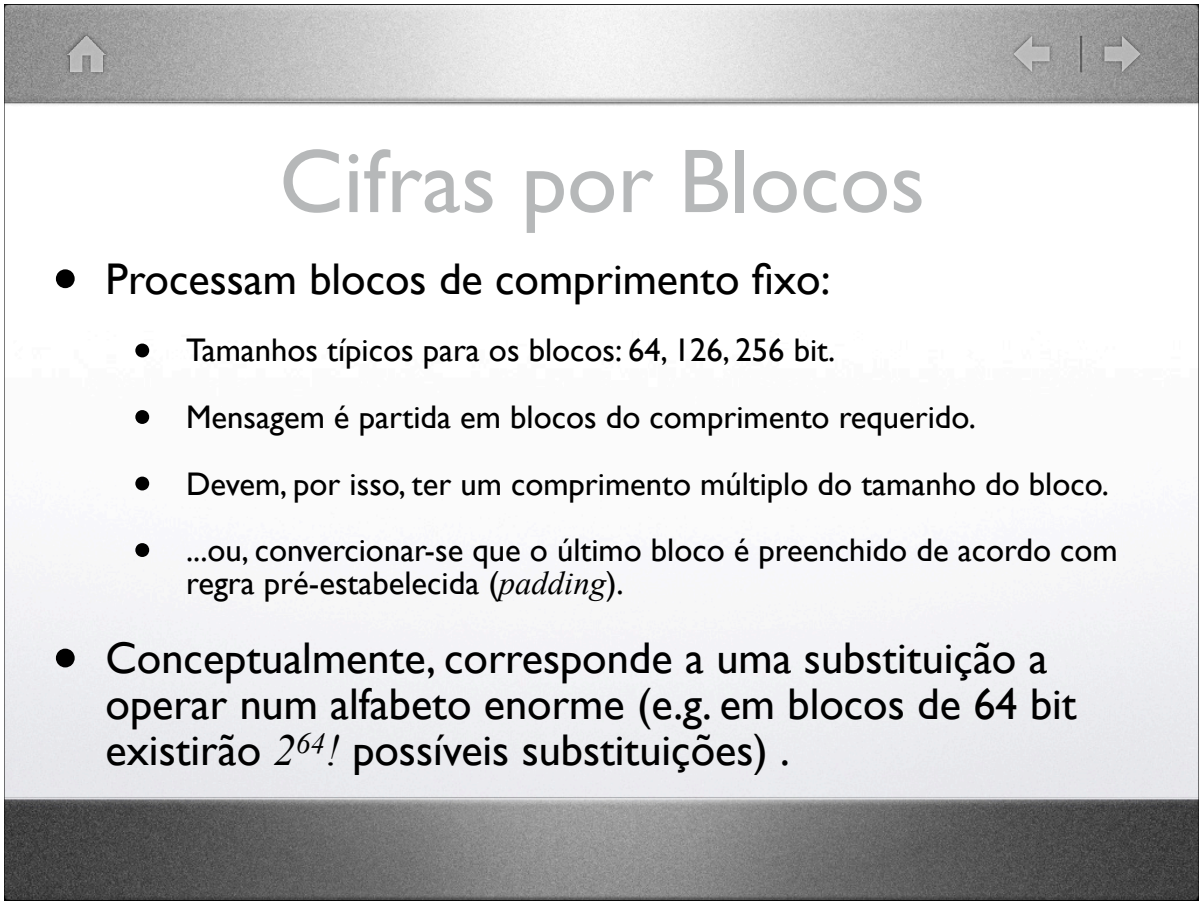
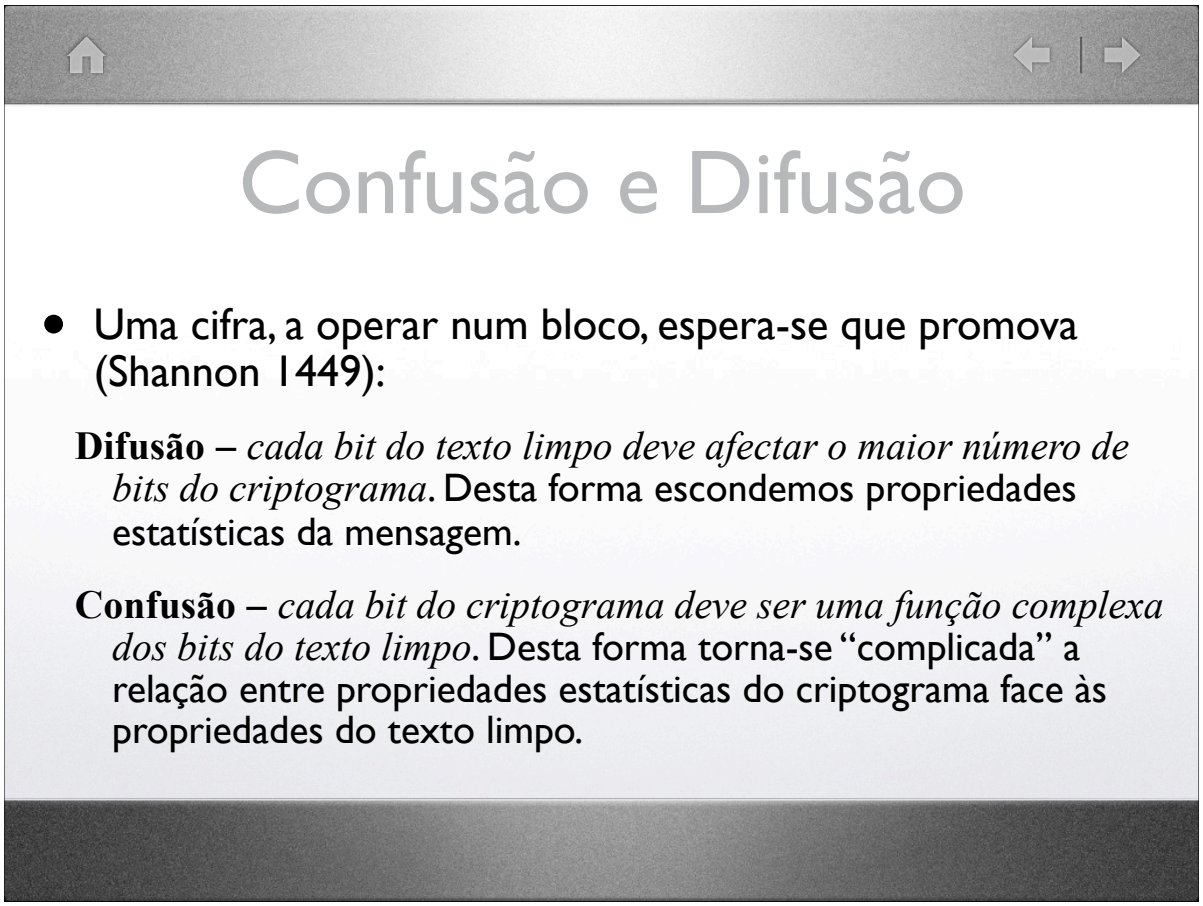

Cifras por Blocos
Criptografia Aplicada
Criptografia e Segurança da Informação


Cifras por Blocos
Processam blocos de comprimento fixo:
Tamanhos típicos para os blocos: 64, 126, 256 bit.
Mensagem é partida em blocos do comprimento requerido.
Devem, por isso, ter um comprimento múltiplo do tamanho do bloco.
...ou, convercionar-se que o último bloco é preenchido de acordo com regra pré-estabelecida (padding).
Conceptualmente, corresponde a uma substituição a operar num alfabeto enorme (e.g. em blocos de 64 bit existirão $2^{64}!$ possíveis substituições) .

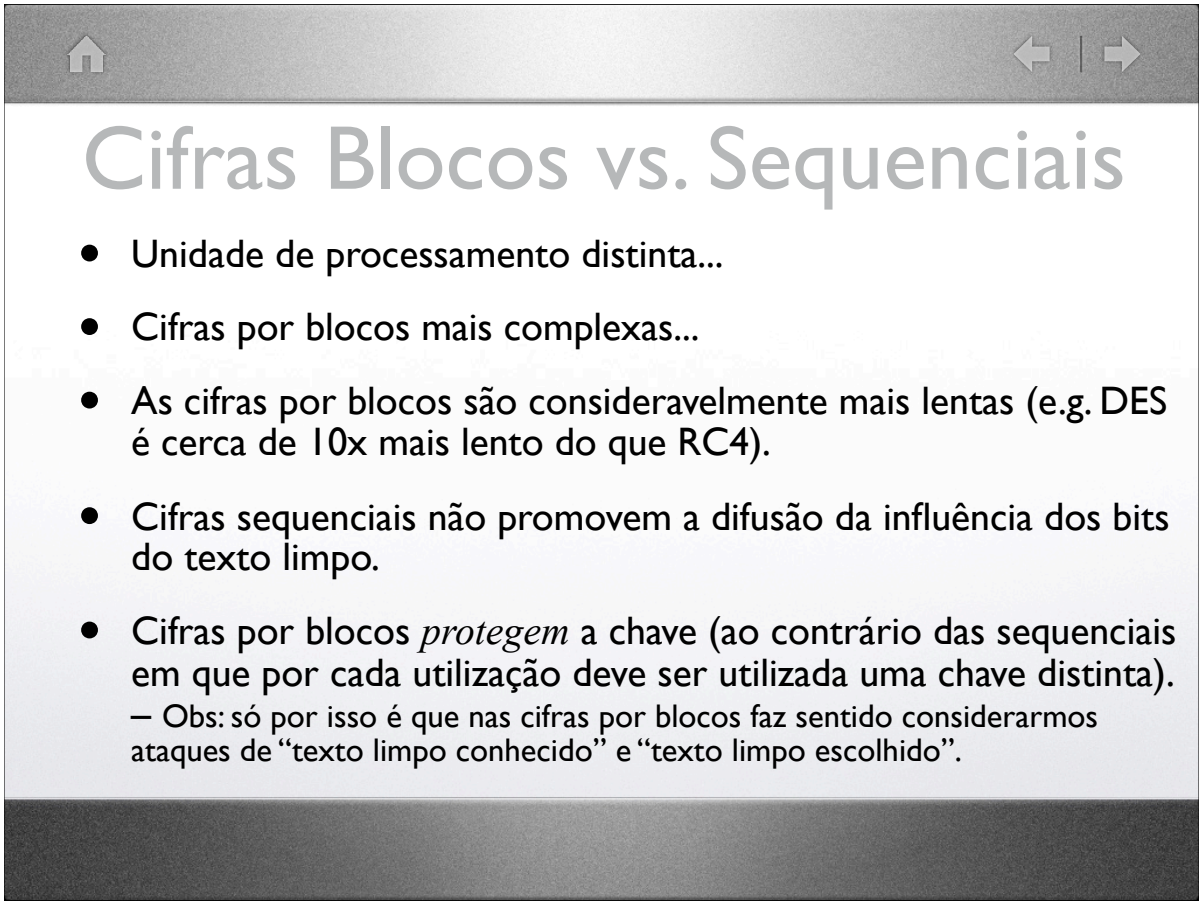


Confusão e Difusão

- Uma cifra, a operar num bloco, espera-se que promova (Shannon 1449):

Difusão – *cada bit do texto limpo deve afectar o maior número de bits do criptograma*. Desta forma escondemos propriedades estatísticas da mensagem.

Confusão – *cada bit do criptograma deve ser uma função complexa dos bits do texto limpo*. Desta forma torna-se "complicada" a relação entre propriedades estatísticas do criptograma face às propriedades do texto limpo.



Cifras Blocos vs. Sequenciais

- Unidade de processamento distinta...
- Cifras por blocos mais complexas...
- As cifras por blocos são consideravelmente mais lentas (e.g. DES é cerca de 10x mais lento do que RC4).
- Cifras sequenciais não promovem a difusão da influência dos bits do texto limpo.
- Cifras por blocos *protegem* a chave (ao contrário das sequenciais em que por cada utilização deve ser utilizada uma chave distinta).
  – Obs: só por isso é que nas cifras por blocos faz sentido considerarmos ataques de "texto limpo conhecido" e "texto limpo escolhido".

Padding

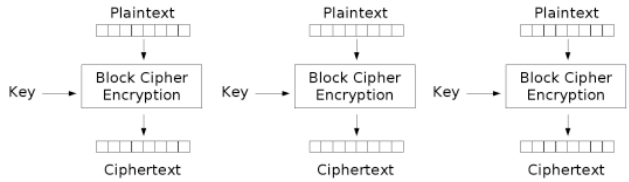
- Estratégia para completar o último bloco de texto-limpo...
- ...sem perder informação sobre comprimento efectivo da mensagem.
- Várias metodos:
  - bit 1 seguido de 0s;
  - *n* bytes *n;*
  - ...
- Por vezes (cifras assimétricas) é explorado para introduzir aleatoriedade na mensagem.

Modos de Operação

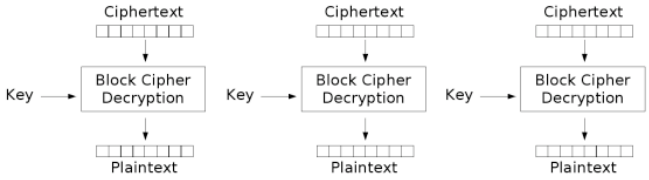
- Dependendo da aplicação, podemos cifrar uma mensagem com uma cifra por blocos de diversos **modos**:
  - Electronic Code Book (ECB)
  - Cipher Block Chaining (CBC)
  - Cipher FeedBack (CFB)
  - Output FeedBack (OFB)
  - Counter Mode (CTR)

...

Electronic Code Book (ECB)

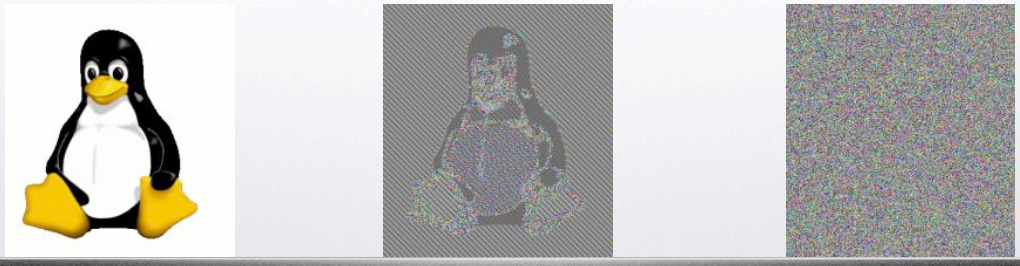


Electronic Codebook (ECB) mode encryption

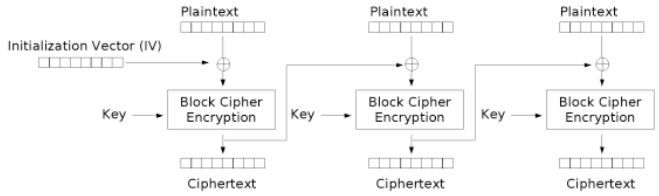


Electronic Codebook (ECB) mode decryption

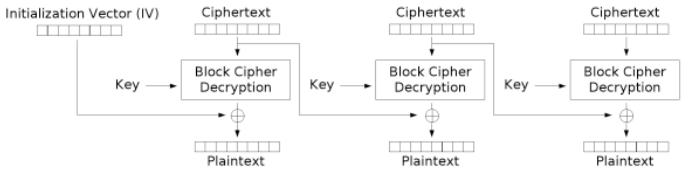
- Eventual repetição de blocos é detectável - *code book attack*.
- Vulnerável a ataques por repetição/substituição.
- Só deve ser utilizado para cifrar mensagem de um só bloco (ou poucos...).
- Um erro de um bit num bloco do criptograma afecta todo o bloco após a decifragem.



Cipher Block Chaining (CBC)



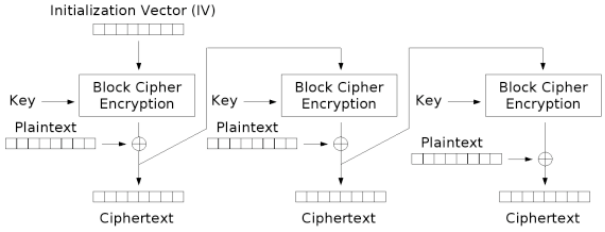
Cipher Block Chaining (CBC) mode encryption



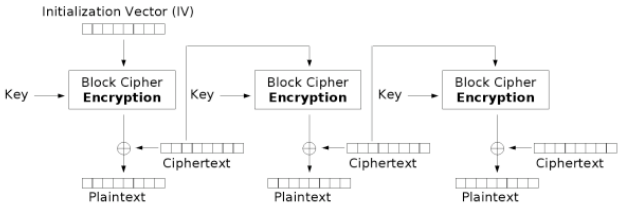
Cipher Block Chaining (CBC) mode decryption

- É utilizado um "vector de inicialização" (IV) previamente conhecido para iniciar o processo (fixo ou, preferencialmente, enviado cifrado em ECB). Obs.: se o IV for enviado em claro, o intruso pode alterar bits do primeiro bloco alterando os respectivos bits do IV. Obs2.: Se IV for fixo, os criptogramas de duas mensagens com um prefixo comum vão preservar essa propriedade.
- Um erro num bloco do criptograma corrompe dois blocos após a decifragem (mais precisamente, um bloco e um bit).
- Encadeamento do processo faz depender a operação de cifra de um bloco de todos os que o antecedem.
- Pode assim ser utilizado como MAC (utilizando somente o último bloco do criptograma). O problema é que pode ser um código muito pequeno... (e.g. 64 bit no DES).

Cipher FeedBack (CFB)



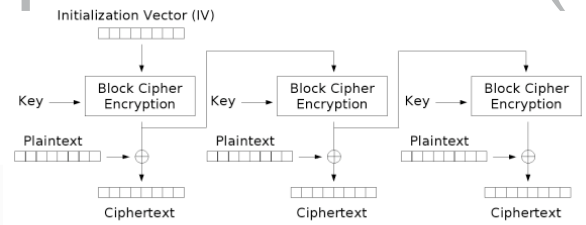
Cipher Feedback (CFB) mode encryption



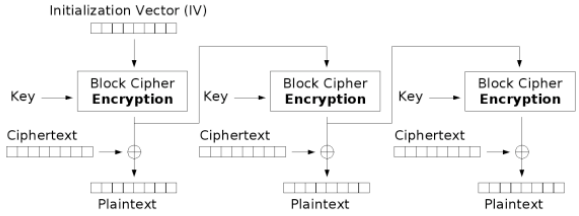
Cipher Feedback (CFB) mode decryption

- Modo que implementa uma cifra sequencial auto-sincronizável com uma cifra por blocos.
- Note-se que se utiliza sempre a operação de "cifrar".
- Número de bits no FeedBack é variável (*CFBn - standard* prevê *n*=1, 8 ou 64). A realimentação transfere os *n* bits mais significativos para os menos significativos (com *shift* dos restantes).
- IV deve ser único por cada utilização (c.f. reutilização de chaves em cifras sequenciais). E.g. pode ser enviado em claro.
- Um erro num bit do criptograma afecta o bit respectivo no bloco e todos do bloco seguinte.
- Sequência de chave depende do IV, chave da cifra e de todo o texto limpo já cifrado.

Output FeedBack (OFB)



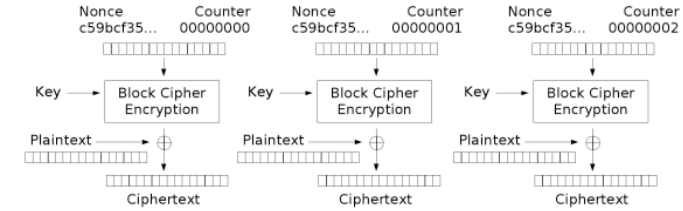
Output Feedback (OFB) mode encryption



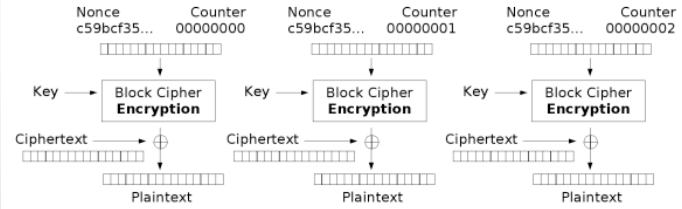
Output Feedback (OFB) mode decryption

- Modo que implementa uma cifra sequencial síncrona com uma cifra por blocos.
- Sequência de chave é obtida iterando a cifra sobre um bloco inicial (IV).
- Sequência de chave é independente da mensagem (pode assim ser processada independentemente de se ter já disponível a mensagem).
- Erros de bits no criptograma só afectam os respectivos bits na mensagem original.

Counter Mode (CTR)



Counter (CTR) mode encryption



Counter (CTR) mode decryption

- Tal como OFB, simula uma cifra sequencial síncrona (mas agora em *counter* mode).
- *Nonce* (IV) e *Counter* podem ser conjugados de diferentes formas (concatenados, xored, ...).
- Único requisito para o *Counter* é produzir valores distintos para todos os blocos (o mais simples é ser mesmo implementado como um contador).
- Não impõe dependência entre processamento dos vários blocos (podem ser processados em paralelo; acesso aleatório; ...)
- Análise teórica da respectiva segurança mais simples...
- Segurança análoga ao modo CBC.

Construção de uma cifra por blocos...

- Procurar implementar uma cifra por blocos directamente como uma substituição arbitrária não é nem exequível nem prático (pense-se no espaço de chaves...). Essa substituição "virtual" terá então de ser construída a partir de blocos mais simples. E.g.:
  - Substituições (de tamanho razoável)
  - Transposições

Substituição

- Uma substituição promove uma permutação entre os símbolos de um alfabeto.
- Quando vista sobre palavras em binário, pode ser entendida como uma permutação (troca de fios) entre um par descodificador/codificador. Obs.: note a correspondência exponencial entre o número de bits de entrada e o número de ligações envolvidas.

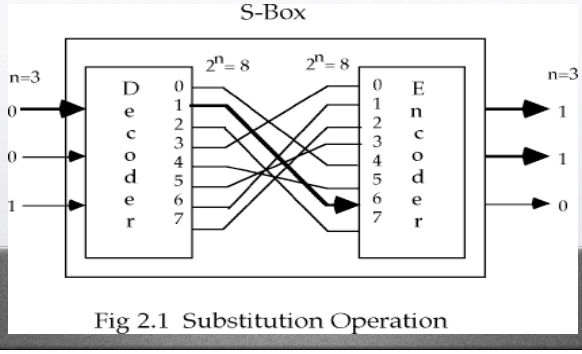


Fig 2.1 Substitution Operation

Permutação

- As transposições são simplesmente uma "troca de fios" entre os bits de entrada e os de saída.

- A sua implementação é muito simples (em *hardware*... já em *software* é uma operação particularmente frustrante) e é comportável operar sobre todo o bloco.

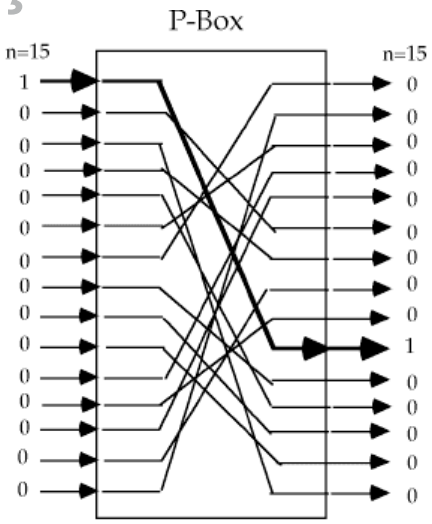


Fig 2.2 - Permutation or Transposition Functior

S-P networks

- Se utilizássemos unicamente uma das construções (substituições de tamanho comportável ou permutações) não teríamos sucesso na construção de uma cifra segura (ambas as técnicas constituem cifras *idempotentes*).

- ...mas, combinando ambas as técnicas, podemos construir uma "cifra produto" não idempotente e que designamos por *round.*

- Um *round,* por si só, não é suficientemente seguro. Mas quando iterado permite obter os níveis de segurança pretendidos.

- É esta a essência das designadas "S-P networks".

...mais um conceito introduzido no artigo basilar de C. Shanon (1949).

S-P *networks* (cont.)
efeito de avalanche...

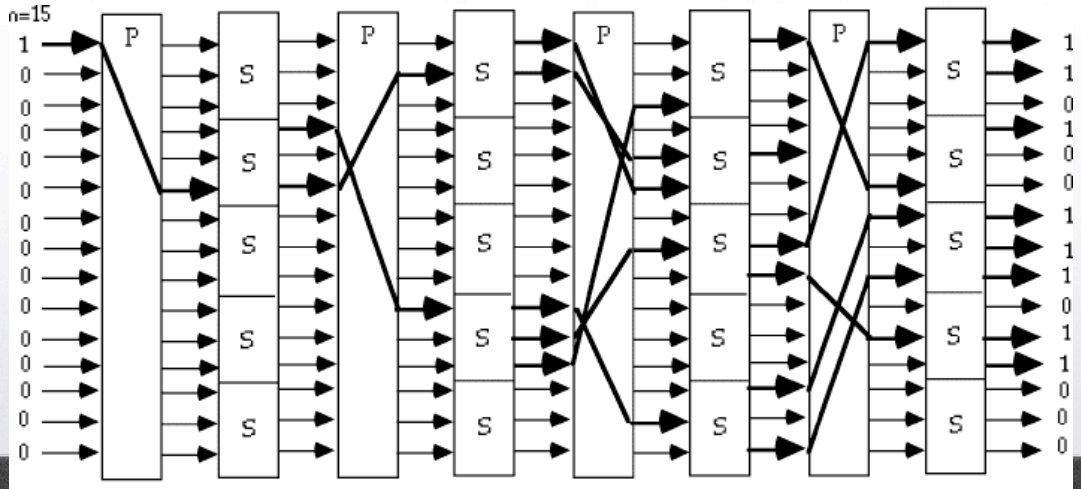


Fig 2.3 - Substitution-Permutation Network, with the Avalanche Characteristic

Circuitos de Feistel

- A utilização de uma S-P *network* numa cifra não é prática porque requer a construção de uma rede inversa para decifrar o criptograma (i.e. obriga à duplicação dos dispositivos/programas)
- Para evitar esse problema devemos encontrar formas expedita de desenhar um *round* por forma o mesmo procedimento seja na sua "inversa"
- Um **circuito de Feistel** é um exemplo de uma tal forma expedita de desenho dos *rounds*: cada bloco é partido em dois sub-blocos - num é aplicada a transformação e o outro é preservado. Os sub-blocos são ainda trocados para que o *round* seguinte afecte agora o sub-bloco que ficou inalterado.

Circuitos de Feistel

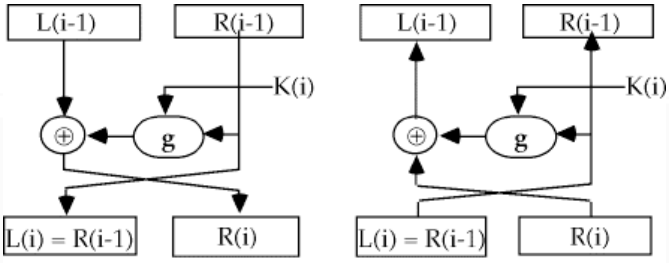


Fig 2.4 - A Round of a Feistel Cipher

$L(i)=R(i-1)$; $R(i)=L(i-1) \oplus g(K(i),R(i-1))$

- Um *round* pode ser considerado a sua própria operação inversa se realizarmos algumas "trocas" nos sub-blocos...

E.g. Lucifer Cipher

- Exemplo da utilização de circuitos de Feistel na cifra Lucifer (o antecessor do DES).
- Da chave da cifra são extraídas 16 sub-chaves (uma por cada *round*)

...note o padrão S-P na construção do *round*...

...e que no último *round* não se procede à troco dos blocos...

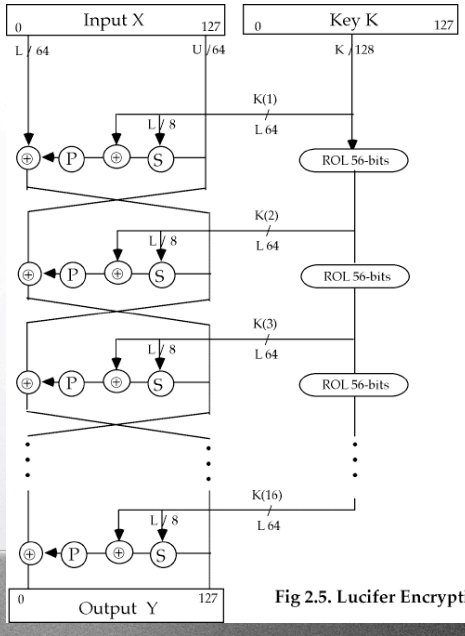


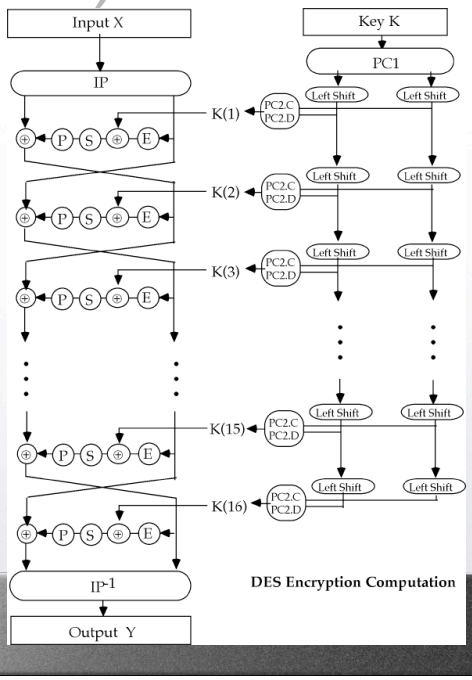
Fig 2.5. Lucifer Encryption Computation

Data Encryption Standard (DES)

- À chamada para propostas de algoritmos de cifras pela NBS (1973) a IBM propôs o LUCIFER.
- O algoritmo foi escolhido e desencadeou-se um processo refinamento (com colaboração da NSA) que resultou Standard DES (1975).
- O algoritmo foi tornado público, mas não os critérios para o seu desenho... Este facto alimentou alguma controvérsia porque, aparentemente, foram enfraquecidos alguns aspectos do LUCIFER relevantes para a segurança (tamanho da chave; tamanho do bloco). Trabalhos publicados posteriormente pela comunidade cientifica vieram demonstrar que as alterações introduzidas foram, de facto, melhoramentos substanciais na segurança do algoritmo.
- Desenvolvimentos recentes em técnicas de cripto-análise (e.g. Cripto-análise diferencial) deparam-se com uma estranha imunidade do DES. Membros da equipa de desenvolvimento do DES afirmaram não se tratar de um facto gratuito já que essas técnicas eram já do conhecimento da NSA...
- (ainda...) De longe, algoritmo de cifra mais utilizado (particularmente pela banca).

DES (cont.)

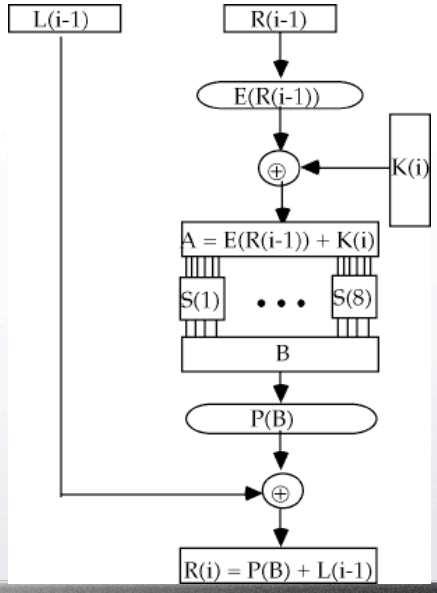
- Ao bloco (64 bit) é aplicada uma permutação inicial e partido em dois sub-blocos (32 bit).
- Seguem-se 16 *rounds* segundo a estrutura típica determinada por um circuito de Feistel.
- A chave (56 bit) é processada para produzir 16 chaves de *round* (48 bit).
- O algoritmo conclui-se com a inversão da permutação inicial.



Round do DES

- Ao sub-bloco a transformar (32 bit) é aplicada uma **permutação de expansão** para 48 bit (alguns bits são duplicados).
- 8 **S-boxes** distintas realizam a operação não linear do *round*.
- Cada **S-box** processa 6 bit de entrada, produzindo 4 de saída. O bloco (*B*) volta a dispor de 32 bit.
- O *round* finaliza com uma permutação.

Obs.: A expansão no *round* DES permite acelerar o *efeito de avalanche*.



S-boxes do DES

- Cada *S-box* é normalmente apresentada como uma matriz de 4 linhas e 16 colunas
- E.g. *S-box 1:*
  - entrada: $x_1..x_6$ (A=x_1x_6 B=$x_2x_3x_4x_5$)
  - saída: y ($y_1y_2y_3y_4$)

y	B															
A	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

- Podemos entender uma *S-box* como 4 permutações independentes dos bits $x_2x_3x_4x_5$, sendo a selecção da permutação da responsabilidade dos bit x_1x_6.

Propriedades e características do DES

- Complementaridade: $\overline{\mathrm{E}_K(X)} = \mathrm{E}_{\overline{K}}(\overline{X})$
- DES não é um grupo: dadas duas chaves K_1 e K_2, não é em geral possível determinar K_3 tal que $\mathrm{E}_{K_3}(X) = \mathrm{E}_{K_2}(\mathrm{E}_{K_1}(X))$
- O DES foi desenhado para permitir implementações eficientes em *hardware.* Em contrapartida, exibe algumas operações particularmente ingratas para implementações em *software* (em particular, permutações e manipulações várias ao nível do bit).

Chaves fracas

- Dado o mecanismo de escalonamento de chaves do DES (produção das chaves de *round* a partir da chave inicial), existem algumas chaves com problemas de segurança associados:
  - dão origem a chaves $K_1..K_{16}$ todas iguais - **chaves fracas**.
  - dão origem a apenas duas sub-chaves diferentes - **chaves semi-fracas**.
  - dão origem a apenas 4 sub-chaves diferentes - **chaves possivelmente fracas**.
- Em chaves fracas (destes três tipos), temos que Ki=K16-i. Assim a operação de cifra é uma involução (cifrar duas vezes permite recuperar o texto original). Estas chaves dão origem a pontos fixos (f(x)=x) e a anti pontos fixos (f(x)=neg(x)).

Obs.: Todos estes conjuntos de chaves estão tabelados, pelo que basta a implementação assegurar que não é escolhida nenhuma destas chaves para não incorrer nos problemas de segurança associados.

Cifra múltipla

- Dado que DES não é um grupo, é razoável aumentar a segurança iterando a operação de cifra.
- Cifra dupla ($E_{K2}(E_{K1}(X))$) oferece segurança análoga à cifra simples - ataque *meet-in-the-middle*.
  - Sabendo que $C=E_{K2}(E_{K1}(P))$,
    - Construímos tabela com $E_K(P)$, para toda a chave K
    - Basta-nos agora encontrar K2, verificando se $D_{K2}(C)$ se encontra na tabela construída.
    - Par determinado é verificado com outro par...
    - ...mas obriga a uma utilização enorme de memória...
- ...por isso, é normal utilizar-se cifra tripla.

Triple DES

- Encadeiam-se operações de cifra com decifragem:

$$E_{(K1,K2,K3)}(X)=E_{K3}(D_{K2}(E_{K1}(X)))$$

- Chave dispõe agora de 168 bit.
- Quando K1=K2=K3=K, é simplesmente uma cifra DES simples.
- Nível de segurança é análogo a duas chaves (modificação de ataque *meet-in-the-middle*).
- Também disponível numa versão com apenas duas chaves:

$$E_{(K1,K2)}(X)=E_{K1}(D_{K2}(E_{K1}(X)))$$

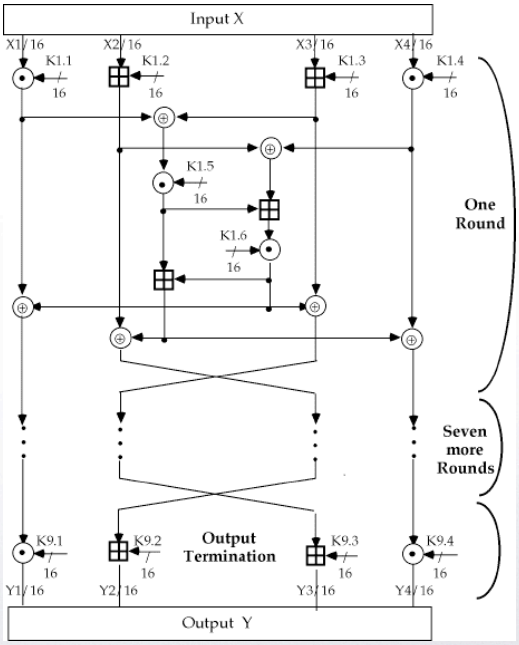

Outras cifras por blocos

- Baseadas em circuitos de Feistel:
  - LOKI (64 bit/bloco; 64 bit/chave)
  - Blowfish (64 bit/bloco; tam. chave variável)
  - ...
- Outros desenhos:
  - IDEA (...)
  - RC5 (tam. bloco, chave e nº *rounds* variável - muito eficiente)
  - AES (...)
  - ...


International Data Encryption Algorithm (IDEA)

- Desenvolvido por *James Massey & Xuejia Lai* (ETH Zurich) - 1990/1/2.
- Utiliza blocos de 64 bit e chaves de 128 bit.
- Utiliza três operações (grupos algébricos) escolhidas pela sua "incompatibilidade"
  - Xor
  - Adição módulo 2^{16}
  - Multiplicação módulo $2^{16}+1$ ($2^{16}+1$ representado por 0)
- Particularmente adaptado para realizações em *Software* (operações em palavras de 16 bit, onde só a multiplicação necessita de código específico)
- Não utiliza permutações (não segue o padrão das S/P-*networks*)

- Bloco é particionado em 4 sub-blocos de 16 bit.
- Cada *round* dispõe de uma *fase difusora* (cada sub-bloco é processado independentemente) e uma *fase misturadora* (onde os sub-blocos se interferem mutuamente).
- Programador de chaves produz 52 sub-chaves (8*6+4).
- Decifragem só difere no programador de chaves (são produzidas as chaves inversas pela ordem adequada).

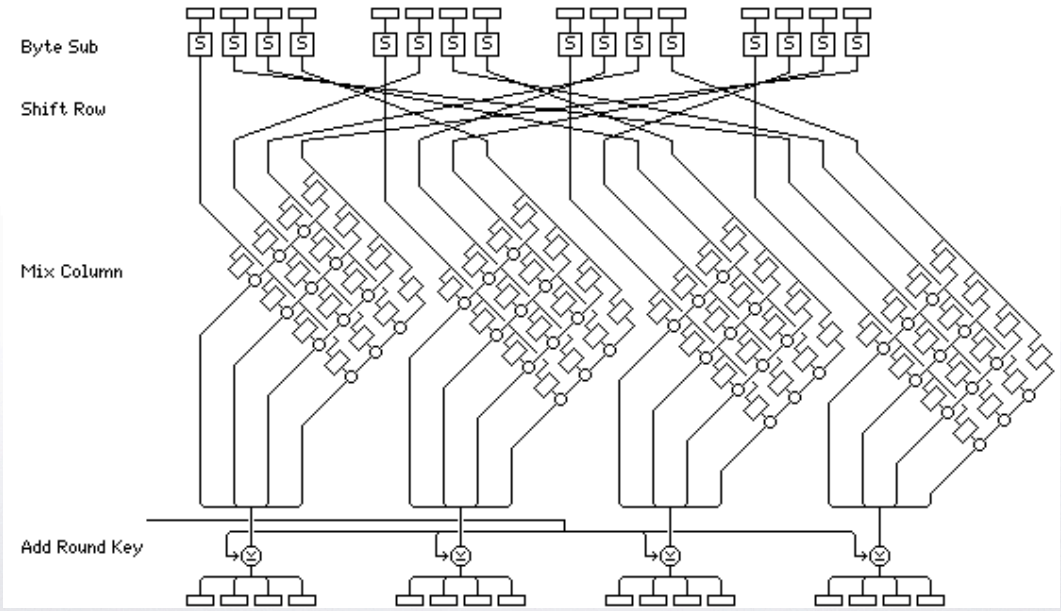


Advanced Encryption Standard (AES)

- Chamada para algoritmos criptográficos pela NIST para elaboração de um standard para substituir o DES - anúncio:1997, submissões: 1998, decisão 2000(1).
- Requisitos:
  - 128+ bit/bloco; 128+ bit chave
  - Mais seguro e mais rápido do que TripleDES
  - Especificação e detalhes de desenho completos.
  - Realizações em C e Java.
- Finalistas:
  - **MARS** - complexo, rápido, alta margem de segurança
  - **RC6** - muito simples, muito rápido, pequena margem de segurança
  - **Rijndael** - limpo, rápido, boa margem de segurança **(VENCEDOR)**
  - **Serpent** - lento, limpo, muito alta margem de segurança
  - **Twofish** - complexo, muito rápido, alta margem de segurança

AES (cont.)

- Desenhado por ***Rij**men**-Dae**men* (Bélgica).
- *Rounds* compostos por 4 camadas:
  - *Byte Substitution*
  - *Shift Rows*
  - *Mix Columns*
  - *Key Addition*
- Todas as operações podem ser realizadas por *Xor* e *lookup* a tabelas (que permite realizações muito eficientes em SW).
- Desenho dispõe de uma forte fundamentação matemática baseada em *corpos finitos* (GF(2^8), GF(2^4),...).
- Particularmente adaptado para processadores modernos (e *smart-cards).*



Rijndael - descrição

$a_{0,0}$	$a_{0,1}$	$a_{0,2}$	$a_{0,3}$	$a_{0,4}$	$a_{0,5}$	$a_{0,6}$	$a_{0,7}$
$a_{1,0}$	$a_{1,1}$	$a_{1,2}$	$a_{1,3}$	$a_{1,4}$	$a_{1,5}$	$a_{1,6}$	$a_{1,7}$
$a_{2,0}$	$a_{2,1}$	$a_{2,2}$	$a_{2,3}$	$a_{2,4}$	$a_{2,5}$	$a_{2,6}$	$a_{2,7}$
$a_{3,0}$	$a_{3,1}$	$a_{3,2}$	$a_{3,3}$	$a_{3,4}$	$a_{3,5}$	$a_{3,6}$	$a_{3,7}$

$k_{0,0}$	$k_{0,1}$	$k_{0,2}$	$k_{0,3}$	$k_{0,4}$	$k_{0,5}$	$k_{0,6}$	$k_{0,7}$
$k_{1,0}$	$k_{1,1}$	$k_{1,2}$	$k_{1,3}$	$k_{1,4}$	$k_{1,5}$	$k_{1,6}$	$k_{1,7}$
$k_{2,0}$	$k_{2,1}$	$k_{2,2}$	$k_{2,3}$	$k_{2,4}$	$k_{2,5}$	$k_{2,6}$	$k_{2,7}$
$k_{3,0}$	$k_{3,1}$	$k_{3,2}$	$k_{3,3}$	$k_{3,4}$	$k_{3,5}$	$k_{3,6}$	$k_{3,7}$

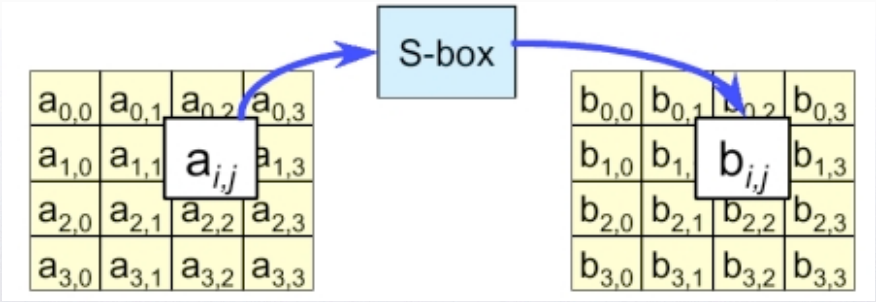
Nr	Nb = 4	Nb = 6	Nb = 8
Nk = 4	10	12	14
Nk = 6	12	12	14
Nk = 8	14	14	14

Number of rounds (Nr) as a function of the block and key length.

- Tamanho de bloco variável (16, 24, 32 bytes) - i.e. 128, 192, 256 bit.
- Tamanho da chave variável (e independente do tamanho do bloco)
- Número de *rounds* dependente do tamanho da chave e do bloco...

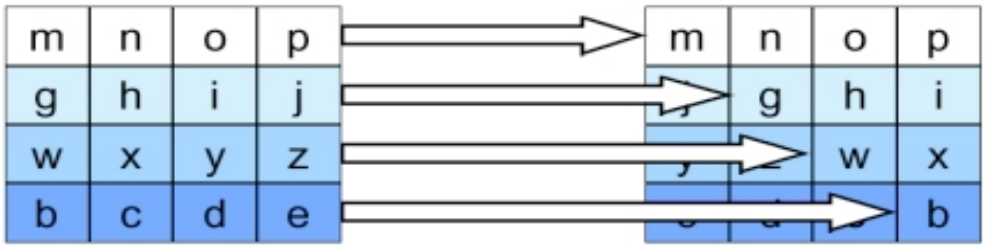
AES - descrição (cont.)

- ByteSub:
  - Fortemente não linear
  - Uma única S-box



AES - descrição (cont.)

- ShiftRow:

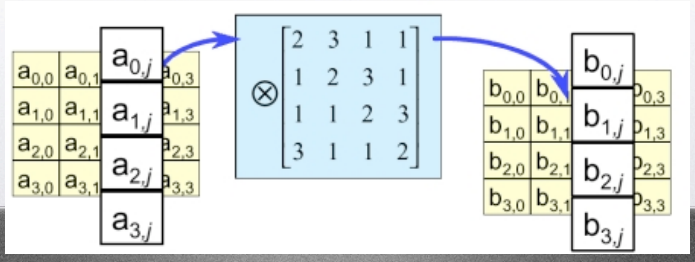


Nb	C1	C2	C3
4	1	2	3
6	1	2	3
8	1	3	4

Table 2: Shift offsets for different block lengths.

AES - descrição (cont.)

- Mix column:
  - Alta difusão intra-coluna
  - Interacção com *ShiftRow* (alta difusão em múltiplos *rounds*)
  - c(x) = '03'x^3+'01'x^2+'01'x+'02'



AES - descrição (cont.)

- Key addition:

$$\begin{bmatrix} a_{0,0} & a_{0,1} & a_{0,2} & a_{0,3} \\ a_{1,0} & a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,0} & a_{2,1} & a_{2,2} & a_{2,3} \\ a_{3,0} & a_{3,1} & a_{3,2} & a_{3,3} \end{bmatrix} + \begin{bmatrix} k_{0,0} & k_{0,1} & k_{0,2} & k_{0,3} \\ k_{1,0} & k_{1,1} & k_{1,2} & k_{1,3} \\ k_{2,0} & k_{2,1} & k_{2,2} & k_{2,3} \\ k_{3,0} & k_{3,1} & k_{3,2} & k_{3,3} \end{bmatrix} = \begin{bmatrix} b_{0,0} & b_{0,1} & b_{0,2} & b_{0,3} \\ b_{1,0} & b_{1,1} & b_{1,2} & b_{1,3} \\ b_{2,0} & b_{2,1} & b_{2,2} & b_{2,3} \\ b_{3,0} & b_{3,1} & b_{3,2} & b_{3,3} \end{bmatrix}$$

AES - descrição (cont.)

- Round transformation...

$$\begin{bmatrix} e_{0,j} \\ e_{1,j} \\ e_{2,j} \\ e_{3,j} \end{bmatrix} = \begin{bmatrix} d_{0,j} \\ d_{1,j} \\ d_{2,j} \\ d_{3,j} \end{bmatrix} \oplus \begin{bmatrix} k_{0,j} \\ k_{1,j} \\ k_{2,j} \\ k_{3,j} \end{bmatrix} \text{ and } \begin{bmatrix} d_{0,j} \\ d_{1,j} \\ d_{2,j} \\ d_{3,j} \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} c_{0,j} \\ c_{1,j} \\ c_{2,j} \\ c_{3,j} \end{bmatrix}.$$

For the ShiftRow and the ByteSub transformations, we have:

$$\begin{bmatrix} c_{0,j} \\ c_{1,j} \\ c_{2,j} \\ c_{3,j} \end{bmatrix} = \begin{bmatrix} b_{0,j} \\ b_{1,j-C1} \\ b_{2,j-C2} \\ b_{3,j-C3} \end{bmatrix} \text{ and } b_{i,j} = \mathrm{S}\left[a_{i,j}\right].$$

In this expression the column indices must be taken modulo `Nb`. By substitution, the above expressions can be combined into:

$$\begin{bmatrix} e_{0,j} \\ e_{1,j} \\ e_{2,j} \\ e_{3,j} \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} \mathrm{S}\left[a_{0,j}\right] \\ \mathrm{S}\left[a_{1,j-C1}\right] \\ \mathrm{S}\left[a_{2,j-C2}\right] \\ \mathrm{S}\left[a_{3,j-C3}\right] \end{bmatrix} \oplus \begin{bmatrix} k_{0,j} \\ k_{1,j} \\ k_{2,j} \\ k_{3,j} \end{bmatrix}.$$

AES - descrição (cont.)

The matrix multiplication can be expressed as a linear a combination of vectors:

$$\begin{bmatrix} e_{0,j} \\ e_{1,j} \\ e_{2,j} \\ e_{3,j} \end{bmatrix} = \mathrm{S}[a_{0,j}] \begin{bmatrix} 02 \\ 01 \\ 01 \\ 03 \end{bmatrix} \oplus \mathrm{S}[a_{1,j-C1}] \begin{bmatrix} 03 \\ 02 \\ 01 \\ 01 \end{bmatrix} \oplus \mathrm{S}[a_{2,j-C2}] \begin{bmatrix} 01 \\ 03 \\ 02 \\ 01 \end{bmatrix} \oplus \mathrm{S}[a_{3,j-C3}] \begin{bmatrix} 01 \\ 01 \\ 03 \\ 02 \end{bmatrix} \oplus \begin{bmatrix} k_{0,j} \\ k_{1,j} \\ k_{2,j} \\ k_{3,j} \end{bmatrix}.$$

The multiplication factors S[$a_{i,j}$] of the four vectors are obtained by performing a table lookup on input bytes $a_{i,j}$ in the S-box table S[256].

We define tables T_0 to T_3 :

$$\mathrm{T}_0[a] = \begin{bmatrix} \mathrm{S}[a] \bullet 02 \\ \mathrm{S}[a] \\ \mathrm{S}[a] \\ \mathrm{S}[a] \bullet 03 \end{bmatrix} \quad \mathrm{T}_1[a] = \begin{bmatrix} \mathrm{S}[a] \bullet 03 \\ \mathrm{S}[a] \bullet 02 \\ \mathrm{S}[a] \\ \mathrm{S}[a] \end{bmatrix} \quad T_2[a] = \begin{bmatrix} \mathrm{S}[a] \\ \mathrm{S}[a] \bullet 03 \\ \mathrm{S}[a] \bullet 02 \\ \mathrm{S}[a] \end{bmatrix} \quad \mathrm{T}_3[a] = \begin{bmatrix} \mathrm{S}[a] \\ \mathrm{S}[a] \\ \mathrm{S}[a] \bullet 03 \\ \mathrm{S}[a] \bullet 02 \end{bmatrix}.$$

These are 4 tables with 256 4-byte word entries and make up for 4KByte of total space. Using these tables, the round transformation can be expressed as:

$$e_j = \mathrm{T}_0[a_{0,j}] \oplus \mathrm{T}_1[a_{1,j-C1}] \oplus \mathrm{T}_2[a_{2,j-C2}] \oplus \mathrm{T}_3[a_{3,j-C3}] \oplus k_j \,.$$

Técnicas de Cripto-Análise

- Força Bruta – na prática inviável... (e.g. DES dispõe de 2^{56} hipóteses de chaves)
- Ataques à estrutura do algoritmo:
  - Cripto-análise Diferencial
    - Ataque de texto-limpo escolhido. Compara o comportamento da cifra para blocos com diferenças determinadas.
  - Cripto-análise Linear
    - Ataque de texto-limpo conhecido. Aproxima as *S-boxes* por funções lineares.
  - Cripto-análise Algébrica
    - Ataque de texto-limpo conhecido. Explora a estrutura algébrica da cifra.
- Ataques à implementação:
  - *Side-Channel attacks*
    - Timming information, power consumption, electromagnetic leaks, etc.

Cripto-análise Diferencial

- Desenvolvida em 1990 por *Eli Biham* e *Adi Shamir* para atacar o DES.
- Ataque de texto limpo escolhido.
- Analisa as diferenças no processamento das *S-boxes* do último *round* por forma a determinar a chave utilizada nesse *round*.
- Restantes bits da chave (8) podem ser obtidos por força bruta...

- No desenho das *S-boxes* existiu a preocupação de garantir que é produzida uma saída aleatória face a uma entrada aleatória.
- Mas quando comparamos a diferença dos resultados de uma S-box para pares de entradas com uma diferença determinada o resultado não é mais aleatório.
- Exemplo: para S-1 - quando analisamos a frequência da diferença dos resultados nos pares de entrada com diferença 0x01 (64 possíveis pares) obtemos:

	Output														
Input	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
01	0	0	0	6	0	2	4	4	0	10	12	4	10	6	2	4

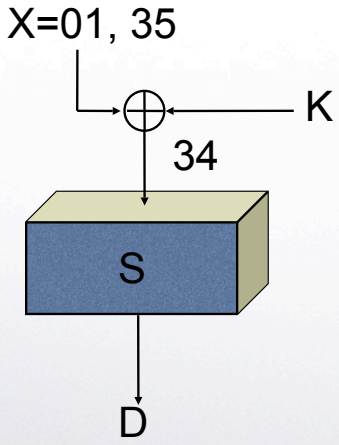
- ...este comportamento está longe de ser aleatório: muitas das possibilidades de resultado estão excluidas (0,1,2,4,8) e podemos identificar uma maior probabilidade de ocorrer *A* como diferença dos resultados.
- Dado que as *S-boxes* são fixas, podemos construir uma tabela que agregue a informação referida...

Características dos *rounds*

Input XOR	Output XOR 0_	1_	2_	3_	4_	5_	6_	7_	8_	9_	A_	B_	C_	D_	E_	F_
0_	64	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1_	0	0	0	6	0	2	4	4	0	10	12	4	10	6	2	4
2_	0	0	0	8	0	4	4	4	0	6	8	6	12	6	4	2
3_	14	4	2	2	10	6	4	2	6	4	4	0	2	2	2	0
4_	0	0	0	6	0	10	10	6	0	4	6	4	2	8	6	2
5_	4	8	6	2	2	4	4	2	0	4	4	0	12	2	4	6
6_	0	4	2	4	8	2	6	2	8	4	4	2	4	2	0	12
7_	2	4	10	4	0	4	8	4	2	4	8	2	2	2	4	4
8_	0	0	0	12	0	8	8	4	0	6	2	8	8	2	2	4
9_	10	2	4	0	2	4	6	0	2	2	8	0	10	0	2	12
A_	0	8	6	2	2	8	6	0	6	4	6	0	4	0	2	10
B_	2	4	0	10	2	2	4	0	2	6	2	6	6	4	2	12
C_	0	0	0	8	0	6	6	0	0	6	6	4	6	6	14	2
D_	6	6	4	8	4	8	2	6	0	6	4	6	0	2	0	2
E_	0	4	8	8	6	6	4	0	6	6	4	0	0	4	0	8
F_	2	0	2	4	4	6	4	2	4	8	2	2	2	6	8	8
10_	0	0	0	0	0	0	2	14	0	6	6	12	4	6	8	6
⋮																
27_	10	4	2	0	2	4	2	0	4	8	0	4	8	8	4	4
28_	12	2	2	8	2	6	12	0	0	2	6	0	4	0	6	2
29_	4	2	2	10	0	2	4	0	0	14	10	2	4	6	0	4
2A_	4	2	4	6	0	2	8	2	2	14	2	6	2	6	2	2
2B_	12	2	2	2	4	6	6	2	0	2	6	2	6	0	8	4
2C_	4	2	2	4	0	2	10	4	2	2	4	8	8	4	2	6
2D_	6	2	6	2	8	4	4	4	2	4	6	0	8	2	0	6
2E_	6	6	2	2	0	2	4	6	4	0	6	2	12	2	6	4
2F_	2	2	2	2	2	6	8	8	2	4	4	6	8	2	4	2
30_	0	4	6	0	12	6	2	2	8	2	4	4	6	2	2	4
31_	4	8	2	10	2	2	2	2	6	0	0	2	2	4	10	8
32_	4	2	6	4	4	2	2	4	6	6	4	8	2	2	8	0
33_	4	4	6	2	10	8	4	2	4	0	2	2	4	6	2	4
34_	0	8	16	6	2	0	0	12	6	0	0	0	0	8	0	6
35_	2	2	4	0	8	0	0	0	14	4	6	8	0	2	14	0
36_	2	6	2	2	8	0	2	2	4	2	6	8	6	4	10	0
37_	2	2	12	4	2	4	4	10	4	4	2	6	0	2	2	4
38_	0	6	2	2	2	0	2	2	4	6	4	4	4	6	10	10
39_	6	2	2	4	12	6	4	8	4	0	2	4	2	4	4	0
3A_	6	4	6	4	6	8	0	6	2	2	6	2	2	6	4	0
3B_	2	6	4	0	0	2	4	6	4	6	8	6	4	4	6	2
3C_	0	10	4	0	12	0	4	2	6	0	4	12	4	4	2	0
3D_	0	8	6	2	2	6	0	8	4	4	0	4	0	12	4	4
3E_	4	8	2	2	2	4	4	14	4	2	0	2	0	8	4	4
3F_	4	8	4	2	4	0	2	4	4	2	4	8	8	6	2	2

- Tabela com comportamento diferencial para *S-box 1*.
- Uma leitura da tabela consiste em atribuir probabilidades da (diferença) das saidas em função da entrada (n/64). Dessa forma podemos modelar estatisticamente um *round* - a **característica do round** - que nos diz que dispomos de uma dada probabilidade de obter uma saída para uma determinada entrada.
- Podemos ainda "juntar" essas características por forma a construir características de *n-rounds*.

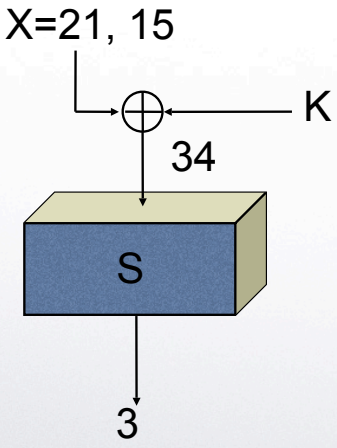
Exemplo



- Pela tabela verificamos que existem 8 formas de mapear 34 com D.
- Pela construção da tabela determinamos que K⊕X tem de ser: 06, 10, 16, 1C, 22, 24, 28, 32.

06 ⊕ 01 = 07	06 ⊕ 35 = 33
10 ⊕ 01 = 11	10 ⊕ 35 = 25
16 ⊕ 01 = 17	16 ⊕ 35 = 23
1C ⊕ 01 = 1D	1C ⊕ 35 = 29
22 ⊕ 01 = 23	22 ⊕ 35 = 17
24 ⊕ 01 = 25	24 ⊕ 35 = 11
28 ⊕ 01 = 29	28 ⊕ 35 = 1D
32 ⊕ 01 = 33	32 ⊕ 35 = 07

Exemplo (cont.)



- Um raciocínio análogo permite-nos concluir que...
- ... K⊕X só pode ser um dos valores 01, 02, 15, 21, 35, 36

01 ⊕ 21 = 20	01 ⊕ 15 = 14
02 ⊕ 21 = 23	02 ⊕ 15 = 17
15 ⊕ 21 = 34	15 ⊕ 15 = 00
21 ⊕ 21 = 00	21 ⊕ 15 = 34
35 ⊕ 21 = 14	35 ⊕ 15 = 20
36 ⊕ 21 = 17	36 ⊕ 15 = 23

- Chave: determinada pela intersecção dos conjuntos de possibilidades - "17".

Ataque a n-rounds

- A cripto-análise diferencial baseia-se no conhecimento das diferenças na entrada/saída das S-boxes...
- As diferenças nas entradas da S-box do último *round* são directamente calculadas dos criptogramas disponíveis.
- Já as diferenças na saida terão de ser calculadas com base em estimativas construídas por via das características...

Este é o motivo pelo qual a viabilidade da cripto-análise diferencial é fortemente influenciada pelo número de *rounds*.

- ...para 3 *rounds* é possível conduzir uma análise determinística. Para mais do que isso é necessário uma abordagem probabilística que aumenta muito o número de pares texto limpo/criptograma a analisar e complica substancialmente o ataque...

Resumo de resultados - Cripto-análise diferencial

Nº Rounds	Pares escolhidos
3	2/3
6	120
8	$\approx 2^{14}$
10	$\approx 2^{24}$
13	$\approx 2^{39}$
16	$\approx 2^{47}$

Cripto-análise Linear

- Desenvolvida em 1993 por *Mitsuru Matsui*.
- Utiliza aproximações lineares para descrever as acções da cifra.
- Explora "desvios" (*bias*) das probabilidades com que essas aproximações se verificam ($p \neq 1/2$).
- Exemplo: A acção da S-box 5 permite-nos derivar a aproximação (que se verifica com probabilidade 1/2-5/16)

$$K_{26}=X_{17} \oplus Y_3 \oplus Y_8 \oplus Y_{14} \oplus Y_{25}$$

- As aproximações lineares de diferentes *rounds* podem ser associadas...
- Permite recuperar uma chave DES com 2^{43} pares conhecidos...


Cripto-análise Algébrica

- Introduzido por *Nicolas Courtois* and *Josef Pieprzyk* em 2002.
- Aproxima a cifra por um enorme sistema equações quadráticas (e.g. AES é aproximado por 8000 equações com 1600 variáveis).
- Os autores propões um *método heurístico* para resolverem tais sistemas (aplicável em *toy-ciphers*, mas desapropriado para cifras como o AES).
- ...mas existe a esperança/risco de vir a estender esses métodos por forma a permitirem comprometer cifras actuais...
- Em geral, a comunidade científica mantém-se muito céptica relativamente à viabilidade deste ataque em cifras como o AES.


Side-Channel attacks

- Muitas técnicas de cripto-análise não atacam directamente o algoritmo criptográfico mas procuram retirar informação do "ambiente" onde este é executado...
  - Power analysis
  - Timming analysis
  - Differential fault analysis
  - ...
- Estas técnicas não são, de forma alguma, específicas do DES. Algumas são até particularmente adequadas para atacar cifras assimétricas...

Timming analysis...

- Analisa tempos de execução das operações criptográficas por forma a retirar informação sobre os parâmetros do algoritmo.
- Particularmente efectiva para algoritmos assimétricos, onde as operações são fortemente dependentes dos dados (e.g. Exponenciação modular)
- Cuidados especiais nas implementações dificultam (ou impedem) ataques (e.g. dummy code; randomized delays; etc.)

Power analysis...

- Retira informação dos parâmetros dos algoritmos por análise do consumo de energia.
  - Execuções condicionais;
  - ...escalonamento das chaves; permutações; etc.

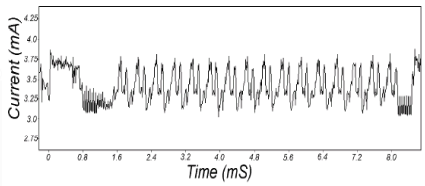


Figure 1: SPA trace showing an entire DES operation.

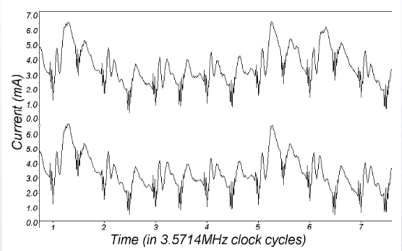


Figure 3: SPA trace showing individual clock cycles.

- Divergência do relógio no ciclo 6 indica execução condicional...

Differential Fault analysis...

- Comparam a execução do algoritmo com uma onde são induzidos erros na sua execução.
  - Induz-se um erro num dos bits do último round.
  - Por analise diferencial, determina-se chave do round
  - Restantes bits por força bruta...
- Vocacionada para atacar dispositivos "tamper proof" (e.g. Smartcards).

Referências

- Block Ciphers – *M. J. Robshaw*, RSA Labs TR-601, 1995.
- Cryptography: theory and practice – D. Stinson. [cap. 3]
- Applied Cryptography – *Bruce Schneier*. [cap. 12,13,14,15]
- Basic Methods of Cryptography – *Jan C. A. van der Lubbe*, Cambridge Press, 1997.