CHAPTER 12

Data Encryption Standard (DES)

Background

- The Data Encryption Standard (DES), also known as Data Encryption Algorithm (DEA) by ANSI and the DEA-1 by ISO, has been a worldwide standard for a twenty years.

Development of the Standard

- In the May 15, 1973 Federal Register, the NBS issued a public request for proposals for a standard cryptographic algorithm. They specified a series of design criteria:
  1. The algorithm must provide a high level of security.
  2. The algorithm must be completely specified and easy to understand.
  3. The security of the algorithm must reside in the key; the security should not depend on the secrecy of the algorithm.
  4. The algorithm must be available to all users.
  5. The algorithm must be adaptable for use in diverse applications.
  6. The algorithm must be economically implementable in electronic devices.
  7. The algorithm must be efficient to use.
  8. The algorithm must be able to be validated.
  9. The algorithm must be exportable.

Description of DES

- DES is a block cipher.

- It encrypts data in 64-bit blocks.
- It is a symmetric algorithm.
- The key length is 56 bits.

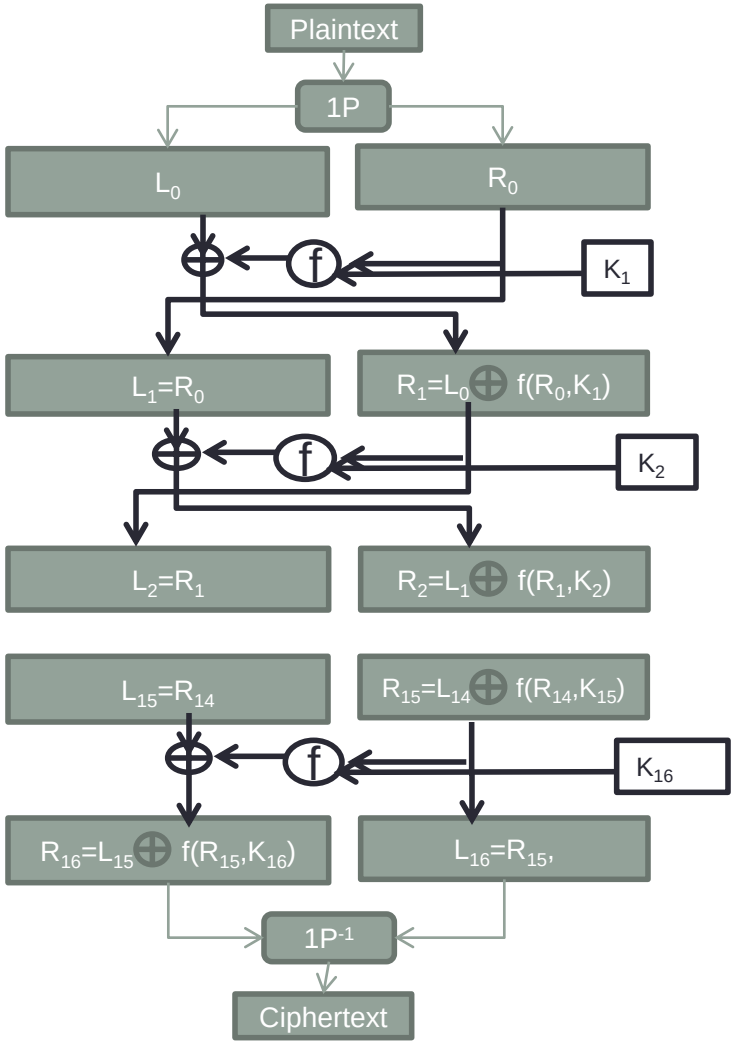
- The key usually expressed as a 64 bit number, but every 8th bit is used for parity checking and is ignored.
- The algorithm is nothing more than a combination of the two basic techniques of encryption: Confusion and Diffusion.

- The fundamental building of DES is a single combination of these techniques (a substitution followed by a permutation) on the test based on the key.

- This is known as a round.
- DES has 16 rounds. (Figure on the next slide).

Figure: DES

- DES:

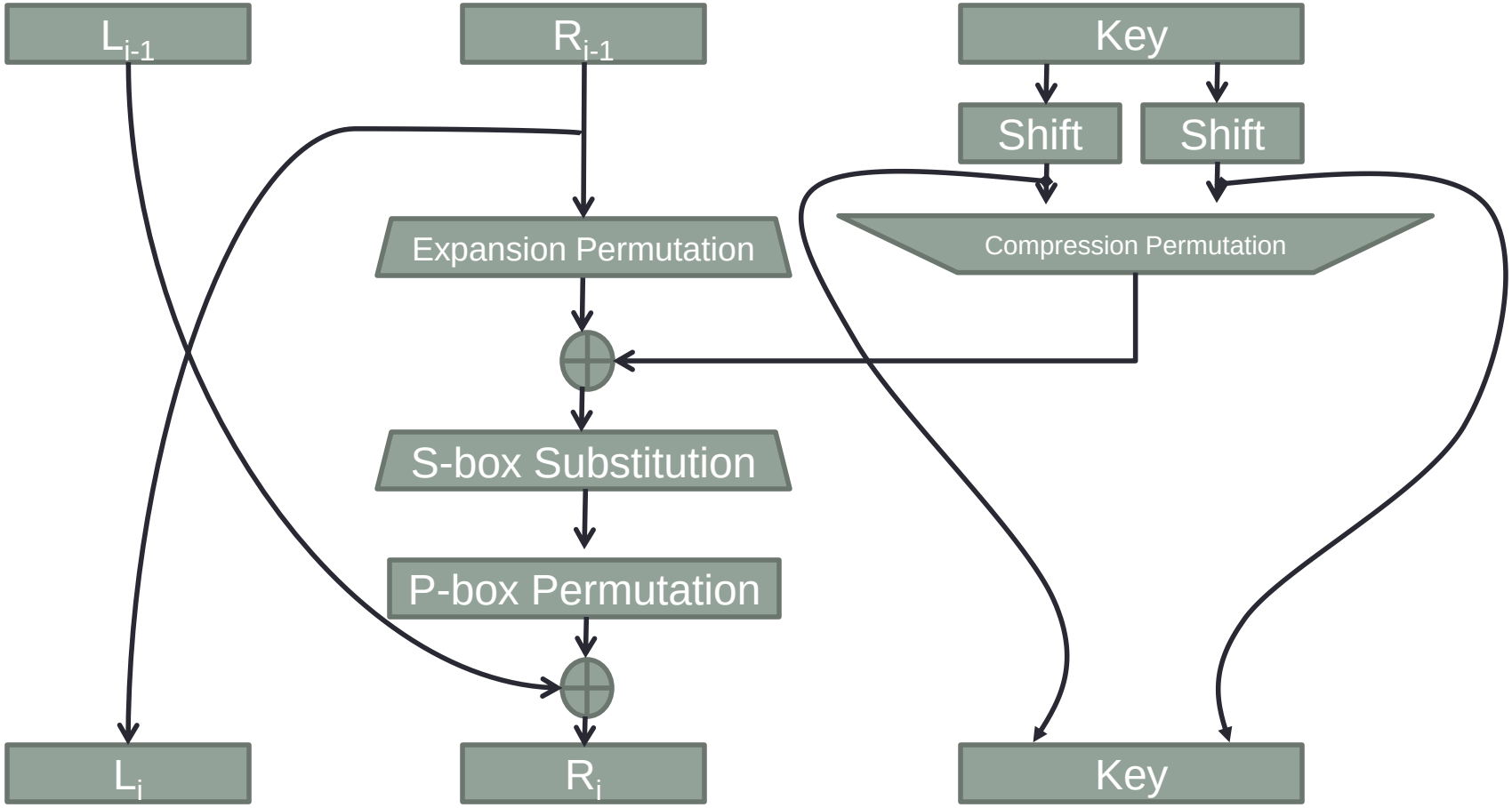


Outline of the Algorithm

- After an initial permutation, the block is broken into a right half and a left half, each 32 bits long.
- Then there are 16 rounds of identical operations, called function f, in which the data are combined with the key.
- After the 16 round the right and left halves are joined, and a final permutation (the reverse of initial permutation) are performed.
- In each round(figure on the next slide), the key bits are shifted and then 48 bits are selected from the 56 bits of the key.
- The right half of the data is expanded 48 bits via an expansion permutation, combined with 48 bits of a shifted and permuted key via an XOR, sent through 8 S-boxes producing 32 new bits, and permuted again.
- These four operations make up the function f.
- The output of function f is then combined with the left half via another XOR.
- The result of these operations becomes the new right half.
- The old right half becomes the new left half.
- These operations are repeated 16 times, making 16 rounds of DES.

Figure: DES one round

- DES one Round:



The Initial Permutation

- This permutation occurs before round 1; it transposes the input block as described in the following table:

58,	50,	42,	34,	26,	18,	10,	2,	60,	52,	44,	36,	28,	20,	12, 4
62,	54,	46,	38,	30,	22,	14,	6,	64,	56,	48,	40,	32,	24,	16, 8
57,	49,	41,	33,	25,	17,	9,	1,	59,	51,	43,	35,	27,	19,	11, 3
61,	53,	45,	37,	29,	21,	13,	5,	63,	55,	47,	39,	31,	23,	15, 7

- This table is read left to right, top to bottom.

- In the table we see, the initial permutation moves bit 58 of the plaintext to bit position 1, bit 50 to position 2, bit 42 to bit position 3 and so forth.

The Key Transformation

- Initially 64 bit DES key is reduced to a 56 bit key by ignoring every eight bit. Table below shows this.

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

- The bits 8, 16, 24… can be used as a parity check to ensure the key is error free.
- After a 56 bit key is extracted a different 48 bit subkey k_i is generated for each of the 16 rounds of DES.

- For this, 56 bit key is divided into two 28 bit halves. Then, the halves are circularly shifted left by either one or two bits, depending on the round (table on next slide shown).

The Key Transformation Continue…

Round	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Number Of bit shifted	1	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1

- After being shifted, 48 out of 56 bits are selected.
- Because this operation permutes the order of the bits as well as selects a subset of bits, it is called a compression permutation.

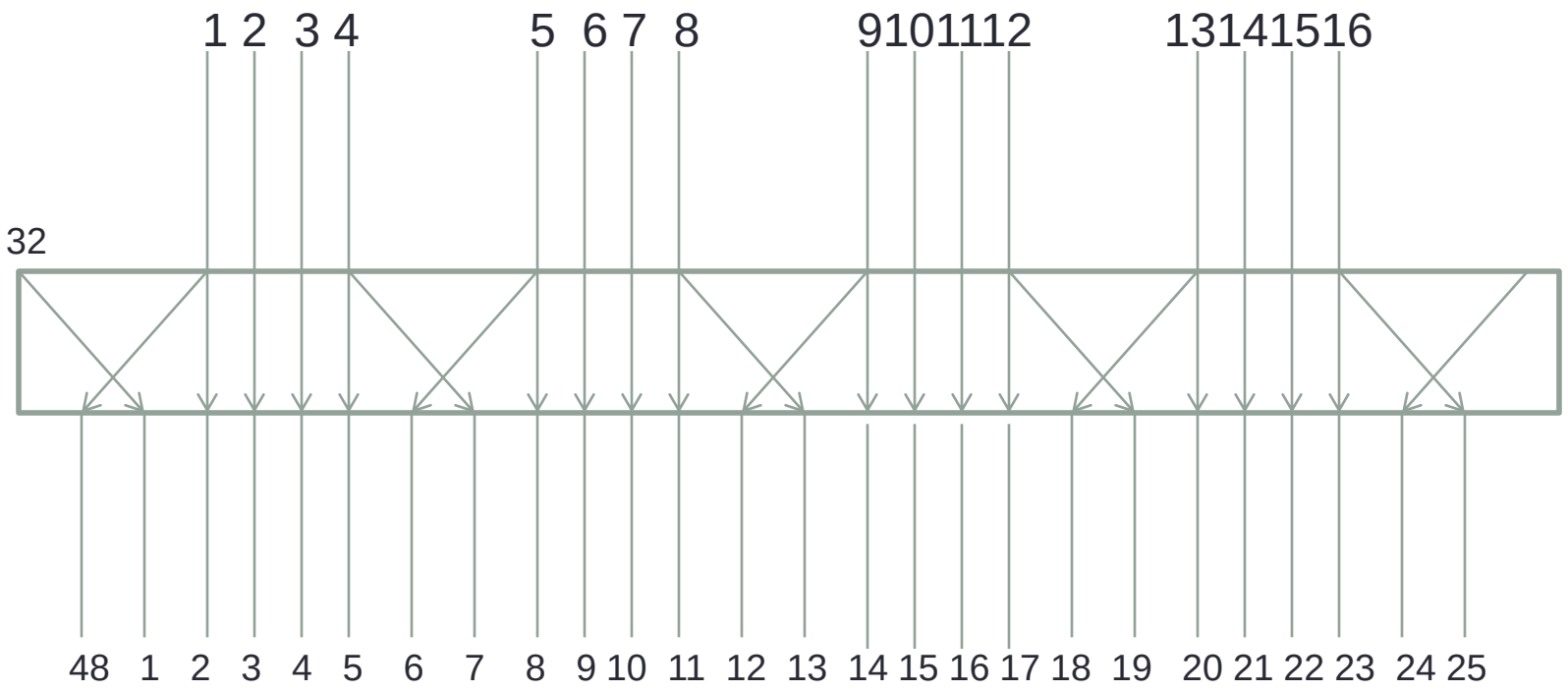
The Expansion Permutation

- The operations expands the right half of the data, R_i, from 32 bits to 48 bits.
- Figure on the next slide show the expansion permutation.
- This is also known as E-box.
- For each 4 bit input block, the first and fourth bits each represent two bit of the output block, while the second and third bits each represent one bit of the output block.
- Table below shows which output positions correspond to which input positions.

Expansion Permutation
32, 1, 2, 3, 4, 5, 4, 5, 6, 7, 8, 9 8, 9, 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, 30, 31, 32, 1

Figure: The Expansion Permutation

- The Expansion Permutation:



The S-box Substitution

- The substitutions are performed by eight substitution boxes, or S-boxes.
- Each S-box has a six bit input and a four bit output.
- The first 6 bit block is operated on by S-box 1, the next six bit block is by S-box 2, and so on (Figure on the next slide).
- Each S-box is a table of 4 rows and 16 columns.
- Each entry in the box is a 4 bit number.
- The 6 input bits of the S-box specify under which row and column number to look for the output (See table on the slide after next).
- Bits b1 and b6 are combined to form a 2 bit number, from 0 to 3 which corresponds to a row in the table.
- The middle 4 bits b2 through b5 are combined to form a 4 bit number, from 0 to 15, which corresponds to a column in the table.
- For example: Assume input to sixth S-box (bit 31 through 36 of the XOR function) is 110011, the first and last bits combine to 11, which corresponds to row 3, and the middle four bits 1001 corresponds to column 9 of the sixth S-box so the entry is 14 (row/column start from 0).
- The 1110 is substituted for 110011.

Figure: S-box Substitution

- S-box Substitution:

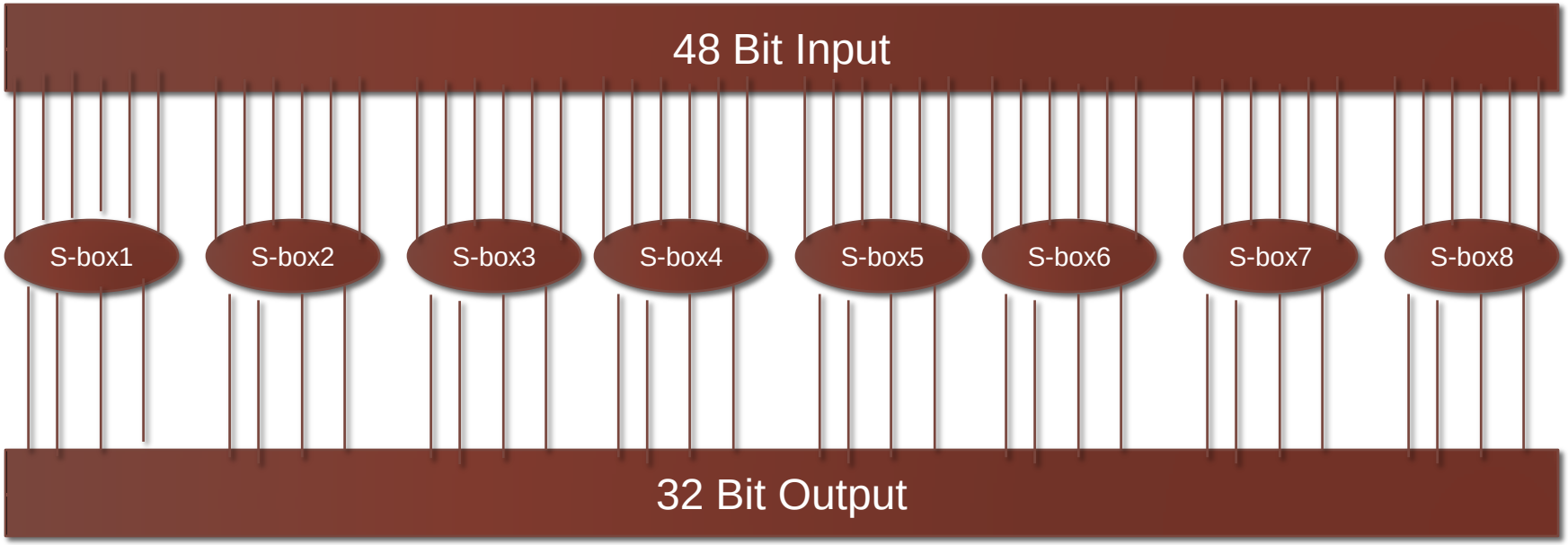


Table: S-Boxes

S-Box 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
S-Box 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
S-Box 3	10 0 9 14 6 3 15 5 1 13 12 7 11 4 2 8 13 7 0 9 3 4 6 10 2 8 5 14 12 11 15 1 13 6 4 9 8 15 3 10 11 1 2 12 5 10 14 7 1 10 13 0 6 9 8 7 4 15 14 3 11 5 2 12
S-Box 4	7 13 14 3 0 6 9 10 1 2 8 5 11 12 4 15 13 8 11 5 6 15 0 3 4 7 2 12 1 10 14 9 10 6 9 0 12 11 7 13 15 1 3 14 5 2 8 4 3 15 0 6 10 1 13 8 9 4 5 11 12 7 2 14
S-Box 5	2 12 4 1 7 10 11 6 8 5 3 15 13 0 14 9 14 11 2 12 4 7 13 1 5 0 15 10 3 9 8 6 4 2 1 11 10 13 7 8 15 9 12 5 6 3 0 14 11 8 12 7 1 14 2 13 6 15 0 9 10 4 5 3
S-Box 6	12 1 10 15 9 2 6 8 0 13 3 4 14 7 5 11 10 15 4 2 7 12 9 5 6 1 13 14 0 11 3 8 9 14 15 5 2 8 12 3 7 0 4 10 1 13 11 6 4 3 2 12 9 5 15 10 11 14 1 7 6 0 8 13
S-Box 7	4 11 2 14 15 0 8 13 3 12 9 7 5 10 6 1 13 0 11 7 4 9 1 10 14 3 5 12 2 15 8 6 9 14 15 5 2 8 12 3 7 0 4 10 1 13 11 6 4 3 2 12 9 5 15 10 11 14 1 7 6 0 8 13
S-Box 8	13 2 8 4 6 15 11 1 10 9 3 14 5 0 12 7 1 15 13 8 10 3 7 4 12 5 6 11 0 14 9 2 7 11 4 1 9 12 14 2 0 6 10 13 15 3 5 8 2 1 14 7 4 10 8 13 15 12 9 0 3 5 6 11

P-Box Permutation

- The 32 bits output of the S-box substitution is permuted according to P-box.
- Table below shows the position to which each bit moves.

- Example: bit 21 moves to bit 4, while bit 4 moves to bit 31.
- The result of P-box permutation is XORed with the left half of the initial 64 bit block.

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

The Final Permutation

- The final permutation is the inverse of the initial permutation as told earlier.
- Table below show this final permutation.
- Also note the left and right halves are not exchanged after the last round of DES; the concatenated block $R_{16}L_{16}$ is used as the input to the final permutation.

Initial Permutation
58, 50, 42, 34, 26, 18, 10, 2, 60, 52, 44, 36, 28, 20, 12, 4
62, 54, 46, 38, 30, 22, 14, 6, 64, 56, 48, 40, 32, 24, 16, 8
57, 49, 41, 33, 25, 17, 9, 1, 59, 51, 43, 35, 27, 19, 11, 3
61, 53, 45, 37, 29, 21, 13, 5, 63, 55, 47, 39, 31, 23, 15, 7

Final Permutation
40, 8, 48, 16, 56, 24, 64, 32, 39, 7, 47, 15, 55, 23, 63, 31
38, 6, 46, 14, 54, 22, 62, 30, 37, 5, 45, 13, 53, 21, 61, 29
36, 4, 44, 12, 52, 20, 60, 28, 35, 3, 43, 11, 51, 19, 59, 27
34, 2, 42, 10, 50, 18, 58, 26, 33, 1, 41, 9, 49, 17, 57, 25

Decryption of DES

- Actually the same algorithm can be used for both encryption and decryption.
- The only difference is that the key must be used in reverse order.

- That is, if the encryption keys for each round are K_1, K_2, K_3,…,K_{16}, then the decryption keys are K_{16}, K_{15}, K_{14},…,K_1.
- The algorithm that generates the key used for each round is circular as well.

- The key shift is a right shift and the number of positions shifted is 0, 1, 2, 2, 2, 2, 2, 2, 1, 2, 2, 2, 2, 2, 2, 1.

DES Variants

Multiple DES:

- Some DES implementation use triple-DES (Figure on the next slide).
- Since DES is not a group, then the resultant ciphertext is much harder to break using exhaustive search.

DES with Independent Subkey:

- Another variation is to use a different subkey for each round, instead of generating them from a 56 bit key.
- Since 48 key bits are used in each of 16 rounds, this means that the key length for this variant is 768 bits.
- This variant would drastically increase the difficulty of a Brute-force attack against the algorithm; that attack would have a complexity of 2^{768}.

Figure: Triple DES

- Triple DES: