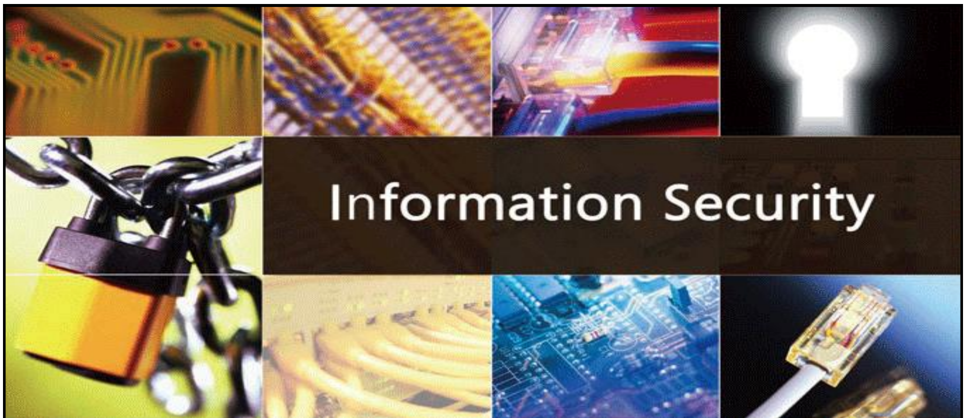


Symmetric encryption

Lecturer: Nguyễn Thị Thanh Vân – FIT - HCMUTE

Objective

- Modern Symmetric-Key Ciphers
  - Modern block ciphers
  - Modern stream ciphers
- Encipherment Using Modern Symmetric-Key Ciphers
  - Use of Modern block ciphers
  - Use of Modern stream ciphers
- Data Encryption Standard - DES
  - Double DES
  - Triple DES
- Advanced Encryption Standard - AES

Taxonomy of Cryptography

The type of operations used for transforming plaintext to ciphertext
- Substitution
- Transposition

The number of keys used
- Symmetric, single-key, secret-key, conventional encryption
- Asymmetric, two-key, or public-key encryption

The way in which the plaintext is processed
- Block cipher
- Stream cipher

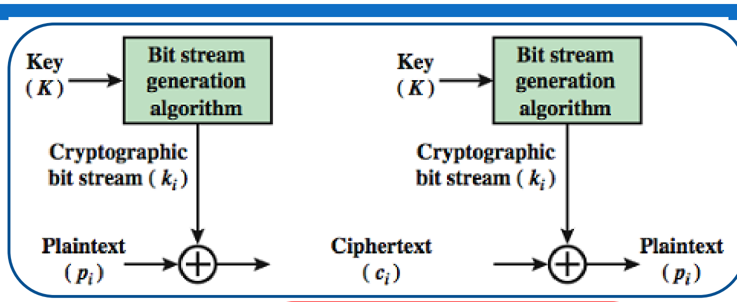
Modern Symmetric-Key Ciphers

Modern Symmetric-Key Ciphers

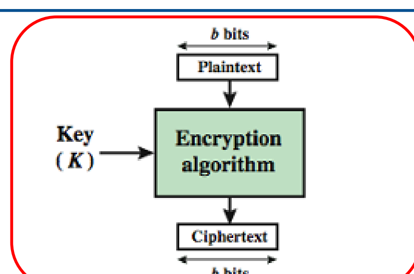
- MODERN BLOCK CIPHERS
  - Substitution or Transposition
  - Block Ciphers as Permutation Groups
  - Components of a Modern Block Cipher
  - S-Boxes
  - Product Ciphers
  - Two Classes of Product Ciphers
  - Attacks on Block Ciphers
- MODERN STREAM CIPHERS
  - Synchronous Stream Ciphers
  - Nonsynchronous Stream Ciphers

Block Cipher vs. Stream Cipher

Stream Cipher

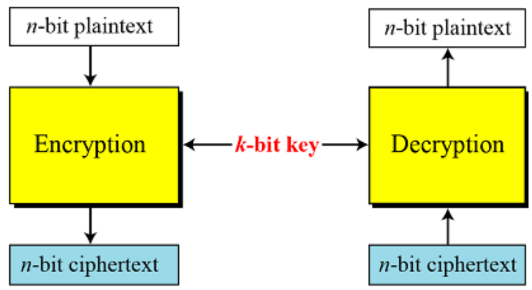


Block Cipher



Modern block cipher

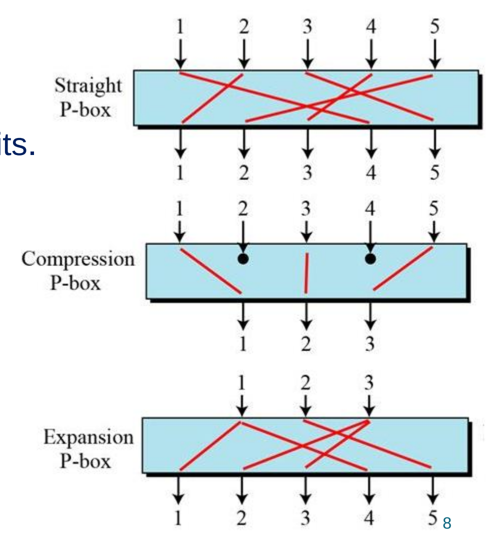
- A symmetric-key modern block cipher: Encryption & Decryption



- Modern block ciphers normally are keyed substitution ciphers
- the key allows only partial mapping from the possible inputs to the possible outputs.

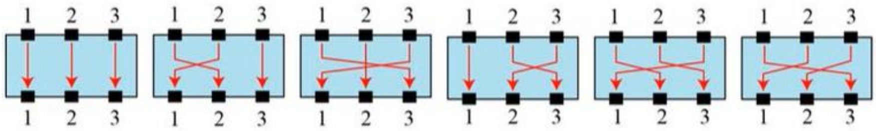
P-Boxes

- A P-box (permutation box) parallels the traditional transposition cipher for characters. It transposes bits.
- Three types of P-boxes



Straight P-box

- Ex: The possible (6) mappings of a 3 × 3 P-box



- Example of a permutation table for a straight P-box

58	50	42	34	26	18	10	02	60	52	44	36	28	20	12	04
62	54	46	38	30	22	14	06	64	56	48	40	32	24	16	08
57	49	41	33	25	17	09	01	59	51	43	35	27	19	11	03
61	53	45	37	29	21	13	05	63	55	47	39	31	23	15	07

Straight P-box

- Example :
  - Design an 8 × 8 permutation table for a straight P-box thatmoves the two middle bits (bits 4 and 5) in the input word to the two ends (bits 1 and 8) in the output words. Relative positions of other bits should not be changed.
- Solution:
  - We need a straight P-box with the table [4 1 2 3 6 7 8 5]. The relative positions of input bits 1, 2, 3, 6, 7, and 8 have not been changed, but the first output takes the fourth input and the eighth output takes the fifth input.

Compression P-box

- A compression P-box is a P-box with n inputs and m outputs where m < n.
- Example of a 32 × 24 permutation table
  - Note that inputs 7, 8, 9, 15, 16, 23, 24, and 25 are blocked

01	02	03	21	22	26	27	28	29	13	14	17
18	19	20	04	05	06	10	11	12	30	31	32

- Compression P-boxes are used when we need to permute bits and the same time decrease the number of bits for the next stage.

Expansion P-box

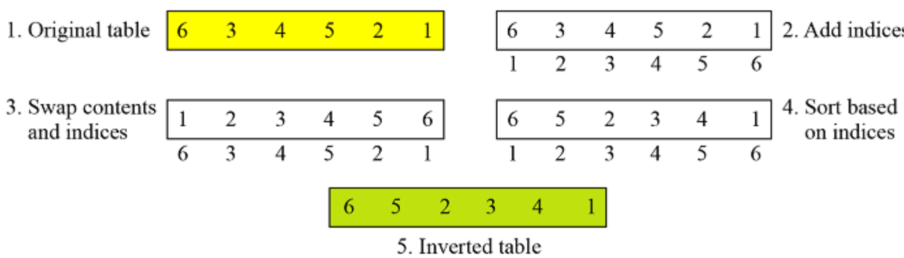
- An expansion P-box is a P-box with n inputs and m outputs where m > n.
- Example of a 12 × 16 permutation table
  - Note that each of the inputs 1, 3, 9, and 12 is mapped to 2 outputs.

01	09	10	11	12	01	02	03	03	04	05	06	07	08	09	12

- The expansion P-boxes:
  - used in modern block ciphers normally are keyless, where a permutation table shows the rule for transposing bits.
  - are used when we need to permute bits and the same time increase the number of bits for the next stage.

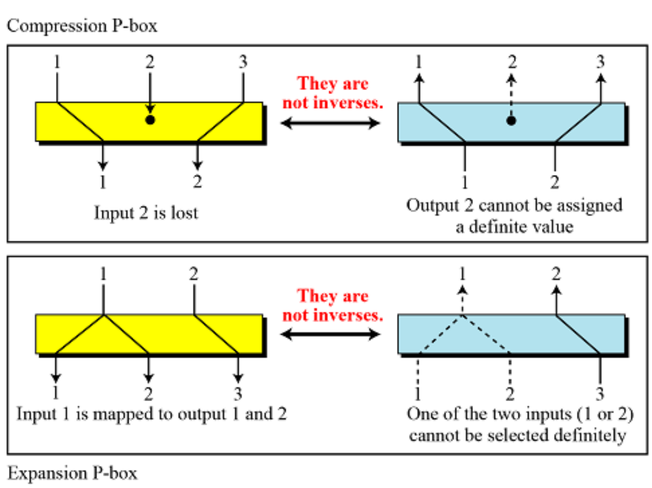
P-boxes - Invertibility

- A straight P-box is invertible.
  - use a straight P-box in the encryption cipher and its inverse in the decryption cipher.
- Compression and expansion P-boxes are not.
- Figure shows how to invert a permutation table represented as a one-dimensional table.



P-boxes - Invertibility

- Compression and expansion P-boxes are non-invertible



S-Box

- An S-box (substitution box) can be thought of as a miniature substitution cipher.
- An S-box is an m × n substitution unit, where m and n are not necessarily the same.
- Linear S-Boxes. The relationship between the inputs and the outputs can be represented as a set of equations

$$
\begin{aligned}
y_1 &= f_1(x_1, x_2, \ldots, x_n) \\
y_2 &= f_2(x_1, x_2, \ldots, x_n) \\
&\cdots \\
y_m &= f_m(x_1, x_2, \ldots, x_n)
\end{aligned}
$$

Linear S-Boxes →

$$
\begin{aligned}
y_1 &= a_{1,1}x_1 \oplus a_{1,2}x_1 \oplus \cdots \oplus a_{1,n}x_n \\
y_2 &= a_{2,1}x_1 \oplus a_{2,2}x_1 \oplus \cdots \oplus a_{2,n}x_n \\
&\cdots \\
y_m &= a_{m,1}x_1 \oplus a_{m,2}x_1 \oplus \cdots \oplus a_{m,n}x_n
\end{aligned}
$$

S-Box

- Ex: In an S-box with 3 inputs and 2 outputs, we have

$$y_1 = x_1 \oplus x_2 \oplus x_3 \qquad y_2 = x_1$$

  - S-box is linear:

$$a_{1,1} = a_{1,2} = a_{1,3} = a_{2,1} = 1 \text{ and } a_{2,2} = a_{2,3} = 0$$

  - The relationship:

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 0 & 0 \end{bmatrix} \times \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

S-boxes - Invertibility

- Invertibility S-boxes are substitution ciphers
  - relationship between input and output is defined by a table or mathematical relation.
- An S-box may or may not invertible.
  - In an invertible S-box, the number of input bits should be the same a number of output bits.
- Ex: A S-box tables

Input: 001 (r1,c2)
Output: 101
input 101 (r2,c2)
output 001

3 bits

	00	01	10	11
0	011	101	111	100
1	000	010	001	110

Table used for encryption

3 bits

3 bits

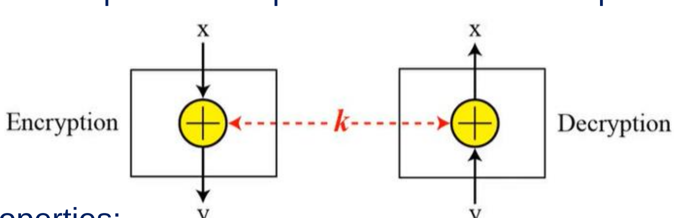
	00	01	10	11
0	100	110	101	000
1	011	001	111	010

Table used for decryption

3 bits

Exclusive-Or

- It is an important component in most block ciphers



- Properties:

$x \oplus (y \oplus z) \leftrightarrow (x \oplus y) \oplus z$

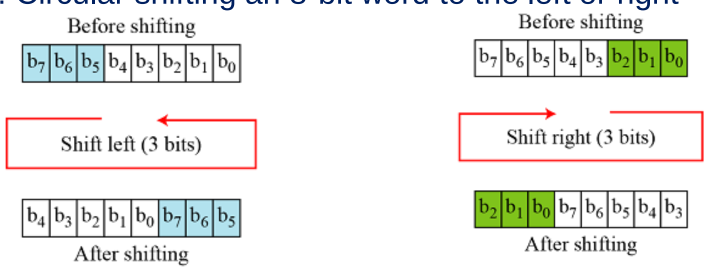
$x \oplus y \leftrightarrow y \oplus x$

$x \oplus (00\ldots0) = x$

$x \oplus x = (00\ldots0)$

Circular Shift

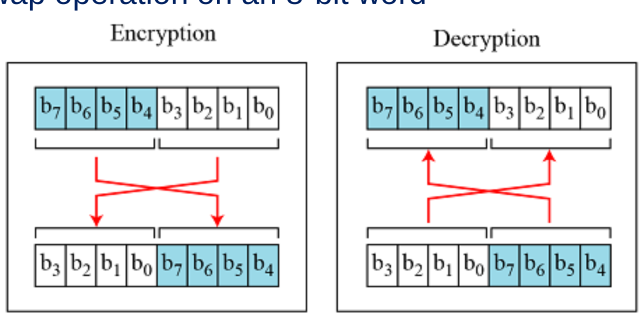
- Circular Shift is another component found in some modern block ciphers
  - It mixes the bits in a word and helps hide the patterns in the original word
- Ex: Circular shifting an 8-bit word to the left or right



- A circular left-shift operation is the inverse of the circular right-shift operation.

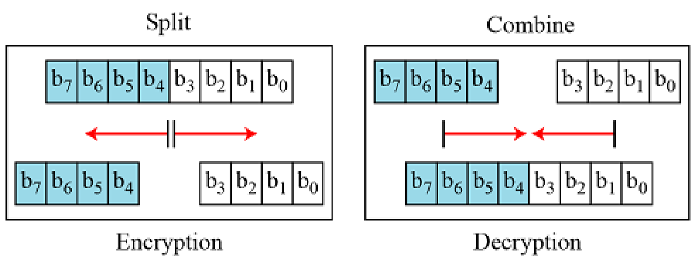
Swap

- The swap operation is a special case of the circular shift operation where k = n/2.
- Ex: Swap operation on an 8-bit word



Split and Combine

- Both operations are found in some block ciphers
  - The split: splits an n-bit word in the middle, creating two equal-length words.
  - The combine: concatenates two equal-length words to create an n-bit word.
- These two operations are inverses of each other and can be used as a pair to cancel each other out.



Product Ciphers

- Product cipher is introduced by Shannon
  - a complex cipher combining substitution, permutation, and other components.
- It have two important properties: diffusion and confusion.
  - Diffusion: hide the relationship between the ciphertext & the plaintext.

  => frustrate the adversary who uses ciphertext statistics to find the plaintext.
  - Confusion: hide the relationship between the ciphertext & the key.

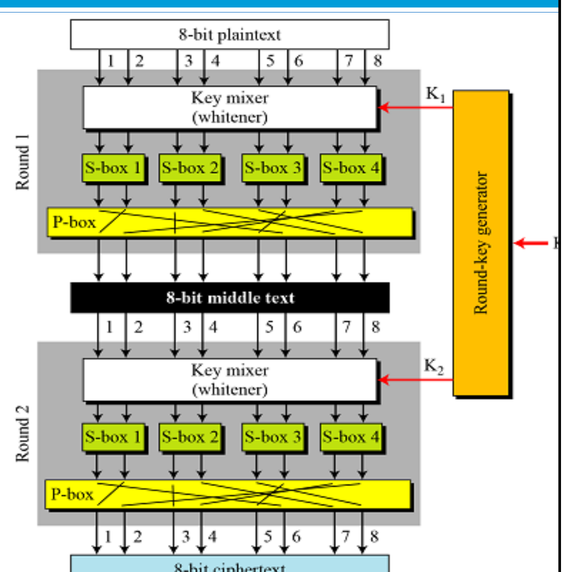
  => frustrate the adversary who tries to use the ciphertext to find the key.

Product Ciphers: Rounds

- Diffusion and confusion can be achieved using iterated product ciphers where each iteration is a combination of S-boxes, P-boxes, and other components.
- The block cipher uses a key schedule or key generator that creates different keys for each round from the cipher key.
- In an N-round cipher, the plaintext is encrypted N times to create the ciphertext; the ciphertext is decrypted N times to create the plaintext.

Product cipher: ex, made of 2 rounds

- 3 transformations happen at each round
  - Key mixer
  - S-boxes
  - P-box

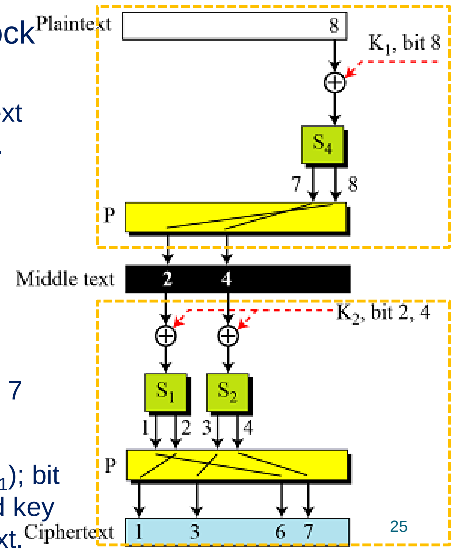


Product Ciphers

- Diffusion and confusion in a block cipher. Ex:
  - changing a single bit in the plaintext affects many bits in the ciphertext.
- Diffusion. Round1
  - XOR with K1
  - S-box 4: affects bits 7 and 8
  - P-box: bit7 -> bit2, bit8 -> bit4.

  Round 2: ….

  => bit 8 has affected bits 1, 3, 6, and 7
- Confusion:
  - bits 1,3,6,7 are affected by bit8 (K_1); bit 2,3 (K_2). => each bit in each round key affects several bits in the ciphertext.

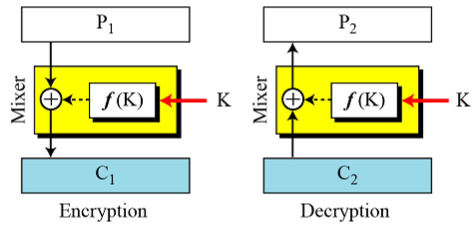


Product Ciphers: Two Classes

- Modern block ciphers are all product ciphers, but are divided into two classes.
- Feistel ciphers
  - Feistel designed a very intelligent and interesting cipher that has been used for decades.
  - A Feistel cipher can have three types of components: self-invertible, invertible, and noninvertible.
- Non-Feistel ciphers
  - A non-Feistel cipher uses only invertible components.
  - A component in the encryption cipher has the corresponding component in the decryption cipher.

Feistel cipher design

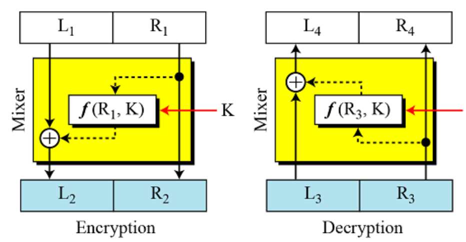
- The first thought
  - Mixer ((XOR, F(K)): self-invertible
  - Same key: encryption and decryption are inverses of each other



- Improvement
  - Use left and right blocks
  - the inputs to the function must be exactly the same inencryption & decryption.

  Means: The right half of the plain-text never changes. => 1 flaw

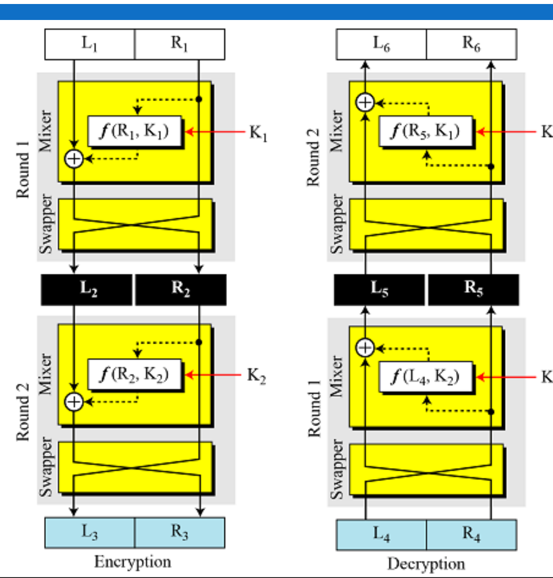
  => Attack can intercepting the ciphertext and extracting the right half



Final design of a Feistel cipher with 2 rounds

Improvement.

- increase the number of rounds.
- add a new element to each round: a swapper. => it allows us to swap the left and right halves in each round.
- K_1 K_2 are used in reverse order in the encryption and decryption



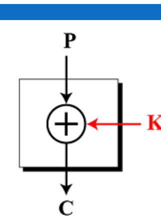
24/09/2021

Attacks on Block Ciphers

- Attacks on traditional ciphers can also be used on modern block ciphers, but today's block ciphers resist most of the attacks
- Differential Cryptanalysis
  - Eli Biham and Adi Shamir introduced the idea of differential cryptanalysis. This is a chosen-plaintext attack
- Linear Cryptanalysis
  - Linear cryptanalysis was presented by Mitsuru Matsui in 1993. The analysis uses known plaintext attacks.

Ex, Differential Cryptanalysis

- Using XOR: Without knowing the value of the key, attack can easily find the relationship between plaintext differences and ciphertext differences if by plaintext/ ciphertext difference $P_1 \oplus P_2$ and $C_1 \oplus C_2$.
- The following proves:



$$C_1 = P_1 \oplus K \qquad C_2 = P_2 \oplus K \qquad \rightarrow \qquad C_1 \oplus C_2 = P_1 \oplus K \oplus P_2 \oplus K = P_1 \oplus P_2$$

Because:

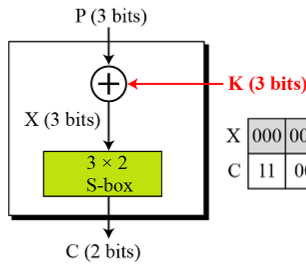
$x \oplus x = (00\ldots0)$

$x \oplus (00\ldots0) = x$

=> So simple

Ex, Differential Cryptanalysis

- Solution: add one S-box.
  - prevents atacker from finding a definite relationship btw P and C
- SBut, attack can create a probabilistic relationship:
- Make table from information about S-box input/output table with $P_1 \oplus P_2 = X_1 \oplus X_2$.



X	000	001	010	011	100	101	110	111
C	11	00	10	10	01	00	11	00

S-box table

$C_1 \oplus C_2$

$P_1 \oplus P_2$	00	01	10	11
000	8			
001	2	2		4
010	2	2	4	
011		4	2	2
100	2	2	4	
101		4	2	2
110	4		2	2
111			2	6

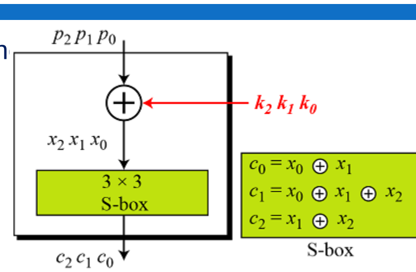
Ex, Linear Cryptanalysis

- 3 linear equations between plaintext an ciphertext bits

$$c_0 = p_0 \oplus k_0 \oplus p_1 \oplus k_1$$
$$c_1 = p_0 \oplus k_0 \oplus p_1 \oplus k_1 \oplus p_2 \oplus k_2$$
$$c_2 = p_1 \oplus k_1 \oplus p_2 \oplus k_2$$

- Solving for three unknowns, we get.

$$k_1 = (p_1) \oplus (c_0 \oplus c_1 \oplus c_2)$$
$$k_2 = (p_2) \oplus (c_0 \oplus c_1)$$
$$k_0 = (p_0) \oplus (c_1 \oplus c_2)$$



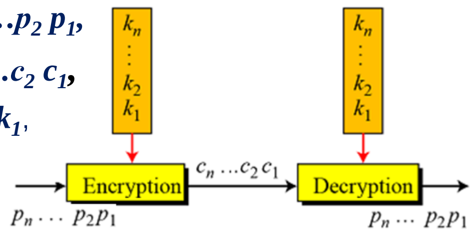
- This means: 3known-plaintext attacks can find the values of k_0, k_1, and k_2
- In some modern block ciphers, some S-boxes are not totally nonlinear; they can be approximated, probabilistically, by some linear functions.

$$(k_0 \oplus k_1 \oplus \cdots \oplus k_x) = (p_0 \oplus p_1 \oplus \cdots \oplus p_y) \oplus (c_0 \oplus c_1 \oplus \cdots \oplus c_z)$$

where $1 \le x \le m$, $1 \le y \le n$, and $1 \le z \le n$.

Modern stream ciphers

- In a modern stream cipher, encryption and decryption are done r bits at a time.
- We have: p_i, c_i, k_i are r-bit words.
  - a plaintext bit stream $P = p_n \ldots p_2\, p_1$,
  - a ciphertext bit strea $C = c_n \ldots c_2\, c_1$,
  - a key bit stream $K = k_n \ldots k_2\, k_1$,



- 2 types:
  - Synchronous stream cipher
  - Nonsynchronous stream cipher

Synchronous stream cipher

- In a synchronous stream cipher the key is independent of the plaintext or ciphertext.
- Ex: Enc & Dec with stream cipher

```
  11001100  plaintext          10100000  ciphertext
⊕ 01101100  key stream       ⊕ 01101100  key stream
  10100000  ciphertext         11001100  plaintext
```

Use the XOR function and the given key to encrypt the word “Hi”.

key = FA F2

Hi =

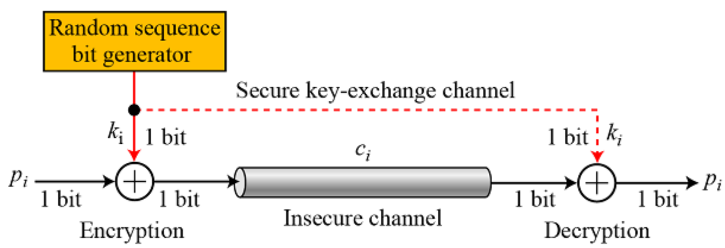
FA F2 =

Hi encrypted =

Synchronous stream cipher

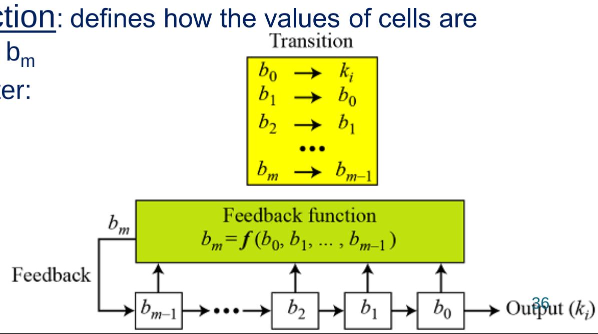
- One-time pad: The simplest and the most secure type
  - cannot guess the key or the plaintext and ciphertext statistics.
  - no relationship between the plaintext and ciphertext, either.
  - ? How can the sender and the receiver share a one-time pad key

  => this perfect and ideal cipher is very difficult to achieve



Synchronous stream cipher

- The feedback shift register (FSR)
  - can be implemented in either software or hardware
  - A feedback shift register is made of a shift register and a feedback
- The shift register: a sequence of m cells, b_0 to b_{m-1}, each cell holds a single bit: b_i receives value from b_{i-1}, give value to b_{i+1}
- The feedback function: defines how the values of cells are combined to calculate b_m
- A feedback shift register:
  - linear or
  - nonlinear.

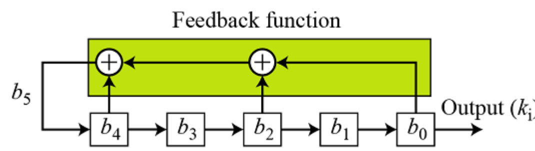


Synchronous stream cipher

- Linear feedback shift register (LFSR)
  - b_m is a linear function of $b_0, b_1, \ldots, b_{m-1}$

$$b_m = c_{m-1}\, b_{m-1} + \cdots + c_2\, b_2 + c_1\, b_1 + c_0\, b_0 \qquad (c_0 \neq 0)$$

- Ex: Create a linear feedback shift register with 5 cells in which $b_5 = b_4 \oplus b_2 \oplus b_0$
- LSFR for Example
  - c_i=0 -> b_i not connected
  - 3 connections



- vulnerable to attacks mainly because of its linearity

Synchronous stream cipher

- NonLinear feedback shift register (NLFSR)
  - b_m is a nonlinear function of $b_0, b_1, \ldots, b_{m-1}$
- Combination:
  - A stream cipher can use a combination of linear and nonlinear structures.
  - Some LFSRs can be made with the maximum period and then combined through a nonlinear function.

Nonsynchronous stream cipher

- the key depends on either the plaintext or ciphertext.

Encipherment Using Modern Symmetric-Key Ciphers

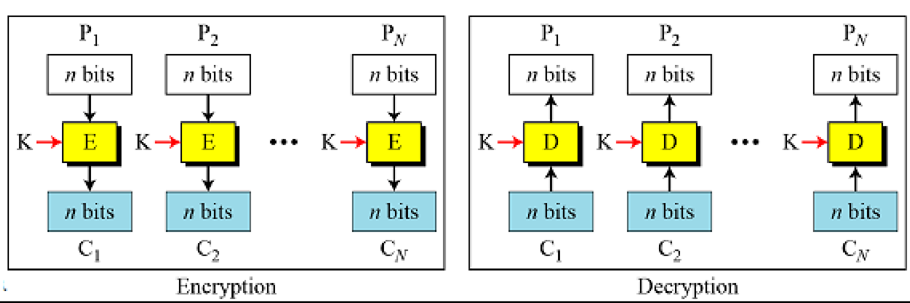
Use modern block ciphers

- 5 Modes of operations have been standardized by NIST for use with symmetric block ciphers such as DES and AES:
  - electronic codebook mode - ECB
  - cipher block chaining mode - CBC
  - cipher feedback mode - CFB
  - output feedback mode – OFB
  - counter mode - CRT

24/09/2021 41

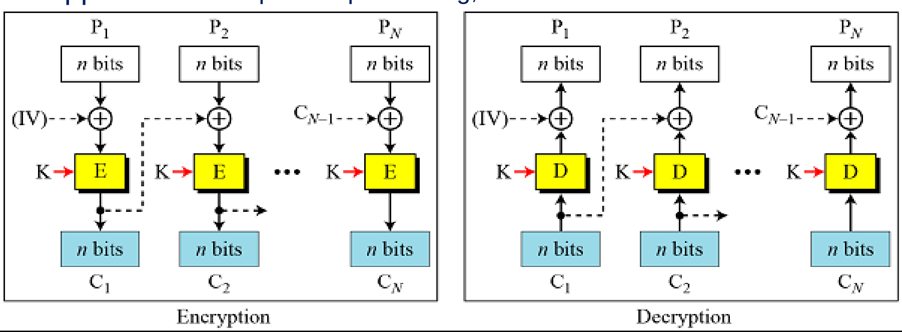
Electronic Codebook - ECB

- The simplest mode: Each block of 64 plaintext bits is encoded independently using the same key.
- Security Issues:
  - blocks are the same => need decrypt 1 block
  - exchange some ciphertext blocks without knowing the key
- Application: Secure transmission of single values; parallel processing



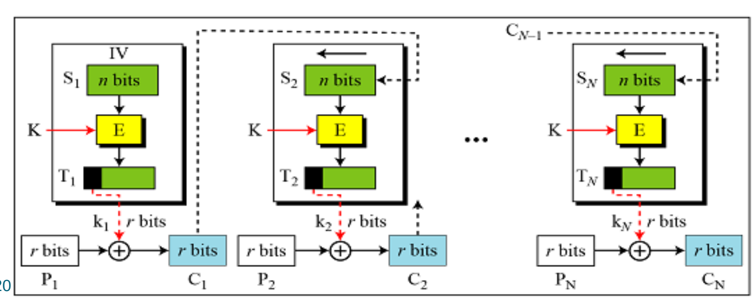
Cipher Block Chaining - CBC

- initialization vector (IV): is an arbitrary number, part of the secret key for data encryption: important role in the security -> difficult to hack
- Security issues:
  - same P -> same IV;
  - Error Propagation: error in Cj => error in most bits in Pj during decryption
- Application: Not parallel processing; Authentication



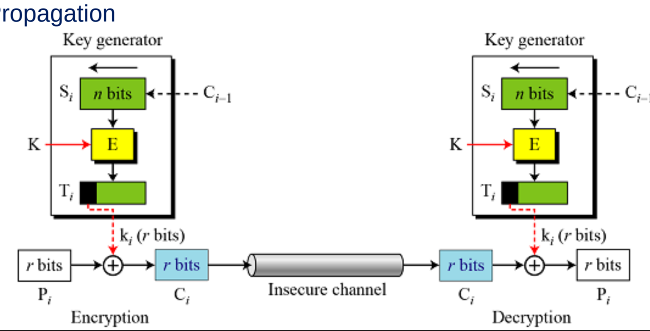
r-bit Cipher Feedback (CFB)

- Both encipherment and decipherment use the encryption function of the underlying block cipher (DES or AES) => result is a stream cipher
  - no padding is required
- CFB is less efficient than CBC or ECB, because it needs to apply the encryption function of underlying block cipher for each small block of size r
- Application: used to encipher blocks of small size, ex: 1 character or bit at a time



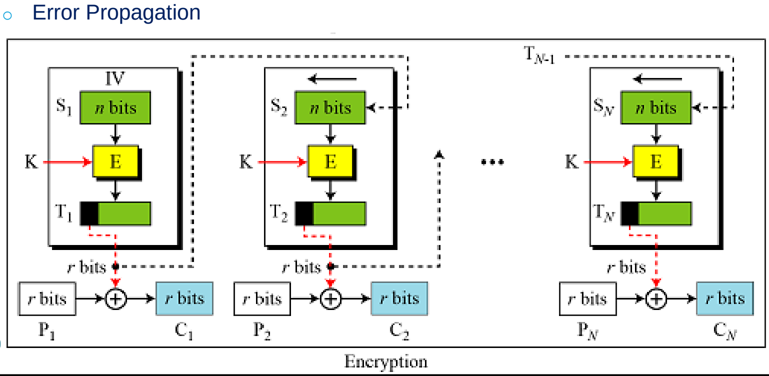
r-bit Cipher Feedback (CFB)

- CFB mode as a stream cipher: it is a nonsynchronous stream cipher.
- Security Issues
  - Just like CBC, the patterns at the block level are not preserved.
  - Many P can be encrypted with the same key, but the value of the IV should be changed for each message.
  - Attacker can add some C block to the end of the ciphertext stream.
  - Error Propagation



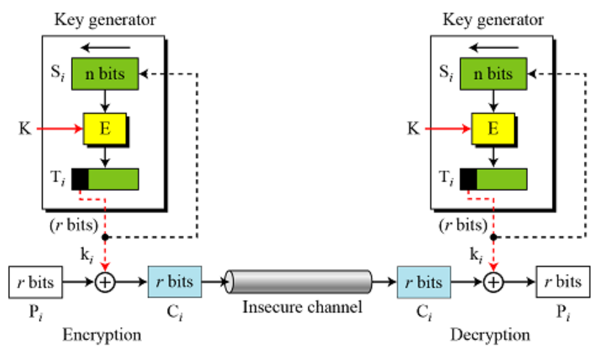
Output Feedback - OFB

- Similar to CFB, except that the input to the encryption algorithm is the preceding encryption output, and full blocks are used.
- Security Issues
  - Just like the CFB mode, patterns at the block level are not preserved.
  - Any change in the ciphertext affects the plaintext encrypted at the receiver side.
  - Error Propagation



Output Feedback - OFB

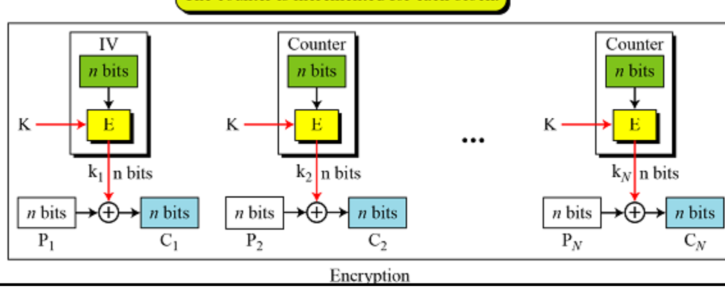
- OFB as a Stream Cipher:
  - OFB creates a stream cipher out of the underlying block cipher.
  - The key stream is independent from the plaintext or ciphertext,
- Application: Stream-oriented transmission over noisy channel (satellite communication)



Counter (CTR)

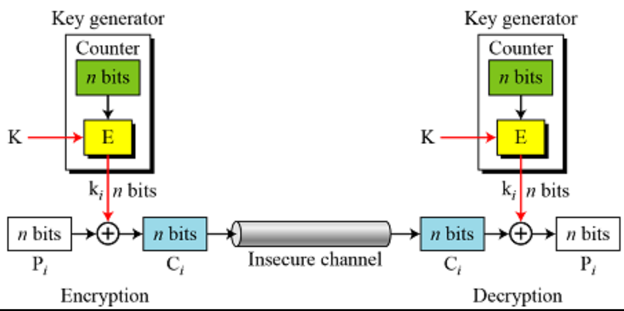
- CRT: there is no feedback.
  - The pseudorandomness in the key stream is achieved using a counter
  - Like ECB, CTR creates n-bit ciphertext blocks that are independent from each other
  - cannot be used for real-time processing
  - can be used to encrypt and decrypt random-access files as long as the value of the counter

The counter is incremented for each block.



Counter (CTR)

- CTR as a Stream Cipher
  - Like CFB and OFB, CTR is actually a stream cipher (different block are exclusive-ored with different keys).
- Security Issues
  - the same as the those for OFB mode.
  - Error Propagation: A single error in the ciphertext affects only the corresponding bit in the plaintext.



24/09/2021 49

Comparison of Different Modes

Operation Mode	*Description*	*Type of Result*	*Data Unit Size*
ECB	Each n-bit block is encrypted independently with the same cipher key.	Block cipher	n
CBC	Same as ECB, but each block is first exclusive-ored with the previous ciphertext.	Block cipher	n
CFB	Each r-bit block is exclusive-ored with an r-bit key, which is part of previous cipher text	Stream cipher	$r \le n$
OFB	Same as CFB, but the shift register is updated by the previous r-bit key.	Stream cipher	$r \le n$
CTR	Same as OFB, but a counter is used instead of a shift register.	Stream cipher	n

24/09/2021 50

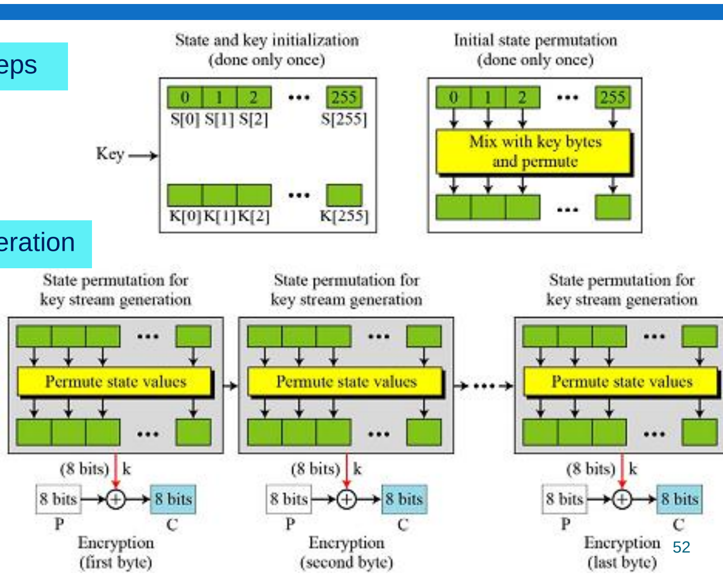
Use of stream ciphers

- RC4:
  - was designed in 1984 by Ronald Rivest for RSA Data Security.
  - is used in many data communication & network protocols, SSL/TLS
  - RC4 is based on the concept of a state: S[0] S[1] S[2] …. S[255]
- A5/1
  - a member of the A5 family of ciphers is used in the Global System for Mobile Communication (GSM), a network for mobile telephone communication

RC4 - The idea

Initialization: 2 steps

Key Stream Generation

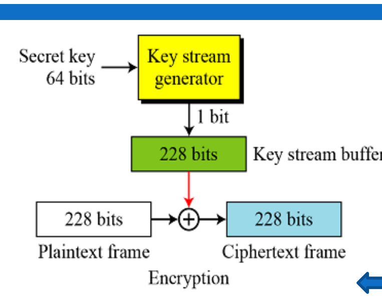


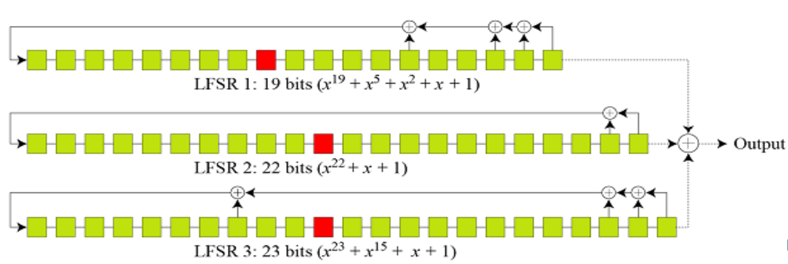
CR4

- Encryption or Decryption:
  - After k has been created, the plaintext byte is encrypted with k to create the ciphertext byte. Decryption is the reverse process.
- Security Issues
  - It is secure if the key size is at least 128 bits (16 bytes).
  - There are some reported attacks for smaller key sizes (less than 5 bytes), but the protocols that use RC4 today all use key sizes that make RC4 secure.
  - It is recommended the different keys be used for different sessions. This prevents attacker from using differential cryptanalysis on the cipher.

A5/1 (a member of the A5 family of ciphers)

- General outline of A5/1
- Key Generator A5/1: ex
  - Use 3 LFSRs with 19, 22, and 23 bits.
  - The LFSRs - Linear Feedback Shift Register, the characteristic polynomials, and the clocking bits
  - Figure: note 3 red boxes: majority func





A5/1

- Encryption/Decryption
  - The bit streams created from the key generator are buffered to form a 228-bit key that is XOR with the plaintext frame to create the ciphertext frame. Encryption/ decryption is done one frame at a time.
- Security Issues
  - Although GSM continues to use A5/1, several attacks on GSM have been recorded.
  - With some new attacks on the horizon, GSM may need to replace or fortify A5/1 in the future.

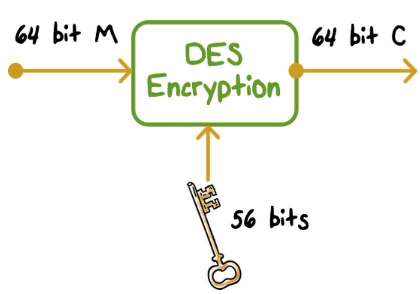
Data Encryption Standard

DES

- Introduction to DES
- DES Structure
  - Initial and Final Permutations
  - Rounds
  - Cipher and Reverse Cipher
  - Examples
- Properties of DES
- Security of DES

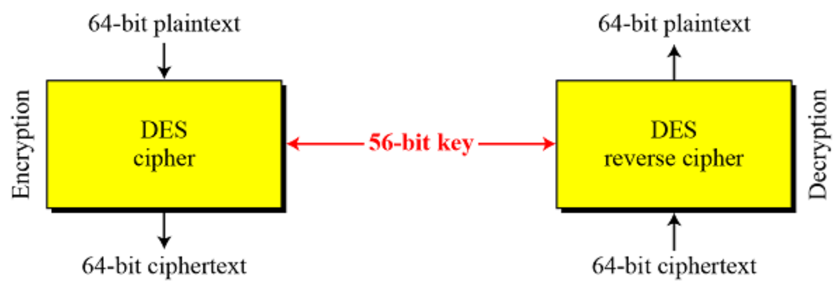
Data Encryption Standard

- **DES:** Data Encryption Standard
  - published in 1977 by the National Bureau of Standards
  - is referred to as the Data Encryption Algorithm (DEA).
  - data are encrypted in ***64-bit blocks* using a *56-bit key*.**
  - **Key:** 64 bit quantity=8-bit parity+56-bit key
    - Every 8^{th} bit is a parity bit



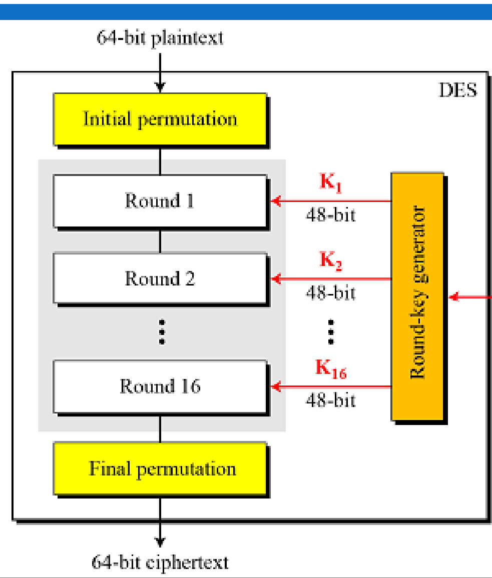
DES

- Encryption and decryption with DES



- DES uses a 56 bit key.
  - the initial key consists of 64 bits.
  - before the DES process even starts, every 8th bit of the key is discarded to produce a 56 bit key. That is bit position 8, 16, 24, 32, 40, 48, 56 and 64 are discarded.

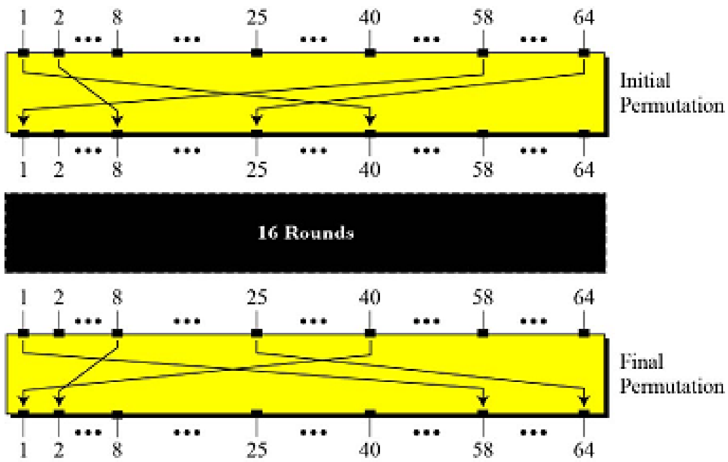
DES Structure



The process of encrypting a 64-bit block with DES:

- Initial permutation - IP
- 16 calculation loops using key
- Permutation end (be the inverse of IP)

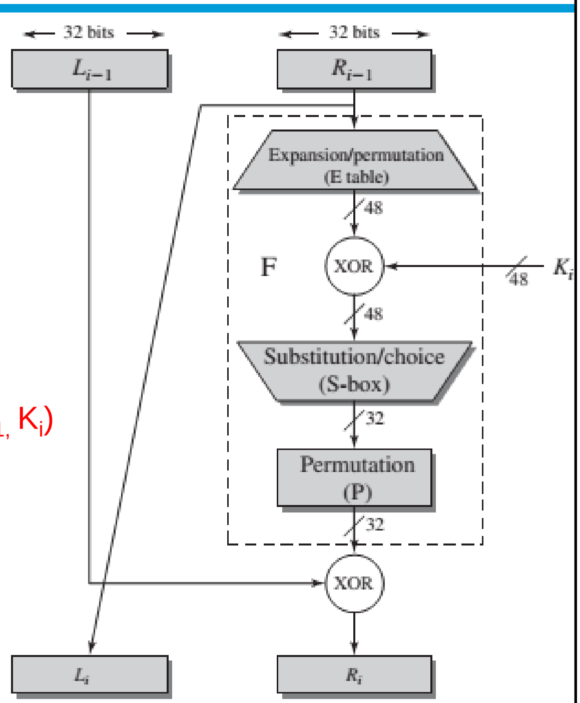
Initial and final permutation



A round in DES

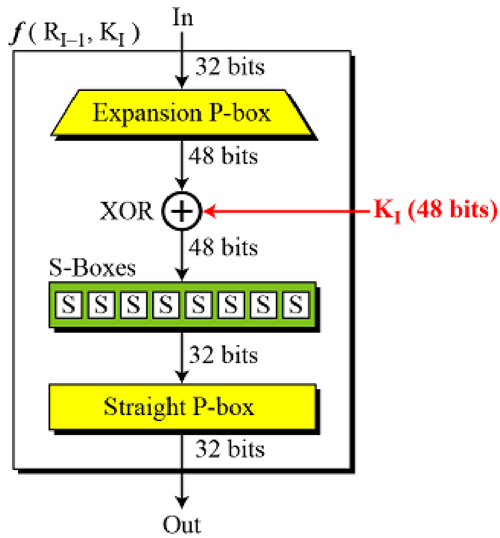
- full
- 1 Round:

$L_i=R_{i-1}$; $R_i=L_{i-1}$ XOR $F(R_{i-1}, K_i)$



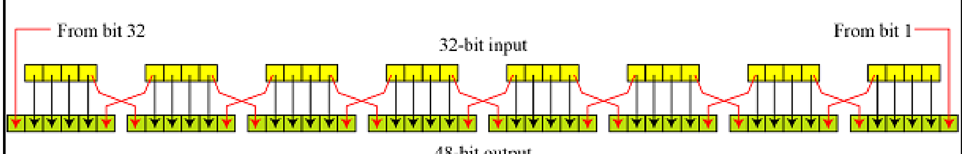
DES function

- DES function:
  - Expansion P-box
  - Straight P-box
  - XOR
  - S-Boxes



Expansion P-box

- Since R_{I-1} is a 32-bit input and K_I is a 48-bit key, we first need to expand R_{I-1} to 48 bits.

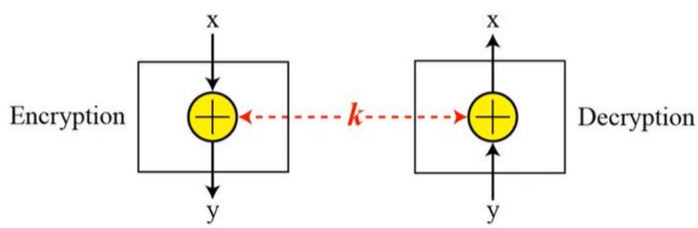


- Expansion P-box table

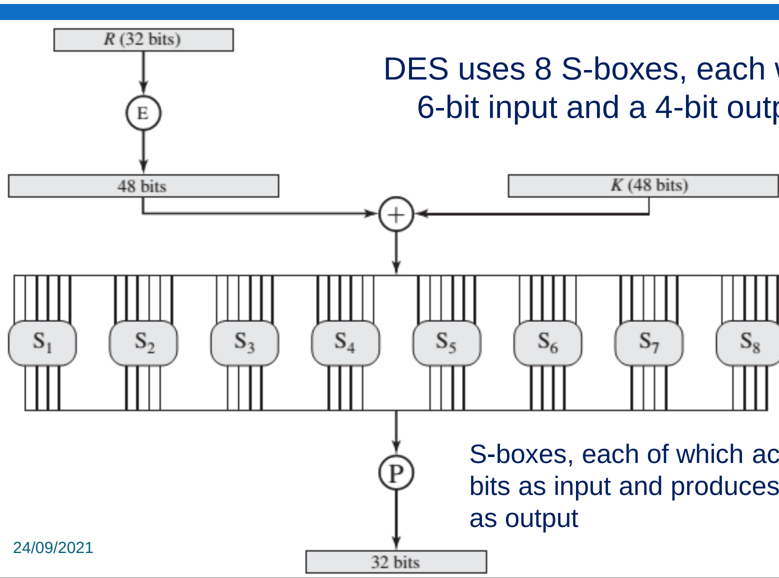
32	01	02	03	04	05
04	05	06	07	08	09
08	09	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	31	31	32	01

XOR

- DES uses the XOR operation on the expanded right section and the round key.
- Note that:
  - both the right section and the key are 48-bits in length.
  - the round key is used only in this operation
- Invertibility of the exclusive-or operation



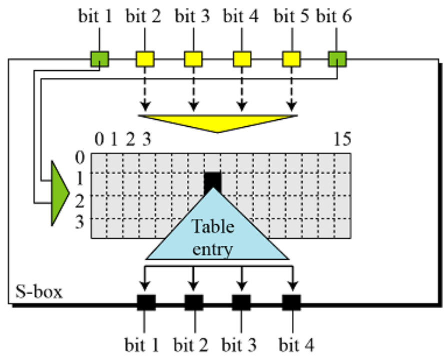
S-boxes



DES uses 8 S-boxes, each with a 6-bit input and a 4-bit output.

S-boxes, each of which accepts 6 bits as input and produces 4 bits as output

S-Box Rules



	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	14	04	13	01	02	15	11	08	03	10	06	12	05	09	00	07
1	00	15	07	04	14	02	13	10	03	06	12	11	09	05	03	08
2	04	01	14	08	13	06	02	11	15	12	09	07	03	10	05	00
3	15	12	08	02	04	09	01	07	05	11	03	14	10	00	06	13

EX: S-Box

For the given input, determine the output.

S_5		Middle 4 bits of input															
		0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
Outer bits	00	0010	1100	0100	0001	0111	1010	1011	0110	1000	0101	0011	1111	1101	0000	1110	1001
	01	1110	1011	0010	1100	0100	0111	1101	0001	0101	0000	1111	1010	0011	1001	1000	0110
	10	0100	0010	0001	1011	1010	1101	0111	1000	1111	1001	1100	0101	0110	0011	0000	1110
	11	1011	1000	1100	0111	0001	1110	0010	1101	0110	1111	0000	1001	1010	0100	0101	0011

Input: 011011 **Output:**

Straight Permutation

- Straight Permutation The last operation in the DES function is a straight permutation with a 32-bit input and a 32-bit output.
- For example, The input/output relationship
  - the seventh bit of the input becomes the second bit of the output.

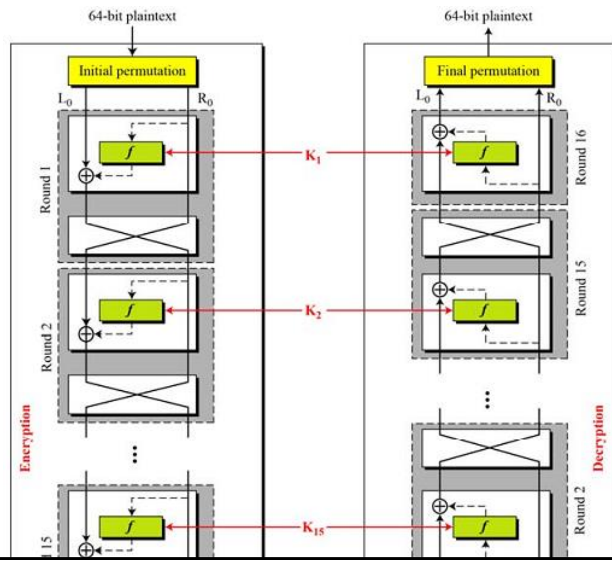
16	07	20	21	29	12	28	17
01	15	23	26	05	18	31	10
02	08	24	14	32	27	03	09
19	13	30	06	22	11	04	25

Cipher and Reverse Cipher

- Using mixers and swappers, we can create the cipher and reverse cipher, each having 16 rounds.
- First Approach
  - To achieve this goal, one approach is to make the last round (round 16) different from the others; it has only a mixer and no swapper.

DES cipher and reverse cipher
for the first approach

- 16 rounds
- no swapper in the last round



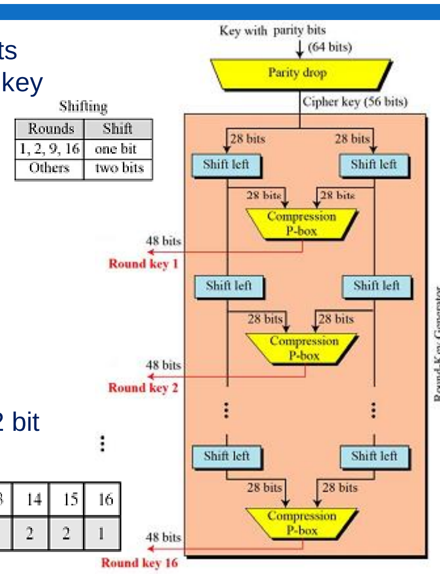
Cipher and Reverse Cipher

- Alternative Approach
  - We can make all 16 rounds the same by including one swapper to the 16th round and add an extra swapper after that (two swappers cancel the effect of each other).
- Key Generation
  - The round-key generator creates sixteen 48-bit keys out of a 56-bit cipher key.

Key generation

- Parity Drop: drops the parity bits (bits 8, 16, 24, 32, …, 64) from the 64-bit key and permutes the rest of the bits
- Parity-bit drop table

57	49	41	33	25	17	09	01
58	50	42	34	26	18	10	02
59	51	43	35	27	19	11	03
60	52	44	36	63	55	47	39
31	23	15	07	62	54	46	38
30	22	14	06	61	53	45	37
29	21	13	05	28	20	12	04



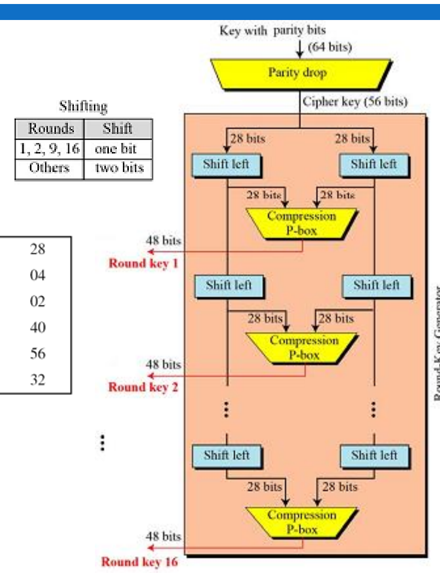
- Shift left: two 28-bit parts, shift 1 or 2 bit
- Number of bits shifts

Round	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Bit shifts	1	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1

Key generation

- Compression permutation (P-box) changes the 58 bits to 48 bits, which are used as a key for a round.
- The compression permutation is shown in Table

14	17	11	24	01	05	03	28
15	06	21	10	23	19	12	04
26	08	16	07	27	20	13	02
41	52	31	37	47	55	30	40
51	45	33	48	44	49	39	56
34	53	46	42	50	36	29	32



ex

Plaintext: 123456ABCD132536
CipherText: C0B7A8D05F3A829C

Key: AABB09182736CCDD

Plaintext: 123456ABCD132536

After initial permutation: 14A7D67818CA18AD
After splitting: L_0=14A7D678 R_0=18CA18AD

Round	*Left*	*Right*	*Round Key*
Round 1	18CA18AD	5A78E394	194CD072DE8C
Round 2	5A78E394	4A1210F6	4568581ABCCE
Round 3	4A1210F6	B8089591	06EDA4ACF5B5
Round 4	B8089591	236779C2	DA2D032B6EE3
Round 5	236779C2	A15A4B87	69A629FEC913
Round 6	A15A4B87	2E8F9C65	C1948E87475E
Round 7	2E8F9C65	A9FC20A3	708AD2DDB3C0
Round 8	A9FC20A3	308BEE97	34F822F0C66D
Round 9	308BEE97	10AF9D37	84BB4473DCCC
Round 10	10AF9D37	6CA6CB20	02765708B5BF
Round 11	6CA6CB20	FF3C485F	6D5560AF7CA5
Round 12	FF3C485F	22A5963B	C2C1E96A4BF3
Round 13	22A5963B	387CCDAA	99C31397C91F
Round 14	387CCDAA	BD2DD2AB	251B8BC717D0
Round 15	BD2DD2AB	CF26B472	3330C5D9A36D
Round 16	19BA9212	CF26B472	181C5D75C66D

After combination: 19BA9212CF26B472

Ciphertext: C0B7A8D05F3A829C *(after final permutation)*

Properties of DES

- Avalanche Effect
  - means a small change in the plaintext (or key) should create a significant change in the ciphertext.
  - DES has been proved to be strong with regard to this property.
  - Ex:

Plaintext: 0000000000000000 Key: 22234512987ABB23
Ciphertext: 4789FD476E82A5F1

Plaintext: 0000000000000001 Key: 22234512987ABB23
Ciphertext: 0A4ED5C15A63FEA3

  - Number of bit differences

Rounds	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*14*	*15*	*16*
Bit differ	1	6	20	29	30	33	32	29	32	39	33	28	30	31	30	29

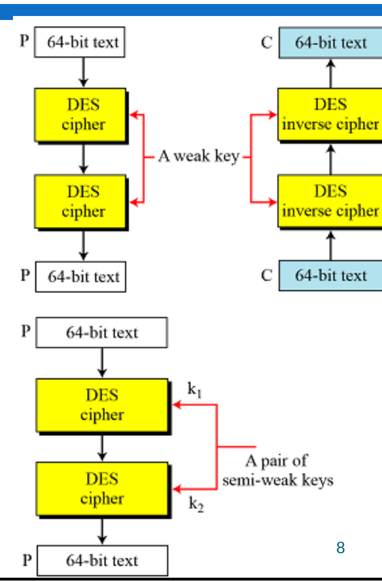
- Completeness effect
  - means that each bit of the ciphertext needs to depend on many bits on the plaintext.
  - The diffusion and confusion produced by P-boxes and S-boxes in DES, show a very strong completeness effect.

Design Criteria of DES

- S-Boxe:
  - The design provides confusion and diffusion of bits from each round to the next.
- P-Boxes:
  - They provide diffusion of bits.
- Number of Rounds:
  - DES uses sixteen rounds of Feistel ciphers.
  - The ciphertext is thoroughly a random function of plaintext and ciphertext.

DES Weaknesses

- Weaknesses in S-boxes
- Weaknesses in P-boxes
- Weaknesses in Key
  - Key Size => brute-force attack
  - Weak keys
  - Semi-weak Keys
  - Possible Weak Keys
  - Key Complement
  - Key Clustering

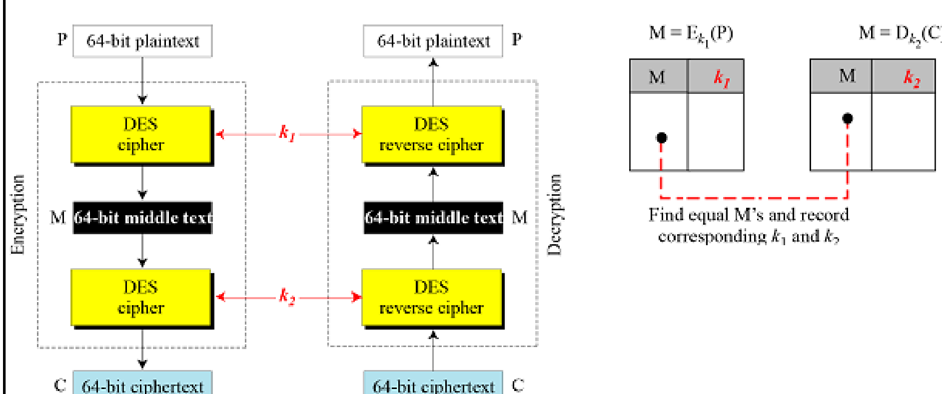


Multiple DES

- The major criticism of DES regards its key length.
- This means that we can use double or triple DES to increase the key size.
- Double DES
  - use two instances of DES ciphers for encryption and two instances of reverse ciphers for decryption.
  - Each instance uses a different key, which means that the size of the key is now doubled (112 bits).
  - However, double DES is vulnerable to a known-plain text attack,
- Triple DES (3DES):
  - uses three stages of DES for encryption and decryption.
  - Two versions of triple DES are in use today:
    - triple DES with two keys and
    - triple DES with three keys.

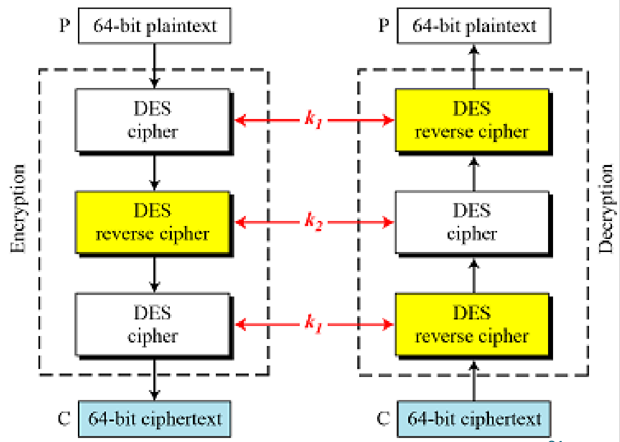
2DES: Meet-in-the-Middle Attack

- The middle text, the text created by the first encryption or first decryption, M, should be the same for encryption and decryption to work. In other words, we have 2 relationships:



Triple DES with two keys

- The 1^{st} and 3^{rd} stages use k_1 The 2^{nd} stage uses k_2.
- Uses
  - reverse cipher
- Attack
  - known-plaintext



Security of DES

- Brute-Force Attack
  - **Key space & weak:** With a key length of 56 bits, there are 2^{56} possible keys, which is approximately $7.2 * 10^{16}$ keys.
- Differential Cryptanalysis
  - Attack on S-boxes: design 16 rounds to make DES specifically resistant to this type of attack
- Linear Cryptanalysis
  - S-boxes are not very resistant to linear cryptanalysis. It has been shown that DES can be broken using 2^{43} pairs of known plaintexts.
  - However, from the practical point of view, finding so many pairs is very unlikely.

Advanced Encryption Stand AES

AES

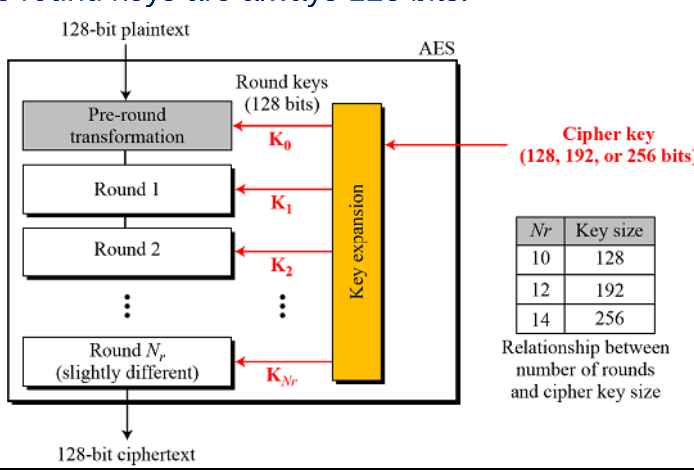
- Introduction to AES
- Structure of Each Round
- Transformations in AES:
  - substitution,
  - permutation,
  - mixing,
  - key-adding
- Cipher with AES
- Security of AES

AES - Advanced Encryption Standard

- AES:
  - intended to replace DES for commercial applications.
  - A symmetric-key block cipher published by the NIST in 12/2001.
  - It uses a 128-bit block size and a key size of 128, 192, or 256 bits.
  - does not use a Feistel structure (take ½ block data >< entire data).
- The criteria defined by NIST for selecting AES
  - Security: 128-bit key, resistance to cryptanalysis attacks other than brute-force attack
  - Cost: covers the computational efficiency and storage
  - Implementation: flexibility (on any platform) and simplicity

General design of AES

- AES has defined three versions, with 10, 12, 14 rounds.
- The round keys are always 128 bits.

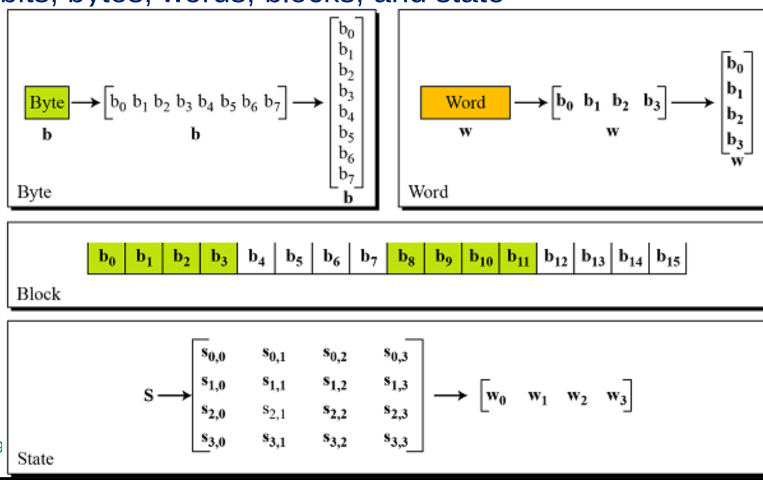


Nr	Key size
10	128
12	192
14	256

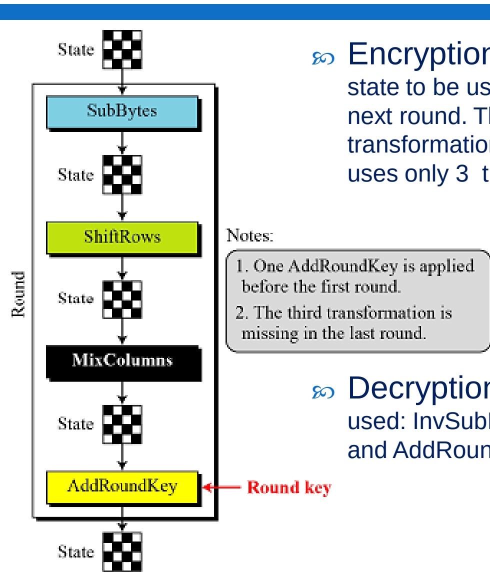
Relationship between number of rounds and cipher key size

Data units used in AES

- AES uses five units of measurement to refer to data: bits, bytes, words, blocks, and state



Structure of a round



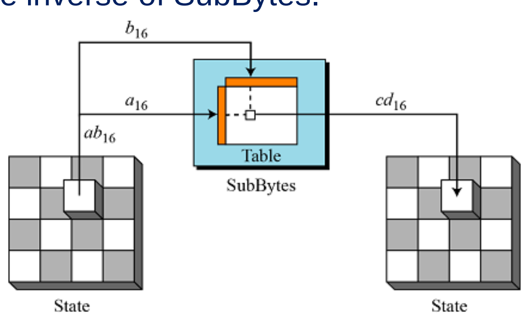
- **Encryption:** takes a state and creates another state to be used for the next transformation or the next round. The pre-round section uses only one transformation (AddRoundKey); the last round uses only 3 transformations
- **Decryption**: the inverse transformations are used: InvSubByte, InvShiftRows, InvMixColumns, and AddRoundKey (this one is self-invertible)

Transformations in AES

- To provide security, AES uses 4 types of transformations:
  - substitution,
  - permutation,
  - mixing,
  - key-adding.

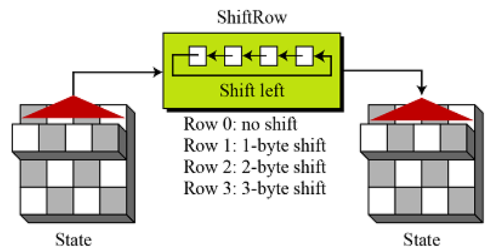
Transformations in AES: Substitution

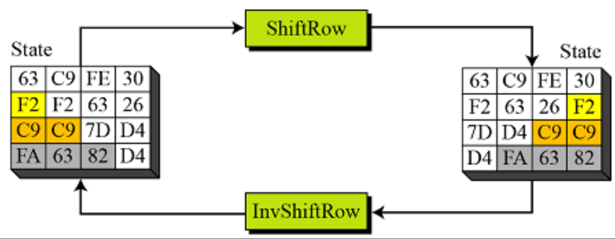
- AES uses substitution. uses 2 invertible transformations.
- SubBytes: The first transformation, is used at the encryption site.
  - To substitute a byte, we interpret the byte as 2 hexadecimal digits.
  - The SubBytes operation involves 16 independent byte-to-byte transformations.
- InvSubBytes: is the inverse of SubBytes.



Transformations in AES: Permutation

- Permutes the bytes: shifting
- ShiftRows: in encryption
  - is done at the byte level
  - Shift to the left
- InvShiftRows: decryption
  - Shift to the right
- Ex:

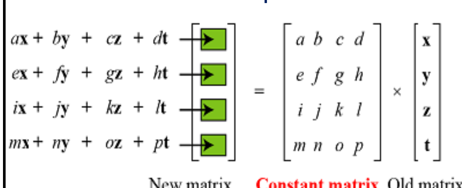


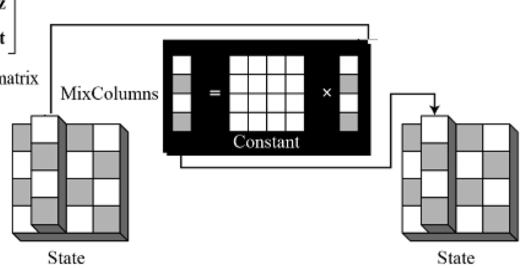


Transformations in AES: mixing

- Mixing: changes the value of the byte based only on original value and an entry in the table; does not include the neighboring bytes
- MixColumns: transforms each column of the state to a new column.

=> the matrix multiplication of a state column by a constant square matrix.

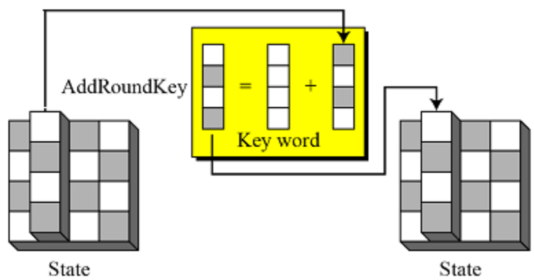




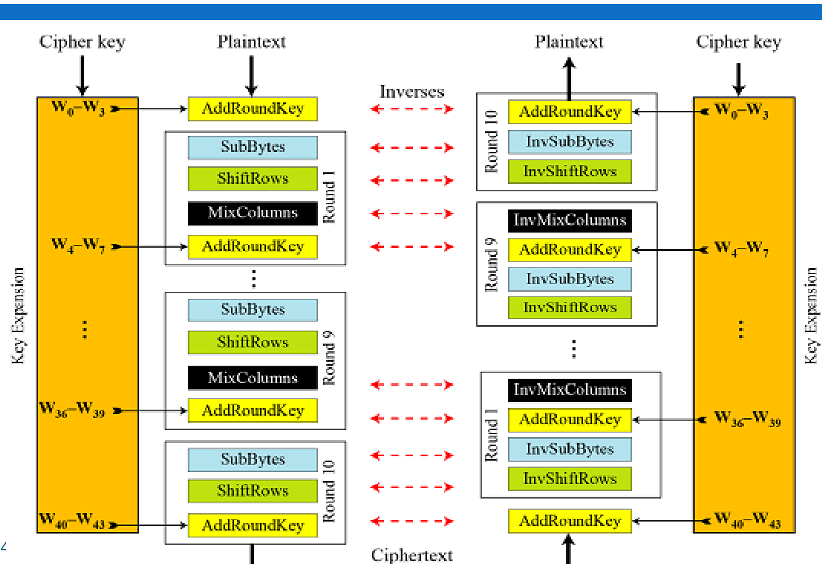
The InvMixColumns:
Inverses of the MixColumns

Transformations in AES: key-adding

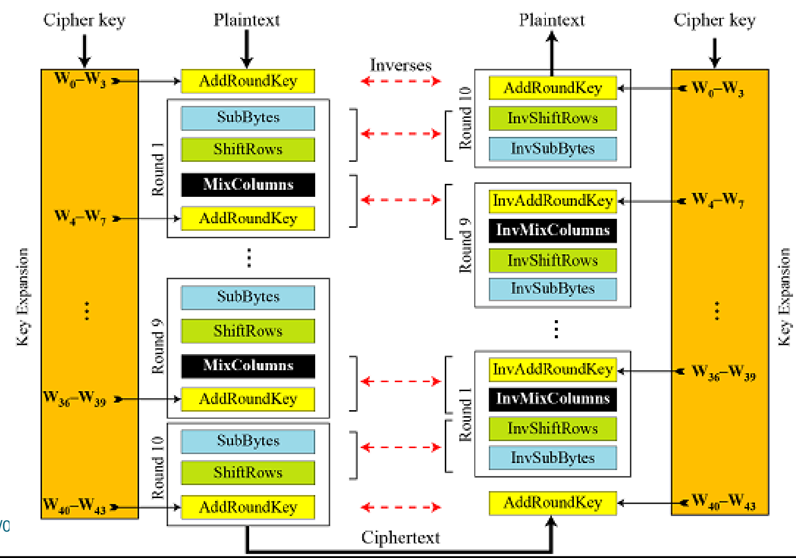
- AddRoundKey:
  - proceeds one column at a time.
  - adds a round key word with each state column matrix;
  - the operation in AddRoundKey is matrix addition.



AES: Original Design



AES: Alternate design



Security of AES

- Most of the known attacks on DES were already tested on AES; none of them has broken the security of AES so far.
- Brute-Force Attack
  - the larger-size key (128, 192, and 256 bits). DES with 56-bit
- Statistical Attacks
  - The strong diffusion and confusion provided by the combination of the SubBytes, ShiftRows, and MixColumns transformations removes any frequency pattern in the plaintext.
  - Numerous tests have failed to do statistical analysis of the ciphertext.
- Differential and Linear Attacks
  - Differential and linear cryptanalysis attacks were no doubt taken into consideration.
  - There are no differential and linear attacks on AES as yet.

Comparison of Encryption Algorithms

	DES	Triple DES	AES
Plaintext block size (bits)	64	64	128
Ciphertext block size (bits)	64	64	128
Key size (bits)	56	112 or 168	128, 192, or 256

DES = Data Encryption Standard
AES = Advanced Encryption Standard

A block cipher:
- processes the plaintext input in fixed-size blocks
- produces a block of ciphertext of equal size for each plaintext block.

Practice

- OpenSSL