DES: The Data Encryption Standard (1)

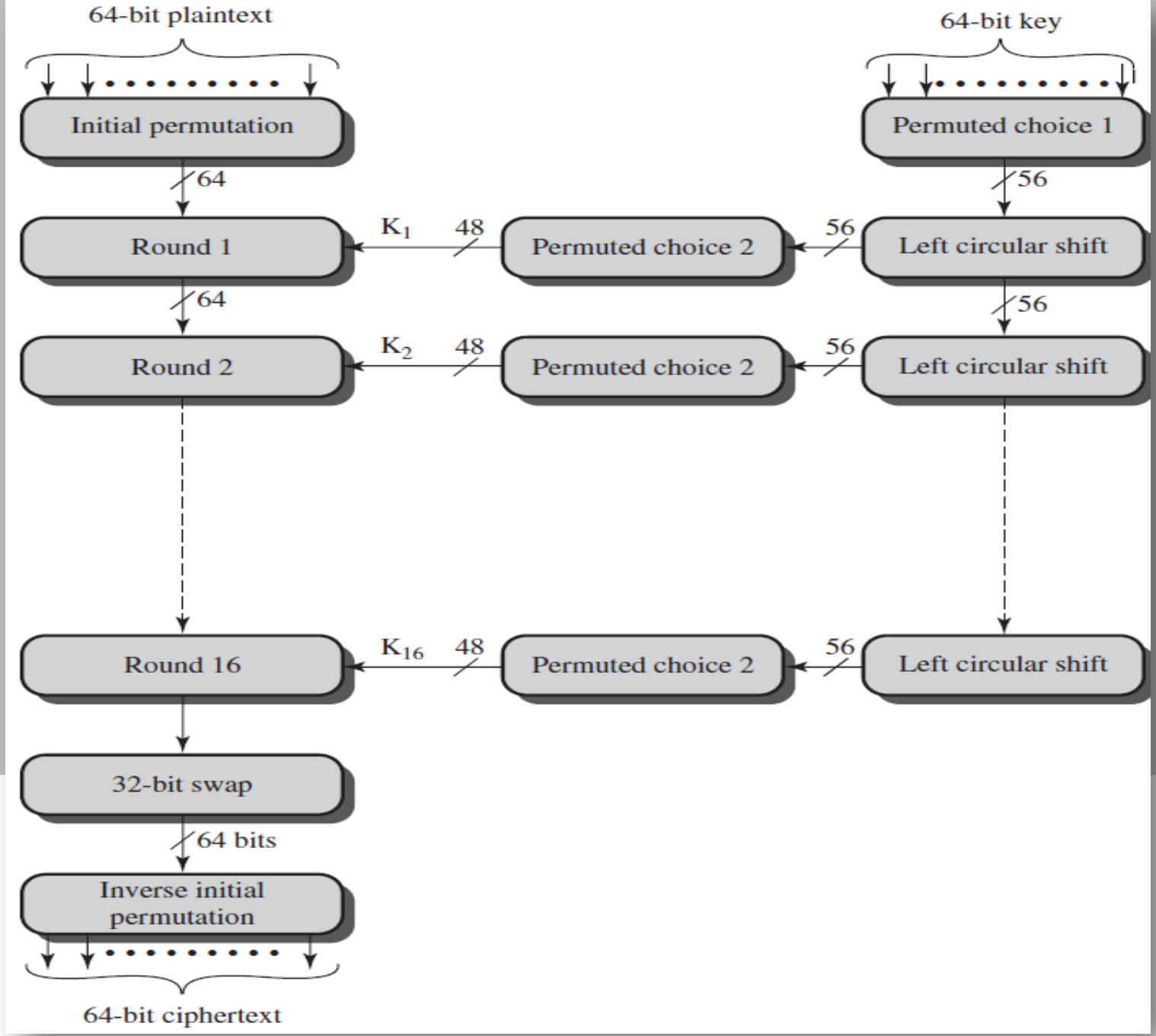
- Adopted by NIST in 1977.
- Based on a cipher (Lucifer) developed earlier by IBM.
- DES uses the Feistel cipher structure with 16 rounds of processing.
- DES uses a 56-bit encryption key. (The key size was apparently dictated by the memory and processing constraints imposed by a single-chip implementation of the algorithm for DES.)
- The key itself is specified with 8 bytes, but one bit of each byte is used as a parity check.

DES (2)

- DES encryption was ***broken*** in 1999 by Electronics Frontiers Organization. This resulted in NIST issuing a new directive that year that required organizations to use ***Triple DES***, that is three consecutive applications of DES.
- That DES was found to be not as strong as originally believed also prompted NIST to initiate the development of new standards for data encryption in 2001. The result is AES (Advanced Encryption Standard).
- Triple DES continues to enjoy wide usage in commercial applications. To understand Triple DES, you must first understand the basic DES encryption.

DES (3)

- As mentioned, DES uses the Feistel structure with 16 rounds.
- What is specific to DES is:
  1. the implementation of ***the F function*** in the algorithm
  2. and how ***the round keys are derived*** from the main encryption key.
- The round keys are generated from the main key by a sequence of permutations. Each round key is 48 bits in length.
- The figure in the next page shows the general description of DES encryption process.


64-bit plaintext
64-bit key
Initial permutation
Permuted choice 1
64
56
Round 1
K1
48
Permuted choice 2
56
Left circular shift
64
56
Round 2
K2
48
Permuted choice 2
56
Left circular shift
Round 16
K16
48
Permuted choice 2
56
Left circular shift
32-bit swap
64 bits
Inverse initial permutation
64-bit ciphertext

DES Decryption

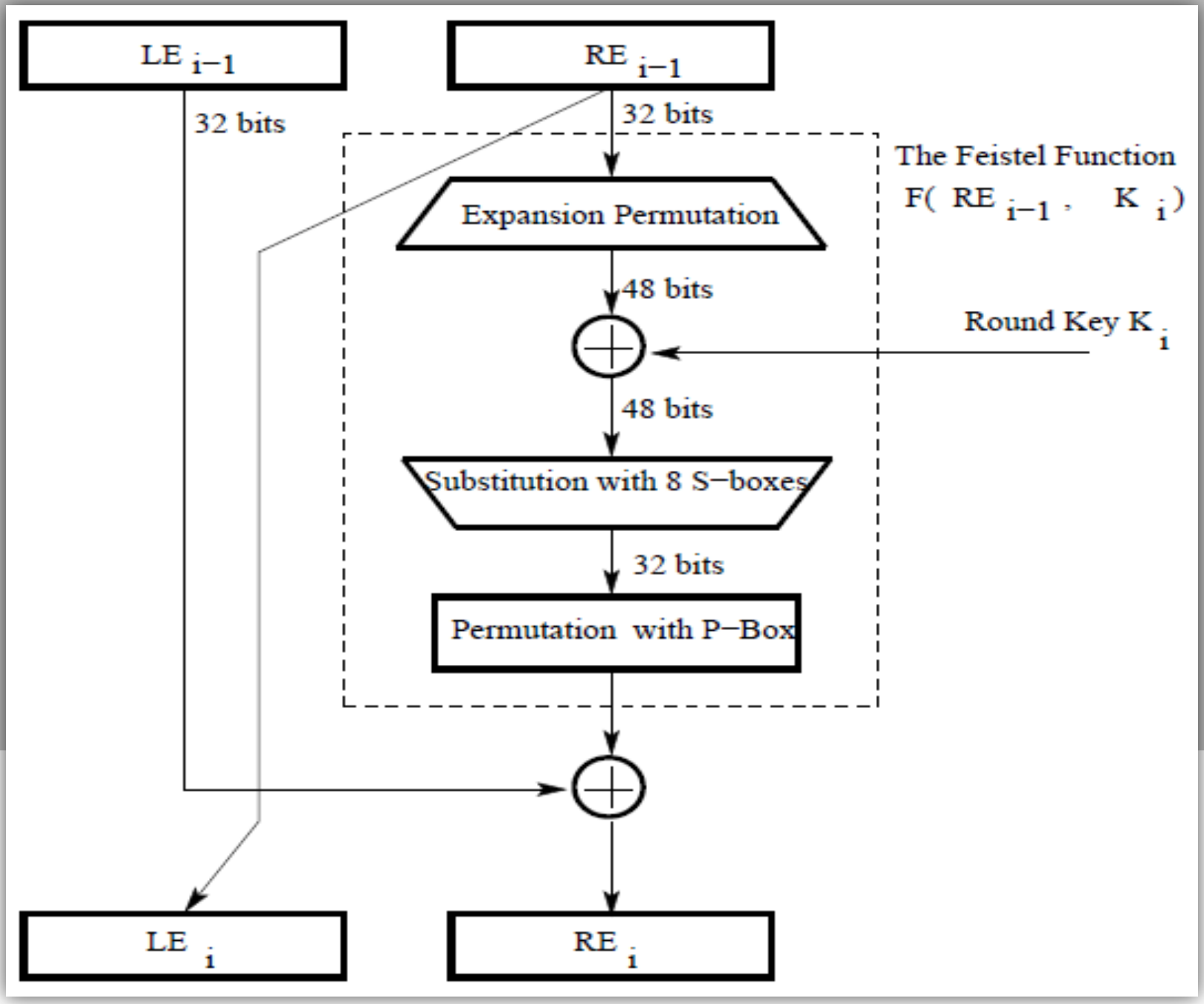
1. As with any Feistel cipher, decryption uses the same algorithm as encryption, except that the application of the subkeys is ***reversed***.

2. Additionally for DES, the initial and final permutations are ***reversed*** (Note that these two permutations can be omitted from DES without security degradation).

One Round of Processing in DES (1)

- The next Figure a single round of processing in DES. The dotted rectangle constitutes the F function.
- The 32-bit right half of the 64-bit input data block is expanded by into a 48-bit block. This is referred to as the ***expansion permutation step*** (E-step).
- The above-mentioned E-step entails the following:

1. first divide the 32-bit block into eight 4-bit words
2. attach an additional bit on the left to each 4-bit word that is the last bit of the previous 4-bit word
3. attach an additional bit to the right of each 4-bit word that is the beginning bit of the next 4-bit word.


LE i–1
RE i–1
32 bits
32 bits
The Feistel Function
F(RE i–1 , K i)
Expansion Permutation
48 bits
Round Key K i
48 bits
Substitution with 8 S–boxes
32 bits
Permutation with P–Box
LE i
RE i

One Round of Processing in DES (2)

- Note that what gets prefixed to the first 4-bit block is the last bit of the last 4-bit block. By the same token, what gets appended to the last 4-bit block is the first bit of the first 4-bit block.
- The 56-bit key is divided into two halves, each half shifted separately, and the combined 56-bit key permuted/contracted to yield a 48-bit round key.
- The 48 bits of the expanded output produced by the E-step are XORed with the round key. This is referred to as ***key mixing***.

One Round of Processing in DES (3)

- The output produced by the previous step is broken into eight six-bit words. Each six-bit word goes through a ***substitution step***; its replacement is a 4-bit word. The substitution is carried out with an **S-box**.
- So after all the substitutions, we again end up with a 32-bit word.
- The 32-bits of the previous step then go through a ***P-box*** based permutation, to be shown later.
- What comes out of the P-box is then XORed with the left half of the 64-bit block that we started out with. The output of this XORing operation gives us the ***right half*** block for the next round.

One Round of Processing in DES (4)

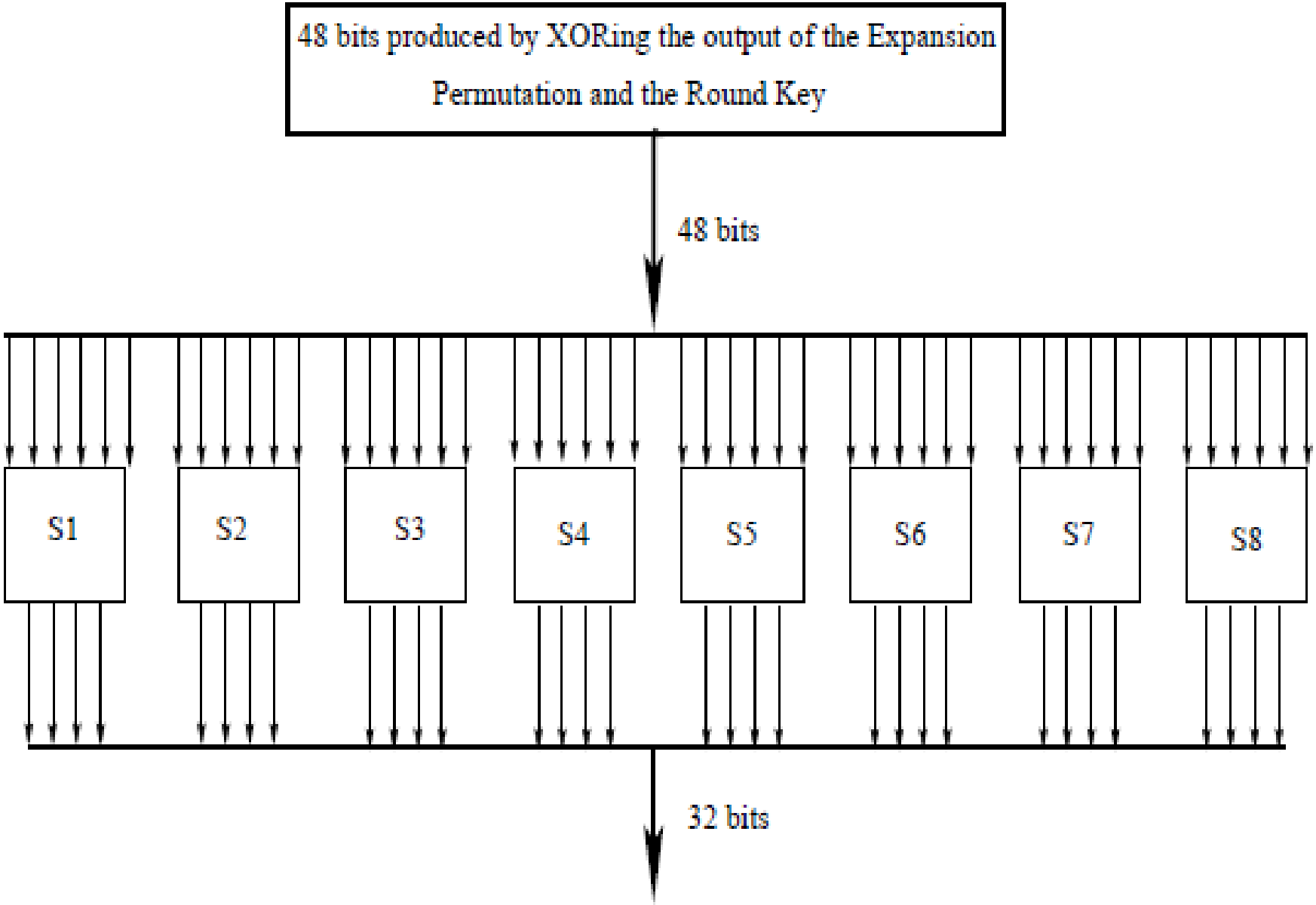
- Note that the goal of the substitution step implemented by the **S-box** is to introduce diffusion in the generation of the output from the input. ***Diffusion means that each plaintext bit must affect as many ciphertext bits as possible***.
- The strategy used for creating the **different round keys** from the main key is meant to introduce confusion into the encryption process. ***Confusion in this context means that the relationship between the encryption key and the ciphertext must be as complex as possible***.
- ***Diffusion*** and ***confusion*** are the two cornerstones of block cipher design.

The S-boxs Step in Each Round (1)

- As shown in the next Figure, the 48-bit input word is divided into eight 6-bit words and each 6-bit word fed into a separate S-box. Each S-box produces a 4-bit output. Therefore, the ***8 S-boxes*** together generate a 32-bit output. As you can see, the overall substitution step takes the 48-bit input back to a 32-bit output.
- Each of the eight S-boxes consists of a 4×16 table lookup for an output 4-bit word. The first and the last bit of the 6-bit input word are decoded into one of ***4 rows*** and the middle 4 bits into one of ***16 columns*** for the table lookup.

The S-boxs Step in Each Round (2)

- The goal of the substitution carried out by an S-box is to ***enhance diffusion***.
- As mentioned previously, the expansion-permutation step (the E-step) expands a 32- bit block into a 48-bit block by attaching a bit at the beginning and a bit at the end of each 4-bit sub-block, the two bits needed for these attachments belong to the adjacent blocks.
- Thus, the row lookup for each of the eight S-boxes becomes a function of the input bits for the previous S-box and the next S-box.


48 bits produced by XORing the output of the Expansion Permutation and the Round Key
48 bits
S1
S2
S3
S4
S5
S6
S7
S8
32 bits

The Substitution Tables

The 4×16 substitution table for S_1															
14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

The 4×16 substitution table for S_2															
15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9

- One can similarly specify tables for the other six substitution boxes (S_3, ... , S_8).

The P-box Permutation in the F Function

P-Box Permutation							
16	7	20	21	29	12	28	17
1	15	23	26	5	18	31	10
2	8	24	14	32	27	3	9
19	13	30	6	22	11	4	25

- As with ***all permutation tables***, this permutation table simply means that the first output bit will be the 16^{th} bit of the input, the second output bit the 7^{th} bit of the input, and so on, for all of the 32 bits of the output that are obtained from the 32 bits of the input. *Note that bit indexing starts with 1 and not with 0.*

Round Key Generation

1. The initial 56-bit key may be represented as 8 bytes, with the last bit of each byte used as ***a parity bit***.
2. The relevant 56 bits are subject to a permutation at the beginning before any round keys are generated. This is our ***Permutation Choice 1.***
3. At the beginning of each round, we divide the 56 relevant key bits into two 28 bit halves and ***circularly shift*** each half by one or two bits.
4. For generating round key, we join together the two halves and apply a 56 bit to 48 bit contracting permutation (***Permutation Choice 2)*** to the joined bit pattern. The resulting 48 bits constitute our round key.
5. The two halves generated in each round ***are fed*** as the two halves going into the next round.

Initial Permutation of the Encryption Key

Permutation Choice 1						
57	49	41	33	25	17	9
1	58	50	42	34	26	18
10	2	59	51	43	35	27
19	11	3	60	52	44	36
63	55	47	39	31	23	15
7	62	54	46	38	30	22
14	6	61	53	45	37	29
21	13	5	28	20	12	4

- Note that the bit positions assume that the key bits are addressed 1 through 64 in an 8-byte bit pattern. But note that the *last bit of each byte is used as a parity bit*. Also note that the permutation shown is not a table, in the sense that the rows and the columns do not carry any special and separate meanings. The permutation order for the bits is given by reading the entries shown *from the upper left corner to the lower right corner*.

Contraction-permutation that Generates the 48-bit Round Key from the 56 Key

Permutation Choice 2							
14	17	11	24	1	5	3	28
15	6	21	10	23	19	12	4
26	8	16	7	27	20	13	2
41	52	31	37	47	55	30	40
51	45	33	48	44	49	39	56
34	53	46	42	50	36	29	32

- Also note that the permutation shown is not a table, in the sense that the rows and the columns do not carry any special and separate meanings. The permutation order for the bits is given by reading the entries shown from the upper left corner to the lower right corner.

Security of DES (1)

- ✓ The substitution step is very effective as far as ***diffusion*** is concerned. It has been shown that if you change just one bit of the 64-bit input data block, on the average that alters 34 bits of the ciphertext block.
- ✓ The manner in which the round keys are generated from the encryption key is also very effective as far as ***confusion*** is concerned. It has been shown that if you change just one bit of the encryption key, on the average that changes 35 bits of the ciphertext.
- ✓ Both effects mentioned above are referred to as the ***avalanche effect***.

Security of DES (2)

- And, of course, the 56-bit encryption key means a ***key space*** of size $2^{56} \approx 7.2 \times 10^{16}$.
- Assuming that, on the average, you'd need to try half the keys in a *brute-force attack*, a machine trying one key per ***microsecond*** would take 1142 years to break the code.
- However, a parallel-processing machine trying ***1 million keys simultaneously*** would need only about 10 hours.

Security of DES (3)

- In the design of the DES, the S-boxes were tuned to enhance the resistance of DES to what is known as the ***differential cryptanalysis attack***. Even a slight modification of the S-boxes can weaken the DES to differential cryptanalysis attack.

- For more details on this issue, the student is requested to refer to the textbook.

Finally . . .

- **Acknowledgment:** These lecture notes are based on the textbook by William Stallings and notes prepared by Avinash Kak, Purdue University. My sincere thanks are devoted to them and to all other people who offered the material on the web.
- Students are advised to study and solve the problems and answer the questions in **Assignment-4**.
- Students are also advised to read the following:

1. The appendix on Simplified DES (S-DES)
2. Tutorial by Howard Heys on differential cryptanalysis