CO-INS:Information and Network Security

UNIT-II (Part-II) Modern Symmetric-Key Ciphers

Course Instructors:
Soma Saha
Veerendra Srivastava

Soma Saha (PhD)

Department of Computer Engineering
SGSITS Indore, India

March 31, 2021

Soma Saha

UNIT-II (Part-II): Learning Objectives

Upon completion of this unit, you should be able to

LO1 Explain the concept of modern block ciphers and discuss their characteristics

LO2 Discuss the components of a modern block cipher

LO3 Relate the concept of product ciphers and distinguish between the two classes of product ciphers

L04 Explain modern stream ciphers and discuss two broad categories—synchronous and non-synchronous

Cryptography: Modern Symmetric-Key Ciphers

- The traditional/classical symmetric-key ciphers (that we have studied so far) are **character-oriented ciphers**.
- With the advent of the computer, we need **bit-oriented ciphers**.
- Why??

Soma Saha

Cryptography: Modern Symmetric-Key Ciphers

- The traditional/classical symmetric-key ciphers (that we have studied so far) are **character-oriented ciphers**.
- With the advent of the computer, we need **bit-oriented ciphers**.
- Why??
  - The information to be encrypted is not just text; it can also consists of numbers, graphics, audio, and video data.

Cryptography: Modern Symmetric-Key Ciphers

- The traditional/classical symmetric-key ciphers (that we have studied so far) are **character-oriented ciphers**.
- With the advent of the computer, we need **bit-oriented ciphers**.
- Why??
  - The information to be encrypted is not just text; it can also consists of numbers, graphics, audio, and video data.
  - It is convenient to convert these types of data into stream of bits, to encrypt the stream, and then to send the encrypted stream.

Cryptography: Modern Symmetric-Key Ciphers

- The traditional/classical symmetric-key ciphers (that we have studied so far) are **character-oriented ciphers**.
- With the advent of the computer, we need **bit-oriented ciphers**.
- Why??
  - The information to be encrypted is not just text; it can also consists of numbers, graphics, audio, and video data.
  - It is convenient to convert these types of data into stream of bits, to encrypt the stream, and then to send the encrypted stream.
  - Additionally, when text is treated at the bit level, each character is replaced by 8 (or 16) bits, which means that the number of symbols becomes 8(or 16) times larger. **Mixing a larger number of symbols increases security**.

Modern Block Cipher

- A symmetric-key **modern block cipher** encrypts an n-bit block of plaintext or decrypts an n-bit block of ciphertext. The encryption or decryption algorithm uses a k-bit key.

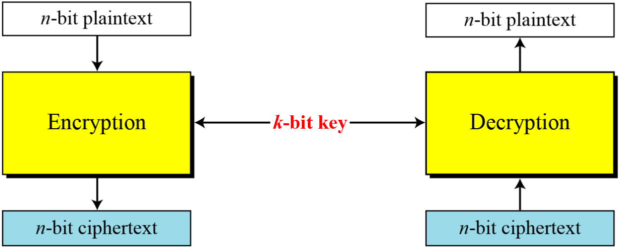


Figure 1: A modern block cipher.

Modern Block Cipher.. contd…1

- If the message has fewer than n bits, padding must be added to make it an n-bit block; if the message has more than n bits, it should be divided into n-bit blocks and appropriate padding must be added to the last block if necessary.
- **Common values for n?**

Modern Block Cipher.. contd…1

- If the message has fewer than n bits, padding must be added to make it an n-bit block; if the message has more than n bits, it should be divided into n-bit blocks and appropriate padding must be added to the last block if necessary.
- **Common values for n?**

– The common values for n are 64, 128, 256, or 512 bits.

Modern Block Cipher.. contd…2

- How many padding bits must be added to a message of 100 characters if 8-bit ASCII is used for encoding and the block cipher accepts blocks of 64 bits?

Modern Block Cipher.. contd…2

- How many padding bits must be added to a message of 100 characters if 8-bit ASCII is used for encoding and the block cipher accepts blocks of 64 bits?

– Encoding 100 characters using 8-bit ASCII results in an 800-bit message. The plaintext must be divisible by 64. If |M| and |Pad| are the length of the message and the length of the padding,

$$|M| + |Pad| = 0 \bmod 64 \quad \rightarrow \quad |Pad| = -800 \bmod 64 \quad \rightarrow \quad 32 \bmod 64$$

Soma Saha

Substitution or Transposition

- A modern block cipher can be designed to act as a **substitution cipher** or a **transposition cipher**.
- **Example:** If the cipher is designed as a substitution cipher, a 1-bit or a 0-bit in the plaintext can be replaced by either 0 or 1. This signifies that the ciphertext and plaintext can have a different number of 1's.

Substitution or Transposition

- A modern block cipher can be designed to act as a **substitution cipher** or a **transposition cipher**.
- **Example:** If the cipher is designed as a substitution cipher, a 1-bit or a 0-bit in the plaintext can be replaced by either 0 or 1. This signifies that the ciphertext and plaintext can have a different number of 1's.
  - a 64 bit plaintext block of 12 0's and 52 1's can be encrypted to a ciphertext of 34 0's and 30 1's.

Substitution or Transposition

- A modern block cipher can be designed to act as a **substitution cipher** or a **transposition cipher**.
- **Example:** If the cipher is designed as a substitution cipher, a 1-bit or a 0-bit in the plaintext can be replaced by either 0 or 1. This signifies that the ciphertext and plaintext can have a different number of 1's.
  - a 64 bit plaintext block of 12 0's and 52 1's can be encrypted to a ciphertext of 34 0's and 30 1's.
- If the cipher is designed as a transposition cipher, the bits are only reordered (transposed); there is same number of 1's in the plaintext and in the ciphertext.
- **Conclusion:** Modern block ciphers are designed as substitution ciphers to be resistant to exhaustive-search attack.

Modern block cipher: Substitution or Transposition:: Example

- Suppose that we have a block cipher where n = 64. If there are 10 1's in the ciphertext, how many trial-and-error tests does Eve need to do to recover the plaintext from the intercepted ciphertext in each of the following cases?
  a. The cipher is designed as a substitution cipher.
  b. The cipher is designed as a transposition cipher.
- **Solution:**

Modern block cipher: Substitution or Transposition:: Example

- Suppose that we have a block cipher where n = 64. If there are 10 1's in the ciphertext, how many trial-and-error tests does Eve need to do to recover the plaintext from the intercepted ciphertext in each of the following cases?
  a. The cipher is designed as a substitution cipher.
  b. The cipher is designed as a transposition cipher.
- **Solution:**
  a. In the first case, Eve has no idea how many 1's are in the plaintext. Eve needs to try all possible 2^{64} 64-bit blocks to find one that makes sense.
     - If eve could try 1 billion blocks per second, it would still take hundreds of years, on average, before she could be successful.

Modern block cipher: Substitution or Transposition:: Example

- Suppose that we have a block cipher where n = 64. If there are 10 1's in the ciphertext, how many trial-and-error tests does Eve need to do to recover the plaintext from the intercepted ciphertext in each of the following cases?
  a. The cipher is designed as a substitution cipher.
  b. The cipher is designed as a transposition cipher.
- **Solution:**
  a. In the first case, Eve has no idea how many 1's are in the plaintext. Eve needs to try all possible 2^{64} 64-bit blocks to find one that makes sense.
     - If eve could try 1 billion blocks per second, it would still take hundreds of years, on average, before she could be successful.
  b. In the second case, Eve knows that there are exactly 10 1's in the plaintext. Eve can launch an exhaustive-search attack using only those 64-bit blocks that have exactly 10 1's.

$$\binom{64}{10} = \frac{64!}{(10!)(54!)} = 151{,}473{,}214{,}816$$

(Less than 3 min...)

Block Ciphers as Permutation Groups

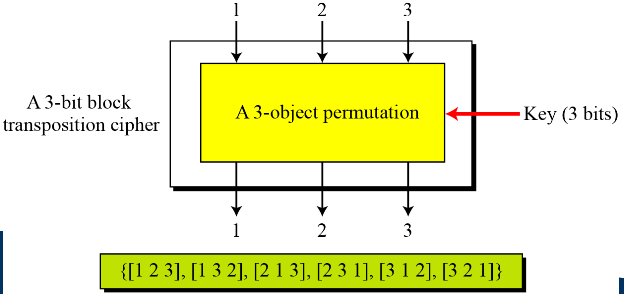
- **Full-size key ciphers:** the key is long enough to choose every possible mapping from input to output. In practice, the key is smaller (partial-key), only some mappings from the input to output are possible.
  - Full-size key ciphers are not used in practice, only partial-key ciphers are used.

Block Ciphers as Permutation Groups

- **Full-size key ciphers:** the key is long enough to choose every possible mapping from input to output. In practice, the key is smaller (partial-key), only some mappings from the input to output are possible.
  - Full-size key ciphers are not used in practice, only partial-key ciphers are used.
- Full-Size Key Transposition Block Ciphers: In a full-size key transposition cipher We need to have n! possible keys, so the key should have $\lceil log_2 n! \rceil$ bits.
  - Only transposes bits without changing their values.
  - So, it can be modeled as an n-object permutation with a set of **n!** permutation tables in which the key defines which table is used by Alice and Bob.

Full-Size Key Transposition Block Ciphers: Example

- **Show the model and the set of permutation tables for a 3-bit block transposition cipher where the block size is 3 bits.**
- The set of permutation tables has 3! = 6 elements, as shown below:



The set of permutation tables with 3! = 6 elements

Soma Saha

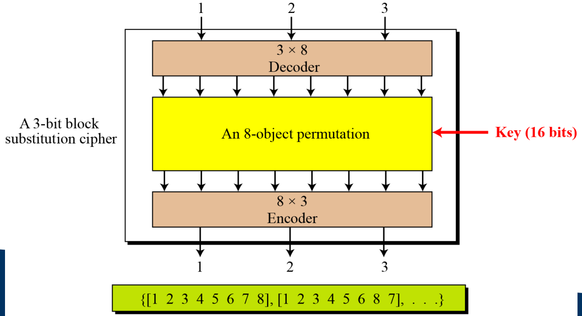
Full-Size Key Substitution Block Ciphers

- **Full-Size Key Substitution Block Ciphers:** A full-size key substitution cipher does not transpose bits; it substitutes bits. We can model the substitution cipher as a permutation if we can decode the input and encode the output.
  -
  - We can model the substitution cipher as a permutation if we can decode the input and encode the output.
  - **Decoding** means transforming an n-bit integer into a 2^n-bit string with only a single 1 and $2^n - 1$ 0's.
  - The position of the single 1 is the value of the integer, in which the positions range from 0 to $2^n - 1$.
  - **Encoding** is the reverse process. As the new input and output have always a single 1, the cipher can be modeled as a permutation of 2^n objects.

Soma Saha

Full-Size Key Substitution Block Ciphers: Example

- Example: Show the model and the set of permutation tables for a 3-bit block substitution cipher.
- The figure shows the model and the set of permutation tables. The key is much longer, $\lceil log_2 40,320 \rceil$ = 16 bits.



The set of permutation tables with 8! = 40,320 elements

NOTE:

- **A full-size key n-bit transposition cipher or a substitution block cipher can be modeled as a permutation, but their key sizes are different:**
  - Transposition: the key is $\lceil log2n! \rceil$ bits long.
  - Substitution: the key is $\lceil log_2(2^n)! \rceil$ bits long.

Partial-Size Key Cipher

- **Two or more cascaded permutations can be always replaced with a single permutation. Hence it is useless to have more than one stage of full-size key ciphers, because the effect is the same as having a single stage.**
- Modern block ciphers normally are keyed substitution ciphers in which the key allows only partial mappings from the possible inputs to the possible outputs.

Partial-Size Key Cipher

- **Two or more cascaded permutations can be always replaced with a single permutation. Hence it is useless to have more than one stage of full-size key ciphers, because the effect is the same as having a single stage.**
- Modern block ciphers normally are keyed substitution ciphers in which the key allows only partial mappings from the possible inputs to the possible outputs.
- **For example**, a common substitution cipher is DES (Data Encryption Standard) which uses a 64-bit block cipher. If the designer of DES had used a full-size key, the key would have been $log_2(2^{64})! = 2^{70}$ bits. The key size for DES is only 56 bits which is only a very small fraction of the full-size key. This means that DES uses only 2^{56} mappings out of approximately $2^{2^{70}}$ possible mappings.

Components of a Modern Block Cipher

- In cryptography, **confusion** and **diffusion** are two properties of the operation of a secure cipher identified by **Claude Shannon** in his 1945 classified report: “A Mathematical Theory of Cryptography”.

Diffusion

- **Diffusion:** Diffusion means that if we change a single bit of the plaintext, then (statistically) half of the bits in the ciphertext should change, and similarly, if we change one bit of the ciphertext, then approximately one half of the plaintext bits should change. Since a bit can have only two states, when they are all re-evaluated and changed from one seemingly random position to another, half of the bits will have changed state.
  - **The idea of diffusion is to hide the relationship between the ciphertext and the plain text.**
  - This will make it hard for an attacker who tries to find out the plain text and it increases the redundancy of plain text by spreading it across the rows and columns; it is achieved through transposition of algorithm and it is used by block ciphers only.

Confusion

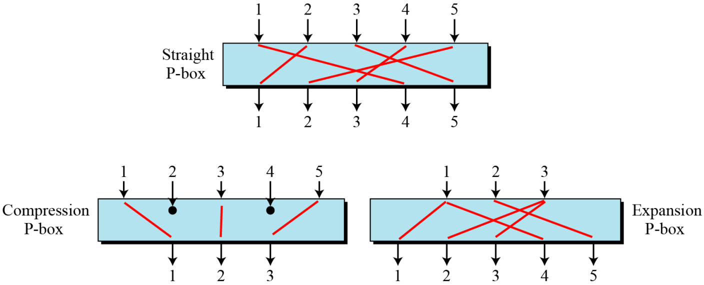
- Confusion means that each binary digit (bit) of the ciphertext should depend on several parts of the key, obscuring the connections between the two.
- **The property of confusion hides the relationship between the ciphertext and the key.**
- This property makes it difficult to find the key from the ciphertext and if a single bit in a key is changed, the calculation of the values of most or all of the bits in the ciphertext will be affected.
- Confusion increases the ambiguity of ciphertext and it is used by both block and stream ciphers.

Components of a Modern Block Cipher

- To provide required properties of a modern block cipher, such as **diffusion** and **confusion**, a modern block cipher is made of a combination of
  - transposition units for diffusion (called **D-boxes** or **P-boxes** for permutation),
  - substitution units (called **S-boxes**),
  - and some other units.

P-boxes or D-boxes

* A P-box (permutation box) or D-box (diffusion box) parallels the traditional transposition cipher for characters. It transposes bits.



Straight P-boxes (or D-boxes)

- **straight P-boxes (or D-boxes)** : all 6 possible mappings of a 3 × 3 D-box.

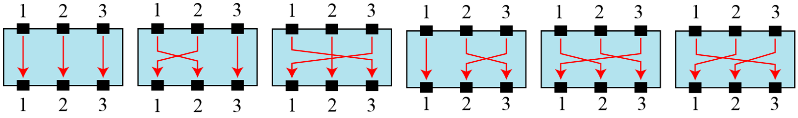


Figure 2: Although a P-box can use a key to define one of the n! mappings, P boxes are normally keyless, which means the mapping is predetermined. .

Straight P-boxes (or D-boxes)

- **straight P-boxes (or D-boxes)** : all 6 possible mappings of a 3 × 3 D-box.

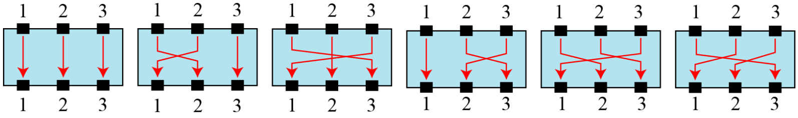


Figure 2: Although a P-box can use a key to define one of the n! mappings, P boxes are normally keyless, which means the mapping is predetermined. .

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

Soma Saha

Example of a permutation table for a straight D-box/P-box for a 64 x 64 D-Box.

Example

- Design an 8 × 8 permutation table for a straight P-box that moves 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.

Example

- Design an 8 × 8 permutation table for a straight P-box that moves 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-Boxes (or D-boxes

- A compression P-box is a P-box with n inputs and m outputs where m < n.

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

Figure 4: Example of a 32 × 24 permutation table.

Expansion P-Boxes (or D-boxes)

- An expansion P-box is a P-box with n inputs and m outputs where m > n

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

Figure 5: Example of a 12 × 16 permutation table.

Soma Saha

P-Boxes/D-Boxes: Invertibility

- **A straight P-box is invertible, but compression and expansion P-boxes are not.**

P-Boxes/D-Boxes: Invertibility

- **A straight P-box is invertible, but compression and expansion P-boxes are not.**
- How can you invert a permutation table to be represented as a one-dimensional table?

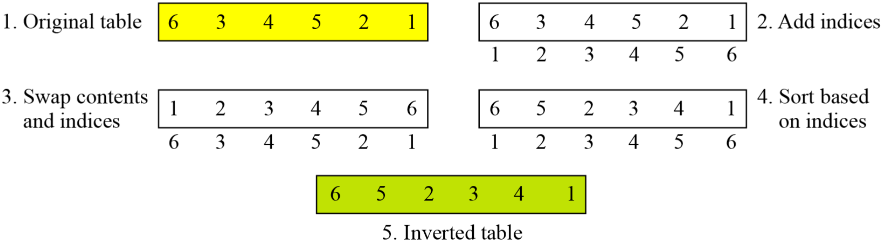
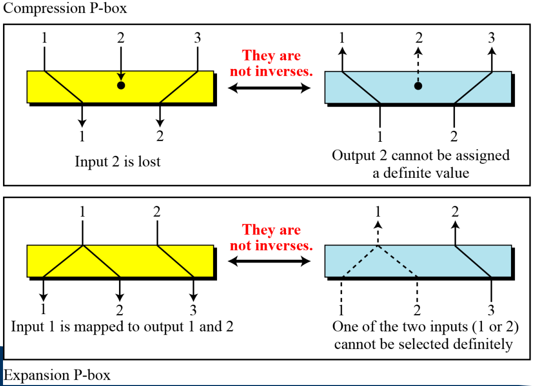


Figure 6: Inverting a permutation table.

Compression and expansion P-boxes are non-invertible



Soma Saha

S-boxes

- 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.**
- For example: The following table defines the input/output relationship for an S-box of size 3 × 2. The leftmost bit of the input defines the row; the two rightmost bits of the input define the column. The two output bits are values on the cross section of the selected row and column.

Leftmost bit

Rightmost bits

	00	01	10	11
0	00	10	01	11
1	10	00	11	01

Output bits

Figure 6. **Based on the table, an input of 010 yields the output 01. An input of 101 yields the output of 00.**

S-Boxes: Invertibility

- An S-box may or may not be invertible. In an invertible S-box, the number of input bits should be the same as the number of output bits.

S-Boxes: Invertibility.. contd

- The following shows an example of an invertible S-box. For example, if the input to the left box is 001, the output is 101. The input 101 in the right table creates the output 001, which shows that the two tables are inverses of each other.

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

XOR (Exclusive-Or)

- An important component in most block ciphers is the exclusive-or operation.

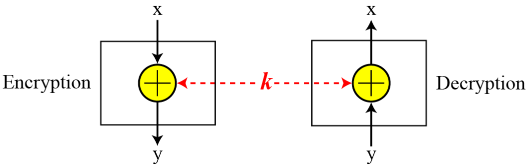


Figure 10: Invertibility of the exclusive-or operation.

XOR.. Contd..

- An important component in most block ciphers is the exclusive-or operation.
- **The five properties of the exclusive-or operation makes this operation a very interesting component for use in a block cipher: closure, associativity, commutativity, existence of identity, and existence of inverse.**

X EXOR 0 = X
X EXOR 1 = $\overline{X}$
X EXOR $\overline{X}$ = 1
X EXOR X = 0

Exclusive OR ...Contd

- The inverse of a component in a cipher makes sense if the component represents a unary operation (one input and one output).

Exclusive OR …Contd

- The inverse of a component in a cipher makes sense if the component represents a unary operation (one input and one output).
- For example, a keyless P-box or a keyless S-box can be made invertible because they have one input and one output.
- An exclusive-or operation is a binary operation. The inverse of an exclusive-or operation can make sense only if one of the inputs is fixed (is the same in encryption and decryption).

Exclusive OR ...Contd

- The inverse of a component in a cipher makes sense if the component represents a unary operation (one input and one output).
- For example, a keyless P-box or a keyless S-box can be made invertible because they have one input and one output.
- An exclusive-or operation is a binary operation. The inverse of an exclusive-or operation can make sense only if one of the inputs is fixed (is the same in encryption and decryption).
- For example, if one of the inputs is the key, which normally is the same in encryption and decryption, then an exclusive-or operation is self-invertible, as shown in Last Figure. 10.

Circular Shift

- Another component used in some modern block ciphers is the **circular shift** operation.

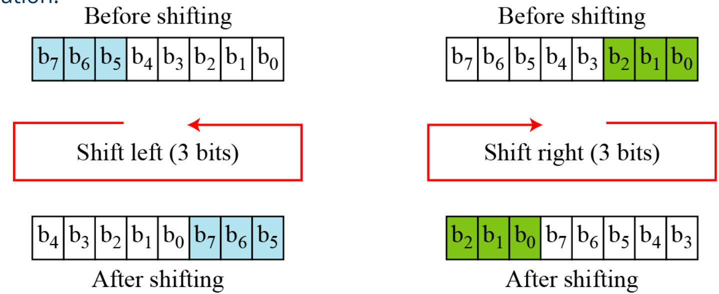


Figure 11: Circular shifting an 8-bit word to the left or right.

Swap

- The **swap** operation is **a special case of the circular shift operation** where **$k = n/2$**.

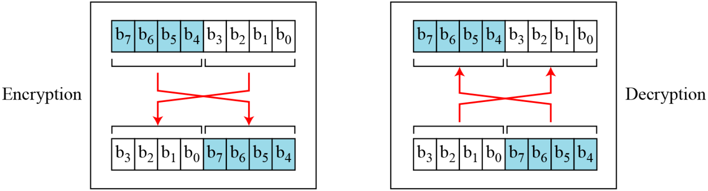


Figure 12: Swap operation on an 8-bit word.

Split and Combine

- Two other operations found in some block ciphers are split and combine.

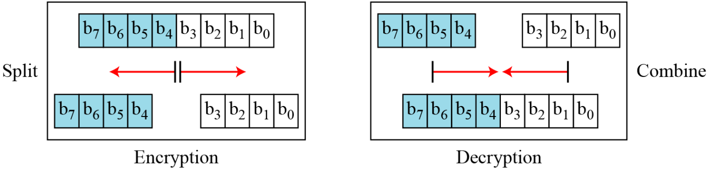


Figure 13: Split and combine operations on an 8-bit word.

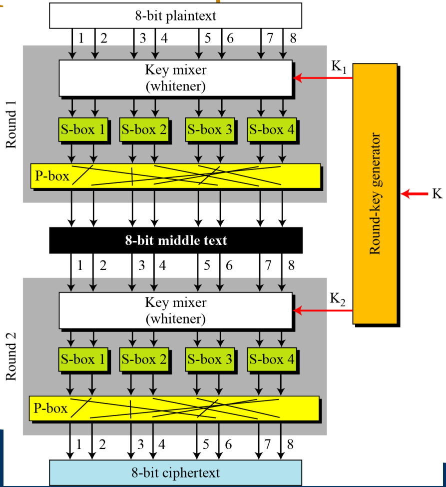
Product Cipher

- **Shannon** introduced the concept of **product cipher**.
- A **product cipher** is a complex cipher combining substitution, permutation, and other components discussed in previous sections.

Diffusion, confusion, and Rounds

- **Diffusion:** The idea to hide the relationship between the ciphertext and the plaintext.
- **Confusion:** The idea to hide the relationship between the ciphertext and the key.
- **Rounds:** Diffusion and confusion can be achieved using iterated product ciphers where each iteration is a combination of S-boxes, P-boxes/D-boxes, and other components.

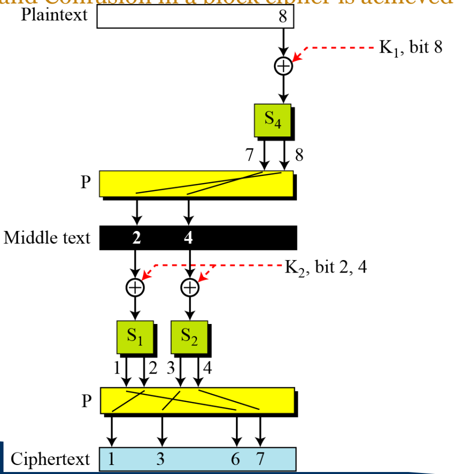
Product cipher Example



Soma Saha

A product cipher made of two rounds

How does Diffusion and Confusion in a block cipher is achieved?



Diffusion and confusion in a block cipher.

Soma Saha

Classification of Product Ciphers

- Modern block ciphers are all product ciphers, but they are divided into two classes.
  - **Feistel ciphers**
  - **Non-Feistel ciphers**

Classification of Product Ciphers

- Modern block ciphers are all product ciphers, but they are divided into two classes.

- **Feistel ciphers**
- Has been used for decades.
- Can have three types of components : self-invertible, invertible, and non-invertible.
- Example: **DES**

- **Non-Feistel ciphers:**
- Uses only invertible components.
- A component in the encryption cipher has the corresponding component in the decryption cipher.
- Example: **AES**

Feistel Ciphers: First Thought

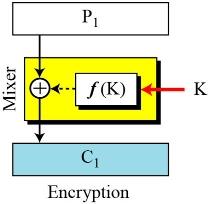


Figure 16: The first thought in Feistel cipher design: **f(K) is a non-invertible function**.

- **Encryption:** $C_1 = P_1 \oplus f(K)$
- **Decryption:**
  $P_2 = C_2 \oplus f(K) = C_1 \oplus f(K) = P_1 \oplus f(K) \oplus f(K) = P_1 \oplus (00\ldots0) = P_1$

Soma Saha

Feistel Ciphers: First Thought

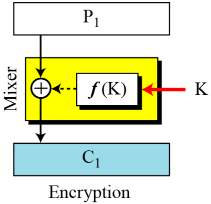


Figure 16: The first thought in Feistel cipher design: **f(K) is a non-invertible function**.

- **Encryption:** $C_1 = P_1 \oplus f(K)$
- **Decryption:**
  $P_2 = C_2 \oplus f(K) = C_1 \oplus f(K) = P_1 \oplus f(K) \oplus f(K) = P_1 \oplus (00\ldots0) = P_1$
- **The mixer(combination of function and ex-or operation) in the Feistel design is self-invertible.**

Soma Saha

Example

- This is a trivial example. The plaintext and ciphertext are each 4 bits long and the key is 3 bits long. Assume that the function takes the first and third bits of the key, interprets these two bits as a decimal number, squares the number, and interprets the result as a 4-bit binary pattern. Show the results of encryption and decryption if the original plaintext is 0111 and the key is 101.

Example

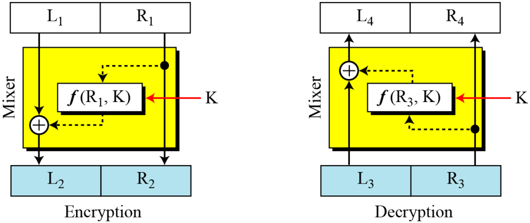
- This is a trivial example. The plaintext and ciphertext are each 4 bits long and the key is 3 bits long. Assume that the function takes the first and third bits of the key, interprets these two bits as a decimal number, squares the number, and interprets the result as a 4-bit binary pattern. Show the results of encryption and decryption if the original plaintext is 0111 and the key is 101.
- **Solution:**
  - The function extracts the first and second bits to get 11 in binary or 3 in decimal. The result of squaring is 9, which is 1001 in binary.

$$\textbf{Encryption: } C = P \oplus f\,(K) = 0111 \oplus 1001 = 1110$$

$$\textbf{Decryption: } P = C \oplus f\,(K) = 1110 \oplus 1001 = 0111$$

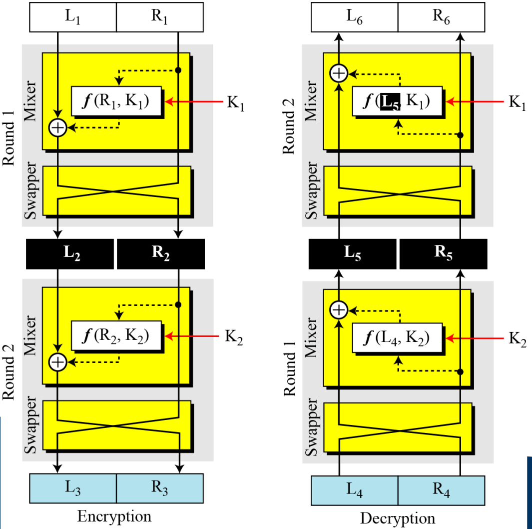
Soma Saha

Improvement of the previous Feistel design



- $R_4 = R_3 = R_2 = R_1$
- $L_4 = L_3 \oplus f(R_3, K) = L_2 \oplus f(R_2, K) = L_1 \oplus f(R_1, K) \oplus f(R_1, K) = L_1$
- **Flaw in this design: Right half of the plaintext never changes.**

Final design of a Feistel cipher with two rounds



Soma Saha

Feistel Cipher design.. contd..

- Let us see if $L_6 = L_1$ and $R_6 = R_1$, assuming that $L_4 = L_3$ and $R_4 = R_3$ (no change in ciphertext during transmission).
- We first prove the equality for the middle text.
  $L_5 = R_4 \oplus f(L_4, K_2) = R_3 \oplus f(R_2, K_2) = L_2 \oplus f(R_2, K_2) \oplus f(R_2, K_2) = L_2$
  $R_5 = L_4 = L_3 = R_2$
- Then it is easy to prove that the equality holds for two plaintext blocks.
  $L_6 = R_5 \oplus f(L_5, K_1) = R_2 \oplus f(L_2, K_1) = L_1 \oplus f(R_1, K_1) \oplus f(R_1, K_1) = L_1$
  $R_6 = L_5 = L_2 = R_1$

Feistel Ciphers

- Blowfish.
- Camellia.
- CAST-128.
- DES.
- FEAL.
- GOST 28147-89.
- ICE.

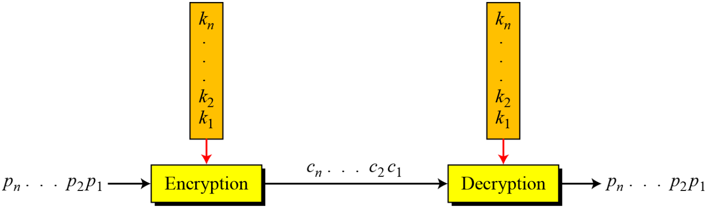
Modern Block Cipher: Secure??

- Attacks on traditional ciphers can also be used on modern block ciphers, but today's block ciphers resist most of the attacks discussed in classes/unit2_Cryptography_partI slides.
- **Linear cryptanalysis and Differntial cryprtanalysis are the two most widely used attacks on block ciphers.**
- Eli Biham and Adi Shamir introduced the idea of **differential cryptanalysis**. This is a chosen-plaintext attack.
  - Differential cryptanalysis is based on a nonuniform differential distribution table of the S-boxes in a block cipher.
- **Linear cryptanalysis** was presented by Mitsuru Matsui in 1993. The analysis uses known plaintext attacks.

Modern Stream Ciphers

- In a modern stream cipher, encryption and decryption are done **r** bits at a time. We have a plaintext bit stream $P = p_n \ldots p_2\, p_1$, a ciphertext bit stream C = $c_n \ldots c_2\, c_1$, and a key bit stream K = $k_n \ldots k_2\, k_1$, in which p_i , c_i, and k_i are **r**-bit words.
- Encryption is $c_i = E(k_i, p_i)$, and
- Decryption is $p_i = D(k_i, c_i)$.
- **Classification:**
  1. Synchronous Stream Ciphers
  2. Nonsynchronous Stream Ciphers

Stream Cipher



- In a modern stream cipher, each r-bit word in the plaintext stream is enciphered using an r-bit word in the key stream to create the corresponding r-bit word in the ciphertext stream.

Synchronous Stream Ciphers

- In a synchronous stream cipher the key is independent of the plaintext or ciphertext.

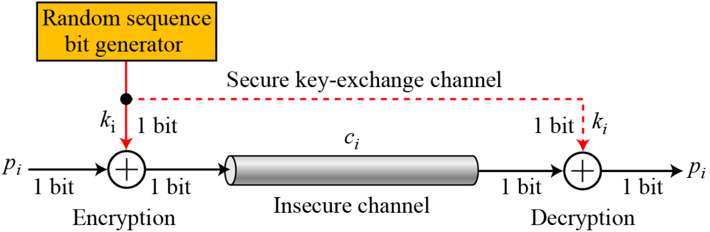


Figure 17: **One-time pad**: one-time pad invented and patented by Gilbert Vernam..

Soma Saha

Example

- What is the pattern in the ciphertext of a one-time pad cipher in each of the following cases?

  a. The plaintext is made of n 0's.
  b. The plaintext is made of n 1's.
  c. The plaintext is made of alternating 0's and 1's.
  d. The plaintext is a random string of bits.

Example

- What is the pattern in the ciphertext of a one-time pad cipher in each of the following cases?
  a. The plaintext is made of n 0's.
  b. The plaintext is made of n 1's.
  c. The plaintext is made of alternating 0's and 1's.
  d. The plaintext is a random string of bits.
- **solution**
  a. Because $0 \oplus k_i = k_i$, the ciphertext stream is the same as the key stream. If the key stream is random, the ciphertext is also random. The patterns in the plaintext are not preserved in the ciphertext.

Example..contd...

- Because $1 \oplus k_i = \bar{k}_i$ where $\bar{k}_i$ is the complement of k_i , the ciphertext stream is the complement of the key stream. If the key stream is random, the ciphertext is also random. Again the patterns in the plaintext are not preserved in the ciphertext.
- In this case, each bit in the ciphertext stream is either the same as the corresponding bit in the key stream or the complement of it. Therefore, the result is also a random string if the key stream is random.
- In this case, the ciphertext is definitely random because the exclusive-or of two random bits results in a random bit.

Feedback Shift Register

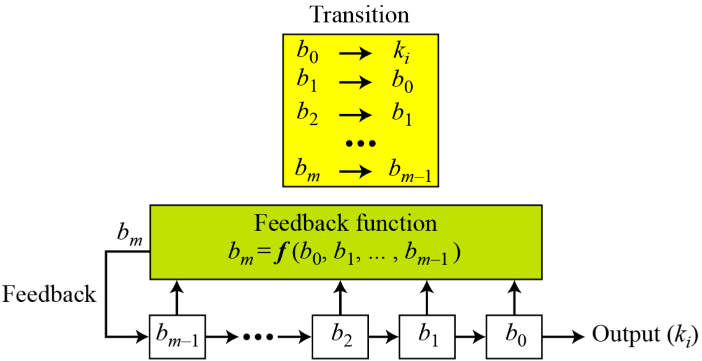
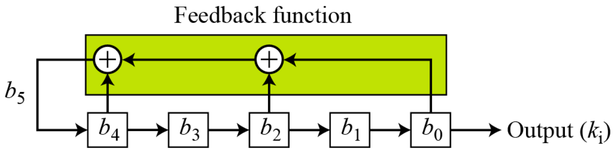


Figure 18: Feedback shift Register.

Example..1

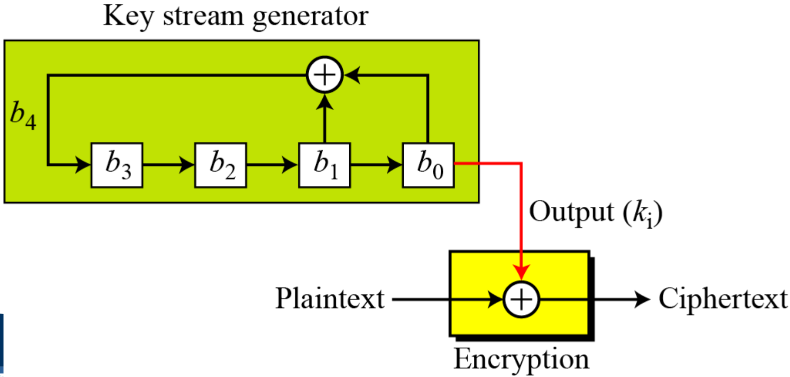
- Create a linear feedback shift register with 5 cells in which $b5 = b4 \oplus b2 \oplus b0$.
- If $c_i = 0$, b_i has no role in calculation of b_m. This means that b_i is not connected to the feedback function. If $c_i = 1$, b_i is involved in calculation of b_m. In this example, c1 and c3 are 0's, which means that we have only three connections. following figure shows the design.



Soma Saha

Example..2

- Create a linear feedback shift register with 4 cells in which $b_4 = b_1 \oplus b_0$. Show the value of output for 20 transitions (shifts) if the seed is $(0001)_2$.



Soma Saha

Example2: Cell values and key sequence..

States	b_4	b_3	b_2	b_1	b_0	k_i
Initial	1	0	0	0	1	
1	0	1	0	0	0	1
2	0	0	1	0	0	0
3	1	0	0	1	0	0
4	1	1	0	0	1	0
5	0	1	1	0	0	1
6	1	0	1	1	0	0
7	0	1	0	1	1	0
8	1	0	1	0	1	1
9	1	1	0	1	0	1
10	1	1	1	0	1	0

Example2: Cell values and key sequence... cont..2

11	1	1	1	1	0	1
12	0	1	1	1	1	0
13	0	0	1	1	1	1
14	0	0	0	1	1	1
15	1	0	0	0	1	1
16	0	1	0	0	0	1
17	0	0	1	0	0	0
18	1	0	0	1	0	0
19	1	1	0	0	1	0
20	1	1	1	0	0	1

Nonsynchronous Stream Cipher

- In a nonsynchronous stream cipher, each key in the key stream depends on previous plaintext or ciphertext.

Data Encryption Standard (DES)

- The most widely used cipher in civilian applications.
- Developed by IBM; Evolved from Lucifer.
- Accepted as an US NBS standard in 1977, and later as an international standard, the National Institute of Standards and Technology (NIST).
- A block cipher with **N = 64 bit blocks**.
- **56-bit keys** (eight bytes, in each byte seven bits are used; the eighth bit can be used as a parity bit).
- Exhaustive search requires 2^{56} encryption steps (2^{55} on average).
- Iterates a round-function 16 times in **16 rounds**. The round-function mixes the data with the key.
- Each round, the key information entered to the round function is called a subkey. The subkeys $K_1, \ldots, K_{16}$ are computed by **a key scheduling algorithm**.

DES Overview

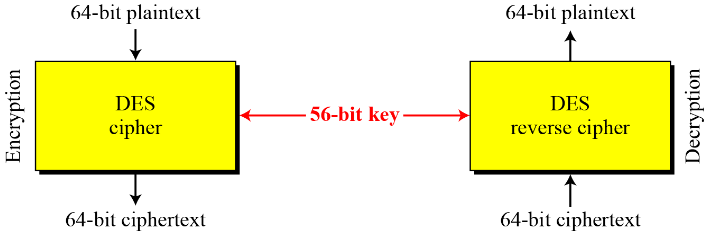
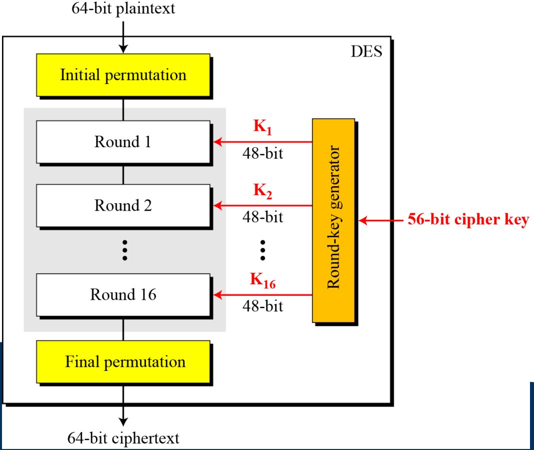


Figure 19: Encryption and decryption with DES.

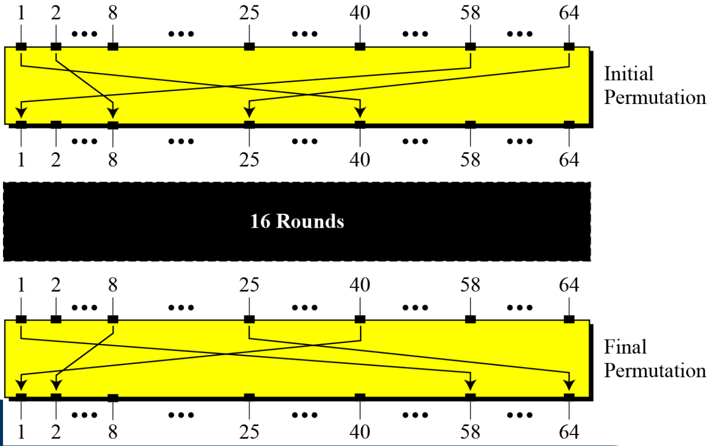
General Structure of DES

- The encryption process is made of two permutations (P-boxes), which we call initial and final permutations, and sixteen Feistel rounds.



Soma Saha

Initial and final permutation in DES



Initial and final permutation tables

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

Figure 22: Initial and final permutation tables in DES.

Example1

- Find the output of the final permutation box when the input is given in hexadecimal as:

  0x0000 0080 0000 0002

- **Solution:**
  Only bit 25 and bit 63 are 1s; the other bits are 0s. In the final permutation, bit 25 becomes bit 64 and bit 63 becomes bit 15. The result is

  0x0002 0000 0000 0001

Example2

- Prove that the initial and final permutations are the inverse of each other by finding the output of the initial permutation if the input is

0x0002 0000 0000 0001

- **Solution:** The input has only two 1s; the output must also have only two 1s. Using Table 6.1, we can find the output related to these two bits. Bit 15 in the input becomes bit 63 in the output. Bit 64 in the input becomes bit 25 in the output. So the output has only two 1s, bit 25 and bit 63. The result in hexadecimal is

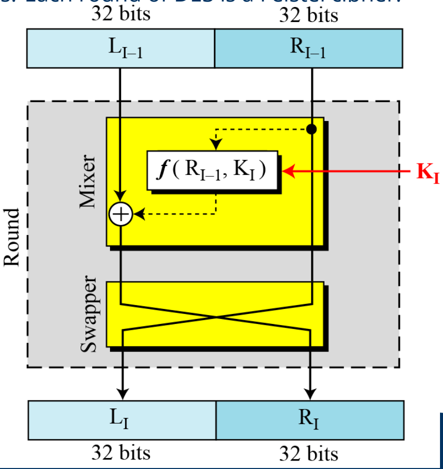
0x0000 0080 0000 0002

Cryptographic significance of initial and final permutations

- **The initial and final permutations are straight P-boxes that are inverses of each other. They have no cryptography significance in DES.**

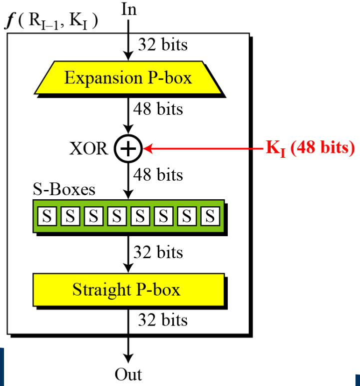
Rounds

- DES uses 16 rounds. Each round of DES is a Feistel cipher.



DES function

- The heart of DES is the DES function. The DES function applies a 48-bit key to the rightmost 32 bits to produce a 32-bit output.



Soma Saha

Expansion P-Box in DES

- 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.

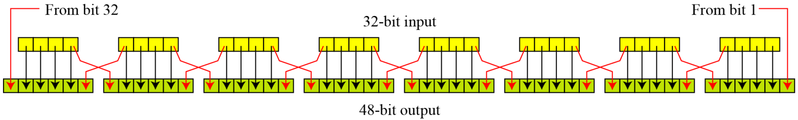


Figure 25: Expansion P-box

Contd...

- Although the relationship between the input and output can be defined mathematically, DES uses Table 6.2 to define this P-box.

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

Figure 26: Expansion P-box Table

Whitener (XOR) in DES

- After the expansion permutation, 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. Also note that the round key is used only in this operation.

S-Boxes in DES

- The S-boxes do the real mixing (confusion). DES uses 8 S-boxes, each with a 6-bit input and a 4-bit output.

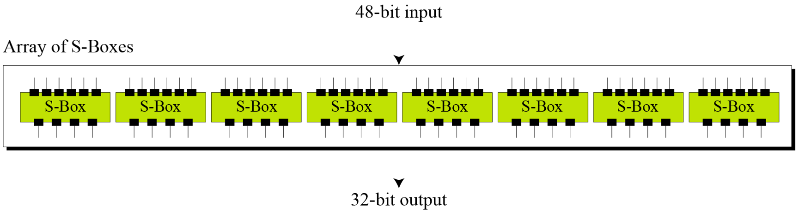
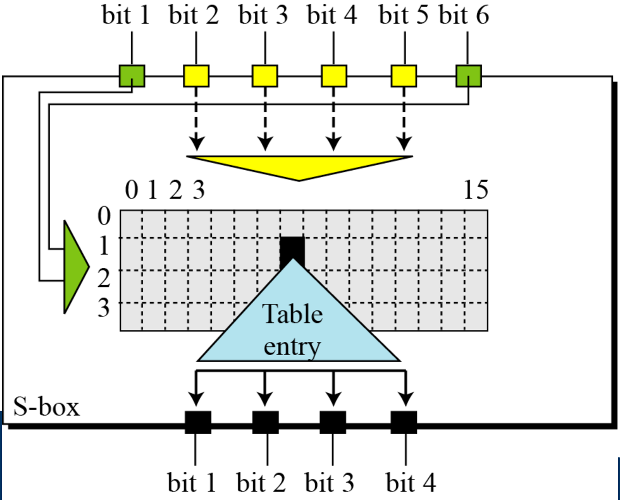


Figure 27: S-boxes

S-box rule for DES



Soma Saha

permutation for S-box 1

	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

Figure 29: permutation for S-box 1, rest can be checked from textbook.

Example 1

- When the S-box 1 (in book Table 6.3) is referred and the input to S-box 1 is 100011. What is the output?
- **Solution:**
  If we write the first and the sixth bits together, we get 11 in binary, which is 3 in decimal. The remaining bits are 0001 in binary, which is 1 in decimal. We look for the value in row 3, column 1, in Table 6.3 (S-box 1). The result is 12 in decimal, which in binary is 1100. So the input 100011 yields the output 1100.

Example 2

- When the S-box 8 (in book Table 6.10) is referred and the input to S-box 8 is 000000. What is the output?
- **Solution:**
  If we write the first and the sixth bits together, we get 00 in binary, which is 0 in decimal. The remaining bits are 0000 in binary, which is 0 in decimal. We look for the value in row 0, column 0, in Table 6.10 (S-box 8). The result is 13 in decimal, which is 1101 in binary. So the input 000000 yields the output 1101.

Straight Permutation table in DES function

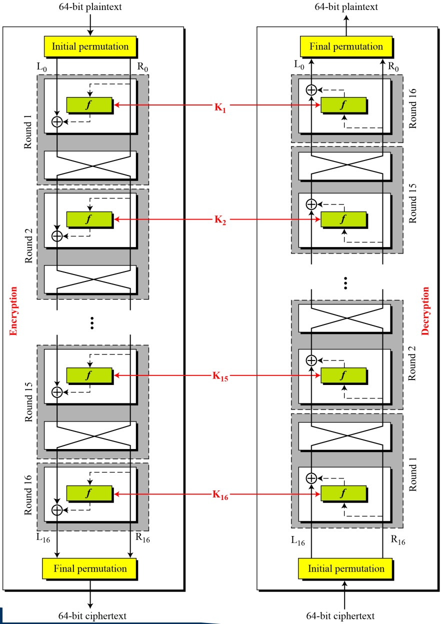
16	07	20	21	29	12	28	17
01	15	23	26	05	18	31	10
02	08	24	14	32	27	03	09
19	13	30	06	22	11	04	25

Figure 30: Straight Permutation table

Soma Saha

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.
  - *In the first approach, there is no swapper in the last round.*



Soma Saha

DES cipher and reverse cipher for the first approach.

Pseudocode for DES cipher

```
Cipher (plainBlock[64], RoundKeys[16, 48], cipherBlock[64])
{
    permute (64, 64, plainBlock, inBlock, InitialPermutationTable)
    split (64, 32, inBlock, leftBlock, rightBlock)
    for (round = 1 to 16)
    {
        mixer (leftBlock, rightBlock, RoundKeys[round])
        if (round!=16) swapper (leftBlock, rightBlock)
    }
    combine (32, 64, leftBlock, rightBlock, outBlock)
    permute (64, 64, outBlock, cipherBlock, FinalPermutationTable)
}
```

Soma Saha

Pseudocode for DES cipher.. Contd…1

```
mixer (leftBlock[48], rightBlock[48], RoundKey[48])
{
    copy (32, rightBlock, T1)
    function (T1, RoundKey, T2)
    exclusiveOr (32, leftBlock, T2, T3)
    copy (32, T3, rightBlock)
}
swapper (leftBlock[32], rigthBlock[32])
{
    copy (32, leftBlock, T)
    copy (32, rightBlock, leftBlock)
    copy (32, T, rightBlock)
}
```

Pseudocode for DES cipher.. Contd...2

```
substitute (inBlock[32], outBlock[48], SubstitutionTables[8, 4, 16])
{
    for (i = 1 to 8)
    {
       row ← 2 × inBlock[i × 6 + 1] + inBlock [i × 6 + 6]
       col  ← 8 × inBlock[i × 6 + 2] + 4 × inBlock[i × 6 + 3] +
              2 × inBlock[i × 6 + 4] + inBlock[i × 6 + 5]

       value = SubstitutionTables [i][row][col]


       outBlock[[i × 4 + 1] ← value / 8;        value ← value mod 8
       outBlock[[i × 4 + 2] ← value / 4;        value ← value mod 4
       outBlock[[i × 4 + 3] ← value / 2;        value ← value mod 2
       outBlock[[i × 4 + 4] ← value
    }
}
```

Soma Saha

Alternative approach

- We can make all 16 rounds the same by including one swapper to the 16^{th} round and add an extra swapper after that (two swappers cancel the effect of each other).

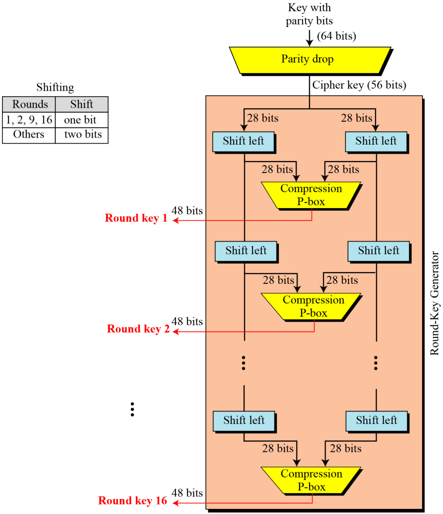


Figure 32: **Key Generation:The round-key generator creates sixteen 48-bit keys out of a 56-bit cipher key.**

Parity-bit Drop Table

- The preprocess before key expansion is a compression transposition step that we call **parity-bit drop**.
- It drops the parity bits (bits 8, 16, 24, 32,..., 64) from 64 -bit key and permutes the rest of the bits according to the following 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

- After the straight permutation, the key is divided into two 28-bit parts. Each part is shifted left (circular shift) one or two bits.
- In rounds 1, 2, 9, and 16, shifting is one bit; in the other rounds, it is two bits.
- The two parts are then combined to form 56-bit part.

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

Figure 33: **Number of shifts for each round.**

Soma Saha

Key-compression in Key Generation in DES

- The compression D-box or P-box changes the 58 bits to 48 bits, which are used as a key for a round.

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

Figure 34: **Key-compression table**

Algorithm for round-key generation..part1

```
Key_Generator (keyWithParities[64], RoundKeys[16, 48], ShiftTable[16])
{
    permute (64, 56, keyWithParities, cipherKey, ParityDropTable)
    split (56, 28, cipherKey, leftKey, rightKey)
     for (round = 1 to 16)
     {
          shiftLeft (leftKey, ShiftTable[round])
          shiftLeft (rightKey, ShiftTable[round])
          combine (28, 56, leftKey, rightKey, preRoundKey)
          permute (56, 48, preRoundKey, RoundKeys[round], KeyCompressionTable)
     }
}
```

Algorithm for round-key generation..part2

```
shiftLeft (block[28], numOfShifts)
{
    for (i = 1 to numOfShifts)
    {
        T ← block[1]
        for (j = 2 to 28)
        {
            block [j−1] ← block [j]
        }
        block[28] ← T
    }
}
```

Soma Saha

Example1

- We choose a random plaintext block and a random key, and determine what the ciphertext block would be (all in hexadecimal):

Plaintext: 123456ABCD132536 Key: AABB09182736CCDD
CipherText: C0B7A8D05F3A829C

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

Soma Saha

Example1..Trace of Data.. continued..

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)*

Soma Saha

Decryption/Deciphering at Receiver's end

- Let us see how Bob, at the destination, can decipher the ciphertext received from Alice using the same key. The following Table shows some interesting points.

Ciphertext: C0B7A8D05F3A829C			
After initial permutation: 19BA9212CF26B472 After splitting: L_0=19BA9212 R_0=CF26B472			
Round	*Left*	*Right*	*Round Key*
Round 1	CF26B472	BD2DD2AB	181C5D75C66D
Round 2	BD2DD2AB	387CCDAA	3330C5D9A36D
. . .	. . .	. . .	. . .
Round 15	5A78E394	18CA18AD	4568581ABCCE
Round 16	14A7D678	18CA18AD	194CD072DE8C
After combination: 14A7D67818CA18AD			
Plaintext:123456ABCD132536 (after final permutation)			

Bibliography: Books and Resources

- Cryptography and Network Security: Principles and Practice by William Stallings
- Cryptography and Network Security by Behrouz A Forouzan and Debdeep Mukhopadhyay
- Principles of Information Security by Michael E. Whitman and Herbert J. Mattord.
- Cisco platform, and Internet.
- Published research papers, study materials from researchers of security domain.