Network Security

3. Block Ciphers and DES

Outline

- Block Cipher Principles
- Simplified-Data Encryption Standard
- Data Encryption Standard
- Block Cipher Modes of Operation

Block Cipher Principles

Stream vs. Block Cipher

- Stream cipher is one that encrypts a digital data stream one bit or one byte at a time
  - e.g. Vigenere cipher
- Block cipher is one in which a block of plaintext is treated as a whole and used to produce a ciphertext block of equal length
  - e.g. S-DES and DES
- The vast majority of network-based cryptographic applications make use of block cipher.

Claude Shannon and Substitution-Permutation Ciphers

- in 1949 Claude Shannon introduced idea of substitution-permutation (S-P) networks
  - modern substitution-transposition product cipher
- these form the basis of modern block ciphers
- S-P networks are based on the two primitive cryptographic operations we have seen before:
  - *substitution* (S-box)
  - *permutation* (P-box)
- provide *confusion* and *diffusion* of message

Confusion and Diffusion

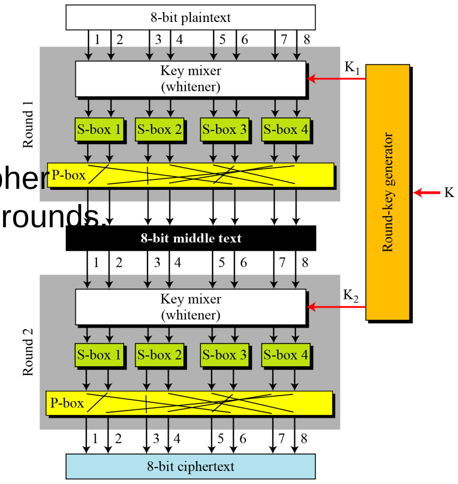
- cipher needs to completely obscure statistical properties of original message
- Shannon suggested combining elements to obtain:
  - **Confusion**
    - This is achieved by the use of a complex substitution algorithm.
    - The idea of confusion is to hide the relationship between the ciphertext and the key.
  - **Diffusion**
    - This is achieved through numerous permutations
    - The idea of diffusion is to hide the relationship between the ciphertext and the plaintext

Continued

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.

Product Cipher



A product cipher made of two rounds.

Two Classes of Product Ciphers

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

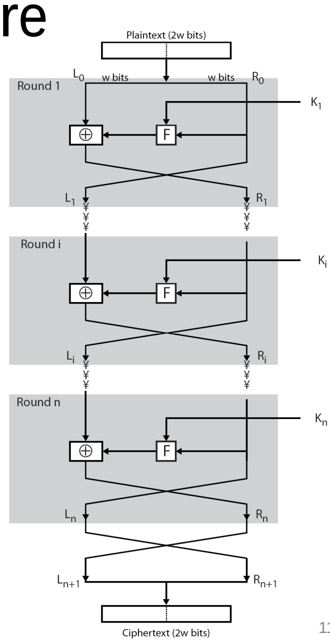
9

Feistel Cipher

- Horst Feistel devised the **feistel cipher**
  - based on concept of invertible product cipher
- partitions input block into two halves
  - process through multiple rounds which
  - perform a substitution on left data half
  - based on round function of right half & subkey
  - then have permutation swapping halves
- implements Shannon's substitution-permutation network concept

10

Feistel Cipher Structure



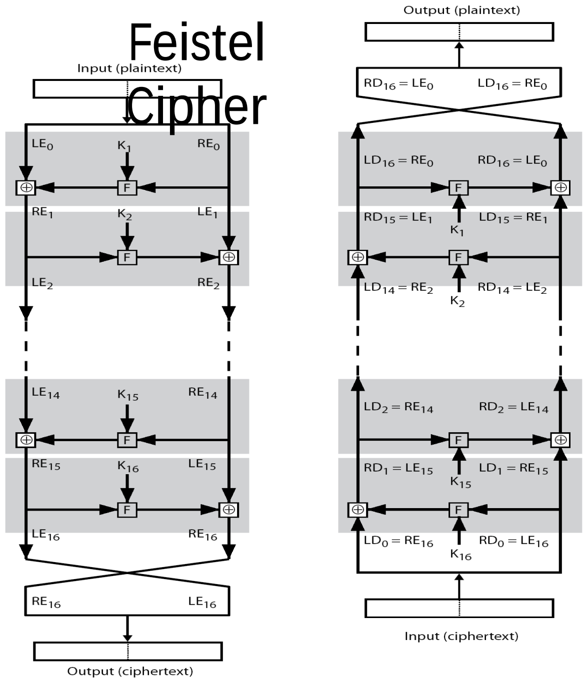
11

Feistel Cipher Design Principles

- **block size**
  - increasing size improves security, but slows cipher
- **key size**
  - increasing size improves security, makes exhaustive key searching harder, but may slow cipher
- **number of rounds**
  - increasing number improves security, but slows cipher
- **subkey generation**
  - greater complexity can make analysis harder, but slows cipher
- **round function**
  - greater complexity can make analysis harder, but slows cipher
- **fast software en/decryption & ease of analysis**
  - are more recent concerns for practical use and testing

12

Feistel Cipher



13

Simplified Data Encryption Standard (S-DES)

S-DES: An Overview

- Similar properties and structure to DES, with much smaller parameters.
- Encryption
  - It takes an 8-bit block of plain text and a 10-bit key as input and produces an 8-bit block of cipher text as output.
- Decryption
  - It takes an 8-bit block of cipher text and the same 10-bit key used to produce that Ciphertext as input and produces the original 8-bit block of plaintext.

15

S-DES Algorithm

Algorithm involves 5 functions

1. An initial permutation (IP).
2. A complex function, f_k , that involves both permutation and substitution operations and depends on a key input.
3. A simple permutation function that switches the two halves of the data (SW).
4. The function f_k again.
5. A permutation function that is the inverse of the initial one (IP^{-1}).

16

S-DES Algorithm

$$C = (IP^{-1} \circ f_{K_2} \circ SW \circ f_{K_1} \circ IP)$$

or

$$Ciphertext = IP^{-1}(f_{K_2}(SW(f_{K_1}(IP(plaintext)))))$$

where

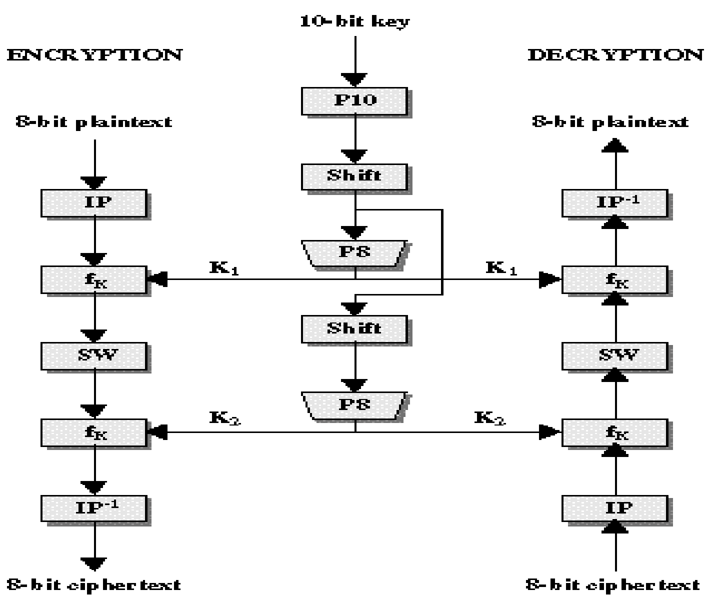
$$K_1 = P8(Shift(P10(key)))$$

$$K_2 = P8(Shift(Shift(P10(key))))$$

and

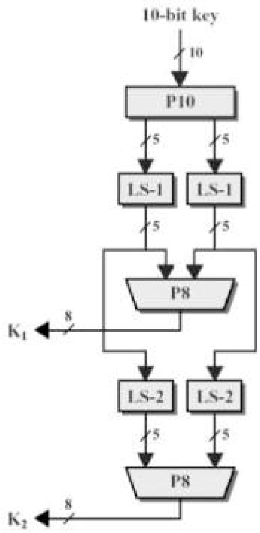
$$Plaintext = IP^{-1}(f_{K_1}(SW(f_{K_2}(IP(ciphertext)))))$$

17



18

S-DES Key Generation



- A 10-bit key is shared between sender and receiver.
- From this key, two 8-bit sub keys are produced for use in particular stages of the encryption and decryption algorithm.
- First permute the key:

$P10(k_1,k_2,k_3,k_4,k_5,k_6,k_7,k_8,k_9,k_{10})=(k_3,k_5,k_2,k_7,k_4,k_{10},k_1,k_9,k_8,k_6)$

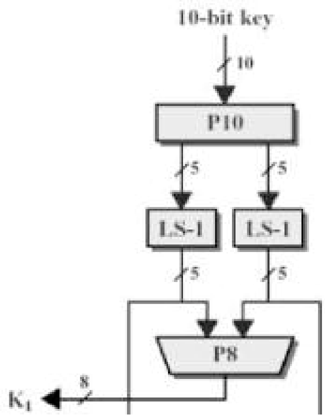
19

S-DES Key Generation (contd.)

- Next perform a circular-left shift (LS-1), or rotation, separately on the first five bits and the second five bits.

  $LS\text{-}1(k_1,k_2,k_3,k_4,k_5)=(k_2,k_3,k_4,k_5,k_1)$
- Next apply P8, which permutes 8 of the 10 bits according to the following rule,

  $P8(k_1,k_2,k_3,k_4,k_5,k_6,k_7,k_8,k_9,k_{10})=(k_6,k_3,k_7,k_4,k_8,k_5,k_{10},k_9)$
- The result is sub key 1 (K_1)

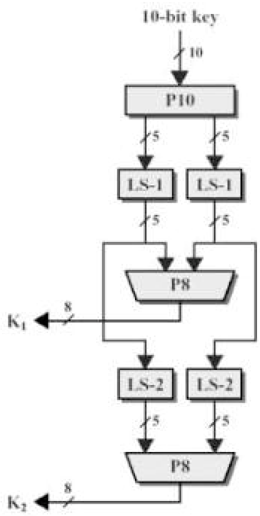


20

S-DES Key Generation (contd.)

- To get the second sub key (K_2), perform again a 2-bit circular-left shift LS-2 on the product of LS-1

  LS-2(LS-1(k_1,k_2,k_3,k_4,k_5)=LS-2(k_2,k_3,k_4,k_5,k_1)=(k_4,k_5,k_1,k_2,k_3)
- Finally, P8 is applied again to produce K_2.



Example - Key Generation

- Permute 10-bit key: 1010000010

  P10(1010000010) = 1000001100
- Perform circular left shift, separately, on first 5 bits and second 5 bits (LS-1).

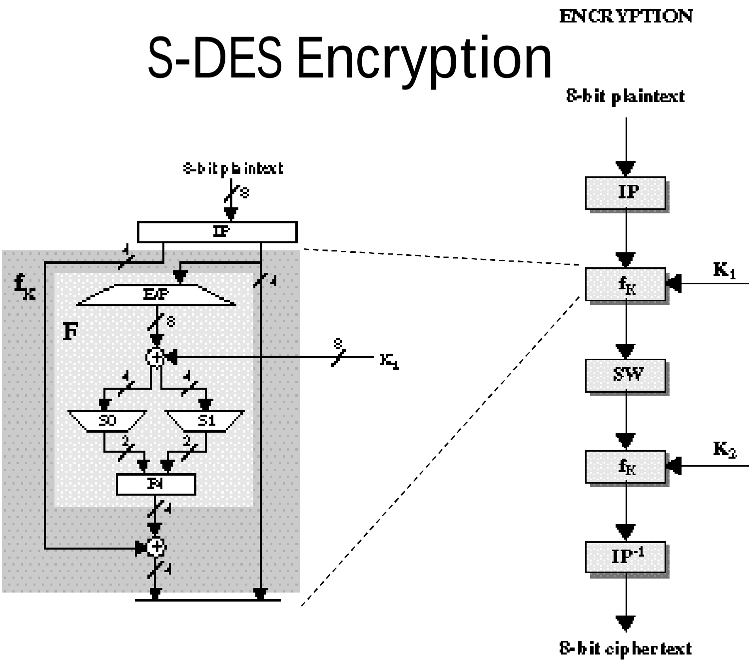
  LS-1(10000) = 00001 and LS-1(01100) = 11000
- Pick out and permute 8 of the 10 bits: (P8)

  P8(0000111000) = 10100100
- Result is K_1
- Now perform circular left shift of 2 bit positions, on first 5 bits and second 5 bits (LS-2) on the result LS-1.

  LS-2(00001) = 00100 and LS-2(11000) = 00011
- Apply (P8) again.

  P8(0010000011) = 01000011
- Result is K_2

S-DES Encryption



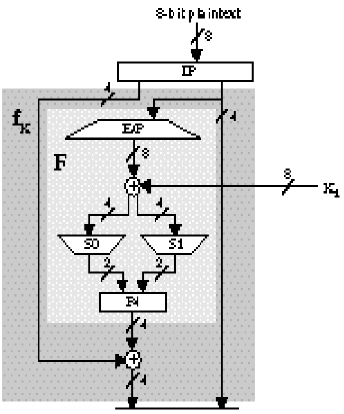
S-DES Encryption

- Initial and final permutations

IP							
2	6	3	1	4	8	5	7

IP^{-1}							
4	1	3	5	7	2	8	6

Mapping F



Input is 4-bit number ($n_1n_2n_3n_4$)

Expansion/permutation operation

E/P							
4	1	2	3	2	3	4	1

$$\begin{array}{c|cc|c} n_4 & n_1 & n_2 & n_3 \\ n_2 & n_3 & n_4 & n_1 \end{array}$$

8-bit sub key K1 is added to output of E/P using exclusive-OR.

Mapping F (contd.)

- First 4 bits fed to S-box S0,second 4 bits fed to S-box S1.
  - S-box uses 1st and 4th bits to specify a row,2nd and 3rd bits to specify a column. Entry in that position (base 2) is 2-bit output.
- 4-bits produced by S-Boxes are permuted using
  P4 = ($k_2k_4k_3k_1$)
- Output of P4 is output of F.

S0

1	0	3	2
3	2	1	0
0	2	1	3
3	1	3	2

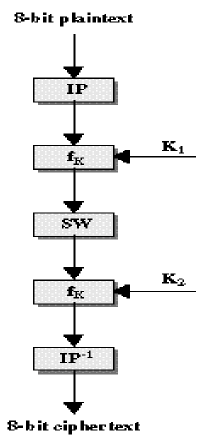
S1

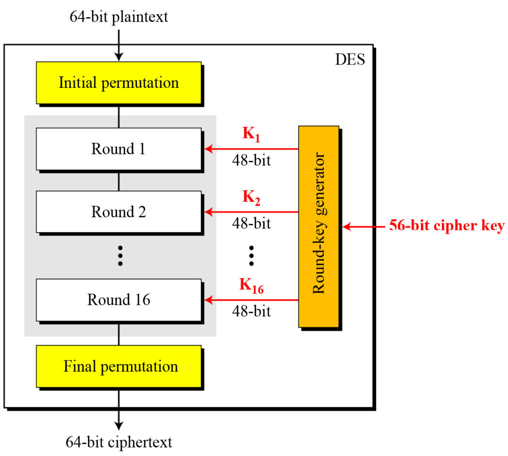
0	1	2	3
2	0	1	3
3	0	1	0
2	1	0	3

The Switch function

- The function f_k only alters the leftmost 4-bits of the input.
- Switch function interchanges left and right 4 bits so that second instance of f_k operates on a different 4 bits.
- In this second instance, the E/P, S0, S1, and P4 functions are same.
- The key input is K2.

S-DES vs. DES





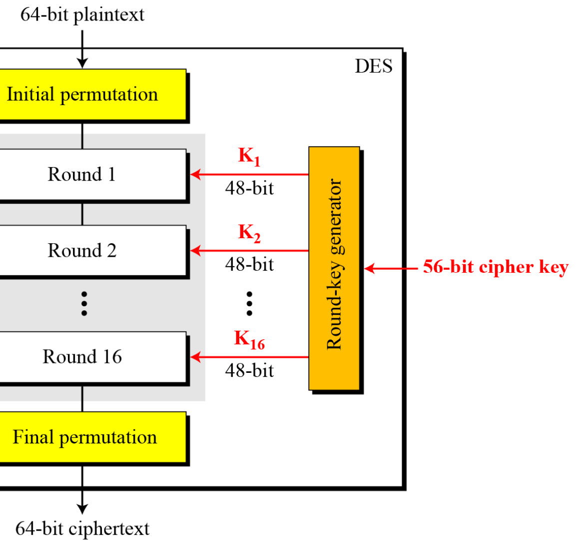
Data Encryption Standard

The Actual DES

About DES

- The original ideas behind the Data Encryption Algorithm were developed by IBM in the 1960's and was based on Claude Shannon’s concept.
- The technique was first called as Lucifer and later refined and renamed as the DEA (Data Encryption Algorithm)
- In 1977 the United States Government chose the Data Encryption Standard (DES)
- DES was widely adopted by industry for secure communication
  - DES is used to encrypt personal identification numbers (PINs) and account transactions in automated teller machines (ATMs)

DES Encryption



31

Data Encryption Standard

- There are two inputs to the encryption function, i.e. the plaintext and the key.
- The plain text must be 64 bits and the key is 56 bits in length.
- Processing of the plaintext proceeds in three phases,
  - The 64 bit plaintext passes through an Initial Permutation (IP)
  - Then a 16 "rounds" of operations which involves both permutation and substitution functions that mix the data and key together in a prescribed manner.
  - The pre-output obtained from swapping the left and right halves of the output in each round, is then passed through a Permutation (IP^{-1}) that is the inverse of the initial permutation function, to produce the 64-bit ciphertext.

32

Why 16 rounds?

- The goal is to completely scramble the data and key so that every bit of the ciphertext depends on every bit of the data and every bit of the key (a 56-bit quantity for the DES).
- After sufficient "rounds" with a good algorithm, there should be no correlation between the ciphertext and either the original data or key.
- The DES uses 16 rounds for some solid reasons.
  - First, a minimum of 12 rounds were needed to sufficiently scramble the key and data together; the others provided a margin of safety.
  - Second, the operation of 16 rounds would return the key back to its original position in an electronic device for the next use when used in accordance with the published algorithm.

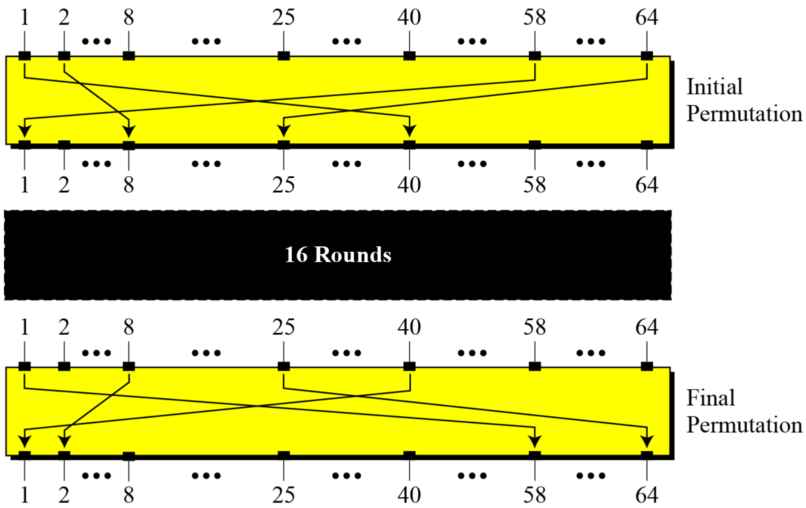
33

Initial Permutation IP

- first step of the data computation
- IP reorders the input data bits
- even bits to LH half, odd bits to RH half
- quite regular in structure (easy in h/w)

34

Initial and Final Permutations



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

DES Round Structure

- uses two 32-bit L & R halves
- as for any Feistel cipher can describe as:

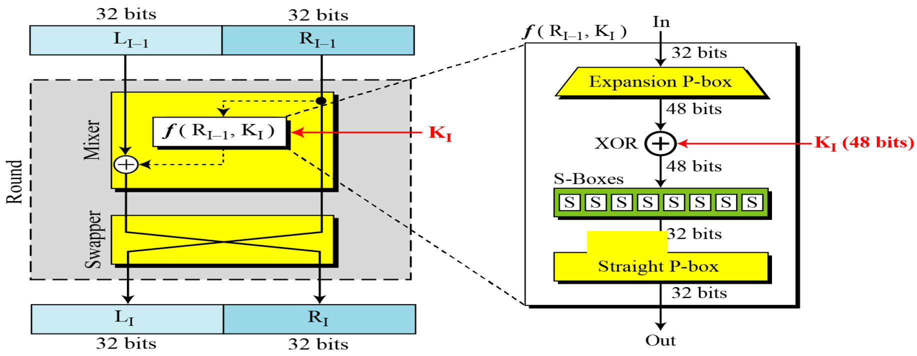
  $L_i = R_{i-1}$

  $R_i = L_{i-1} \oplus F(R_{i-1}, K_i)$
- F takes 32-bit R half and 48-bit subkey:
  – expands R to 48-bits using perm E
  – adds to subkey using XOR
  – passes through 8 S-boxes to get 32-bit result
  – finally permutes using 32-bit perm P

37

DES Round Structure

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

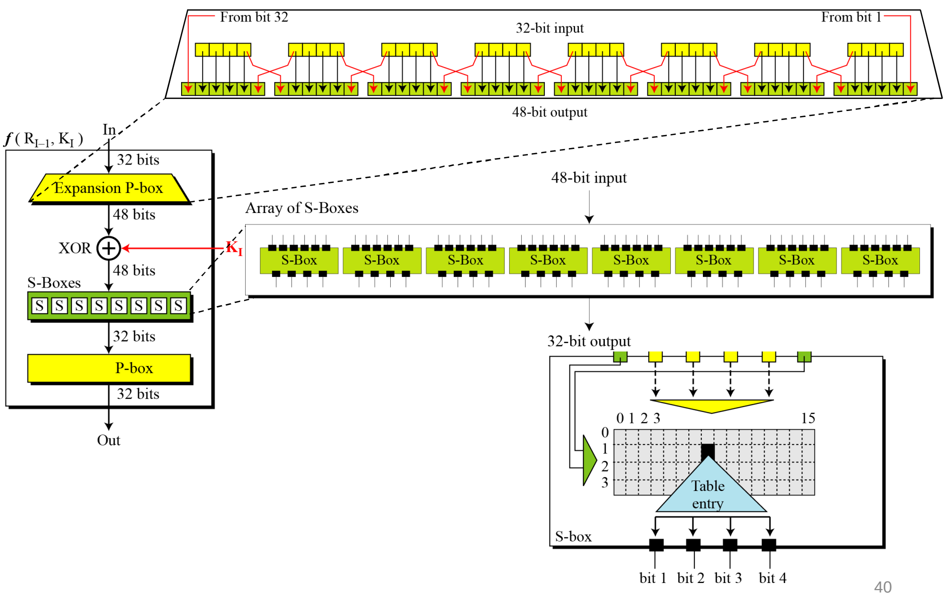


38

Substitution Boxes S

- have eight S-boxes which map 6 to 4 bits
- each S-box is actually 4 little 4 bit boxes
  - outer bits 1 & 6 (**row** bits) select one row of 4
  - inner bits 2-5 (**col** bits) are substituted
  - result is 8 lots of 4 bits, or 32 bits
- row selection depends on both data & key

DES Function



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

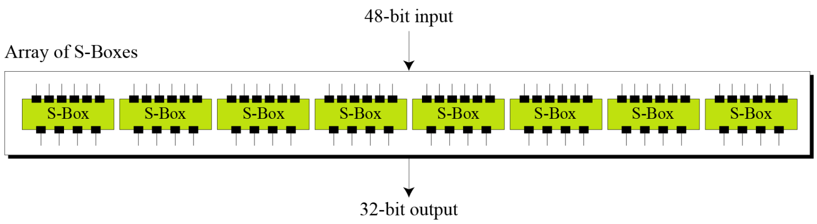
Table Expansion P-Box Table

XOR

- 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

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



43

S-Box Rule

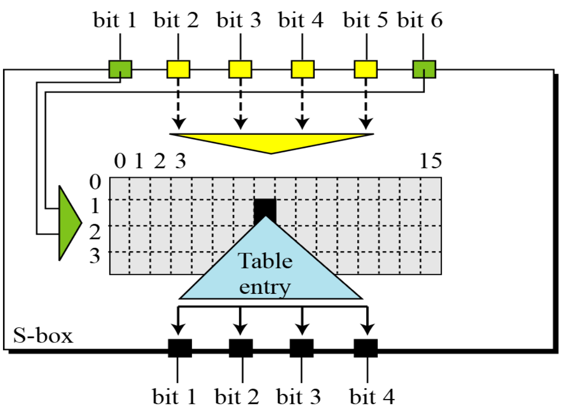


Table: S-Box Rule

44

S-Box 1

Table shows the permutation for S-box 1. For the rest of the boxes see the textbook.

	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

Straight Permutation

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

Table : Straight permutation table

DES Key Schedule

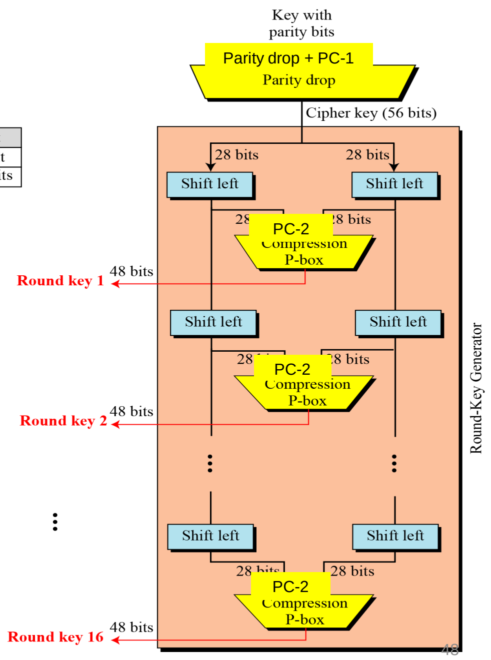
- forms subkeys used in each round
- consists of:
  - initial permutation of the key (PC1) which selects 56-bits in two 28-bit halves
  - 16 stages consisting of:
    - selecting 24-bits from each half
    - permuting them by PC2 for use in function f,
    - rotating **each half** separately either 1 or 2 places depending on the **key rotation schedule** K

47

Key Generation

Shifting

Rounds	Shift
1, 2, 9, 16	one bit
Others	two bits



48

Continued

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

Table 6.12 Parity-bit drop table

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

Table 6.13 Number of bits shifts

Key Compression

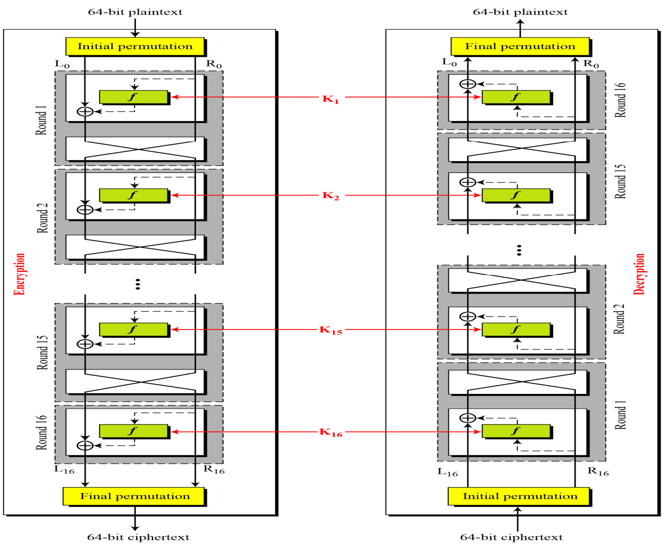
14	17	11	24	01	05	03	28
15	06	21	10	23	19	12	04
26	08	16	07	27	20	13	02
41	52	31	37	47	55	30	40
51	45	33	48	44	49	39	56
34	53	46	42	50	36	29	32

Table : Key-compression table

DES Decryption

- decrypt must unwind steps of data computation
- with Feistel design, do encryption steps again using subkeys in reverse order (SK16 … SK1)
  - IP undoes final FP step of encryption
  - 1st round with SK16 undoes 16th encrypt round
  - ….
  - 16th round with SK1 undoes 1st encrypt round
  - then final FP undoes initial encryption IP
  - thus recovering original data value

DES Cipher and Reverse



Avalanche Effect

- key desirable property of encryption algorithm
- where a change of **one** input or key bit results in changing approx **half** output bits
- making attempts to “home-in” by guessing keys impossible
- DES exhibits strong avalanche

53

Avalanche Effect

- To check the avalanche effect in DES, let us encrypt two plaintext blocks (with the same key) that differ only in one bit and observe the differences in the ciphertext
  - Here changing 1 bit (~1.5%) of the plaintext creates a change of 29 bits (~45%) in the ciphertext

Plaintext: 0000000000000000	Key: 22234512987ABB23
Ciphertext: 4789FD476E82A5F1	
Plaintext: 000000000000000<u>**1**</u>	Key: 22234512987ABB23
Ciphertext: 0A4ED5C15A63FEA3	

54

Triple DES

Multiple Encryption & DES

- A replacement for DES was needed
  - theoretical attacks that can break it
  - demonstrated exhaustive key search attacks
- AES is a new cipher alternative
- prior to this alternative was to use multiple encryption with DES implementations
- Triple-DES is the chosen form

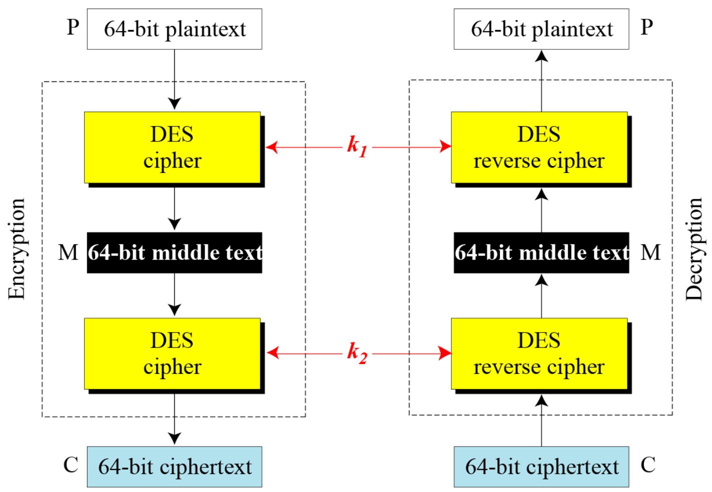
56

Double-DES?

- could use 2 DES encrypts on each block
  - $C = E_{K2}(E_{K1}(P))$
- issue of reduction to single stage
- and have “meet-in-the-middle” attack
  - works whenever use a cipher twice
  - since $X = E_{K1}(P) = D_{K2}(C)$
  - attack by encrypting P with all keys and store
  - then decrypt C with keys and match X value

57

Continued



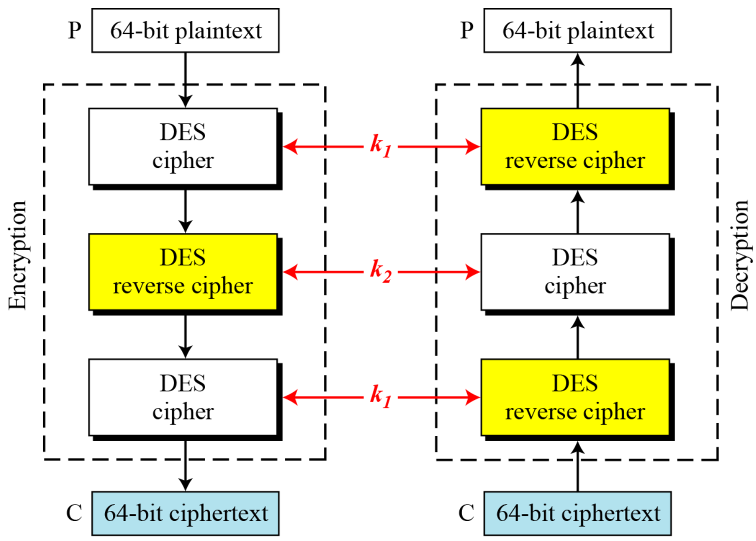
58

Triple-DES with Two-Keys

- hence must use 3 encryptions
  - would seem to need 3 distinct keys
- but can use 2 keys with E-D-E sequence
  - $C = E_{K1}(D_{K2}(E_{K1}(P)))$
  - nb encrypt & decrypt equivalent in security
  - if K1=K2 then can work with single DES
- standardized in ANSI X9.17 & ISO8732
- no current known practical attacks

59

Continued



60

Triple-DES with Three-Keys

- although are no practical attacks on two-key Triple-DES have some indications
- can use Triple-DES with Three-Keys to avoid even these
  - $C = E_{K3}(D_{K2}(E_{K1}(P)))$
- has been adopted by some Internet applications, eg PGP, S/MIME

61

Block Cipher Modes of Operation

Modes of Operation

- block ciphers encrypt fixed size blocks
- eg. DES encrypts 64-bit blocks, with 56-bit key
- need way to use in practise, given usually have arbitrary amount of information to encrypt
- four were defined for DES in ANSI standard **ANSI X3.106-1983 Modes of Use**
- subsequently now have 5modes, intended for any symmetric block cipher
- have **block** and **stream** modes

63

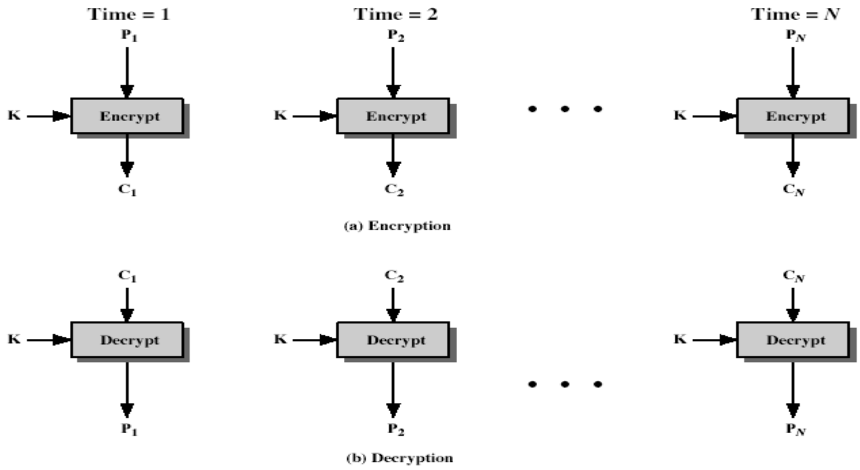
Electronic Codebook Book (ECB)

- message is broken into independent blocks which are encrypted
- each block is a value which is substituted, like a codebook, hence name
- each block is encoded independently of the other blocks

  $C_i = DES_{K1}(P_i)$
- uses: secure transmission of single values

64

Electronic Codebook Book (ECB)



65

Advantages and Limitations of ECB

- repetitions in message may show in ciphertext
  - if aligned with message block
  - particularly with data such graphics
  - or with messages that change very little, which become a code-book analysis problem
- weakness due to encrypted message blocks being independent
- main use is sending a few blocks of data

66

Cipher Block Chaining (CBC)

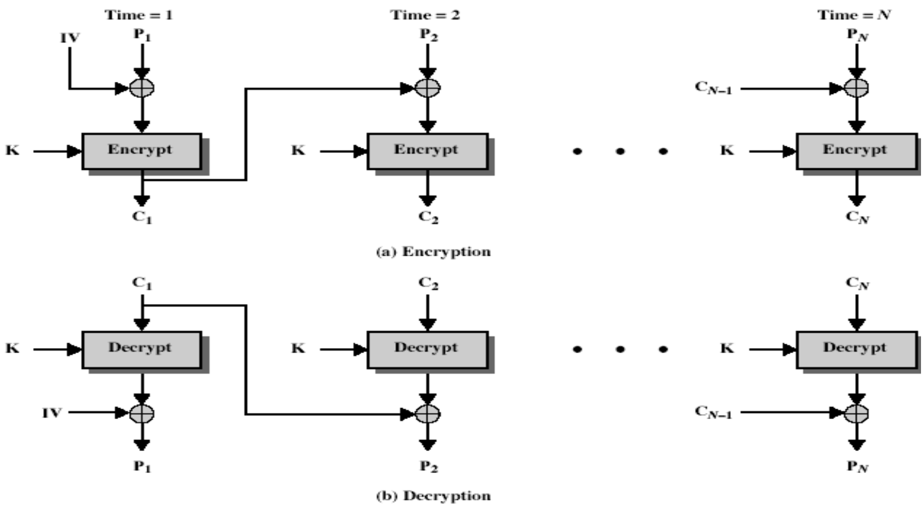
- message is broken into blocks
- but these are linked together in the encryption operation
- each previous cipher blocks is chained with current plaintext block, hence name
- use Initial Vector (IV) to start process

  $C_i = DES_{K1}(P_i \text{ XOR } C_{i-1})$

  $C_{-1} = IV$
- uses: bulk data encryption, authentication

67

Cipher Block Chaining (CBC)



68

Advantages and Limitations of CBC

- each ciphertext block depends on **all** message blocks
- thus a change in the message affects all ciphertext blocks after the change as well as the original block
- need **Initial Value** (IV) known to sender & receiver
  - however if IV is sent in the clear, an attacker can change bits of the first block, and change IV to compensate
  - hence either IV must be a fixed value (as in EFTPOS) or it must be sent encrypted in ECB mode before rest of message
- at end of message, handle possible last short block
  - by padding either with known non-data value (eg nulls)
  - or pad last block with count of pad size
    - eg. [b1 b2 b3 0 0 0 0 5] <- 3 data bytes, then 5 bytes pad+count

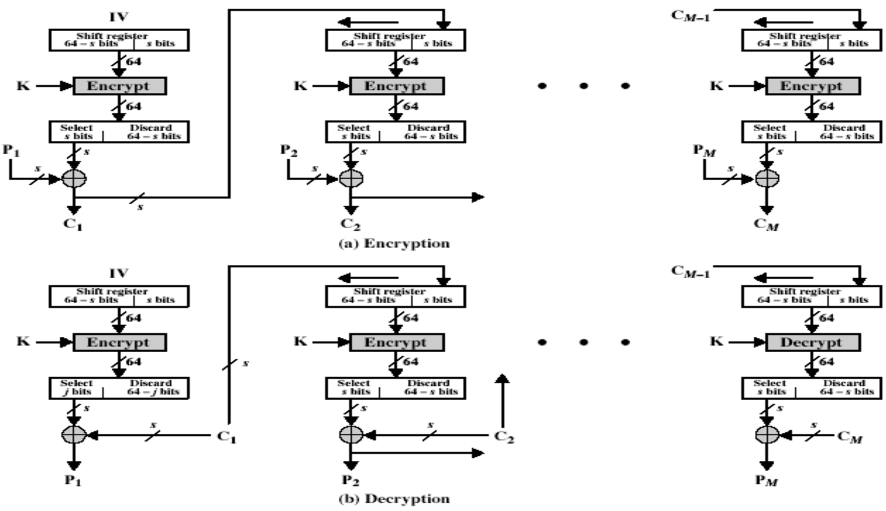
Cipher FeedBack (CFB)

- message is treated as a stream of bits
- added to the output of the block cipher
- result is feed back for next stage (hence name)
- standard allows any number of bit (1,8 or 64 or whatever) to be feed back
  - denoted CFB-1, CFB-8, CFB-64 etc
- is most efficient to use all 64 bits (CFB-64)

  $C_i = P_i \text{ XOR } DES_{K1}(C_{i-1})$

  $C_{-1} = IV$
- uses: stream data encryption, authentication

Cipher FeedBack (CFB)



Advantages and Limitations of CFB

- appropriate when data arrives in bits/bytes
- most common stream mode
- limitation is need to stall while do block encryption after every n-bits
- note that the block cipher is used in **encryption** mode at **both** ends
- errors propagate for several blocks after the error

Output FeedBack (OFB)

- message is treated as a stream of bits
- output of cipher is added to message
- output is then feed back (hence name)
- feedback is independent of message
- can be computed in advance

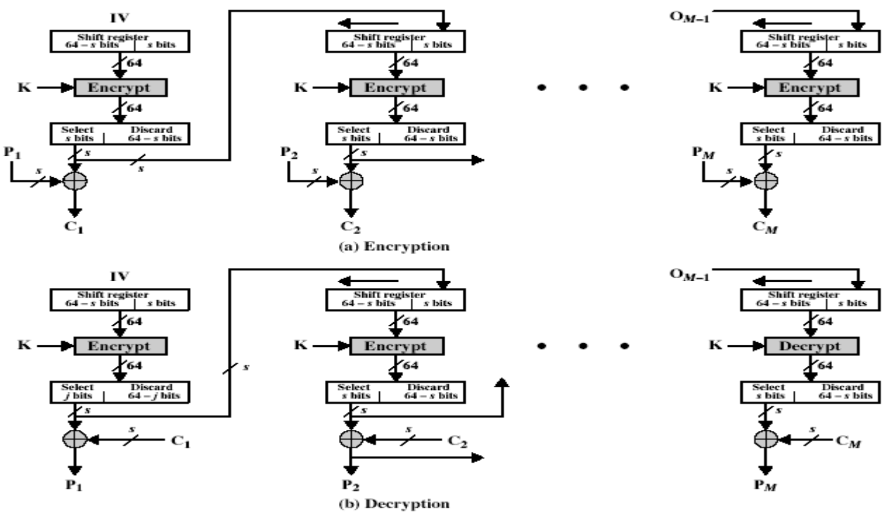
  $C_i = P_i \text{ XOR } O_i$

  $O_i = DES_{K1}(O_{i-1})$

  $O_{-1} = IV$

- uses: stream encryption over noisy channels

Output FeedBack (OFB)



Advantages and Limitations of OFB

- used when error feedback a problem or where need to encryptions before message is available
- superficially similar to CFB
- but feedback is from the output of cipher and is independent of message
- a variation of a Vernam cipher
  - hence must **never** reuse the same sequence (key+IV)
- sender and receiver must remain in sync, and some recovery method is needed to ensure this occurs
- originally specified with m-bit feedback in the standards
- subsequent research has shown that only **OFB-64** should be used

75

Counter (CTR)

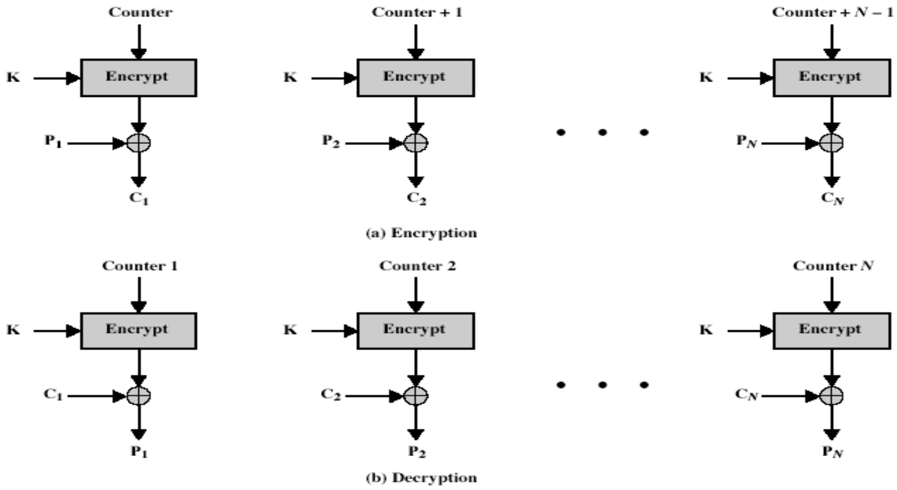
- a "new" mode, though proposed early on
- similar to OFB but encrypts counter value rather than any feedback value
- must have a different key & counter value for every plaintext block (never reused)

  $C_i = P_i \text{ XOR } O_i$

  $O_i = DES_{K1}(i)$
- uses: high-speed network encryptions

76

Counter (CTR)



77

Advantages and Limitations of CTR

- efficiency
  - can do parallel encryptions
  - in advance of need
  - good for bursty high speed links
- random access to encrypted data blocks
- provable security (good as other modes)
- but must ensure never reuse key/counter values, otherwise could break (cf OFB)

78