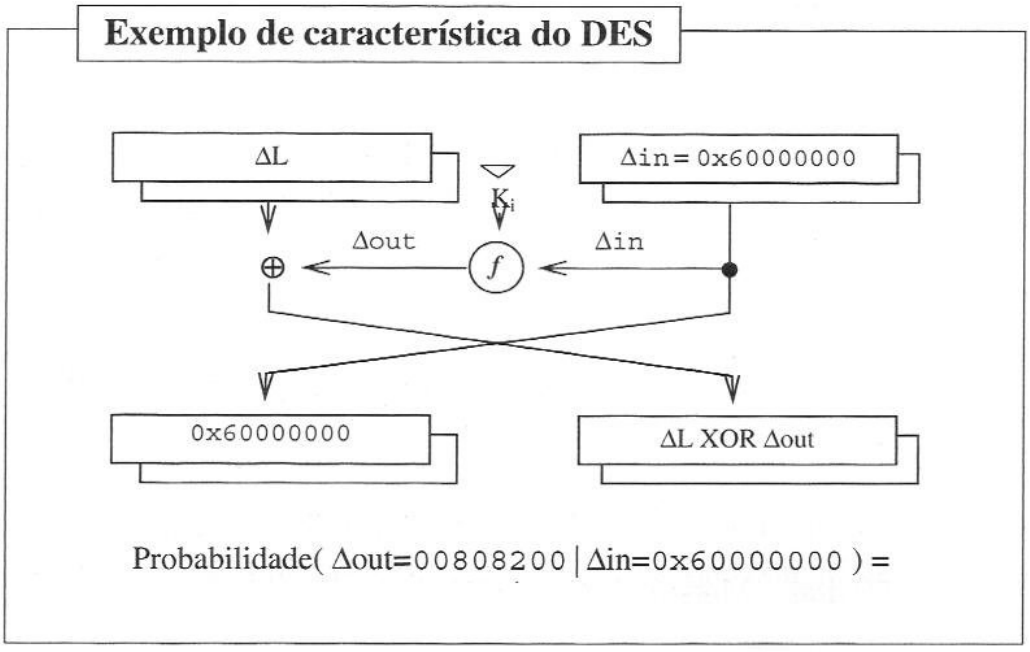
Após 5 rodadas, qualquer bit do criptograma é função dos 64 bits de entrada e dos 56 bits da chave. Porém a técnica de criptonálise diferencial (secretamente conhecida dos projetistas do DES) permite ataque eficiente por texto pleno escolhido, caso o algoritmo execute menos de 16 rodadas.

Criptoanálise diferencial

- **Aplicações -**

Nas redes de Feistel onde ***f*(k,m)** aplica substituição em **k⊕m**, as propriedades estatísticas de ***f*** podem ser exploradas em ataques de texto pleno conhecido, pela análise de diferenças na entrada e saída de ***f***:

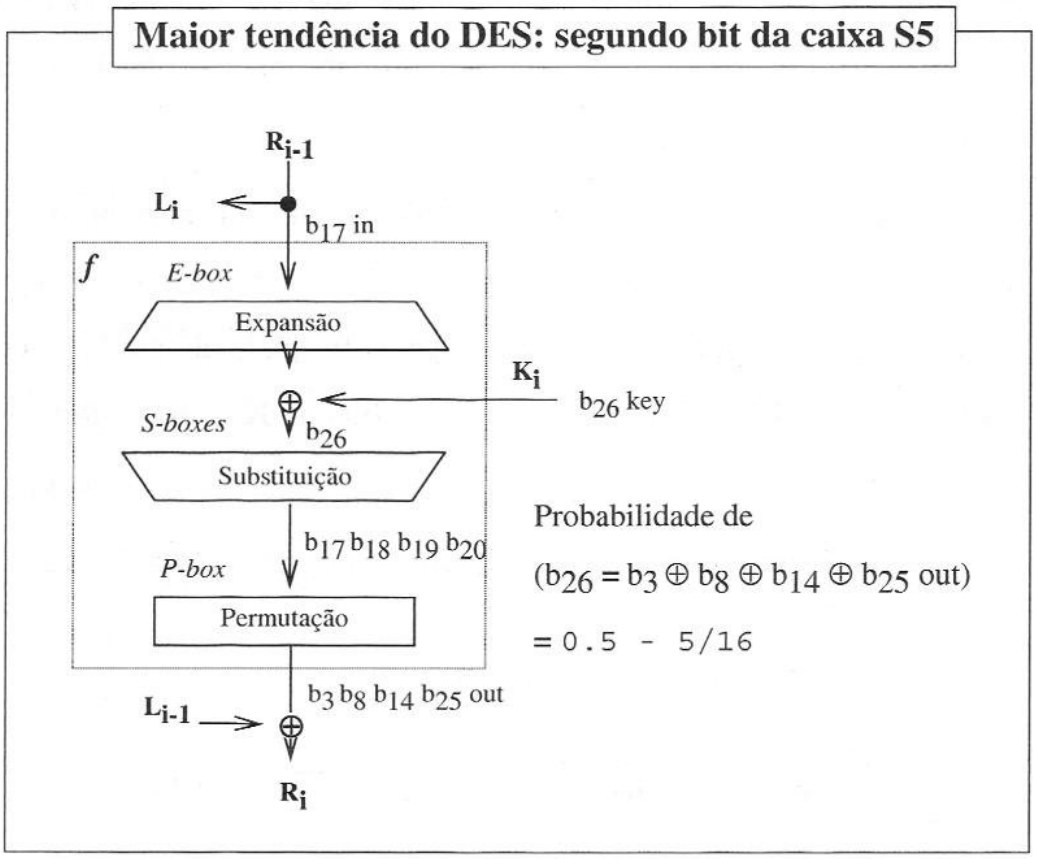
1. Calcula-se a probabilidade de ocorrências de padrões na saída ***f*(k,m) ⊕*f*(k,m')**, para dada diferença fixa entre valores de entrada **m** e **m'**. Constrói-se uma tabela para ***f*** dessas probabilidades, que independem de **k**. (Tabela de características de ***f***)
2. De posse de uma cifra com chave **k** desconhecida, encripta-se vários pares de mensagens **m**, **m'** e mede-se a freqüência de diferenças nos pares de saída na rodada **i**. Compara-se as freqüências medidas com as características de ***f***, para inferir prováveis bits de $\mathbf{K_i}\oplus\mathbf{m}$ e $\mathbf{K_i}\oplus\mathbf{m'}$
3. Combinam-se as probabilidades para valores de bits em subchaves $\mathbf{K_i}$ das várias rodadas, para se estimar os bits mais prováveis da chave **k**.



Ataques ao DES por texto pleno adaptativo

- **Técnica mais eficaz em 1995: criptoanálise linear-** (Matsui 93)

Após a criptoanálise diferencial se tornar pública, foi inventado a técnica de ataque da *criptoanálise linear*, que usa aproximação linear para descrever o comportamento interno de uma cifra de bloco, explorando tendências não randômicas desse comportamento.

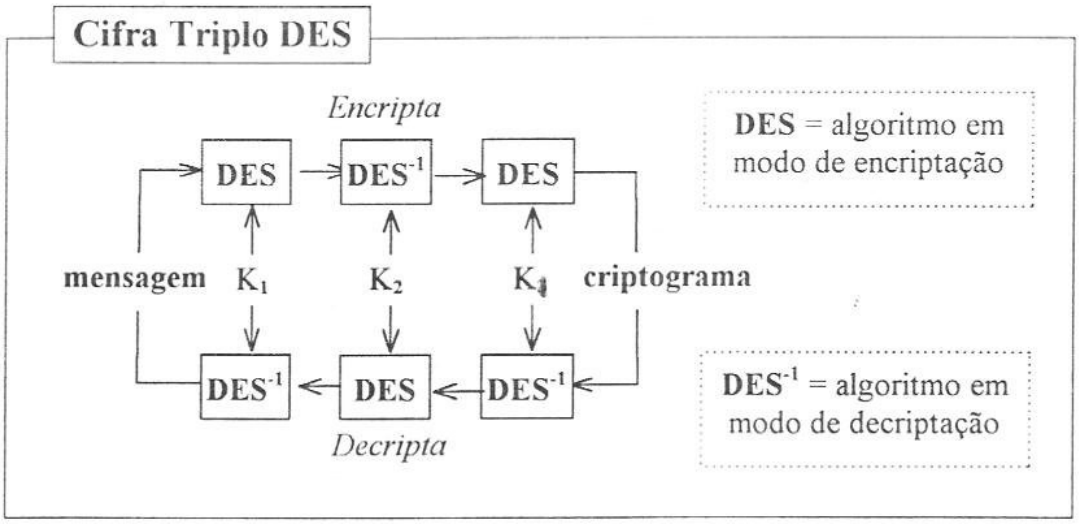


Com refinamentos desenvolvidos até 1995, esta é a melhor técnica publicamente conhecida para quebra do DES, usando em média 2^{43} blocos de texto pleno escolhidos. Uma implementação em software, com o processamento distribuído em 12 estações HP9000/735 conseguiu recuperar a chave em 50 dias. (Matsui, CRIPTO 94)

Técnicas de robustecimento do DES

- **Encriptação tripla -**

Até 1992, não se sabia se essa técnica aumentava realmente a segurança da cifra, ou apenas seu tempo de execução. A demonstração de que o algoritmo criptográfico do DES não forma um grupo (Campbell, CRIPTO '92) confirmou a eficácia da tripla encriptação. Composições menores ou em outras disposições triplas resultam mais frágeis que esta:



- **Caixas de substituição dinâmicas -**

Alguns fabricantes de placas de encriptação DES implementam alocação dinâmica para valores das caixas $\mathbf{S_1..S_8}$ durante a leitura da chave.

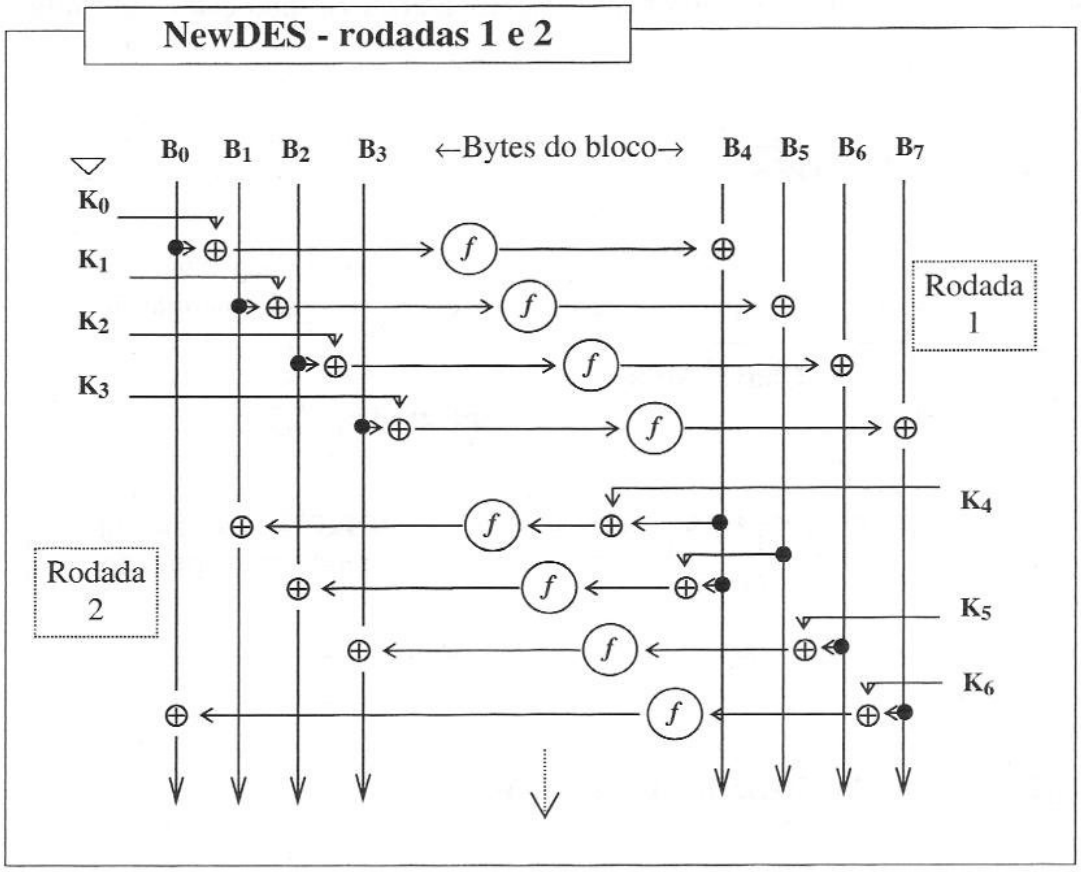
Toda a segurança do DES e os ataques eficientes hoje conhecidos concentram-se nessas caixas. Substituições aleatórias quase sempre tornam o algoritmo bem mais vulnerável à criptoanálise diferencial ou linear. (resistência da cifra a uma dessas técnicas tende a favorecer à outra)

Substituições dinâmicas só devem ser usadas com vida útil curta.

Outros Algoritmos Simétricos

- **NewDES** (Robert Scott, 1985) **-**

Apesar do nome, não é um padrão nem é variante do DES. É um algoritmo iterativo com 17 rodadas, que usa blocos de 64 bits e chave de 120 bits, cujos bytes são as subchaves $\mathbf{K_0}...\mathbf{K_{14}}$, usadas intercaladamente.



- **Análise do NewDES -**

Concebido para implementação em software mais eficiente que o DES, pois opera em bytes. É menos seguro embora use chave maior, sucumbindo ao ataque de *chaves relacionadas,* com 2^{33} chaves e mensagens escolhidas em 2^{33} passos. (parecido à criptoanálise diferencial, onde mede-se a freqüência de diferenças na saída entre pares de chaves)

- **Khufu** (Ralph Merkle, 1990) **-**

Algoritmo cujo projeto explora as deficiências do DES em software. Para cifra de bloco de 64 bits. Iterativo com chave de 512 bits, número de rodadas configurável e S-*boxes* 8x32 dinâmicas, geradas a partir da chave.

Apresenta *overhead* de tempo execução para cálculo das S-*boxes*, com impacto em encriptações curtas e em ataques por força bruta. É resistente à analise diferencial e linear, e patenteado (licenças concedidas pela Xerox Corp, P.O. box 1600, Stamford CT, EUA)

- **Khafre** (Ralph Merkle, 1990) **-**

Algoritmo cujo projeto explora as deficiências do DES em software. Para cifra de bloco de 64 bits, é uma rede de Feistel iterativa com número de rodadas configurável e usa chave de tamanho variável, entre 64 e 128 bits.

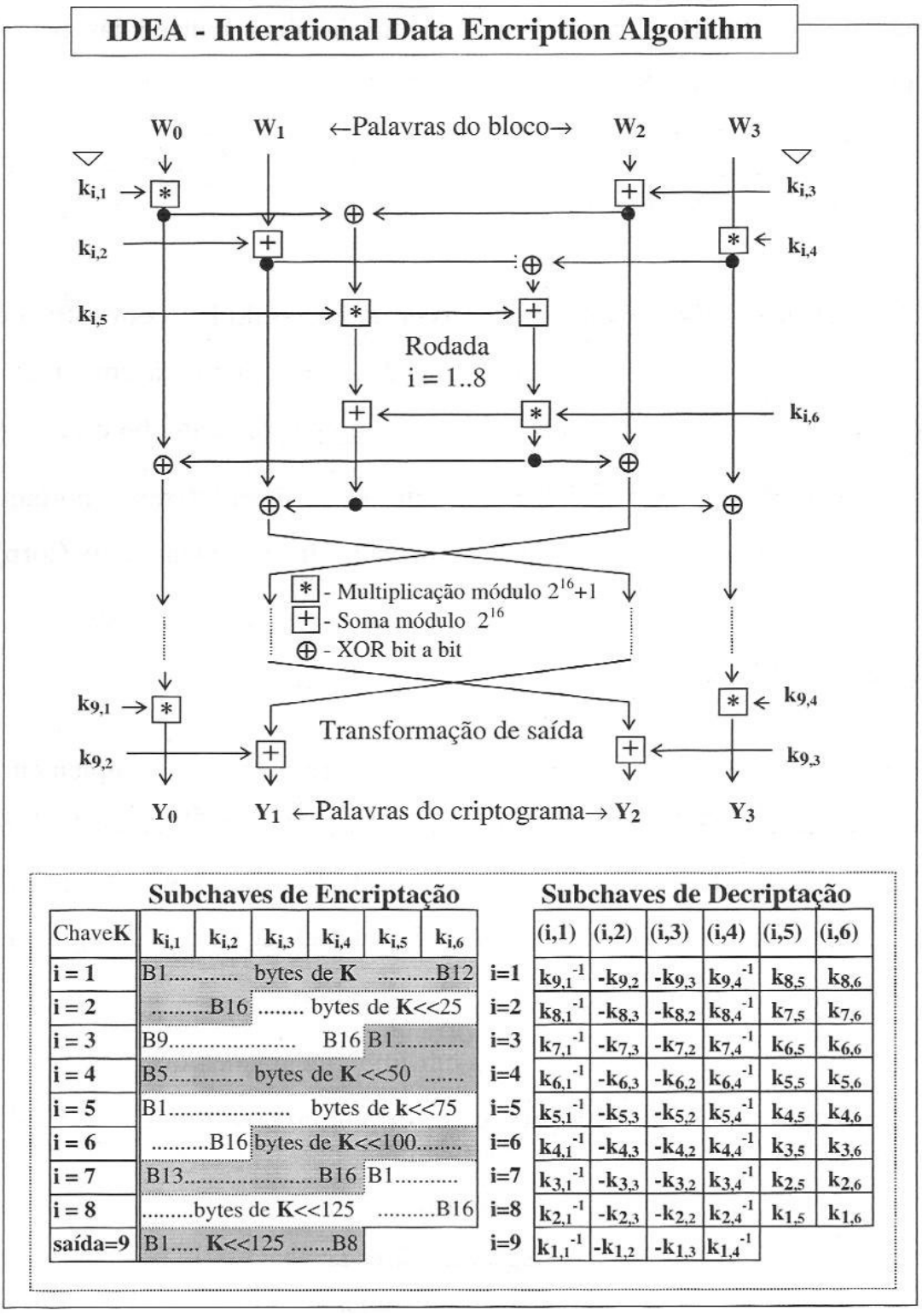
Alternativa ao algoritmo Kuhfu que usa S-*boxes* fixas e portanto sem *overhead* no tempo de execução. É patenteado (licença pela Xerox Corp)

- **RC2** (Ron Rivest, 1990) **-**

RC2 é um algoritmo proprietário, restrito aos implementadores licenciados pela RSA Data Security Inc. Para cifra de bloco de 64 bits. Não iterativo, usa chave de tamanho variável. (até 1024b). Em princípio é resistente à análise diferencial, linear e 3x mais eficiente em SW que o DES.

Um acordo entre a Software Publishers Assoc. e o governo dos EUA autoriza a exportação do algoritmo em implementações que usam 40 bits da chave, empregado em vários protocolos criptográficos para redes TCP/IP.

- **IDEA** (Xuejia Lai & James Massey, 1991) **-**



Subchaves de Encriptação

ChaveK	$k_{i,1}$	$k_{i,2}$	$k_{i,3}$	$k_{i,4}$	$k_{i,5}$	$k_{i,6}$
i = 1	B1..... bytes de **K** B12					
i = 2	.....B16 ..... bytes de **K**<<25					
i = 3	B9..... B16 B1.....					
i = 4	B5..... bytes de **K** <<50					
i = 5	B1..... bytes de **k**<<75					
i = 6	.....B16 bytes de **K**<<100.....					
i = 7	B13.....B16 B1.....					
i = 8	.....bytes de **K**<<125 .....B16					
saída=9	B1..... **K**<<125 .....B8					

Subchaves de Decriptação

	(i,1)	(i,2)	(i,3)	(i,4)	(i,5)	(i,6)
i=1	$k_{9,1}^{-1}$	$-k_{9,2}$	$-k_{9,3}$	$k_{9,4}^{-1}$	$k_{8,5}$	$k_{8,6}$
i=2	$k_{8,1}^{-1}$	$-k_{8,3}$	$-k_{8,2}$	$k_{8,4}^{-1}$	$k_{7,5}$	$k_{7,6}$
i=3	$k_{7,1}^{-1}$	$-k_{7,3}$	$-k_{7,2}$	$k_{7,4}^{-1}$	$k_{6,5}$	$k_{6,6}$
i=4	$k_{6,1}^{-1}$	$-k_{6,3}$	$-k_{6,2}$	$k_{6,4}^{-1}$	$k_{5,5}$	$k_{5,6}$
i=5	$k_{5,1}^{-1}$	$-k_{5,3}$	$-k_{5,2}$	$k_{5,4}^{-1}$	$k_{4,5}$	$k_{4,6}$
i=6	$k_{4,1}^{-1}$	$-k_{4,3}$	$-k_{4,2}$	$k_{4,4}^{-1}$	$k_{3,5}$	$k_{3,6}$
i=7	$k_{3,1}^{-1}$	$-k_{3,3}$	$-k_{3,2}$	$k_{3,4}^{-1}$	$k_{2,5}$	$k_{2,6}$
i=8	$k_{2,1}^{-1}$	$-k_{2,3}$	$-k_{2,2}$	$k_{2,4}^{-1}$	$k_{1,5}$	$k_{1,6}$
i=9	$k_{1,1}^{-1}$	$-k_{1,2}$	$-k_{1,3}$	$k_{1,4}^{-1}$		

Algoritmo baseado em vasta fundamentação teórica, cifra de bloco de 64 bits iterativa com 8 rodadas e chave de 128 bits, alterna operações de 3 grupos algébricos de estruturas distintas e opera em palavras de 2 bytes.

- **Análise do IDEA -**

Algoritmo patenteado, projetado como modelo de cifras de Markov, para as quais a resistência à criptoanálise diferencial pode ser formalizada e quantificada. Pode ser usado em qualquer modo encadeado.

Vários criptólogos já analisaram a versão final do algoritmo (1992), sem nenhum ter divulgado alguma técnica descoberta que o enfraqueça. Há uma classe de chaves fracas (para ataques de texto pleno escolhido) com probabilidade menor que 2^{-96} de serem geradas ao acaso:

k= 0000 0000 00?0 0000 0000 0000 000? ???? ?000

Embora recente, talvez seja hoje a cifra simétrica mais robusta em uso. Implementações do padrão de 8 rodadas em software são em geral 2x mais eficientes que o DES. Com 4 rodadas são aparentemente seguras e dobram a eficiência. Não pode ser expandido para palvras de 32 bits, porque o IDEA explora o fato de $2^{16}+1$ ser primo, enquanto $2^{32}+1$ não é.

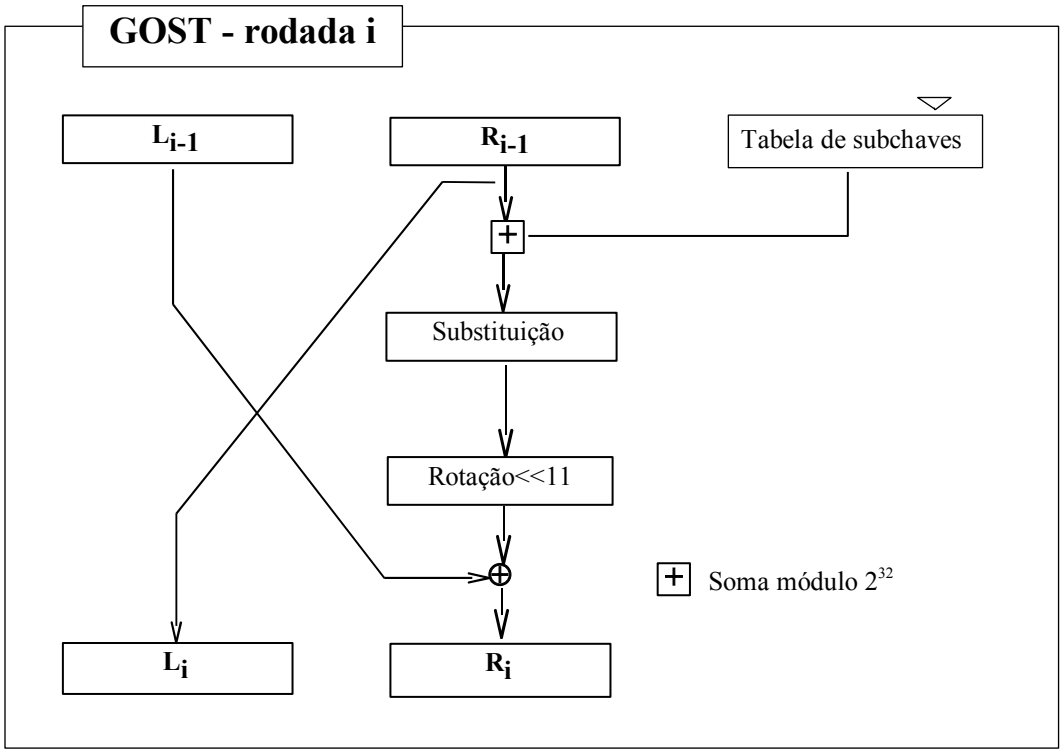
Pode substituir o DES sem muitas modificações em implementações de serviços criptográficos. É mais conhecido por ter sido escolhido para uma implementação do módulo *shareware* de segurança de e-mail, o PGP. IDEA é patenteado na Europa e nos EUA, sendo livre de royalties para implementações sem fins comerciais. Licenças para uso comercial são negociadas por Ascom Sistec AG, Mägenwil, Suíça: idea@ascom.ch

- **MMB** (John Daemen, 1993) **-**

Algoritmo baseado no IDEA, com blocos e chave de 128 bits e multiplicação módulo $2^{32}-1$. É vulnerável à criptoanálise linear e ao ataque de chave escolhida de Biham. É também menos eficiente que o DES para implementações em hardware, embora seja eficiente em software.

- **GOST** (USSR Gosudarstvenyi Standard, 1989) **-**

Padrão de algoritmo criptográfico estabelecido pelo governo da (ex-) União Soviética para cifra de bloco de 64 bits, semelhante ao DES. É uma rede de Feistel iterativa com 32 rodadas que usa chave de 256 bits.



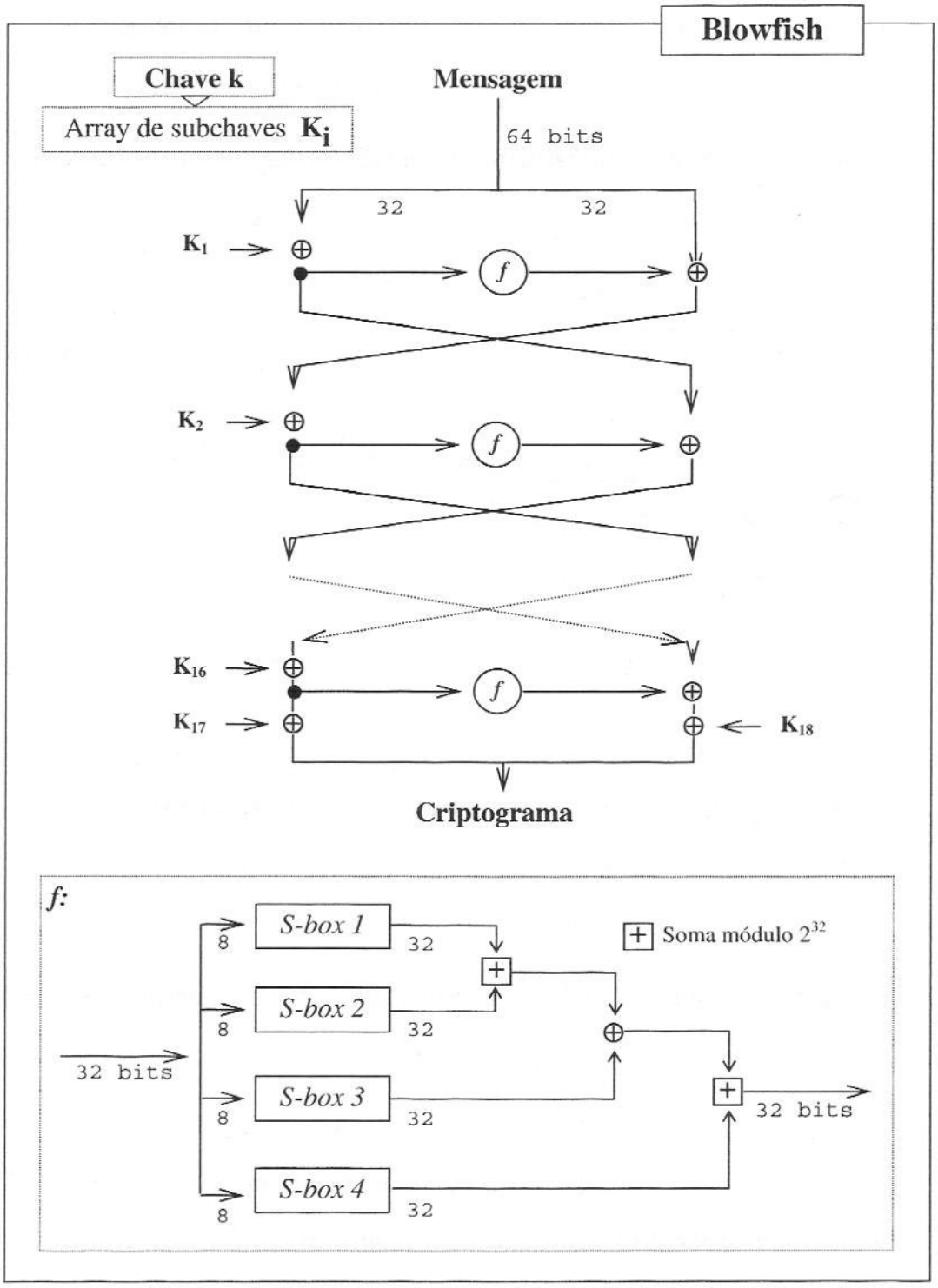
As subchaves são palavras da chave, usadas circularmente: na encriptação no sentido horário até a rodada 24, anti-horário nas rodadas 25 a 32; na decriptação, as subchaves são usadas no sentido inverso.

- **Análise do GOST -**

O DES usa permutações em ***f*** para aumentar a difusão (efeito avalanche da cifra, que propaga a influencia de qualquer bit de entrada em qualquer bit de saída), enquanto o GOST usa um grande número de rodadas, o que também contribui, junto com o tamanho da chave e a ocultação das S-*boxes*, para neutralizar sua análise diferencial e linear.

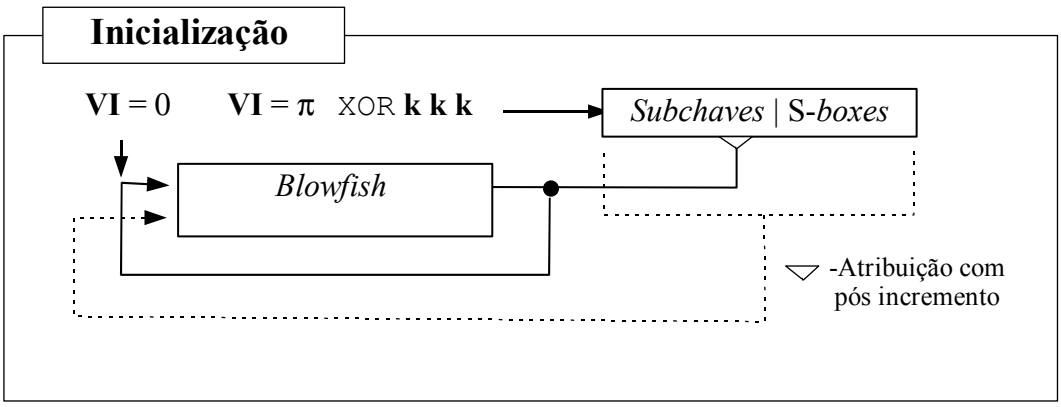
- **Blowfish** (Bruce Schneier, 1994) **-**

Cifra de bloco de 64 bits e chave com tamanho variável de até 448 bits. É uma *rede de Feistel* iterativa de 16 rodadas que usa XOR, consulta a tabela e adição na função ***f***, onde as 4 S-*boxes* 8x32 e o vetor de 18 subchaves são gerados em iterações de inicialização do próprio algoritmo.



- **Inicialização e análise do Blowfish -**

  - Algoritmo não patenteado, de uso inteiramente livre (o autor acredita na criptografia como patrimônio intelectual coletivo da ciência).
  - As subchaves e S-boxes são derivadas na seguinte inicialização:
    1. Os vetores de subchaves e depois os das S-*boxes* são carregados, na ordem em que indexados, com o XOR bit a bit da representação binária do número π e da chave repetidamente concatenada.
    2. A partir de um bloco de zeros na entrada, o algoritmo é iterado com o criptograma, substituindo sucessivas subchaves ou blocos de 8 bytes das S-*boxes* e realimentando a entrada (512 iterações).



  - Chaves fracas (que geram S-boxes duplicadas) podem ser criadas, com probabilidade de 2^{-14}. Estas chaves não podem ser identificadas antes da expansão inicializadora, mas até o momento não se conhece forma de explorá-las com técnicas de análise diferencial e linear.
  - O nº de mensagens escolhidas necessárias para este ataque é da ordem de $2^{8\mathbf{r}+1}$ onde **r** = nº de rodadas da implementação. ($2^{4\mathbf{r}+1}$ com chaves fracas)
  - Em implementações otimizadas para processadores de 32 bits que carregam os vetores em cache, a encriptação de cada bloco pode ser feita em 26 ciclos de *clock* do processador, usando 5K de memória RAM

- **RC5 w/r/b** (Ron Rivest, 1995) **-**

Algoritmos para uma família de cifras de bloco onde o nº de palavras **w** do bloco, bytes **b** da chave e número de rodadas **r** são configuráveis. Projetado para implementação em software, opera com XOR, rotações variáveis e aritmética em 32 bits. Sua patente foi solicitada pela RSA Data Security Inc., que promete vir a cobrar royalties modestos pela licença.

As subchaves $\mathbf{S_i}$ são geradas na inicialização. No caso **w** = 2, o bloco é separado em palavras **A** e **B**, e os algoritmos da cifra são

Encriptação RC5 2/r/b:

$A = A+S_0$
$B = B+S_1$
for i = 1 to r do begin
    $A = ((A \oplus B)<<<B)+S_{2i}$
    $B = ((A \oplus B)<<<A)+S_{2i+1}$
end
{">>>" = shift circular p/ direita}

Decriptação RC5 2/r/b:

for i = r downto 1 do begin
    $B = ((B-S_{2i+1})>>>A) \oplus A$
    $A = ((A-S_{2i})>>>B) \oplus B$
end
$B = B+S_1$
$A = A+S_0$
{"<<<" = shift circular p/ esquerda}

- **Inicialização do RC5 w/r/b -**

A chave é copiada para um vetor **L** de inteiros longos $\mathbf{L_1}...\mathbf{L_m}$ usando a convenção *little endian*. O vetor **S** de subchaves é inicializado com iterações a partir de **P** e **Q,** respectivamente, com bits da representação binária de π e da constante neperiana 'e', e depois mesclado com **L** conforme a especificação:

Carga:

{P=0Xb7e15163}
$S_0 = P$ {Q=0x9e3779b9}
for i = 1 to 2*(r+1)–1 do
    $S_i = (S_{i-1}+Q) \bmod 2^{32}$
i = j = 0
A = B = 0
n = max (2*(r+1), m)

Mescla da chave:

do n times
    $A = S_i = (S_i+A+B) >>>3$
    $B = L_j = (L_j+A+B)>>>(A+B)$
    i = (i+1) mod 2*(r+1)
    j = (j+1) mod m

- **Skipjack** (NIST, 1990) **-**

  - Algoritmo patenteado e restrito, destinado apenas às funções de cifra em hardware resistente a violações. Projetado para implementação nos chips "Clipper", e "Capstone", que incorporam a funcionalidade necessária para uso em protocolos de chaves escrituradas.
  - Foi desenvolvido em 1985 e testado até 1990 pela NSA, e seus detalhes de especificação nunca foram divulgados, exceto algumas de suas propriedades básicas: para cifra de bloco de 64 bits, com chave de 80 bits e iterativo com 32 rodadas.
  - Seus critérios de projeto, teste, e resultados dos testes foram avaliados por um painel de criptólogos não governamentais que o consideraram seguro, estimando para só daqui a 30 anos a equiparação do custo de ataque por força bruta ao SKIPJACK ao custo atual de quebra do DES.

- **Alguns Algoritmos menos conhecidos -**

  1. **FEAL** (1987) - Patenteado pela NNT do Japão, semelhante ao DES com número variável de rodadas e rápido, teoricamente mais frágil que o DES. Algoritmo favorito para teste de novas técnicas de criptoanálise
  2. **LOKI** (1991) - Algoritmo australiano patenteado, semelhante ao DES.
  3. **CA-1.1** (1992) - Algoritmo francês baseado em automata celulares, para cifra de bloco de 384 e chave de 1088 bits em processamento paralelo. Patenteado e livre para uso não comercial (H Gutowitz)
  4. **CAST** (1993) - Patente pendente no Canadá, considerado para adoção como padrão. Usa bloco e chave de 64 bits, 6 S-*boxes* (8x32) geradas da chave. Resistente à análise diferencial e linear. (Adams & Tavares)

Critérios de projeto para cifras de bloco

- **Princípios básicos** (Shannon, 1949) **-**

1. **Difusão** - Espalhamento da influência de bits individuais da chave ou mensagem através da maior parte possível do criptograma.
2. **Confusão** - Ocultação da relação entre mensagem, criptograma e chave no sentido de tornar complexa sua análise estatística.

- **Caso ideal versus caso viável -**

  - A confusão é uma propriedade suficiente para as cifras de bloco serem seguras, e teoricamente tais cifras existem em abundância, já que são apenas permutações do conjunto das cadeias de bits que formam blocos
  - Por outro lado, um algoritmo contendo somente uma S-*box* 64x64 ocuparia $2^{64} * 64$ bits (~ dez mil terabytes) de memória. Na prática intercala-se camadas de confusão (tabelas pequenas) e difusão, geralmente por meio de substituições e permutações, para construi-las.

- **Redes de Feistel** (Horst Feistel, 1973) **-**

  Desenho de intercalação onde a propriedade nilpotente da operação lógica de "ou" exclusivo ($\oplus$) é combinada com a permutação dos operandos para permitir a introdução de uma confusão qualquer na construção da classe de funções inversíveis que constituem a cifra.

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

$$\forall f \; [\; L_{i-1} = L_{i-1} \oplus f(R_{i-1}, K_i) \oplus f(R_{i-1}, K_i) \;]$$

- **Resistência à análise diferencial e linear** (Matsui, EUROCRIPT94)

  Parece haver certa dualidade entre estes dois métodos, tanto nas técnicas de exploração no ataque, como nos critérios de escolha de S-*boxes* que introduzam não-linearidade nas cifras para resistirem a estes ataques.

- **Critérios de definição para as S-*boxes* mxn -**

  1. Quanto maior a saída **n** em relação à entrada **m**, mais efetiva tende a ser a análise linear e menos efetiva a análise diferencial, e vice versa.
  2. Quanto maior as S-*boxes*, mais provável que sua substituição, se escolhida ao acaso, seja resistente aos dois métodos de análise.
  3. *S-boxes* fixas escolhidas para resistirem a estes dois métodos têm segurança desconhecida contra métodos de ataque desconhecido, ao contrário das S-*boxes* randômicas ou geradas a partir da chave.

- **Comparação do impacto no desempenho das escolhas na especificação das cifra** (486SX 33MHz, Schneier 1995) **-**

Velocidade de encriptação			
Algoritmo	KB/Seg	Algoritmo	KB/Seg
Blowfish (12 rodadas)	182	Khufu (16 rodadas)	221
Blowfish (16 rodadas)	135	Khufu (32 rodadas)	115
Blowfish (20 rodadas)	110	NewDES	223
DES	35	RC5 bloco32/8rodadas	127
FEAL (8 rodadas)	300	RC5 32/12	86
FEAL (16 rodadas)	161	RC5 32/16	65
FEAL (32 rodadas)	91	RC5 32/20	52
GOST	53	Triple DES	12

Cifras Encadeadas

- **RC4** (RSA Data Security, 1987) **-**

Algoritmo restrito até 1994, quando teve sua especificação publicada anonimamente na lista de mensagens *Cypherpunk*. Para cifras encadeadas de bytes em modo OFB, usa uma S-*box* 8x8 **S** contendo permutação $\mathbf{S_0}...\mathbf{S_{255}}$ dos valores hexa `00`,..,`FF`, e dois contadores de bytes **m, n**.

Geração da seqüência de *pad:* k

repeat
    n = (n+1) mod 256
    m = (m+ S_n) mod 256
    Troca (S_m ,S_n)
    t = (S_m +S_n) mod 256
    k= S_t

Inicialização de S com a chave semente concatenada = $\mathbf{k_1}...\mathbf{k_{255}}$:

for n=0 to 255 do S_n = n
m=0
for n=0 to 255 do
    m = (m+S_n +k_n) mod 256
    Troca (S_m ,S_n)

- **Análise do RC4 -**

1. A cifra encadeada em modo OFB encripta ou decripta executando o XOR de cada byte **k** gerado pelo algoritmo, com cada byte da mensagem ou do criptograma. A chave semente pode ter até 1024 bits
2. Há $256!*256^2 \approx 2^{1700}$ estados possíveis para o gerador de chaves de *padding* deste algoritmo. RSA afirma parecer não haver sementes $\mathbf{k_1}...\mathbf{k_{255}}$ que gerem ciclos pequenos, e ser o algoritmo imune às análises diferencial e linear. Licenciado para exportação com **H** = 40.
3. Usado em dezenas de produtos (Lotus Notes, Apple AOCE, Oracle Secure SQL, CDPD, etc.) Em princípio pode ser adaptado de 8 para 16 bits, tornando-o mais rápido, mas com 2^{16} iterações de inicialização e ~100K de RAM ocupados pela S-*box* 16x16.

- **SEAL** (Don Coppersmith, 1994) **-**

Algoritmo para construção de famílias de geradores pseudo-randômicos, que pode tanto ser usado para cifras encadeadas como adaptado para cifras de bloco em modo não seqüencial (ECB).

Usa uma chave de 160 bits para inicializar três S-*boxes* 9x32, um índice **n** semelhante ao **VI** das cifras de bloco e quatro registradores de 32 bits, para gerar iterativamente uma seqüência randômica de até 64KB, que pode funcionar como chave de *padding* do **n**-ésimo bloco de uma cifra.

De arquitetura inovadora, foi projetado para implementação eficiente em processadores de 32 bits. Adota as seguintes estratégias:

A chave grande é usada apenas para derivar as três S-boxes.

O índice é usado na mistura de duas S-boxes para escolha da mudança nos registradores de iteração, e na mistura de outras duas para escolha da sequência de operações XOR ou soma dentro função de mistura de cada rodada da iteração.

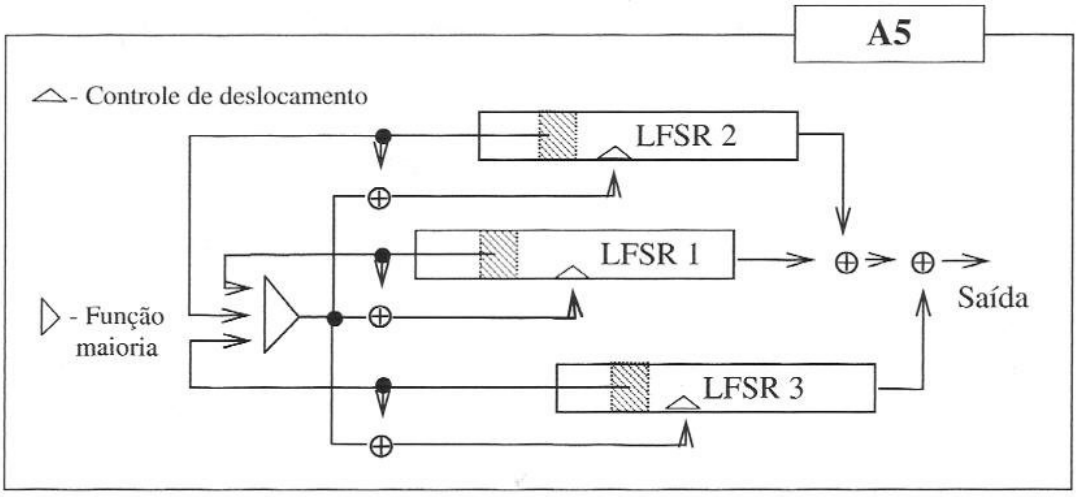
Estados internos mantidos em metade dos registradores não se manifestam na cadeia de saída, sendo usados apenas para modificar o estado dos registradores de iteração.

- **Análise do SEAL -**

Requer 5 operações de máquina para gerar cada byte de *padding*, alcançando 50 MB/seg num PC 486 de 50 MHz. Parece sólido, e até 1996 não havia sido publicado nenhuma criptoanálise independente do algoritmo. Patente pendente e licença para implementação pela IBM.

- **A5 -**

Algoritmo adotado pelo consórcio mundial de telefonia móvel GSM (*Global System for Mobile communication*) para cifragem no elo entre o telefone e a central de comutação. Projetado na França, é uma variação da composição de LFSRs tipo "para e segue" com três registradores de deslocamento linear baseados em polinômios esparsos de grau 19, 22 e 23.



- **Análise do A5 -**

Houve uma disputa durante os anos 80 entre os membros do GSM, sobre o grau de robustez desejada para o padrão (a Alemanha queria criptografia forte, mas sua opinião não prevaleceu). O algoritmo deveria ser restrito, mas um acordo do GAT com *Bradford University* para avaliação do algoritmo omitiu a restrição, e o A5 se tornou público.

Há um ataque simples ao algoritmo que requer 2^{40} encriptações. O algoritmo passa em todos os testes de randomicidade conhecidos, estando sua fragilidade concentrada nos pequenos tamanhos dos registradores e na rarefação dos bits de captura dos polinômios irredutíveis escolhidos.

O algoritmo é bastante eficiente.

- **PKZIP** (Roger Schafly) **-**

A cifragem (não a compressão) neste utilitário, se presentes nas versões até 2.04g, é por cifra encadeada em modo CBC que gera bytes.

O algoritmo da cifra usa uma chave $\mathbf{K_3}$ de 8 bits e registradores de 32 bits $\mathbf{K_0}$, $\mathbf{K_1}$ e $\mathbf{K_2}$, que armazenam o estado interno do gerador, atualizados com o uso de uma tabela de 256 bytes pré-computada, em iterações onde o CRC dos 32 bits anteriores é calculado pelo polinômio `0xedb88320`. Um vetor de inicalização é concatenado ao início da mensagem. Na decriptação, invertem-se criptograma e mensagem no *padding*

```
/*buffer de mensagem M[i], de criptograma C[i] */
    int K0 = 305419896
    int K1 = 591751049
    int K2 = 878082192
    for(i=1, ,i++) {
     C[i] = M[i]^K3;            /* padding */
     K0 = crc32(K0,M[i]);
     K1 = K1*134775813+1;
     K2 = crc32(K2,K1>>24);
     K3 = ((K2|2)*(K2|2)^1))>>8
}
/*crc32(a,b)=(a>>8)^tabela[(a&0xff)^b]*/
```

- **Análise do PKZIP -**

Algoritmo bastante frágil. Um ataque simples de dicionário com 40 a 200 bytes de texto pleno conhecido (cabeçalho de mensagens, por exemplo) desvela a chave em $\sim 2^{27}$ operações, ou algumas horas num PC.

Cript(1) (Unix, 1983) **-**

Algoritmo original de encriptação das primeiras versões do sistema operacional Unix, é uma cifra encadeada baseada na mesma arquitetura da máquina eletromecânica, *Enigma*, usada pelos militares e diplomatas alemães da segunda guerra e quebrada pela equipe inglesa de analistas, liderada por Alan M. Turing.

O algoritmo simula um rotor de 256 elemetos de substituição usado em série com um rotor refletor (a máquina *Enigma* usava 3 dentre cinco rotores de substituição, mais um rotor de reflexão).

Para um analista bem instrumentado, esta cifra é fácil de atacar. Um utilitário de domínio público, o *Crypt Breakers Workbench* (CBW), pode ser usado para quebrar arquivos encriptados com o **Cript(1)**.

RAMBUTAN (Communications Eletronics Security Group) **-**

Algoritmo restrito, vendido apenas em implementações em hardware invioláveis sob licença do governo Britânico para aplicações classificadas como "Confidential", não sendo encontrado no varejo. Usa chave de 112 bits e pode operar nos modos ECB, CBC e CFB de 8 bits.

XPD/KPD (Hughes Aircraft Corporation, 1986) **-**

Algoritmo usado em equipamentos de comunicação e rastreamento de aeronaves vendidas a países estrangeiros aos EUA. Usa um LFSR de 61 bits inicializado com bits de captura de um dentre 2^{10} polinômios primitivos armazenados em ROM, e oito filtros não-lineares na saída.

Funções de Hash

- **Premissas de um hash ou *checksum* criptográfico seguro -**

Como mecanismo principal na autenticação da integridade de dados, funções de hash $\mathbf{h}: M \rightarrow \{0,1\}^n$ devem satisfazer as propriedades:

1. **Propriedades básicas** (unidirecional e livre de colisão)
   - Dado **m**, é fácil calcular **c = h(m)** e dado **c** é difícil calcular **m**;
   - Dado **m** qualquer, é difícil produzir uma colisão com **m** (encontrar **m'** tal que **h(m) = h(m')**). O valor de **n** deve dificultar a produção de colisão através de ataques por dicionário em **m'**.

2. **Propriedade adicional** (resistência à colisão)
   - Para algumas aplicações, a função de hash deve ser *resistente a colisão*: deve ser difícil encontrar, por meio de ataques "de aniversário" ou técnica melhor, qualquer colisão na função **h**.

- **Ataques de aniversário a funções de hash -**

Descrição: Encontrar um par **m** e **m'** tal que **h(m) = h(m')**.

Exemplo: Ao propor um contrato (protocolo tipo VII), o fraudador prepara outra versão que lhe seja vantajosa e prejudicial à outra parte.

Testa o hash de variações de ambas versões, contendo combinações de espaços adicionais (ASCII 32) onde há quebras de linha, até conseguir um par **m** (contrato), **m'** (fraude) que gerem mesmo hash. Assinam o hash de **m** e depois, em juízo, o fraudador protesta **m'**.

Construção de funções de hash

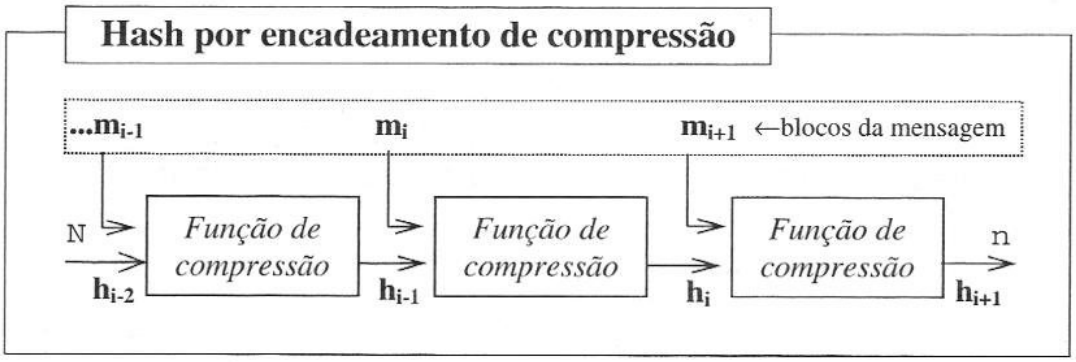
- **Tamanho do hash -**

A entropia **n** da imagem da função de hash $\mathbf{h}: M \rightarrow \{0,1\}^n$ deve ser maior que do espaço de chaves das cifras (ao menos o dobro, $\mathbf{n} \geq 128$ bits)

Se n = nº de testes no ataque por dicionário com probabilidade = 1/2 de acerto, $\approx n^{1/2}$ testes no ataque de aniversário terão igual probabilidade.

- **Encadeamentos de função compressora -**

As funções projetadas para hash, hoje em uso são construídas encandeando-se alguma função de compressão (com entrada e saída de tamanhos fixos, N e n respectivamente), com entrada de blocos da mensagem e valores de compressão retroalimentados.



A mensagem é formatada como concatenação de blocos de compressão, de tamanho $N-n$, e a saída da função de hash (digesto da mensagem) é a última saída da função de compressão. Para melhorar a resistência à colisão, inclui-se $|\mathbf{m}|$ no enchimento do último bloco de **m**.

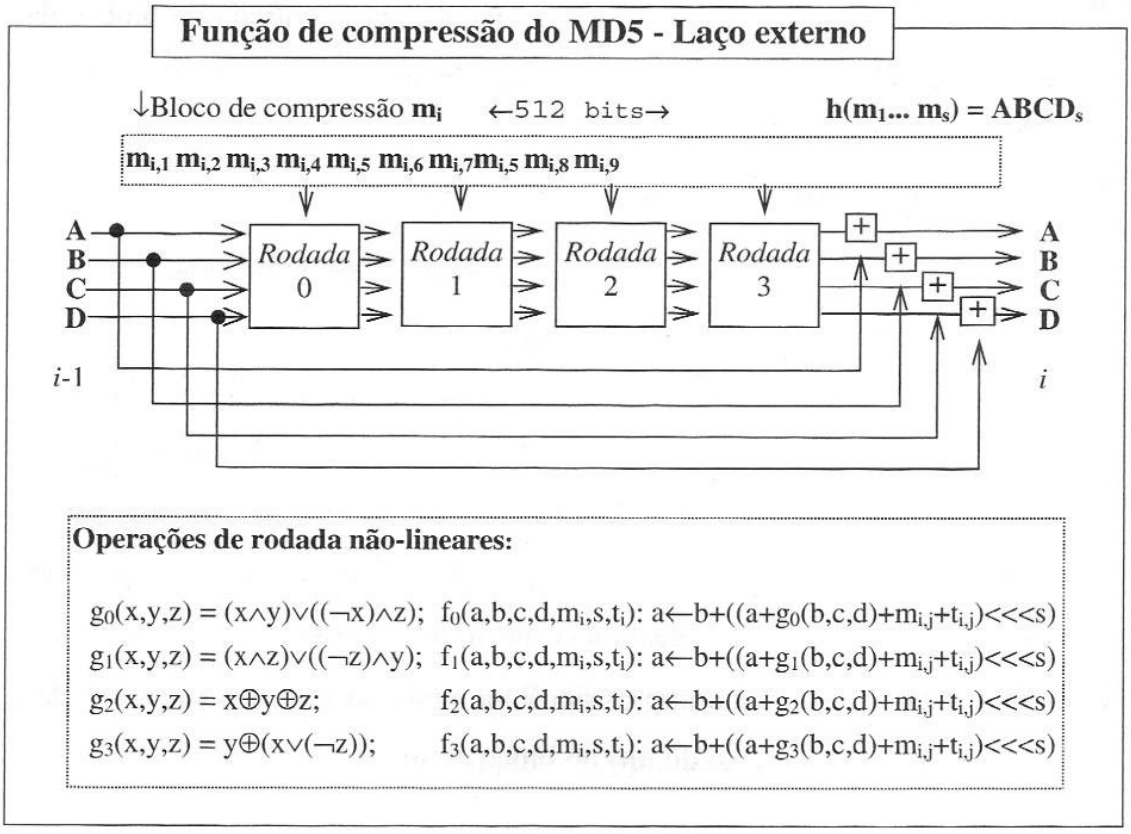
MD5

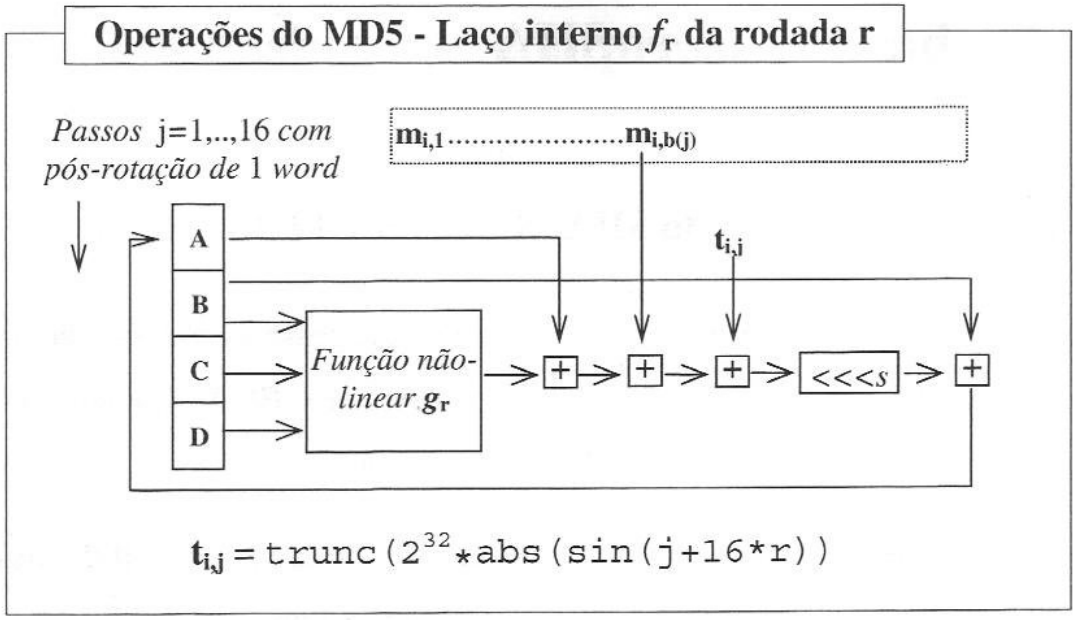
- **Descrição do algoritmo MD5** (Ron Rivest, 1992) **-**

Hash de 128 bits, é a mais recente da série de funções de hash desenvolvidas pela RSA Data Security. MD5 reforça a função anterior MD4, depois da descoberta de formas de ataque a algumas partes desta.

A mensagem é preparada apondo-se sufixo com zeros seguidos da representação binária do comprimento da mensagem original em 64 bits, formando blocos de compressão com 16 subblocos de *words* (32 bits cada)

Nas rodadas i = 0,..,3 (uma a mais que MD4), uma operação não-linear f_i é executada 16 vezes, cada execução envolvendo três das quatro variáveis de encadeamento **A**, **B**, **C** e **D**, um subbloco e duas constantes distintas.





- **Análise da função MD5 -**

O subbloco $\mathbf{m_{i,b(j)}}$ é escolhido segundo tabela b(j), distinta para cada **r**.

Um ano após sua divulgação, den Boer e Bosselaers encontraram uma forma de derivação de colisões mais eficiente que o ataque de aniversário para a função de compressão do MD5. Esta fragilidade na resistência à colisão não tem impacto nas propriedades básicas do hash. (Eurocrypt '93)

- **MD2 -**

Outra função de hash do mesmo autor, usada alternativamente ao algoritmo MD5 nos protocolos PEM para correio eletrônico. É baseada na permutação randômica de bytes, semelhante à cifra encadeada RC4.

É também um hash de 128 bits, onde a função de compressão recebe blocos de 128 bits e a mensagem é preparada para resistir à colisão com um sufixo *checksum* da mensagem de 16 bytes. Embora pareça segura, é mais lenta que a maioria das funções de hash em uso hoje.

SHA

- **Descrição do algoritmo SHA** (NIST - NSA, 1992) **-**

Secure Hash Algorithm, de 160 bits, proposto como padrão para o protocolo de assinatura digital DSA do governo dos EUA. Também baseado em variações no algoritmo MD4, resiste ao ataque de De Boer.

A mensagem é preparada para hash como no algoritmo MD5, mas com um pre-processamento adicional dos subblocos, que são expandidos de 16 $\mathbf{m}_{i,j}$ para 80 *words* $\mathbf{w}_{i,j}$ de 32 bits, através da seguinte rotina:

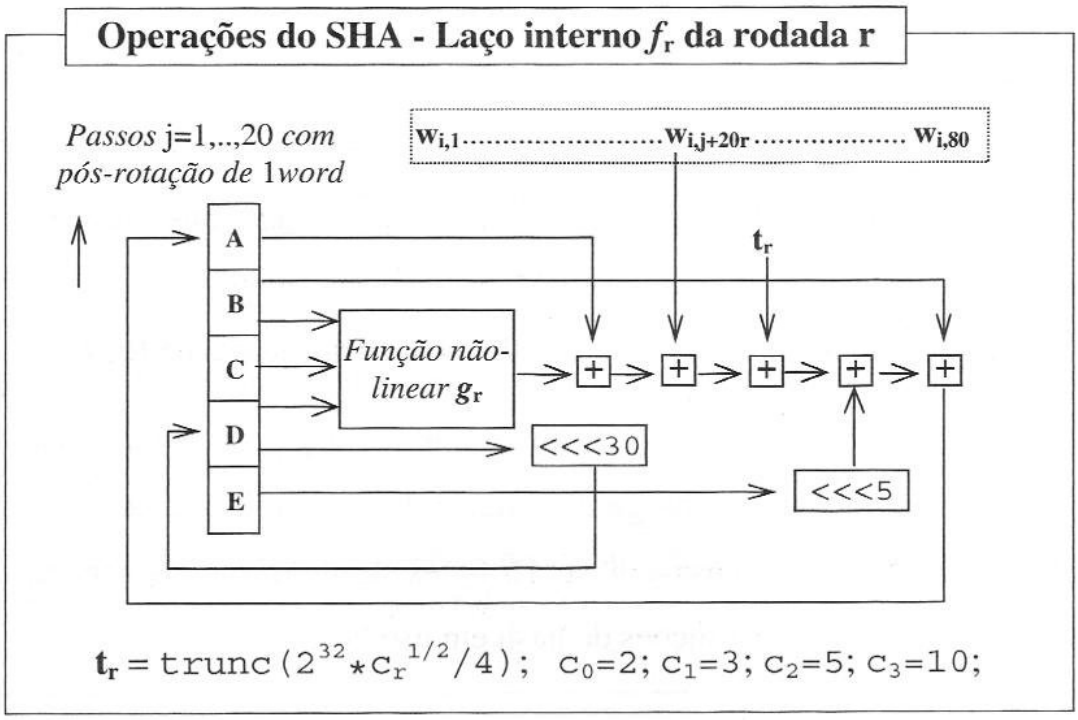
Geração dos subblocos para compressão

for j = 1 to 16 $w_{i,j} = m_{i,j}$

for j = 1 to 16

$w_{i,j} = (w_{i,j-3} \oplus w_{i,j-8} \oplus w_{i,j-14} \oplus w_{i,j-16})<<<1$

Cada uma das 4 rodadas executa 20 operações semelhantes às do MD4, e o aumento na entropia é obtido com uma quinta variável de encadeamento, **E**.



Outros hashes por compressão encadeada

- **RIPE-MD** (RACE - Primitives Evaluation Effort) **-**

Padrão para algoritmo de hash proposto pelo grupo RACE (*Research & Dev. in Advanced Communications Tecnologies in Europe*), de 128 bits.

Duas versões da função de compressão semelhantes ao MD5, com rotações e constantes de operações distintas, executam em paralelo e os resultados são somados na retroalimentação das variáveis de encadeamento, para dificultar a criptoanálise do algoritmo.

Em 1997 foi divulgado uma versão de 160 bits, o **RIPE-MD160**, que tem as duas funções de compressão paralelas semelhantes à do SHA.

- **HAVAL** (Zheng, Pieprick & Seberry, 1993) **-**

Modificação de MD5, com saída de tamanho variável: 128, 160, 192, 224 ou 256 bits. Usa 8 variáveis de encadeamento, número variável de rodadas (3 a 5) de 16 operações, com funções não-linerares de 7 variáveis. Rotações em dois sentidos impedem o ataque de De Boer à compressão.

- **Algoritmos Evitáveis -**

Knapsack hash (Damgard, 1989): quebrável em 2^{32} operações;
Cellular automata hash (Wolfram, 1991): inseguro;
Fast Fourier transform hash (Schnorr, 1992): muito lento;
Galois field GF(2^{593}) hash (U. of Waterloo, 1992): lento.

Hash usando algoritmos para cifra de bloco

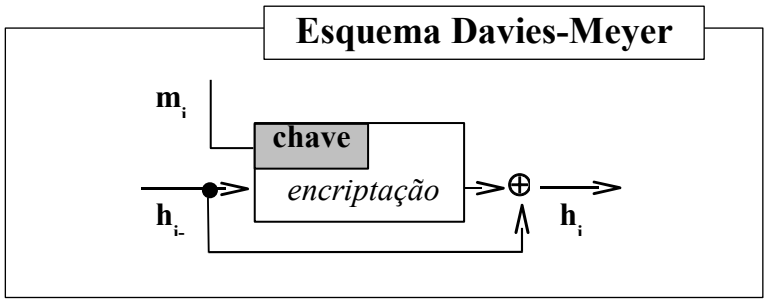
- **Casos para adaptação -**

1. Código de autenticação de mensagens (**MAC**): Em princípio, qualquer algoritmo criptográfico para cifra de bloco poderia fornecer, em modo CBC ou CFB, o último bloco do criptograma como autenticação da mensagem. Neste caso, existem outros tipos de ataque para fraudes.

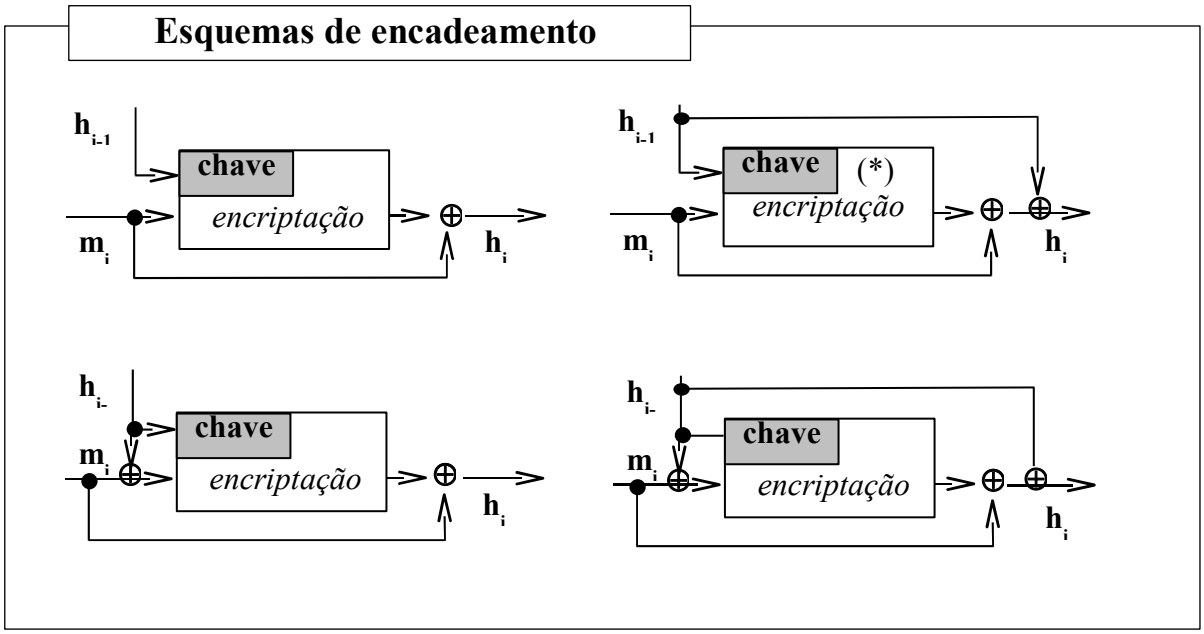
2. Função de **hash**: Um algoritmo criptográfico é mais vulnerável em um hash que em uma cifra. Como no hash a chave é conhecida, vários truques podem ser usados para explorar com mais eficiência a análise diferencial, e a escolha de texto pleno não apresenta dificuldades práticas.

- **Adaptações para hash de mesmo comprimento do bloco -**

Algoritmos criptográficos superdimensionados ou de bloco longo podem ser adaptados para construção de hash, com diferentes esquemas de encadeamento, se o tamanho do bloco previne ataques por aniversário.



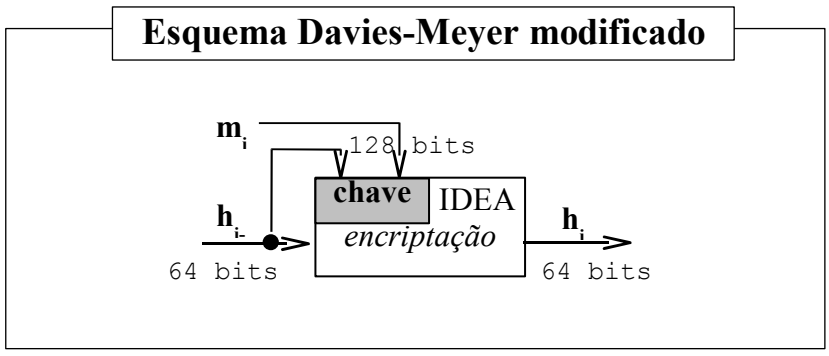
Das 64 possíveis combinações que mapeiam $\mathbf{m_i}$, $\mathbf{h_{i-1}}$, $(\mathbf{m_i} \oplus \mathbf{h_{i-1}})$ e $\mathbf{h_0}$ (Vetor de inicialização) no algoritmo, 13 são impróprios, 37 inseguros, 8 são seguros contra ataques conhecidos exceto o de ponto fixo, e 4 deles, descritos abaixo, são hoje seguros. (B. Preneel, U. Leuven, 1993).



- **Análise dos esquemas de encadeamento -**

Estes quatro esquemas supõem que o algoritmo criptográfico tenha o tamanho do bloco idêntico ao da chave, que será o tamanho do hash. O segundo esquema acima, (*) foi proposto como padrão ISO para hash baseado em algoritmo criptográfico (ISO-IEC/JTC1/SC27/WG2).

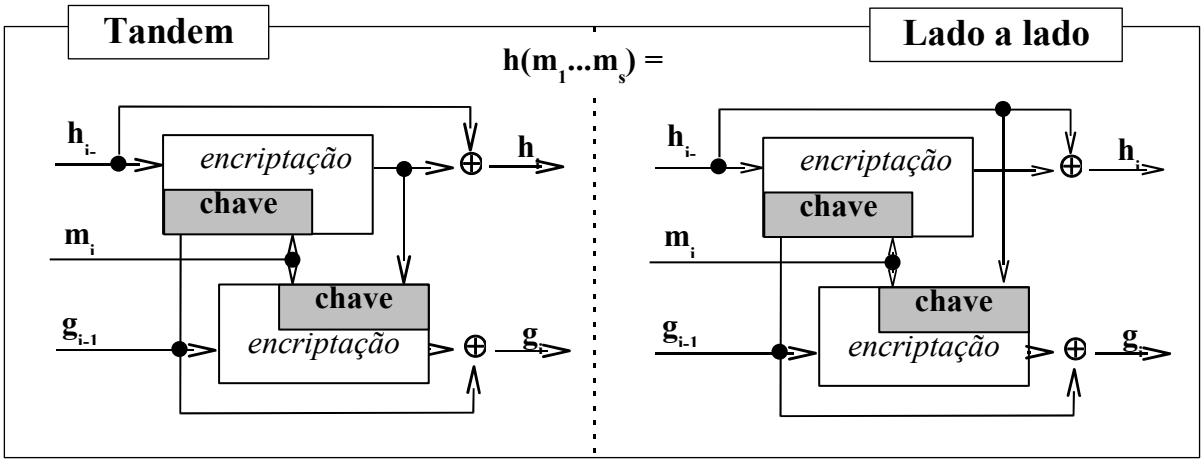
O esquema de Davies-Meyer foi modificado por Lai e Massey para usar o algoritmo IDEA na construção de um hash de 64 bits (Eurocript 92).



Para construção de hash de tamanho maior que o bloco do algoritmo, vários esquemas foram propostos e poucos têm se mostrado seguros.

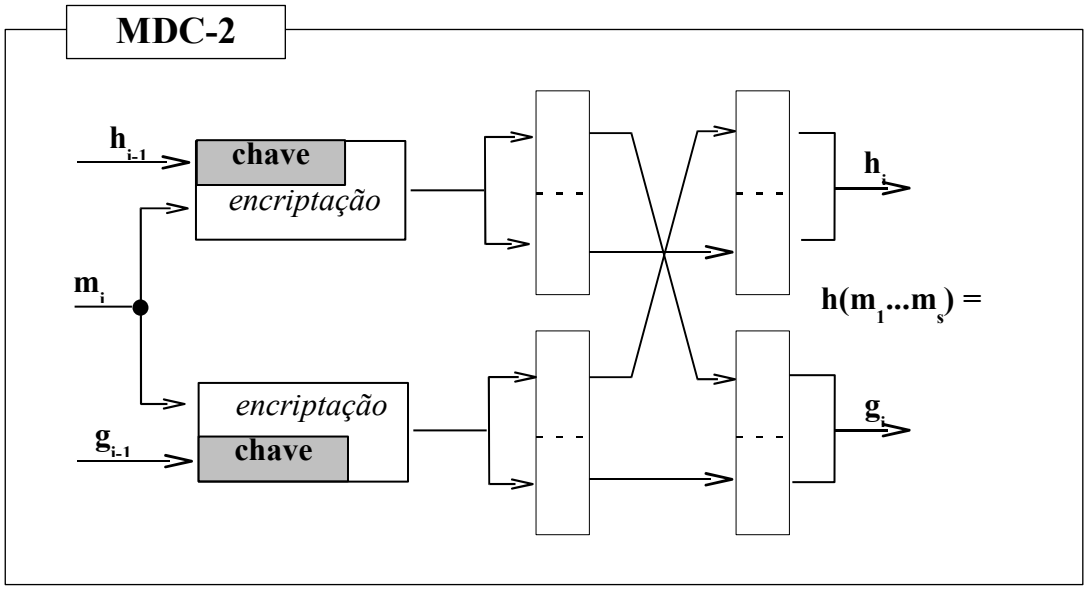
- **Esquema Davies-Meyer em tandem ou lado a lado -**

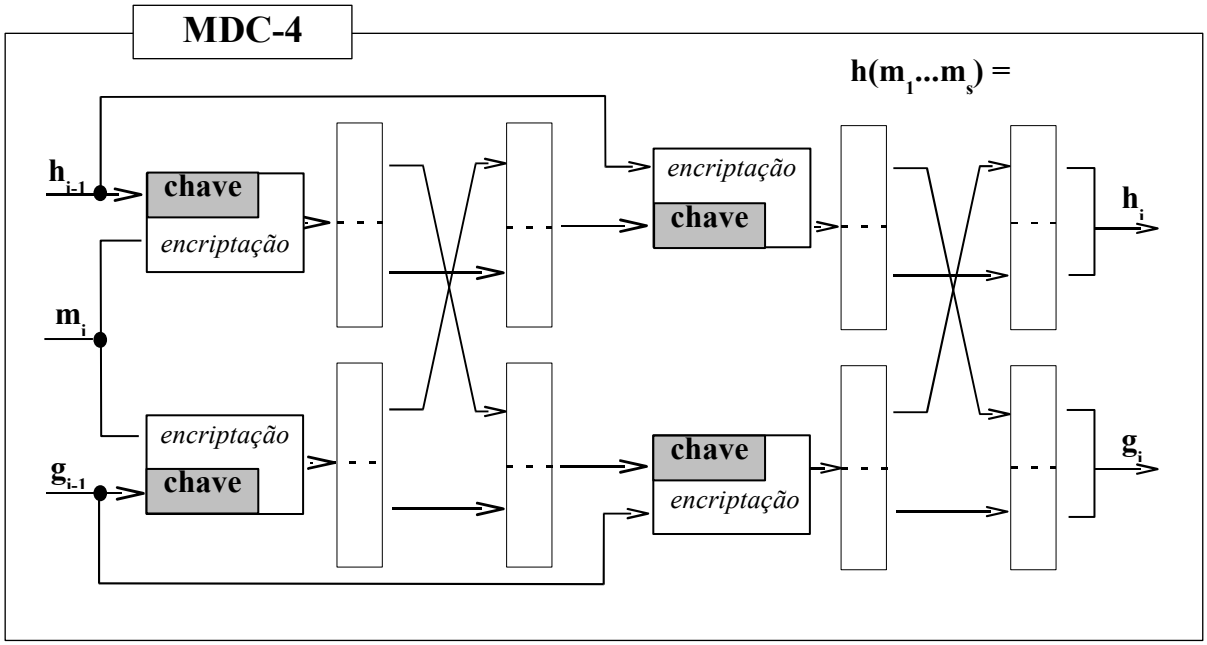
Estes esquemas têm se mostrado seguros para construção de hash que duplica o tamanho do bloco, com algoritmos criptográficos cuja chave tenha o dobro do tamanho do bloco (i.e., IDEA).



- **MDC-2 e MDC-4** (Merley-Schilling, IBM, 1988) **-**

Esquemas patenteados para construção de hash que duplicam o tamanho do bloco, sem restrições ao algoritmo criptográfico simétrico. Estão sendo considerados para padrão ANSI, ISO e avaliados pelo RIPE.





- **Esquemas inseguros -**

Preneel-Bosselaers-Govaerts, *Quaisquater-Girault*, *LOKI Double-block* e *Davies-Meyer paralelo.* Para estes esquemas foram recentemente descobertos métodos de ataque que os tornaram inseguros na criptografia. O segundo desses esquemas foi proposto como padrão ISO em 1989.

- **Comparação de performance** (Schneier, 80386 em 33 MHz) -

Algoritmo de hash	Comprimento	KB/Seg
Davies-Meyer (c/ DES)	64	9
Davies-Meyer lado a lado (c/ IDEA)	128	22
HAVAL (3 rodadas)	variável	168
HAVAL (4 rodadas)	variável	118
HAVAL (5 rodadas)	variável	95
MD2	128	23
MD4	128	236
MD5	128	174
RIPE-MD	128	182
SHA	160	75

Códigos de autenticação de mensagens

- **MACs usando cifras em modo CBC ou CFB -**

MACs são hashes dependentes de chave no cálculo, que podem ser gerados por uma cifra de bloco retroalimentada. O esquema que gera MACs encriptando a mensagem e depois o último bloco do criptograma em modo CBC, constitui os padrões ANSI (X9.9) e ISO (8731-1, 9797).

- **MACs usando funções de hash -**

Dada uma função de hash **h** qualquer, existem adaptações possíveis para torná-la dependente de chave. Dentre os esquemas que concatenam a mensagem **m** à chave **k**, os mais seguros contra fraudes são $\mathbf{h(k_1,h(k_2,m))}$ ou **h(k,_,m,k)**, ou concatenação de bytes da chave a cada bloco de **m**.

- **Análise dos esquemas de geração de MACs -**

MACs gerados por cifras apresentam um problema em potencial, no fato do verificador poder usar a chave para decriptar o hash de trás para frente, buscando construir uma colisão com a mensagem original.

Esquemas com funções de hash que geram MACs por concatenação de chave à mensagem podem permitir fraudes por quem não detém a chave mas conhece **h**. È mais seguro usar ambos, cifrando o hash da mensagem.

Usa-se MACs em situações onde a verificação da integridade, sem sigilo da mensagem, é necessária.

Algoritmos criptográficos de chave pública

- **Histórico -**

A descoberta em 1976 por Diffie, Hellman e Merkle de algoritmos criptográficos assimétricos, onde a segurança se baseia nas dificuldades de

1. Deduzir a mensagem a partir do criptograma;
2. Deduzir uma chave de cifragem a partir da outra chave;

possibilitou o desenvolvimento da criptografia moderna, onde os mecanismos de distribuição de chaves, autenticação de mensagens, e provas de identidade, alcançaram novos patamares de versatilidade.

Protocolos que fazem uso de algoritmos assimétricos, podem dispensar o sigilo de uma das chaves do par. Os que usam esta opção são chamados protocolos de **chave pública**, e devem ser bem projetados para serem seguros

- **Segurança dos sistemas de chave pública -**

Esses sistemas são desenhados para resistir a ataques de texto pleno escolhido, mas podem ser sensíveis a ataques por criptograma escolhido. Portanto, nos sistemas onde a assinatura é operação inversa da cifragem, pares distintos de chaves devem ser usados para estes dois serviços.

Dos algoritmos assimétricos até hoje propostos, apenas quatro são seguros e práticos para ambos serviços: **RSA**, **ElGamal**, **Rabin** e **ECC**. Existe uma família de algoritmos úteis apenas para assinatura, e outros pouco práticos por serem inseguros, muito lentos ou usarem chaves muito longas.

RSA

O mais usado e fácil de implementar dos algoritmos assimétricos, tem o nome formado com iniciais dos descobridores, Rivest, Shamir & Adleman. Resiste a quase 20 anos de análise, sendo sua segurança supostamente baseada na dificuldade de se fatorar números inteiros.

Geração de parâmetros e par de chaves do sistema: {t = tamanho}

p = geraprimo(rand(t))
q = geraprimo(rand(t))
ϕ = (p-1)*(q-1) **{p, q, ϕ secretos}**
n = p*q; e= rand(t)
e = e / mdc(e,ϕ) [>1] e_A **= (e,n)**
d = euclext(e,ϕ,1) d_A **= (d,n)**

Cifragem (começa com e_A pública)

$c_i = m_i^e \bmod n$ **{encripta bloco}**

$m_i = c_i^d \bmod n$ **{decripta bloco}**

Assinatura (começa com d_A privada)

$x = h(m)^d \bmod n$ **{assina hash}**

$h(m) = x^e \bmod n$? **{verifica hash}**

$d = e^{-1} \bmod \phi$: *A segunda chave de um par, inversa da primeira no anel $Z_{\phi(n)}$, é calculada pelo algoritmo de Euclides extendido*:

Algoritmo de Euclides extendido recursivo: Dados a, b, c onde mdc(a,b) divide c, retorna o menor x>0 tal que

```
                          /* a*x = c mod b */
euclext(a, b, c)  begin
     r = b mod a
     se r == 0
        entao retorne( (c div a) mod (b div a) )
        senao retorne( (euclext(r,a,-c)*b+c) div a mod b)
end
```

Fermat: *O algoritmo funciona devido ao Teorema de Fermat*:

$$c_i^d = (m_i^e)^d = m_i^{1+r(p-1)(q-1)} = m_i * m_i^{r(p-1)(q-1)} = m_i * 1 \bmod n$$

A cifra funciona formatando **m** em blocos $\mathbf{m_i}$ de representação binária < **n**.

Análise do RSA

- **Premissas sobre a segurança do algoritmo -**

1. Qualquer dos parâmetros **p**, **q** e **ϕ(n)** permite o cálculo trivial de $\mathbf{d_A}$ a partir de $\mathbf{e_A}$, devendo portanto serem protegidos juntamente com $\mathbf{d_A}$.
2. O ataque por força bruta mais eficiente ao algoritmo consiste em tentar fatorar **n** para se obter **ϕ(n)** e saber em que anel inverter $\mathbf{e_A}$. Pode-se também tentar adivinhar **ϕ(n)**, mas o custo deste ataque é tão alto quanto o de fatorar **n,** sendo maior ainda o custo de se tentar adivinhar $\mathbf{e_A^{-1}}$.
3. Em princípio, poderia existir um método de ataque mais eficiente ao RSA. Porém tal método serviria também para fatoração de **n**, e o problema da fatoração vem sendo extensamente estudado desde 340 A.C., sendo seu melhor algoritmo de complexidade exponencial, $O(e^{c+x_{1/3}\ln_2(x)})$.
4. Números randômicos são selecionados como primos por um algoritmo probabilístico, para o módulo **n**. Existem pseudo-primos, números que passam em todos estes testes sem serem primos (números de Carmichael) Pseudo-primos são muito raros e, se gerados, causarão falha na cifra.

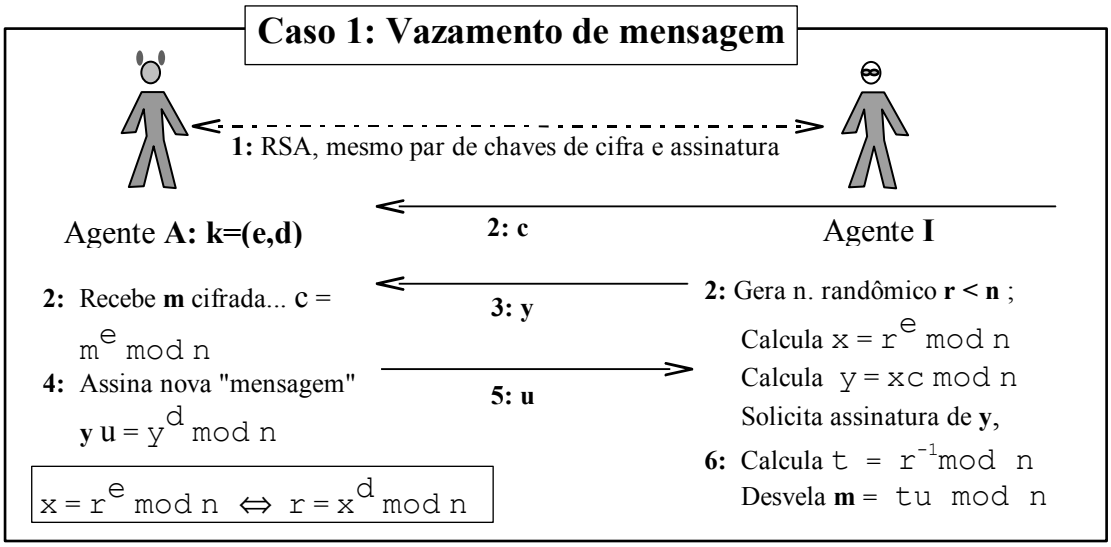
- **Ataques a protocolos que usam o RSA -**

Métodos conhecidos exploram falhas nos protocolos (não diretamente no algoritmo), devido à exponenciação preservar estruturas multiplicativas:

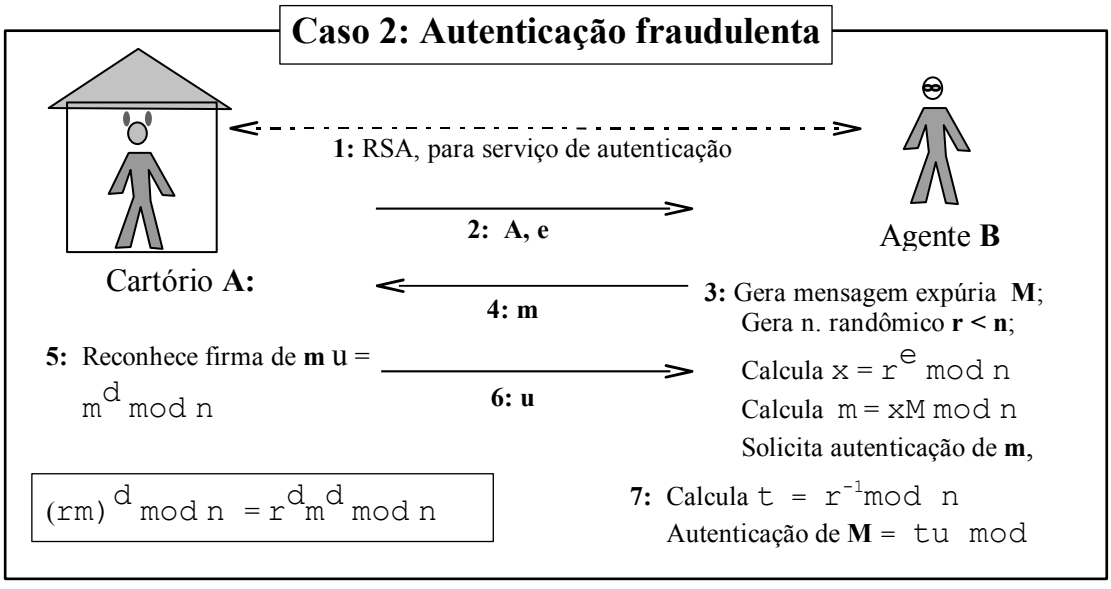
- *Criptograma escolhido contra assinatura;*
- *Módulo comum;*
- *Expoente pequeno para encriptação;*
- *Ordem de operações de cifra e assinatura*.

- **Ataque por criptograma escolhido contra assinatura -**

Este ataque é possível contra protocolos que assinam a mensagem por extenso (e não um hash da mesma), e prescinde da conivência ou negligência do agente fraudado em assinar mensagens sem motivo aparente.



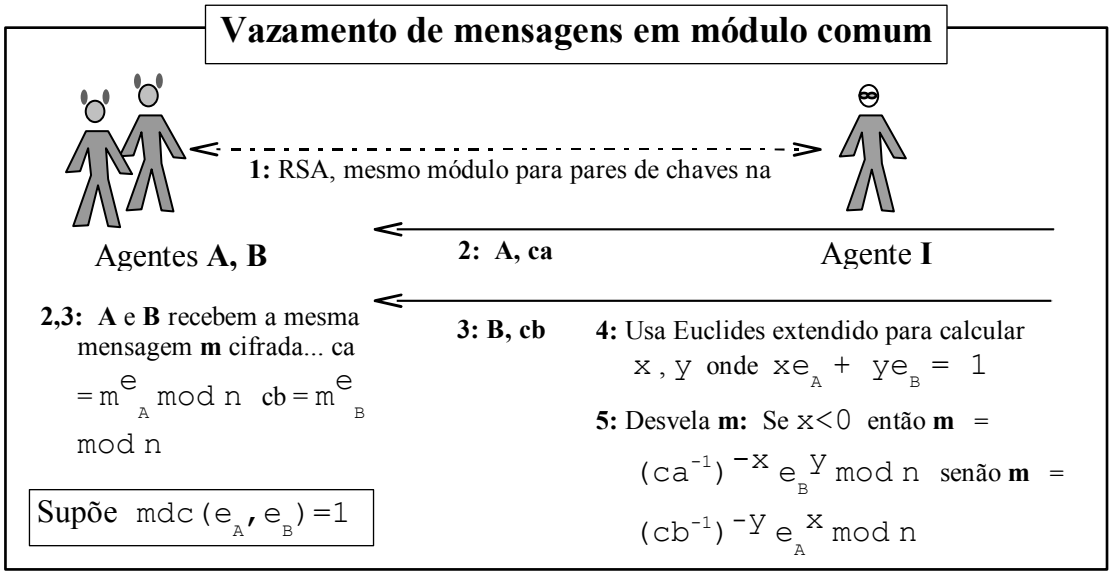
$$tu \bmod n = r^{-1}y^{d} \bmod n = r^{-1}x^{d}c^{d} \bmod n = r^{-1}rc^{d} \bmod n = c^{d} \bmod n = m$$



Em serviços de autenticação, a assinatura deve ser feita sobre o hash.

- **Ataque em módulo comum -**

Este ataque é possível se a distribuição de chaves para a cifra que usa o RSA atribui chaves com o mesmo módulo a usuários distintos. Qualquer mensagem encriptada por mais de um usuário pode ser facilmente vazada.

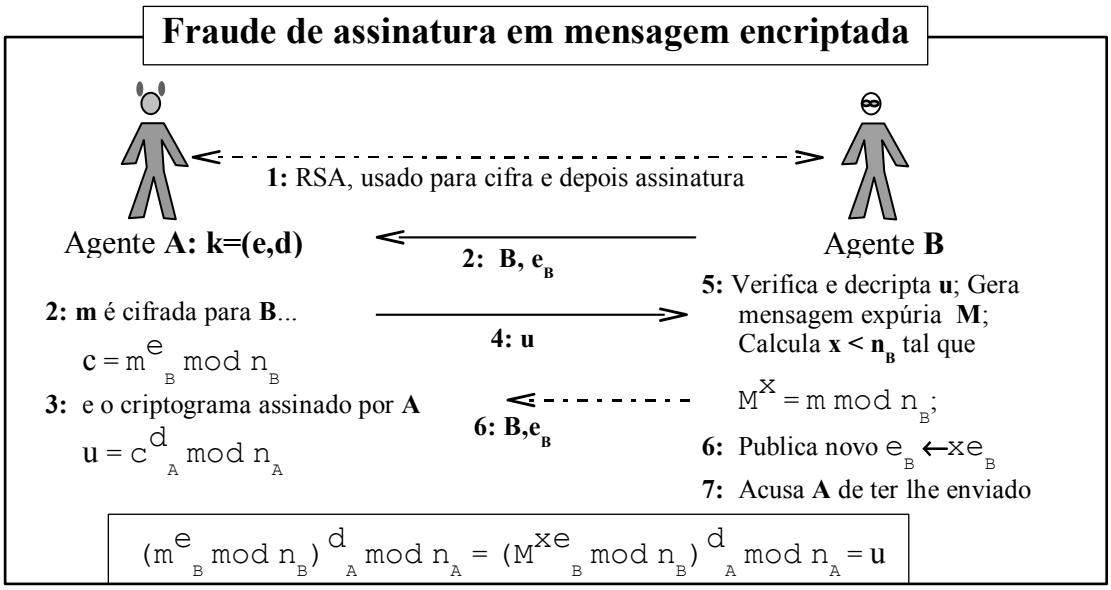


- **Ataque com expoentes pequenos de encriptação -**

Encriptação/verificação de assinatura no RSA é mais rápido quanto menor for a chave pública. Porém este tipo de ataque é possível com a encriptação de $e(e+1)/2$ mensagens linearmente dependentes, caso hajam

- **Ataque com assinatura de criptograma -**

As operações de assinatura e encriptação devem ser executadas nessa ordem, para evitar fraudes decorrentes deste tipo de ataque, onde nem mesmo o uso de função de hash para assinatura pode evitar.



Este ataque é possível porque **B** tem como resolver o problema do logaritmo discreto para encontrar **x**, já que conhece a fatoração de $\mathbf{n_B}$. Se a assinatura antecedesse a encriptação, **B** buscaria **x** sem saber fatorar $\mathbf{n_A}$.

- ## Prevenção contra ataques conhecidos ao RSA -

1. Conhecimento de um par **(e,d)** permite a fatoração do módulo **n**.
2. Conhecimento de um par **(e,d)** permite encontrar outros para mesmo **n**
3. Módulo comum não deve ser usado em serviços de rede.
4. Mensagens devem ser preenchidas com bits randômicos enquanto < **n**.
5. O expoente público deve ser grande, e a assinatura anteceder a cifra.

- ## Padronização e patentes -

O RSA é um padrão *de facto* para criptografia assimétrica: Anexo da norma ISO 9796, *draft* de uma norma ANSI, padrão bancário na França e Austrália. Não é padrão nos EUA por problemas de disputa sobre direitos de patente. A patente, válida somente nos EUA, expira em 20/9/2000.

Rabin

Algoritmo assimétrico cuja segurança é derivada da dificuldade de se extrair raiz quadrada em anéis, com decriptação não determinística. A ordem n do anel deve satisfazer $n = pq$ onde $p, q \equiv 3 \bmod 4$ (M. Rabin, 79).

Geração de parâmetros:

p = gerapr3mod4(rand( ))
q = gerapr3mod4(rand( ))
$n = p*q$ **{público}**
$r = q*(q^{-1} \bmod p)$
$s = p*(p^{-1} \bmod q)$
$e_A = 2$ **{k pública}**
$d_A = (r,s)$ **{k privada}**

Encriptação:

$c = m^2 \bmod n$

Decriptação:

$t = c^{(p+1)/4} \bmod p \quad u = c^{(q+1)/4} \bmod q$

$m = ((\pm r)*t + (\pm s)*u) \bmod n$

- **Análise do algoritmo de Rabin -**

A segurança deste algoritmo é provadamente equivalente à fatoração de inteiros. Entretanto a mensagem só pode ser recuperada, dentre as 4 possíveis decriptações, se contiver algum conteúdo semântico.

Cifras que usam este algoritmo são inseguras contra ataques de criptograma escolhido, o que inviabiliza seu uso para assinatura em texto pleno. O uso de hash no protocolo enfraquece a equivalência acima.

- **Variante de Williams** (Hugh Williams, 1980) **-**

Alternativa do algoritmo com decriptação unívoca, usa $p \equiv 3 \bmod 8$, $q \equiv 7 \bmod 8$ e chave privada $r = ((p-1)(q-1)/4+1)/2$. O criptograma tem três partes, sendo **r** usado na decriptação, como expoente em uma das partes.

ElGamal

Algoritmo assimétrico cuja segurança é derivada da dificuldade de se extrair logaritmos discretos em corpos finitos. (T. ElGamal, 1984).

Geração de parâmetros, chaves assimétricas e chave de sessão:

p = geraprimo(rand( ))
g = rand(|p|)
d_A = x= rand(|p|) **{chave privada}**
$y = g^x \bmod p$
e_A = (p,g,y) **{chave pública}**
k_m = rand(|p|)
$k = k_m / mdc(k_m, p-1)$ **{ch. sessão}**
$a = g^k \bmod p$

Assinatura:

b = euclext(k,p–1,m–x∗a)
{(a,b) = assinatura de m }
$(y^a * a^b) \bmod p \stackrel{?}{=} g^m \bmod p$

Cifragem:

$b = (y^k * m) \bmod p$
{(a,b) = criptograma de m }
$m = (b * a^{-x}) \bmod p$

- ## Análise do algoritmo de ElGamal -

Cada assinatura ou encriptação requer um valor randômico para **k**. O conhecimento de pelo menos duas mensagens encriptadas ou assinadas como o mesmo **k** permite a recuperaração da chave privada **x**.

Este algoritmo não é patenteado, mas sua versão para cifragem é uma variante do algoritmo de Diffie-Helmann. A detentora de patente para o D&H (PKP Inc.) reclama direitos para licenciar seu uso (até abril de 1997).

- ## Variantes e generalizações do algoritmo de ElGamal -

Prova de identidade (T. Beth, EUROCRIPT 88);
Derivação de chaves (W. Jaburek, EUROCRIPT 89);
Autentição de senhas (C. Chang & S. Huang, IEEE Carnahan Conf. 91);
Esquema p/ protocolos de assinatura (Horster, Petersen, ASIACRIPT 94).

Outros algoritmos assimétricos

- **Algoritmos baseados no problema da mochila -**

Merkle-Hellman foi o primeiro algortmo assimétrico divulgado (78). Baseia-se no problema combinatório de se encontrar uma partição em um conjunto fixo de inteiros onde a soma de elementos dê igual ao argumento.

A chave privada é formada por um conjunto fixo supercrescente (versão trivial do problema) com a qual a função decriptadora calcula a partição que representa a mensagem. A chave pública é um conjunto fixo genérico equivalente (obtido daquele por operações modulares), no qual a mensagem mapeia uma partição, cuja soma é seu criptograma.

São inseguros, apesar de NP-completo o problema em que se baseiam: a chave privada pode ser obtida em tempo polinomial a partir da pública.

- **Algoritmos baseados em códigos de recuperação de erros -**

Algoritmos de *McEliece* empregam códigos de *Goppa* como se fossem códigos lineares, baseado em ser NP-completo o problema de se encontrar uma palavra de distância fixa ao argumento, em um código linear.

A chave pública é o produto de três matrizes sobre **GF(2)**: uma permutação, a matriz geradora de um código de *Goppa* e uma matriz não singular. A chave privada é a tripla das matrizes. Apesar de ser rápido e até hoje seguro, é pouco usado por expandir **m** e usar chaves muito grandes.

- **Algoritmos baseados em automata finitos -**

Tao Henji usa automata quasi-lineares em esquema semelhante aos baseados na fatoração de inteiros. Também requerem chaves muito longas.

6: Implementações

- **Cenário atual da criptografia** (1998) **-**
  - **Padronização**: Por normatização ou por forças de mercado (interoperabilidade), a segurança na informática tende naturalmente à busca de padrões. Esta tendência se torna mais imperativa com o advento das redes globais e ambientes de computação distribuída.
  - **Padrões interoperáveis:** Os critérios para escolha de algoritmos criptográficos estão hoje relativamente estabilizados pela prática. Há um senso de urgência para convergência na escolha de protocolos que integrem vários serviços básicos, e mecanismos de implementação independentes de plataforma ou arquitetura (ex: IPSec).
  - **Limitações:** Sistemas legados cuja concepção não contemplava segurança e/ou interoperabilidade, legislação local e internacional omissa, desatualizada ou mal pensada e interesses paroquiais, são os maiores entraves ao avanço do uso da criptografia na informática.
  - **Atualizações:** No cenário da computação global e distribuída de hoje, a criptologia assimétrica é parte fundamental. Como os limites teóricos desta tecnologia ainda não estão bem delineados, suas implementações precisam de constantes reavaliações de risco.
  - **Desafios:.....** Cada nova aplicação ou tecnologia que exija proteção aos dados que trata, acumula desafios de complexidade crescente à criptologia, em especial os sistemas de computação não assistida, viabilizados pela miniaturização eletrônica.

Implementação de serviços de chave pública

- **Algoritmo RSA -**

Existem vários fabricantes licenciados para implementação em chip VLSI. A mais eficiente no mercado em 1995 cifra a 64Kb/seg com módulo de 512 bits (~1000x mais lenta que o DES). Implementações em espaço limitado (*smartcards*) são mais lentas.

Chips com RSA Companhia	Freq. clock	Velocidade (bloco 512)	Ciclos p/ bloco 512	Tecnologia/ transístores	modulo máximo
Alpha Technology	25 MHz	13 Kbits/seg	0.98 M	2.0µ / 180K	1024 bits
AT&T	15 MHz	19 Kb/s	0.4 M	1.5µ / 100K	298 bits
British Telecom	10 MHz	5.1 Kb/s	1 M	2.5µ /	256 bits
Calmos System	20 MHz	28 Kb/s	0.36 M	2.0µ / 95K	593 bits
CNET	25 MHz	5.3 Kb/s	2.3 M	1.0µ / 100K	1024 bits
Cryptech	14 MHz	17 Kb/s	0.4 M	Gate array/ 33K	120 bits
Cylink	30 MHz	6.8 Kb/s	1.2 M	1.5µ / 150K	1024 bits
GEC Marconi	25 MHz	10 Kb/s	0.67 M	1.4µ / 160K	512 bits
Pijnemburg	25 MHz	50 Kb/s	0.25 M	1.0µ / 400K	1024 bits
Sandia	8 MHz	10 Kb/s	0.4 M	2.0µ / 86K	272 bits
Siemens	5 MHz	8.5 Kb/s	0.3 M	1.0µ / 60K	512 bits

- **Aceleração de sistemas que usam o RSA -**

A escolha da chave pública pode influir na velocidade de encriptação ou verificação. Valores fixos de $\mathbf{e}_A$ com 2 bits ligados requerem apenas 17 multiplicações para executar a exponenciação. Recomenda-se $\mathbf{e}_A = 65537 = 2^{16}+1$ (padrão ANSI X.509 e padrão PKCS) ou $\mathbf{e}_A = 3$ (PEM e PKCS)

Implementação de sistemas de chave pública usando aritmética de curvas elípticas

- **Versões distintas do problema do logaritmo discreto -**

Avanços no estado da arte do problema de se fatorar números inteiros comprometem a eficiência dos algoritmos assimétricos que usam aritmética modular, pois demandam dessas implementações chaves maiores para que manterem a mesma segurança.

Alternativamente, pode-se implementar estes algoritmos usando operações algébricas de uma estrutura distinta dos corpos finitos, onde o problema em que se baseiam os algoritmos continua bem posto, mas onde as técnicas avançadas de fatoração de inteiros não se apliquem.

- **Aritmética das curvas elípticas sobre corpos finitos -**

Na geometria analítica, o conjunto de pontos de um espaço vetorial com coordenadas (x,y) que satisfazem uma dada equação da forma.

$$y^2 = x^3+ax+b$$

é chamado de *curva elíptica,* caso os coeficientes satisfaçam $4a^3+27b^2 \neq 0$.

Em 1985 *N. Koblitz* e *V. Miller* descobriram que, se aplicada a um espaço onde as coordenadas são elementos de um corpo finito (ex: $\mathbf{Z}_p$), a definição de curva elíptica seleciona pontos que, incluindo-se um "ponto no infinito", formam um grupo algébrico sob a operação de composição inspirada na geometria das secantes dos espaços métricos. Esta operação substitui a operação de exponenciação em algoritmos assimétricos.

- **Grupos de curvas elípticas $E(Z_p)$ -**

$$E(Z_p) = \{ P=(x,y) \in \mathbf{Z_p} \times \mathbf{Z_p} \mid y^2 = x^3+ax+b \} \cup \{O\}$$

onde a operação algébrica do grupo, denotada por "+", é assim definida:

1. $P+O = O+P = P$
2. Dado $P=(x,y)$, denotamos $-P=(x,-y)$, onde $P+(-P) = O$
3. Dados $P=(x_1,y_1)$, $Q=(x_2,y_2)$, então $P+Q = (x_3,y_3)$ é dado por

   $x_3 = \lambda^2-x_1-x_2$;

   $y_3 = \lambda(x_1-x_3)-y_1$ onde

   $\lambda = (y_2-y_1)/(x_2-x_1)$ se $P \neq Q$, ou

   $\lambda = (3x_1^2+a)/(y_1+y_1)$ se $P=Q$.
4. $nP = P+P+...+P$ (n vezes)

- **Comparações entre aritméticas de Z_p e $E(Z_p)$ -**

Operação	Z_p	$E(Z_p)$
"Produto"	$a*b \bmod p$	$P+Q$
"Exponenciação"	$a^n \bmod p$	nP
Logaritmo discreto	Encontrar n tal que $a^n \bmod p = b$	Encontrar n tal que $nP = Q$

- **Chaves públicas com nível de segurança equivalentes -**

Tempo p/ recuperar chave privada	Fatoração em Z_p: Number field sieve	Logaritmo em $E(Z_p)$: Pollard - Rho
$3x10^8$ MIPS - ano	~960 bits	155 bits
$3x10^{18}$ MIPS - ano	~1820 bits	210 bits
$3x10^{28}$ MIPS - ano	~2500 bits	239 bits

Futuro da criptografia assimétrica

- **Dificuldades técnicas na aritmética de curvas elípticas -**

O grupo de pontos de curva elíptica fornece um código de cifragem semelhante aos códigos de recuperação de erros. Por isso uma porcentagem do espaço de textos não pode ser cifrado, sendo esta porcentagem inversamente proporcional à expansão na encriptação. Esta característica impede seu uso como mero substituto da aritmética modular em algumas aplicações da criptografia.

- **Outras técnicas -**

Estruturas algébricas distintas da aritmética modular e dos grupos de curvas algébricas podem em princípio fornecer formalismos teóricos para a critografia assimética. (ex: grupos semi-lineares). As alternativas que surgiram até hoje na literatura não oferecem apelo prático devido ao grande tamanho das chaves robustas.

De qualquer forma, uma nova área da criptografia – pós quântica – se dedica ao estudo de algoritmos resistentes à paralelização quântica.

- **Computação Quântica -**

Teoricamente um átomo pode funcionar como processador, onde o estado de excitação de um elétron codifica um bit. A superposição linear de estados na teoria quântica significa que uma malha de processamento paralelo pode colapsar para a solução de um probrema massivamente distribuido, como o da fatoração de um inteiro, trivializando a criptografia assimétrica. A construção de computadores quânticos parece ainda remota.

Implementação de serviços de assinatura digital

- **Cenário atual dos protocolos de assinatura digital** (1997) **-**

  - **Padronização**: Embora patenteado nos EUA, o RSA tornou-se um padrão internacional *de facto* para assinatura digital (SSL, ISO 9796). O governo dos EUA (NIST) propôs para si em 1991 um padrão para assinatura digital, o DSS (*Digital Signature Standard*), que emprega o algoritmo DSA (*Digital Signature Algorithm*).

  - **Restrições:..** Apesar da possibilidade de infringir a patente do algoritmo de Schnorr (até 2008), sobre a qual o governo americano não possui nenhum direito, o padrão DSS foi adotado em maio de 1994. O governo dos EUA promete auxílio para defesa legal de quem usar o DSA por força de contrato.

  - **Abrangência:** Existe um esquema para dedução de algoritmos de assinatura digital, do qual ElGamal, Schnorr e DSA são casos particulares. Como este esquema foi publicado mas não patenteado, pode esvaziar disputas sobre o DSA.

  - **Limitações:.**O algoritmo DSA requer um gerador de seqüências pseudo-randômicas seguro. Repetições ou previsão de seqüências podem permitir a fraude de assinaturas.

O padrão DSS especifica a assinatura sobre hash de 1024 bits da mensagem, calculada pelo algoritmo SHA, e sugere a geração de primos seguros através de algoritmo fornecido na especificação.

Digital Signature Algorithm

Algoritmo patenteado e licenciado pelo NIST para uso irrestrito.

Geração de parâmetros, chaves assimétricas e chave de sessão:

q = geraprDSA (rand(160))
repeat
  p=geraprDSA(rand(512+64t))
until q | (p-1)
repeat
  h = rand(512+64t) mod p
  g = $h^{(p-1)/q}$ mod p
until g $\neq$ 1
x = rand(|q|) **{chave privada}**
y = g^x mod p
e_A= (p,q,g,y) **{chave pública}**

Assinatura de m = (r,s):

repeat
  k = rand( ) mod q **{ch. sessão}**
  r = (g^k mod p) mod q
  s = (k^{-1}*(SHA(m)+xr)) mod q
until s $\neq$ 0

Verificação: (r,s) assinatura de m

u = (s^{-1}*SHA(m)) mod q
v = (s^{-1}*r) mod q
r =? (g^u*y^v mod p) mod q

- **Análise do algoritmo DSA -**

Os parâmetros **p**, **q** e **g** podem ser compartilhados entre um grupo de usuários, onde **p** é um primo de tamanho entre 512 e 1024 bits, **q** um primo de 160 bits. A chave de sessão **k** deve ser única para cada assinatura.

Lenstra & Haber descobriram em 1994 a ocorrência de pares **p**, **q** com propriedades que facilitam a recuperação da chave privada **x.** A geração aleatória de pares randômicos com essa propriedade é muito rara. O NIST sugere, na especificação DSS (*Digital Signature Standard*), o uso de um gerador de primos para o DSA que evita estes pares.

O protocolo DSS possibilita a construção de canal subliminar através da escolha da chave **k** do DES para sessão de assinatura. Em algumas implementações, o DES pode ser usado também para emular a cifragem do algoritmo de ElGamal.

Esquema Meta-ElGamal

O Meta-algoritmo de ElGamal é um esquema para se derivar milhares de algoritmos assimétricos para assinatura digital (*Horster, Petersen & Michels: 1994 ACM computer conference on communications security*).

Geração de parâmetros, chaves assimétricas e chave de sessão:

p=geraprimo(rand( ))
repeat
  q = rand( )
until q | (p-1)
repeat
  g = rand( ) mod p
until g^q mod p = 1
x = rand( ) mod q **{chave privada}**
y = g^x mod p
e_A= (p,q,g,y) **{chave pública}**

Assinatura de m = (r,s):

repeat
  k = rand( ) mod q **{ch. sessão}**
until mdc(k,q) = 1 r = g^k mod p
t = r mod q

equação de assinatura

ak = b + cx mod q

equação de verificação:

r^a =? $g^b y^c$ mod q

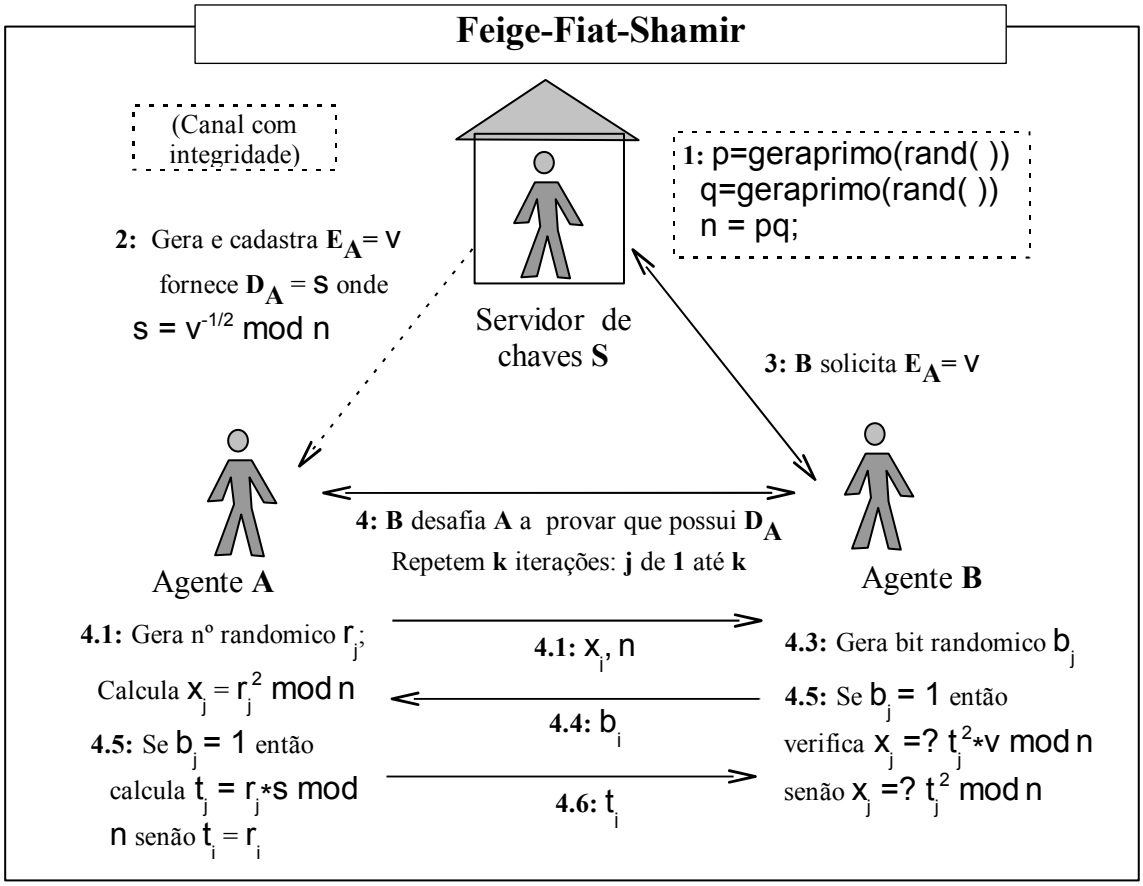
- **Coeficentes no esquema meta-ElGamal -**

Tabela de possíveis valores dos coeficientes a, b, c		
± t	± s	± m
± tm	± s	1
± tm	± ms	1
± tm	± ts	1
± sm	± ts	1

Exemplos: algoritmos derivados da 1ª linha (+) da tabela acima	
Equação de assinatura	**Equação de verificação**
(1) mk = s+tx mod q	$r^t = g^s * y^m$ mod q
(2) tk = m+sx mod q	$r^t = g^m * y^s$ mod q
(3) sk = t+mx mod q	$r^s = g^t * y^m$ mod q
(4) sk = m+tx mod q	$r^s = g^m * y^t$ mod q
(5) mk = s+tx mod q	$r^m = g^s * y^t$ mod q
(6) mk = t+sx mod q	$r^m = g^t * y^s$ mod q

Esquema de autenticação de Feige-Fiat-Shamir

Protocolos derivados deste esquema patenteado implementam provas de identidade baseadas em conhecimento zero e na dificuldade de extração de raizes quadradas em anéis finitos (Weizmann Institute, Israel, 1986).



- **Análise do esquema Feige-Fiat-Shamir -**

O número de iterações pode ser reduzido por paralelização, onde **v, b** se tornam vetores, mantendo-se o mesmo grau de certeza $1-2^{-k}$.

O esquema pode ser adaptado para protocolo de assinatura transformando a função do verificador **B** em uma função de hash, sendo de 20 a 100 vezes menos intensivo em multiplicações que o RSA.

Outros esquemas de autenticação

- **Variações do esquema Feige-Fiat-Shamir** (EUROCRIPT 90)**-**

  O servidor de chaves pode ser abolido se cada usuário escolher seu módulo e publicar sua chave em um banco de dados de acesso seguro.

  Existem implementações paralelizadas com módulos individuais onde o vetor **v** é uma seqüência dos primeiros primos, onde o tempo de verificação é otimizado, e versões para identificação de grupos de pessoas.

- **Ohta-Okamoto** (CRIPTO 88)

  Variante do esquema de Feige-Fiat-Shamir, onde a segurança deriva da dificuldade de fatoração de inteiros.

  Uma de suas aplicações implementa um protocolo de grupo de assinaturas, onde as assinaturas precisam ser aplicadas em ordem específica, proposto para cartões inteligentes (*smartcards*).

- **Guillou-Quaisquater** (EUROCRIPT 88)

  Esquema proposto para catões com processadores, , desenhado para trocas mínimas, implementa protocolos para prova de identidade, assinatura digital individual e múltipla. Emprega multiplicação e exponenciação, semelhante ao de Ohta-Okamoto.

- **Schnorr** (CRIPTO 89)

  Algoritmo patenteado, derivável do exemplo (4) de Meta El-Gamal.

Padrões para assinatura digital e gerenciamento de chaves

- **Histórico -**

A segurança externa (redes e telecomunicações abertas) só é alcançável com a adesão a padrões interoperáveis, desde o registro de algoritmos (ISO/IEC 9979) a esquemas de autenticação e de distribuição de chaves assimétricas. Estes padrões descrevem detalhes sobre a escolha e implementação de protocolos criptográficos independentes do transporte.

A indústria financeira é pioneira nesta padronização, sendo atualmente este esforço liderado pelos comitês ISO, ANSI, ITU, IETF.

- **Assinatura Digital -**

Os padrões de assinatura digital estabelecem formatos para inclusão de autenticadores em arquivos de formato binário ou texto, para a escolha e modo de operação de algoritmo ou função de hash autenticador, visando a interoperabilidade de implementações independentes (ex.: ANSI X9.9).

- **Distribuição de chaves -**

Os padrões de distribuição de chaves criptográficas estabelecem, dentre outros, critérios de segurança para o armazenamento de chaves, ambiente de operação do utilitário de gerência de chaves, técnicas de encriptação de chaves e geração de vetores de inicialização, formato de mensagens de serviços criptográficos, uso pretendido, etc. (ex.: ISO 8732 para bancos)

Principais padrões de protocolos criptográficos

- **Quadro resumo -**

Tópico	ISO/IEC JTC 1 Information Technology	ISO TC68 Instituições Financeiras	ANSI	U.S. Federal Government
Modos de operação	ISO/IEC 8372 ISO/IEC 10116		X3.106	FIPS PUB 81
Message Auth.Code (Geral)	ISO/IEC 9797	ISO 8730 (DSA) ISO 8731 (RSA)	X9.9	FIPS PUB 113
Mess. Auth. Code (Bancos)		ISO 9807	X9.19	
Algoritmos de assinatura digital	ISO/IEC 9594.8 ISO/IEC 9796		X9.30.1 (DSA) X9.31.1 (RSA)	FIPS PUB tba (DSA)
Funções de hash	ISO/IEC 10118		X9.30.2 (SHA) X9.31.2 (MDC)	FIPS PUB 180 (SHA)
Autenticação de agentes	ISO/IEC 9594.8 ISO/IEC 9796	ISO 11131	X9.26	
Gerência de chaves (simétrico)	ISO/IEC 11770.2	ISO 8732 ISO 11568	X9.17 X9.24	FIPS PUB 171
Gerência de chaves (sim. multi-centro)		ISO 11649	X9.28	
Gerência de chaves (assimétrico)	ISO/IEC 9594.8 ISO/IEC 11770.3	ISO 11166	X9.30.3	

- **Outros padrões -**

A interface de uso de cartões inteligentes com 6 pontos de contato foi objeto de padronização pelo comitê técnico do OSI, publicado como ISO/IEC 7816. São especificadas as características físicas, posição dos contatos, natureza e protocolo de intercâmbio dos sinais eletrônicos, dentre outros.

Certificados digitais de chave pública

- **Definição (Abreviação: Certificado Digital)**

  Documento digital destinado à distribuição de uma chave pública titulada, contendo (além da chave) informação sobre o titular, sobre o emitente (certificadora), e sobre o uso pretendido para a chave transportada. Este uso é realizado por um conjunto de protocolos que compõem uma ICP ou PKI (infraestrutura de chaves públicas).

- **Histórico das *Public Key Infrastructures* (PKI)**

  O conjunto de formatos e protocolos PKCS (1 a 13), proposto pela indústria pioneira (RSADSI) tornou-se padrão para PKIs. O padrão ITU-T X.509 da *International Telecommunications Union* (PKCS 9) tem sido aceito como padrão de fato para certificados digitais.

- **Formato básico dos certificados X.509 -**

Versão do modelo de certificado (*v.3*)
Número de série (único para certificados assinados pelo emissor)
Identificador do algoritmo de assinatura **- Algoritmo** **- Parâmetros usados na assinatura**
Emissor *(Certification Authority)*
Período de validade **- Não antes de [DATA]** **- Não depois de [DATA]**
Identificação do Titular da chave transportada (*Distinguished name*)
Chave pública do titular (*public key*) **- Algoritmo a que se destina** **- Parâmetros para uso do algoritmo** **- Chave pública**
Assinatura do emissor neste certificado

Mecanismos para uso de certificados digitais em redes abertas

- ***Certification Authorities* -**

Confiança na integridade sintática de uma chave pública assinada é oferecida pela entidade que assina o certificado usado para distribuí-la. Existem dois modelos básicos para indução de confiança:

1. **Hierárquico**.....*proposto pelo padrão* PEM *para correio eletrônico.*
2. **Malha de confiança**.....*usado pelo utilitário* PGP *p/ correio eletrônico.*

- **Modelo Hierárquico de certificação -**

Este modelo, proposto como padrão *Privacy Enhanced Mail* pela IETF nos RFCs 1421 a 1424, prevê identificação única do sujeito em toda a internet. A verificação de certificados segue uma cadeia de autenticação por entidades certificadoras, com consulta a lista de revogação de certificados, com a assinatura final da IPRA (*Internet Policy Registration Authority*).

- **Redes de confiança -**

O utilitário de domínio público PGP para sigilo e assinatura digital de e-mail (índice de *sites* em http://www.mantis.co.uk/pgp/pgp.html), pressupõe um mecanismo off-line não eletrônico de distribuição fim-a-fim de chave pública, ou a distribuição por e-mail com assinatura de algum outro agente que permite a identificação confiável do titular. Teve aceitação maior que o PEM, devido a sua praticidade, disseminando o uso do RSA pela internet.

Entidades Certificadoras na Internet

- **Modelo híbrido atualmente em uso -**

A necessidade urgente de estabelecimento de infra-estrutura de segurança que viabilize o comércio e governo eletrônicos na internet, tem atropelado os mecanismos formais de padronização e a regulamentação jurídica sobre a responsabilidade civil das entidades certificadoras (a MP 2200 ainda não foi juridicamente testada).

Serviços de emissão e controle de certificados X.509 e PKCS, cujas chaves públicas são na prática confiadas por serem distribuídas junto com os utilitários de navegação web (*browsers*), vem funcionando como autoridades certificadoras, distribuindo certificados de chaves públicas para seus clientes.

- **Entidades Certificadoras em operação** (Verisign, Certisign,etc)

Sites que assinam e distribuem certificados, contendo uma chave pública de cuja inversa a posse foi verificada por meio de desafio.

1. **Certificados de e-mail**.....*validam o vínculo entre uma caixa postal de e-mail e uma chave pública, com dados sobre a conta de e-mail.*
2. **Certificados comercias** . .*validam o vínculo entre um estabelecimento comercial então ativo, com sede e endereço eletrônico (www, e-mail, etc), e uma chave pública, com dados sobre o estabelecimento tais como nome de domínio, endereço, caixa postal, registros cartoriais, etc..*

No âmbito da doutrina jurídica do *common law* (EUA, UK), entidades certificadoras estabeleceram o negócio da certificação de chaves públicas segundo o modelo atuarial (apólice de seguro), incompatível com o conceito de "fé pública".

Uso de *Tokens* em segurança externa

- **Definição -**

Hardware portátil dedicado ao armazenamento de segredo identificador de agente (ex: senha, chave privada) e *firmware* para interoperação com plataformas que executam protocolos de autenticação e serviços.

Esquemas de autenticação baseados em desafio, em sequenciadores, em sincronização de senhas descartáveis ou em verificação de assinatura, para neutralizar ataques de dicionário, escuta ou *replay* podem ser atacados com engenharia reversa, o *firmware* não for projetado para resisti-los.

- **Vulnerabilidade de *tokens* e *smartcards* -**

Estas tecnologias não devem ser usadas como ponto crítico de falha em um sistema de segurança. Mesmo os chips para cartão que trabalham com encriptação interna (ex.: IBM DS5002FP), onde bytes trafegam no barramento e são carregados encriptados em RAM e EPROM, são passíveis de ataque, nos quais o material de chave armazenado ou gerado pode ser extraído. Ataques mais comuns a *tokens* de acesso livre são:

1. **Surtos de clock**.....*usados para introduzir instruções errôneas nos registradores que podem causar o extravazamento em loops de leitura*.
2. **Surtos de potência**.....*podem causar comportamento anômalo em geradores randômicos de uso criptográfico.*
3. ***Probing***.....*violação do lacre eletrostático possibilita avaria nas células de trava da EPROM, para leitura do material da chave*.
4. **Dicionário de código**. *ataque no barramento à criptografia interna.*

Riscos à segurança externa

- **Redes abertas -**

As redes abertas (internet, telefonia celular) que funcionam pela aderência ao conjunto padronizado de protocolos de comunicação denominado (ex: TCP/IP) são, por um lado, intrinsecamente inseguras devido aos objetivos originais na sua concepção, enquanto por outro lado oferecem o potencial de interconxão global através de ambiente legado (redes fechadas).

Qualquer sistema computacional conectado por este tipo de tecnologia pode ser invadido para sabotagem ou exploração em apoio a ataques a outros nós da rede aberta. A busca de segurança neste cenário impulsiona novas direções na ciência da computação, para vencer desafios no controle de tráfego IP através de sistemas legados e interoperáveis (IDSs).

- **Vulnerabilidade do TCP/IP -**

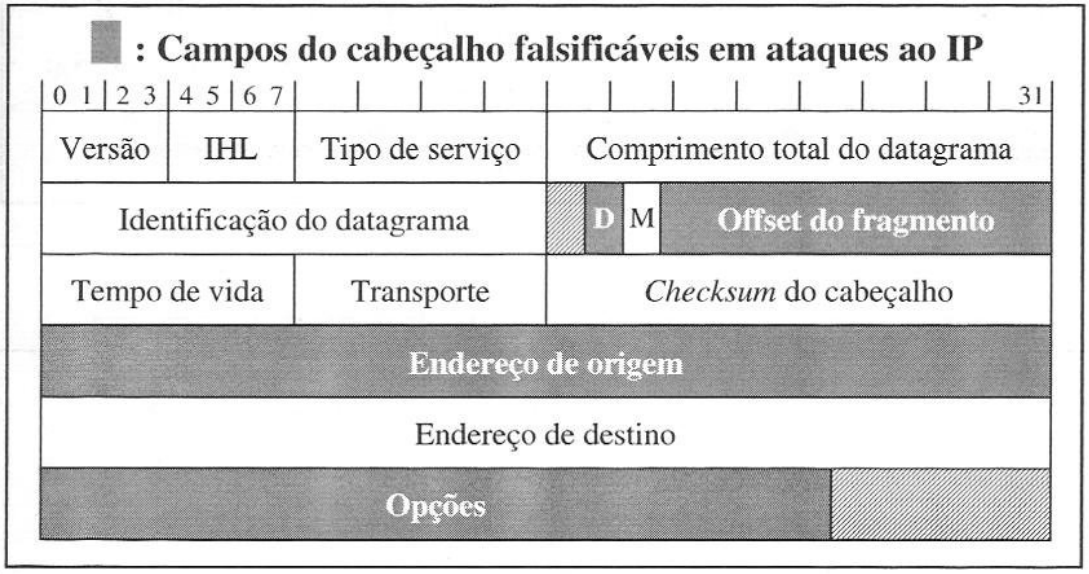
O ambiente TCP/IP é difícil de ser precisamente controlado. Vários tipos de ataque via internet exploram falhas conceituais de segurança em seus serviços, sendo o custo deste controle bastante alto. Busca-se o equilíbrio entre disponibilização de serviços e controles de segurança.

Freqüência relativa de ataques na internet	
1	*sniffers* (kits de análise, troianos, exploits)
2	*spoofing* de IP
3	SMTP (*sendmail*)
4	NFS (*Network File System*)
5	NIS (*Network Information Service*)
6	adivinhação de senhas

(Fonte: *Computer Emergency Response Team´s Anual Report*, 1995)

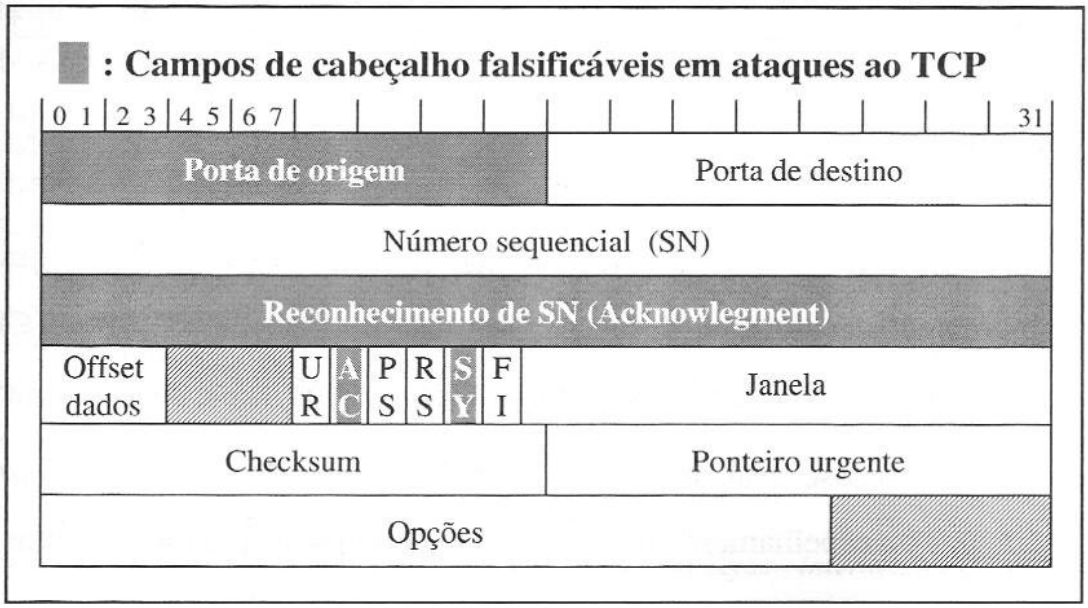
- **IPv4 -**

Protocolo de rede em uso na internet desde 1981, não comporta mecanismo de autenticação. Sua nova versão IPv6 (rfc 1884), em testes desde 1992, permite autenticação simples e comporta uso de criptografia.

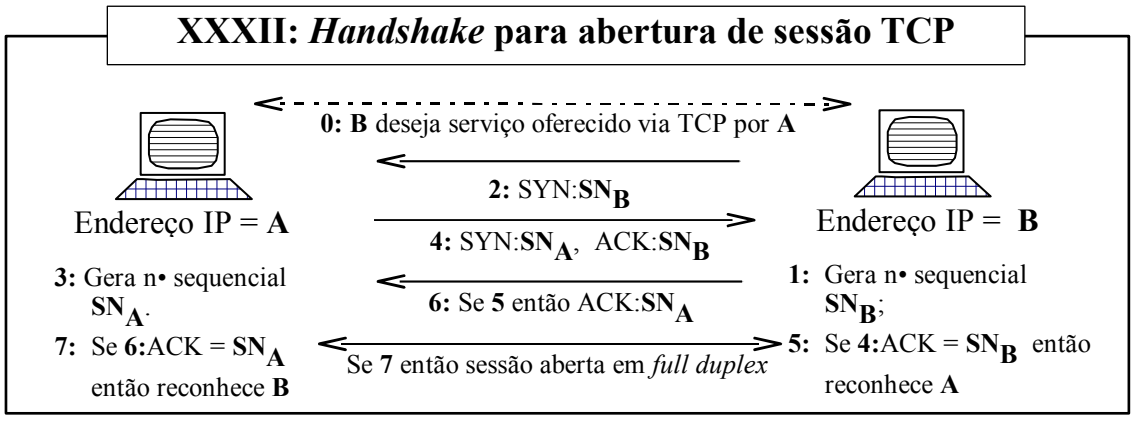


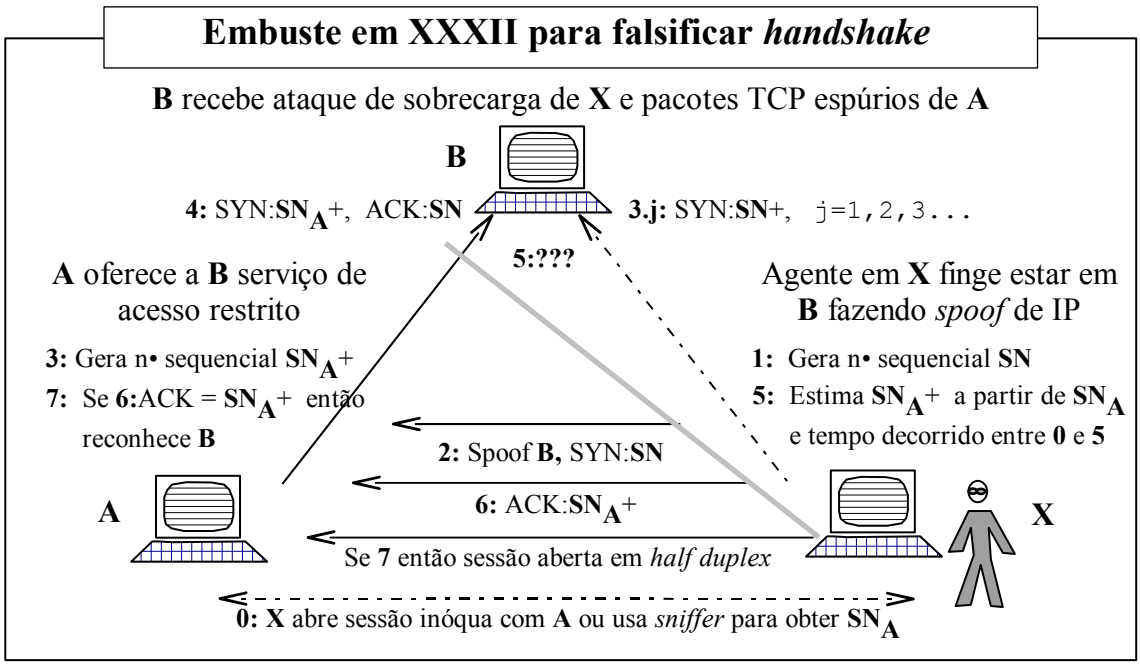
- **TCP -**

Protocolo de transporte orientado a conexão da internet, autentica por *handshake* simples, falsificável se sua implementação for conhecida.



Ataque de número sequencial ao TCP



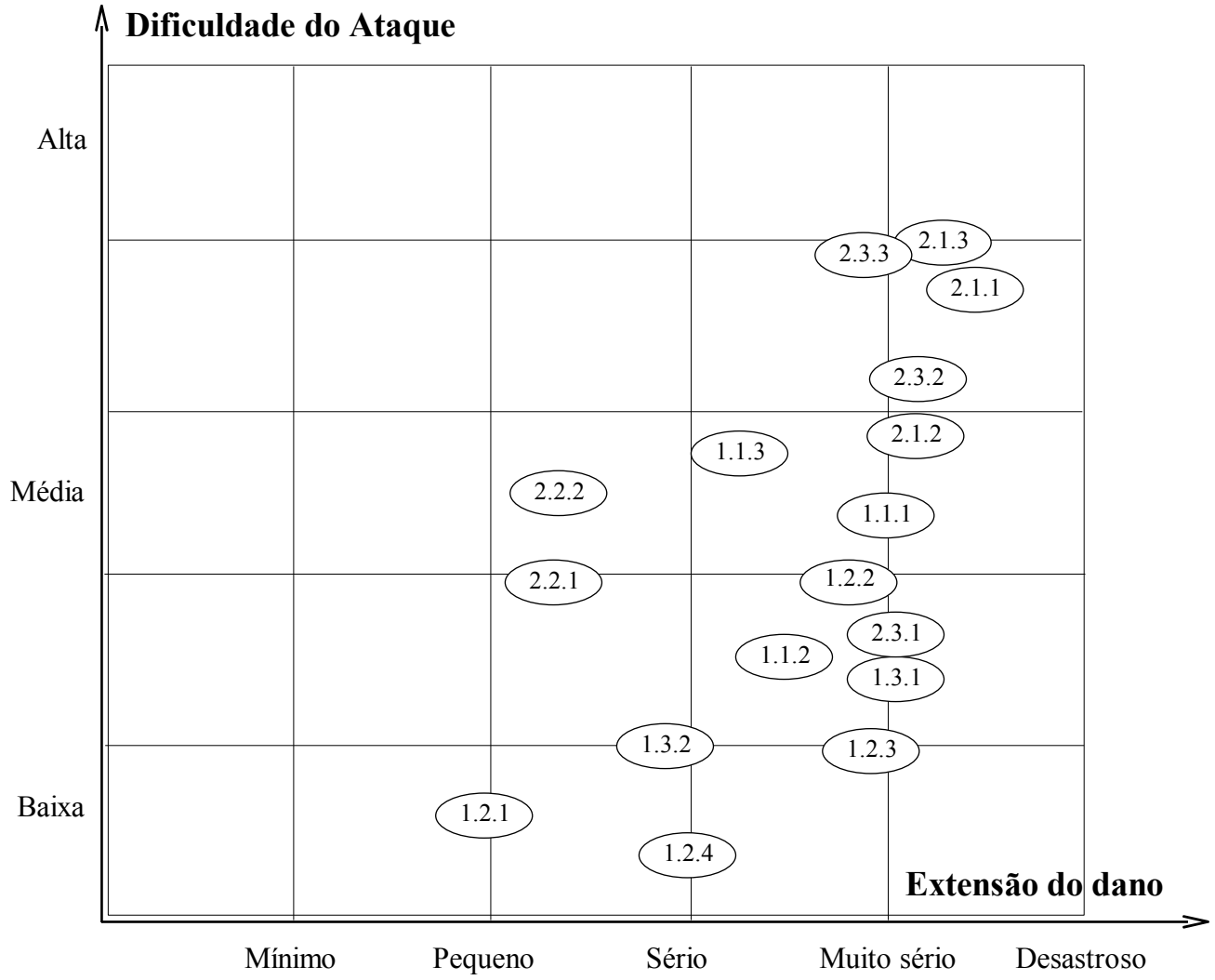


Detalhes:

a - RFC 693 determina que um contador de 32 bits para **SN** deve ser incrementado a cada 4 μ seg, mas nas implementações do TCP dos sistemas Unix BSD, **SN** é somado 1 a cada seg ou 128 a cada chamada.

b - Conjugando-se este ataque aos de roteamento (RIP, *Source Routing*), fraudes ou espelhamento de sessões TCP completas podem ser feitas.

Análise comparativa de riscos (Internet)



1- Ataques internos

1.1- Acesso indevido

1.1.1- Falha de autenticação
1.1.2- NFS
1.1.3- X-windows

1.2- Sabotagem

1.2.1- Sobrecarga
1.2.2- Source Routing, RIP, ICMP
1.2.3- Depredação física
1.2.4- Virus

1.3- Vazamento

1.3.1- *Sniffers*
1.3.2- Engenharia Social

2- Ataques externos

2.1- Acesso indevido

2.1.1- Ataque de número sequencial TCP
Source Routing, RIP, ICMP
Sequestro de sessão TCP

2.2- Sabotagem

2.2.1- Sobrecarga
2.2.2- Fragmentação

2.3- Vazamento

Sniffers
Troianos
Exploits

Fonte: Othmar Kyas - *Internet Security*, 1997

Controles de segurança para a Internet

- **Planejamento integrado -**

As técnicas de proteção para redes TCP/IP, sem a qual investimentos em conectividade poderão não compensar riscos decorrentes, podem amplificar ou neutralizar mutuamente suas funcionalidades, dependendo de como seu uso for planejado e integrado.

- **Mecanismos básicos de controle -**

As técnicas de proteção às redes TCP/IP, sem as quais os protocolos criptográficos podem ser inócuos, devem ser implementadas segundo planejamento de segurança que especifica a natureza do tráfego esperado:

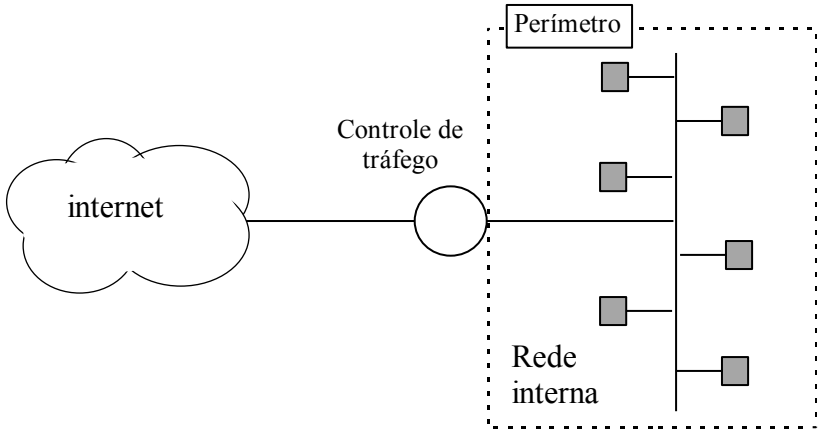
1. *Demarcação do(s) perímetro(s) de segurança da(s) subredes;*
2. *Seleção e alocação criteriosa dos serviços demandados e oferecidos;*
3. *Definição do roteamento e estimativa de volume de tráfego;*
4. *Especificação da filtragem de pacotes;*
5. *Desenho de topologia adequada para as escolhas anteriores;*
6. *Instalação, configuração e testes de filtros, utilitários, patches e demais serviços de segurança, de forma modular e em rede piloto.*
7. *Configuração das contas administrativas nos servidores e hosts;*
8. *Instalação e configuração dos serviços em ambiente de produção*;
9. *Treinamento, acompanhamento e monitoramento dos logs*;

A filtragem deve incluir todo tráfego que cruza o perímetro da rede interna.

Infra-estrutura para controle de tráfego

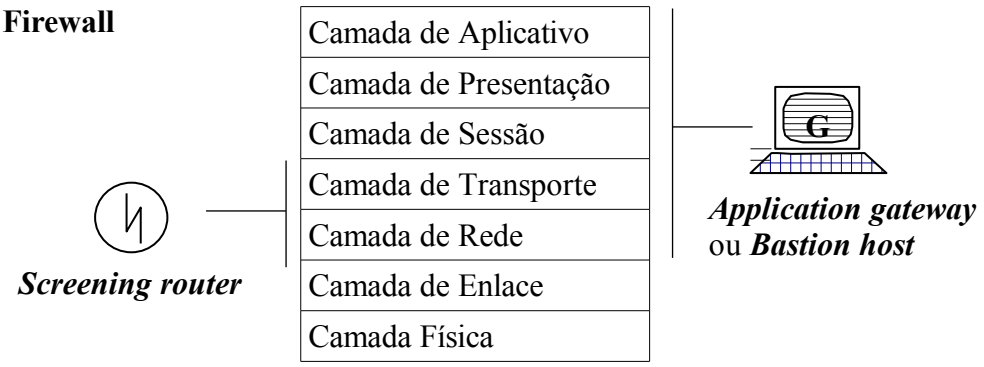
- **Perímetros de segurança -**

A demarcação dos pontos de entrada e saída de tráfego da rede interna para a internet (ou entre subredes) permite a instalação de mecanismos de controle que filtram pacotes e delimitam a área de ação de outros mecanismos de segurança. Por outro lado a filtragem pode introduzir pontos singulares de falha e impactar na capacidade de vazão deste tráfego.



- **Controladores de tráfego -**

O controle de tráfego é feito basicamente, a nível de transporte por um roteador de triagem com componente de filtro (*screening router*), ou a nível de aplicativo por um *gateway* ou *proxy* de aplicativo e/ou uma estação guardiã (*bastion host*). Estes dispositivos tem sido chamados de ***firewalls***.



Técnicas de filtragem

- ## Critérios -

A filtragem de pacotes TCP/IP é guiada por listas de regras. Uma regra de filtragem descreve critérios de decisão, tipicamente baseados em:

- **direção do tráfego:**.....*rede interna para externa, vice versa*
- **interface:**.....*subredes ou enlace de origem e de destino*
- **tipo de protocolo:**.....*IP, ICMP, TCP, UDP, IPX, etc.*
- **endereços:**.....*endereço IP de origem e de destino do pacote*
- **portas:**.....*número de porta de origem e de destino.*
- **Informação sobre o estado do TCP:** *ACK, SYN, RST, PSH, SN, etc*

- ## Mecanismo de filtragem -

1 Operações - qualquer regra determina uma das seguintes ações:

- ***permit***.....*O pacote segue rota se satisfaz as condições descritas*
- ***block***.....*O pacote é descartado se satisfaz as condições descritas*

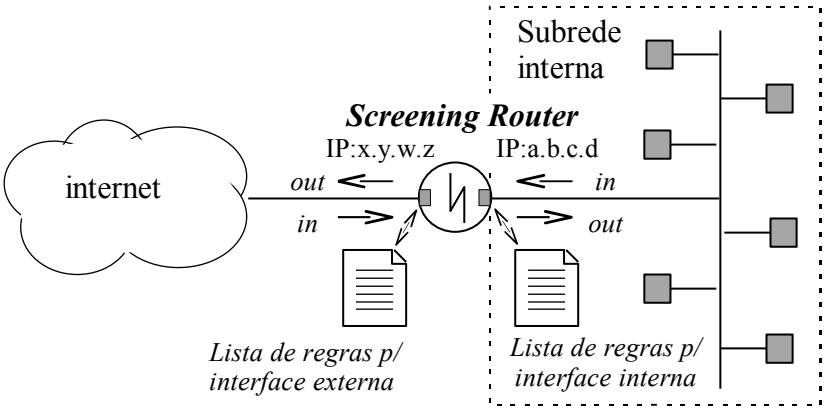
2 Análise - cada pacote é seguidamente submetido às regras de uma lista ordenada, lida de um arquivo (geralmente em formato texto), até que alguma regra determine uma ação sobre o pacote. Se o fim da lista for atingido, é tomada a ação *default* do filtro (geralmente o descarte).

3 Reações - uma regra acionada e/ou um estado do filtro podem acionar:

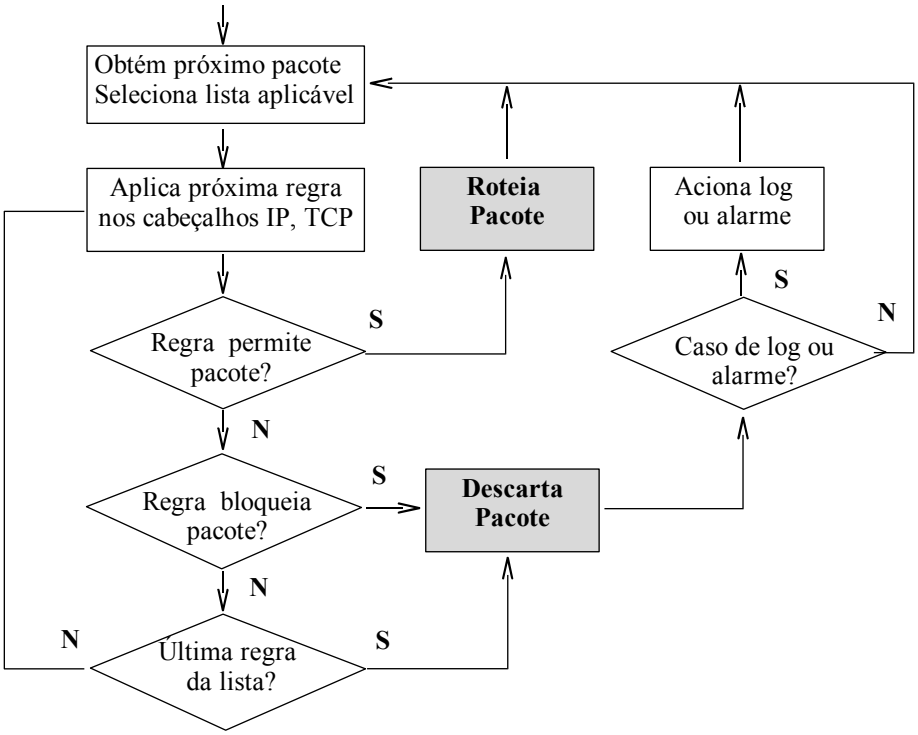
- **log** *pacote descartado mais estado do filtro é registrado.*
- **alarme**.....*e-mail, beep ou mensagem de emergência é disparado.*

Operações de filtragem

- **Modelo básico de filtro de pacotes -**



- **Funcionamento de um filtro típico -**



Exemplo de tabela com lista de regras para filtro TCP:

regra nº	Ação	IP origem	porta orig	IP destino	porta dest	opções IP	flags TCP
1	block	*	*	198.10.12.3	80	3	*
2	permit	200.0.0.0	*	*	25	*	ACK

Arquitetura de *firewalls*

1 Permeabilidade -

Como os serviços TCP são *full duplex,* as regras de filtragem devem distinguir entre o tráfego de serviço demandado e o de serviço oferecido:

- **serviços demandados**.....*iniciado na rede interna* (*mais permeável*)
- **serviços oferecidos**.....*iniciado na rede externa* (*maiores riscos*)

2 Granularidade -

Filtros podem diferir no conjunto de parâmetros que compõem as regras de filtragem, com impacto na capacidade de isolamento de padrões de tráfego.

- **interface**: *enlace de entrada ou saída* (*para filtragem de spoof*)
- **associação**: *protocolo, endereço e porta de entrada e de saída*

3 Complexidade de análise -

Um filtro de pacotes com lista estática de regras não garante segurança na oferta e demanda de alguns serviços que lhe entrecruzem, tais como:

- **transferência de arquivos** .....*protocolo FTP*;
- **X-Windows***protocolo X-11*

Se a granularidade e topologia desejadas demandarem, o filtro pode incluir tabelas dinâmicas, onde são mantidos estados das sessões ativas que filtra.

4 Funcionalidade -

Filtros de pacotes que mantém em tempo real informações sobre sessões exigem ambientes de sistema, e são chamados ***proxies*** ou ***gateways.***

Exemplos de configuração de filtragem

- **Selecionando tráfego para subredes -**

Suponha a política de segurança que especifique permissão de tráfego para as subredes da rede 198.2.0.0 exceto a subrede 198.2.3.0, para a qual o tráfego deve ser bloqueado com exceção daquele destinado ao *host* 198.2.3.4. A seguinte lista de regras implementaria sua filtragem:

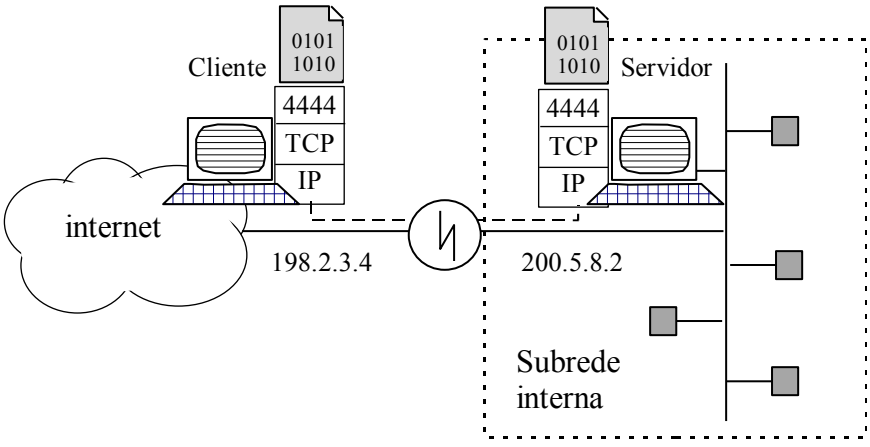
regra nº	Ação	IP origem	porta orig	IP destino	porta dest	protocolo	flags
1	permit	*	*	198.2.3.4	*	TCP	*
2	block	*	*	198.2.3.0	*	TCP	*
3	permit	*	*	198.2.0.0	*	TCP	*

- **Serviço customizado entre dois *hosts* -**

Suponha que uma empresa desenvolveu software proprietário que usa a mesma porta TCP no cliente e no servidor (ex: 4444), e sua política de segurança exige permissão em hosts especificados para cliente e servidor.

- **Num *firewall* que filtra associações:**

regra nº	Ação	IP origem	porta orig	IP destino	porta dest	protocolo	flags
1	permit	198.2.3.4	4444	200.5.8.2	4444	TCP	*

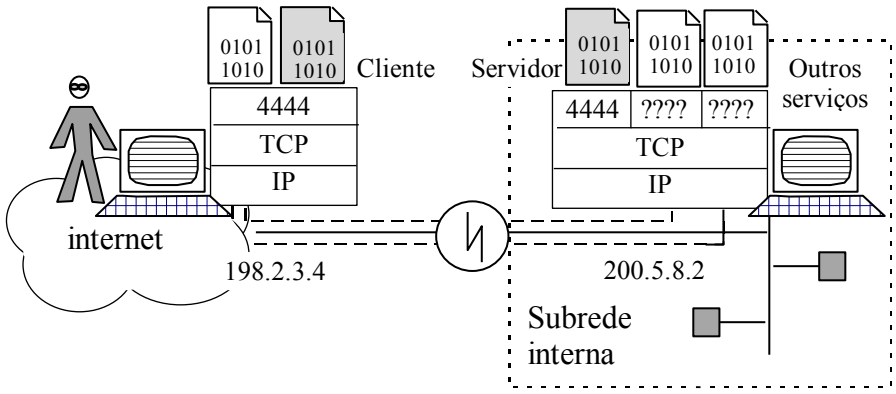


• Serviço customizado entre dois *hosts* (continuação) -

A granularidade de configuração pode afetar sutilmente a capacidade do filtro. Neste exemplo, uma lista que parece atender à especificação de filtragem pode falhar, abrindo acesso não especificado ao host do servidor.

- **Num *firewall* que filtra apenas uma porta:**

regra nº	Ação	IP origem	comentario	IP destino	porta orig	protocolo	flags
1	permit	198.2.3.4	cliente→serv	200.5.8.2	4444	TCP	*
2	permit	200.5.8.2	serv→cliente	198.2.3.4	4444	TCP	*

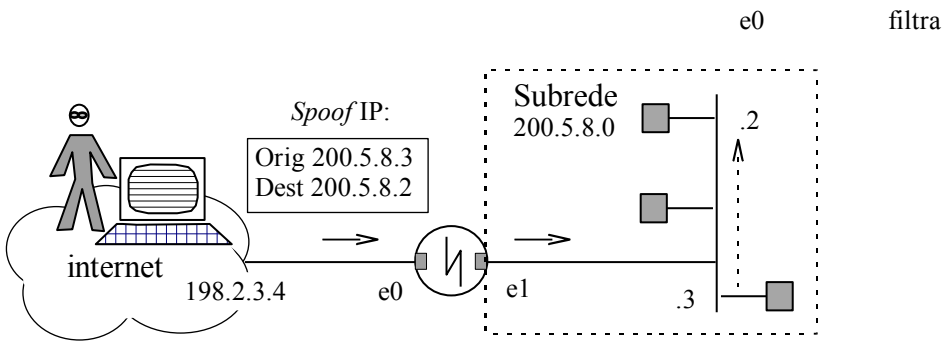


• Filtrando *spoofing* de endereço interno -

Conforme funcione o filtro, bloqueia-se na saída da interface interna ou na entrada da interface externa, endereços de origem da subrede interna.

- **Num *firewall* com apenas duas interfaces:**

regra	Ação	IP origem	port orig	IP destino	port dest	interface	comentário
1	block	200.5.8.0	*	*	*	e1	filtra saída

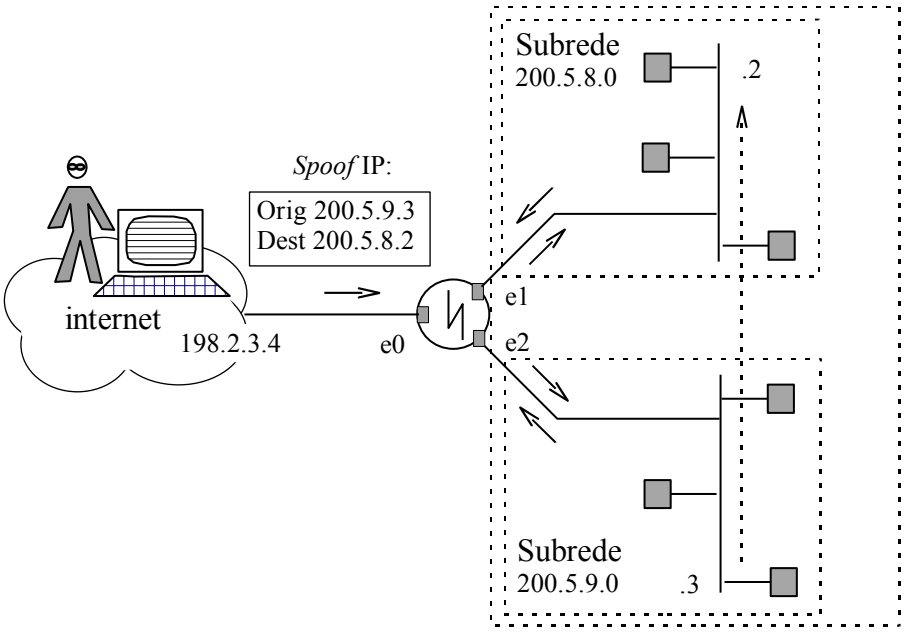


Bloqueando *spoofing* de endereço interno (continuação) -

Num roteador ou *bastion host* com múltiplas interfaces, o bloqueio efetivo de *spoofing* de endereço interno exige do filtro granularidade para especificação da direção do tráfego a ser filtrado.

- **Num *firewall* com múltiplas interfaces:**

regra nº	Ação	IP origem	porta orig	IP destino	porta dest	interface	direçao
1	block	200.5.8.0	*	*	*	e0	in
2	block	200.5.9.0	*	*	*	e0	in



Se o *firewall* só filtra na saída da interface, neste caso perde informação sobre o enlace por onde entra o pacote, e a tentativa de filtrar *spoofing* de endereço interno isolará o tráfego entre subredes internas:

regra	Ação	IP origem	port orig	IP destino	port dest	interface	comentário
1	block	200.5.8.0	*	*	*	e1	filtra saída
2	block	200.5.9.0	*	*	*	e1	filtra saída
3	block	200.5.8.0	*	*	*	e2	filtra saída
4	block	200.5.9.0	*	*	*	e2	filtra saída

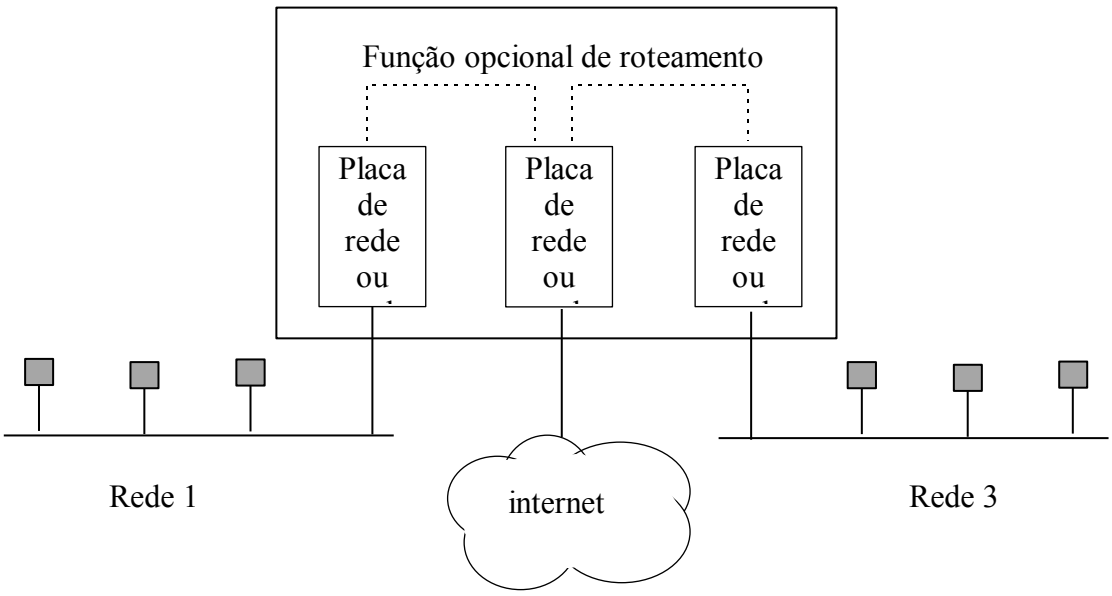
200.5.9.3 não poderá por exemplo estabelecer sessão TCP com 200.5.8.2.

Application gateways

- ***Gateway* de base múltipla (*Dual, multi homed hosts*)**

Computadores que conectam tráfego entre redes por meio de funções de roteamento em uma aplicação que acesse mais de uma interface instalada.

3-*homed host*



- **Configuração da máquina hospedeira de base múltipla -**

O sistema operacional precisa ter seu roteamento automático entre interfaces (*ipforwarding*) desabilitado, e outras precauções necessárias:

- Execução de outros aplicativos que compartilham dados não deve ser nela permitida, pois poderiam desviar tráfego da função roteadora.
- O modo de falha do *gateway* deve ser um estado em que o tráfego é bloqueado. Neste caso o ponto singular de falha (bloqueio de serviço) é preferível à perda de funcionalidade de filtragem do tráfego.

- ## Servidores proxy

São vistos através de um perímetro de segurança como servidor ou cliente do serviço demandado ou oferecido, fazendo intermediação do serviço e acrescentando funcionalidade aos *gateways* de base múltipla.

Sua principal característica de segurança é permitir ocultar ao tráfego externo as máquinas do perímetro interno que hospedam serviços oferecidos, pois ataques baseados em escuta requerem visibilidade do alvo.

- **proxy de aplicação**.....*oferecem serviços de intermediação de acesso a uma aplicação específica, como ftp, telnet, www, etc.*
- **proxy genérico**.....*funcionam como retransmissores de pacote com filtragem a nível de portas baseada no estado das sessões TCP. Pressupõe o uso recomendado de portas para os WKS.*
- **proxy de circuito**.....*acrescentam funcionalidade aos proxys genéricos, estabelecendo um circuito virtual fim-a-fim entre o usuário da rede interna e vários serviços de um destino. Requerem a instalação de programas-cliente que reconheçam o intermediador* (ex.: SOCKS).

- ## Hospedeiro guardião (*Bastion host*)

Uma máquina que hospede um *gateway* ou função de roteamento em um ponto crítico para a segurança da rede, é chamado de ***bastion host***. Para que a proteção desejada seja efetiva, os serviços a serem providos pelo *bastion host* devem ser minima e criteriosamente selecionados.

Um bastion host deve possuir recursos redundantes de armazenamento, não dispor de ferramentas de programação, de contas e serviços desnecessários, nem de programas com permissão SUID e SGID (unix).

Limitações dos *firewalls*

A filtragem de pacotes não garante integridade, autenticidade, sigilo, nem proteção contra ameaças internas ou ataques por implantação: é apenas a primeira linha de defesa, limitada pelas características dos protocolos.

- **Uso de portas:**

  O bloqueio de serviço por filtragem baseado em número de porta não é efetivo, pois o vínculo de portas a serviços que utilizam o TCP está apenas convencionado para os *Well Known Services* (RFC 1700)

- **Portas privilegiadas:**

  A informação visível no cabeçalho do pacote TCP sobre o estado da sessão e o critério de portas privilegiadas (<1024) não são suficientes para bloqueios unidirecionais. (Ex: FTP).

- **Tunelamento:**

  Regras de filtragem podem ser subvertidas por ataques de fragmentação ou tunelamento (RFC 1858), se o tráfego dos serviços demandados ou oferecidos requerer a habilitação destes recursos.

- **Sequestro de sessão TCP:**

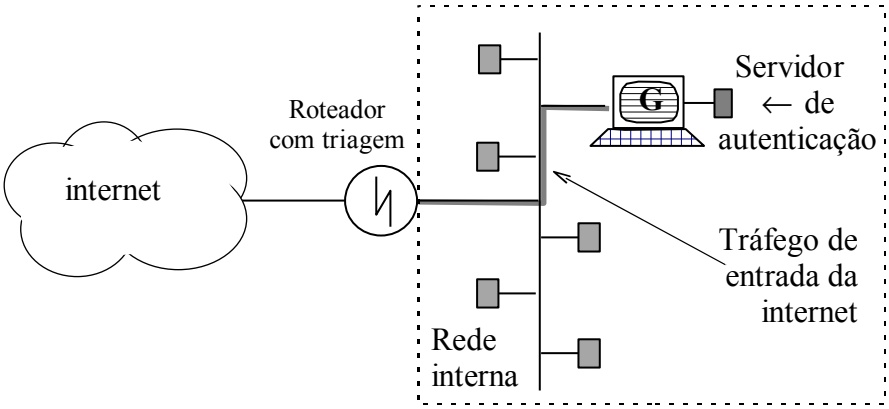
  Este tipo de ataque só pode ser evitado com uso de criptografia, viabilizada pela padronização dos certificados de chave pública, que possibilita autenticação contínua de sessões com envelopes digitais.

- **Exploits:**

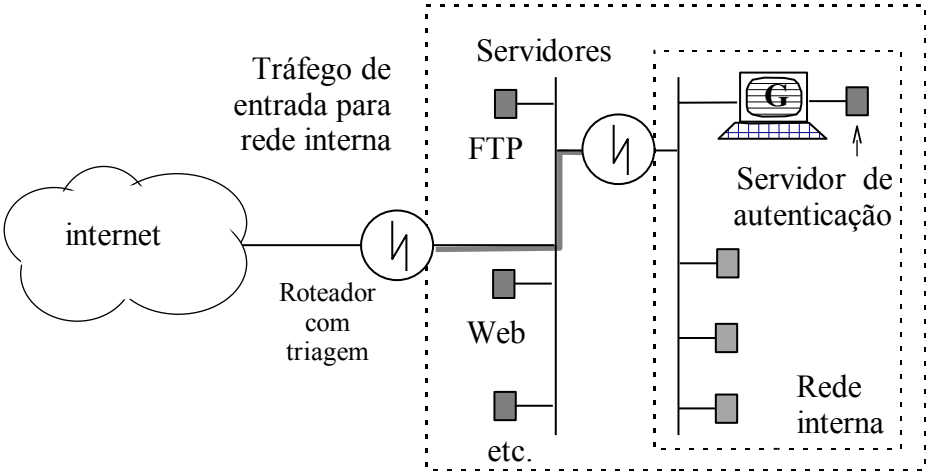
  A autenticação contínua de sessão pode ser subvertida em sistemas pouco seguros pelo vazamento de chaves privadas ou de sessão, através de ataques de *exploits* implantados via www ou e-mail.

Algumas topologias para controle de tráfego

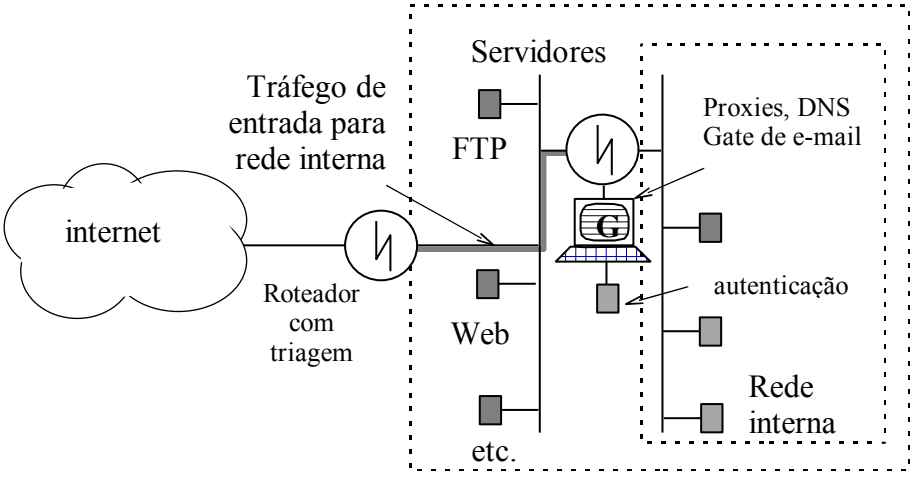
- ***Gateway host* com triagem -**



- **Sub-rede com triagem -**



- **Belt and suspenders -**



7: Exemplos de Aplicações

- **Histórico do e-mail -**

  O PGP foi o primeiro protocolo criptográfico concebido para uso na internet que fazia uso de chaves públicas. Implementado e distribuido pela internet por Phil Zimmerman em 1986, implementa o RSA e o IDEA para negociação de envelopes digitais na trasmissão de e-mail autenticado e encriptado. Devido ao uso de criptograifa robusta e infração de patentes, o autor teve problemas com a justiça americana, tendo posteriorment negociado acordos com as partes envolvidas e licenciado sua distribuição

- **Histórico do www -**

  O NCSA MOSAIC foi o primeiro navegador web a incorporar ganchos para adição de utilitários de segurança, tais como o PGP e as implementações de clientes SMTP com suporte à especificação PEM.

  Em 1994, com o impulso do comércio eletrônico na Web, a consolidação das tentativas de se incorporar sigilo e autenticação aos protocolos nela empregados foi iniciada. Já existem cerca de 50 esquemas de aplicação criptográfica para o comércio eletrônico implementados ou propostos na literatura, em 1998.

- **S-HTTP (Secure HTTP, 1994)**

  Desenvolvido e proposto pela CommerceNet Consortium, usa criptografia de chave pública para sessões http de forma comparável às do padrão PEM, sendo compatível com diversos gerenciadores de chave.

- **SSL (Secure Sockets Layer, 1994) -**

Proposto e implementado pela Netscape em seus *browsers* e servidores Web, sob licença da RSADSI, oferece autenticação de servidor e opcionalmente do cliente com criptografia para serviços genéricos, protegendo toda a pilha TCP/IP para os protocolos de aplicações.

Sua execução verifica certificados de chave pública e negocia a escolha de envelope digital entre cliente e servidor, podendo utilizar certificados X509, algoritmos Diffie-Hellman, RSA, DES, MD5 segundo padrões PKCS.

- **PCT (Private Communication Technology, 1995)**

Proposta da Microsoft para segurança do TCP/IP. Compatível e semelhante ao SSL, difere deste na implementação de um dos mecanismos de autenticação do SSL que continha falhas em sua versão 1.0.

- **SET (Secure Eletronic Transactions, 1996)**

Um dos esquemas criptográficos atualmente em uso para comércio eletrônico, foi concebido com o objetivo de se tornar padrão de fato no suporte da indústia financeira ao comérico eletrônico integrado.

Desenvolvido e proposto em conjunto pela IBM, Visa e MasterCard, faz uso extensivo de certificados X509, sendo um protocolo complexo de 28 passos que procura oferecer garantias adicionais ao consumidor, geralmente o agente mais exposto a vulnerabilidades nos esquemas criptográficos empregados no comércio eletrônico.

Suas vulnerabilidades se concentram em falsificações de certificados.

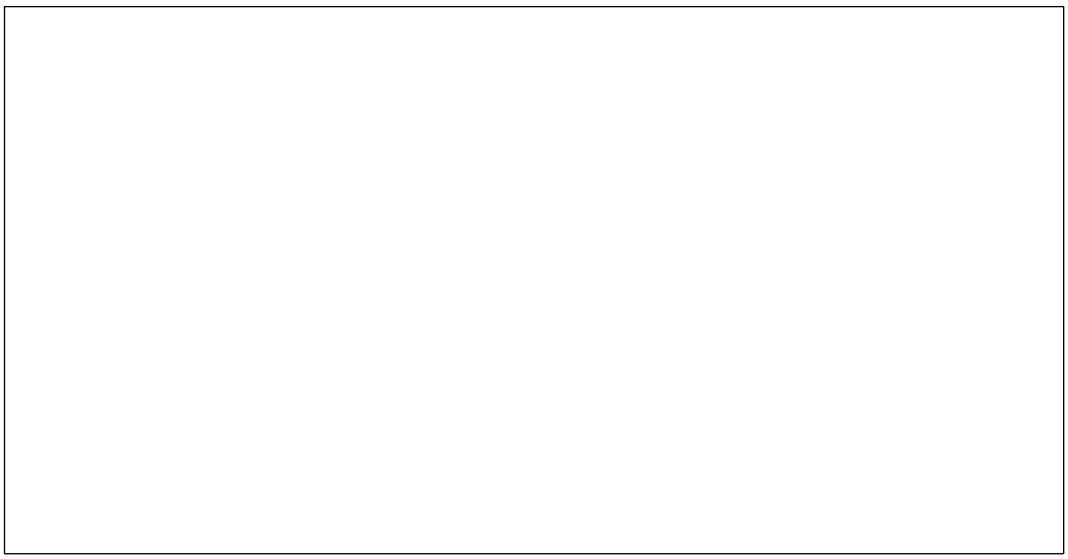
Descrição resumida do SET

- **Participantes devem possuir certificados X.509**

- **Consumidor valida certificados e submete pedido a C**

SET (continuação)

- **Comerciante não tem acesso a número de cartão do cliente**



- **Administradora só tem acesso ao valor da fatura**

Ferramentas e utilitários de segurança

- **Passwd+** (ftp://dartmouth.edu.pub.passw+.tar.Z)

  Um filtro de senhas que impede os titulares de contas unix de criarem senhas fracas, suceptíveis a ataques de dicionário.

- **Crack** (ftp://ftp.uu.net/usenet/comp.sources.misc/volume28/crack)

  Uma ferramenta para ataque de dicionário a arquivos de senhas unix.

- **COPS** (ftp://cert.org/pub/cops)

  Uma ferramenta para inspeção de segurança que verifica se o sistema unix está configurado de maneira segura (trust, rhost, /etc/passwd, etc.)

- **TCP Wrapper** (ftp://cert.org/pub/tools/tcp_wrappers)

  Uma ferramenta para gerenciamento de conexão e log.

- **Xinetd** (ftp://mystique.cs.colorado.edu/pub/xinetd

  Um substituto do inetd que inclui log e gerenciamento de conexões.

- **TAMU** (ftp://net.tamu.edu/pub/security/tamu)

  Uma ferramenta que contém filtro de pacotes, programas de verificação de configuração, de auditoria e de geração de log.

- **TIS Firewall toolkit** (ftp://ftp.tis.com/pub/firewalls/tollkit)

Kit de software para criação e manutenção de firewalls entre redes, distribuído em código- fonte em linguagem C.

- **SOCKS**
  (ftp://ftp.inoc.dl.nec.com/pub/security/socks.cstc.4.0.tar.gz)

  Implementação de um relay de circuitos para filrewalls.

- **WUarchive** (ftp://ftp.uu.net/networking/ftp/wuarchive-ftpd)

  Servidor mais usado na internet para ftp anônimo.

- **CGIWrap** (http://www.umr.edu/:tdciwrap)

  Wrapper de CGI para identificação de scripts de usuários em unix.

- **Web Server Comparison**
  (http://www.proper.com/servers-chat.html)

  Discussão sobre produtos disponíveis na internet para servidores web.

- **COAST archive** (ftp://cs.coast.purdue.edu)

  Repositório variado de programas e informações relacionadas à segurança na internet.

Atualização e informações sobre segurança na internet

- **Request For Comments** (http://www.isi.edu/rfc-editor/)

  Índice dos documentos de discussão de propostas de padrões para internet, e dos sítios onde podem ser acessados.

- **CERT** (http://www.cert.org)

  Computer Emergency Response Team, disponibiliza um conjunto atualizado de patches de segurança, alertas de segurança, e diversas ferramentas distribuídas pelos fornecedores.

- **Web Security FAQ**
  (http://www-genome.wi.mit.edu/WWW/faqs/www-security-faq)

  Roteiro para implementação de segurança em servidores web.

- **Great Circle Associates** (http://www.greatcircle.com)

  Fonte de informações atualizadas e tutoriais sobre firewalls e segurança na internet

- **AT&T Research archive** (ftp:/research.att.com)

  Repositório de informações sobre segurança na internet.