Chronological Description of Simplified Data Encryption Standard for Educational Purposes

Divya Mehta, Chailsee Agrawal, *Ashish Jain
Department of Information Technology, School of Information Technology, Manipal University Jaipur
divyamehta182001@gmail.com, chailseeagrawal@gmail.com, ashishjn.research@gmail.com
***Corresponding Author**

Abstract

A popular block cipher, namely, data encryption standard (DES) is difficult to understand, therefore, this chapter addresses simplified data encryption standard (S-DES), a symmetric encryption approach that was particularly created for educational purposes. For a clear grasp of the concept, the chapter also provides a description of the S-DES process with the aid of an example. S-DES is solely explored to gain a better grasp of the DES procedure; it is not thought to be cryptographically secure.

1. Introduction

In the literature, two categories of cryptosystem exist: symmetric key and asymmetric key [1-7]. In symmetric key cryptosystem a shared key is used for both encryption and decryption [8-10]. Symmetric algorithms run more quickly and require less setup [11, 12]. Because the transmitter and receiver share the same key, they are highly safe and useful [11, 12]. The data encryption standard (DES) is a symmetric key technique that encrypts and decrypts data using a 56-bit key [13]. The algorithm uses a 64-bit input of plaintext that is then transformed into ciphertext. The 64-bit input key for DES is eventually reduced by 8 bits [13]. DES is based on Feistel block cipher and has a 16-round structure, with each round requiring the use of a different key of 48 bits generated from 56-bit key [13]. Its 16-round structure and 56-bit key input make it challenging to comprehend. Consequently, a simplified version of DES known as simplified-DES (S-DES) was created by Schaefer [14] to facilitate understanding of original DES.

2. Description of Simplified Data Encryption Standard

The S-DES is a symmetric cryptography algorithm with the same key for encryption and decryption. S-DES is significantly smaller than the DES technique [14]. It is composed of 8-bit plaintext and a 10-bit key, resulting in an 8-bit ciphertext. S-DES consists of the following five primary functions:

a. An initial permutation
b. Round function that depending on the key input, performs both permutation and substitution operations, it is the most complex part of S-DES.
c. A switch that alternates the left and right halves.
d. Round function is used again.
e. An inverse of the original permutation function.

2.1 Key Generation in S-DES

The key generation method of S-DES is shown in Fig. 1. In the key generation [14] procedure, two 8-bit keys are formed from a 10-bit secret key. A symmetric key cipher uses the same key for both encryption and decryption. This key is available to both the sender and the recipient. Now, we illustrate the procedure to generate two 8-bit round keys from a 10-bit secret key in S-DES.

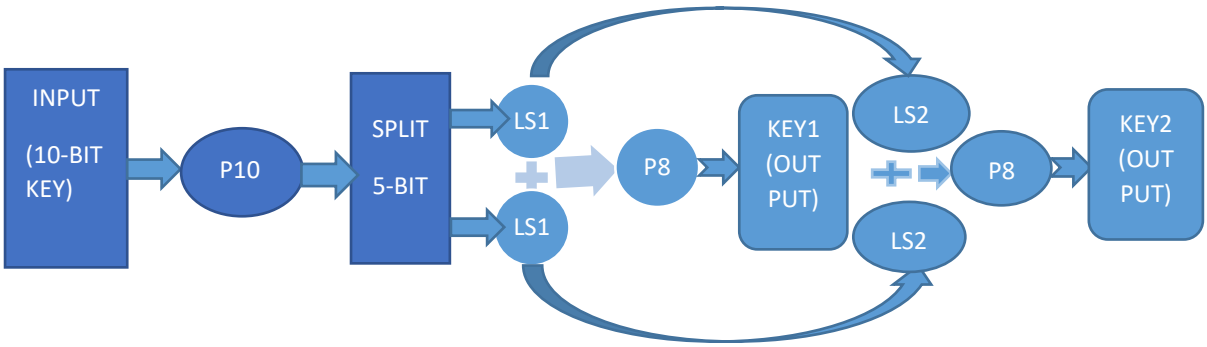

Figure 1: Key Generation in S-DES

(P10: Permutation Box 10, LS: Left Shift, P8: Permutation Box 8)

From Fig. 1, we can understand the following:

KEY1 = P8(LeftShift(P10(Key)))

KEY2 = P8(LeftShift(LeftShift(LeftShift(P10(Key)))))

In Fig. 1, three functions are involved, namely, 10-bit permutation (P10), 8-bit permutation (P8), and shifting of each 5 bits by left (LeftShift: LS1 and LS2, see Fig. 1).

Example:

Let 10-bit secret key is “1010010110”

Step 1: The first step entails entering a 10-bit key into a P10 table, which produces a permuted version of the 10-bit secret key.

Input	1010010110
P10	3527401986 (0 indicates 10th bit) Description: 3rd bit of input becomes 1st bit of output, 5th bit of input becomes 2nd bit of output, and so on.
Output	1000001111

Step 2: Divide the key into left and right halves.

Left	10000
Right	01111

Step 3: Apply left shift by one bit on the left and right halves that have been obtained in step 2, i.e., apply LS1(see Fig 1.), we get the following:

Left	00001
Right	11110

Step 4: Combine both previous step's halves and place them in the P8 table. In this step, the bits are permuted using P8 permutation. Note that in this process the 1st and the 2nd bits are ignored.

Input	0000111110
P8	63748509 (0 indicates 10th bit)

	Description: 6th bit of input becomes 1st bit of output, 3rd bit of input becomes 2nd bit of output, and so on.
Output	10101101

Note: The result that we have obtained is the first key (i.e., KEY1 is "10101101").

Step 5: The output from Step 3 that had a 1-bit left shift will now undergo a 2-bit left shift again. That is apply LS2 (see Fig 1.), we get the following:

Left	00100
Right	11011

Step 6: To obtain the KEY2, the result from Step-5 would be merged and placed back into the P8 table.

Input	0010011011
P8	63748509 (0 indicates 10th bit)
Output	11100011

Note: The result that we have obtained is the second key (i.e., KEY2 is "11100011").

2.2 Encryption in S-DES

The encryption method of S-DES is shown in Fig. 2. In S-DES algorithm two rounds are used for both encryption and decryption [14]. The procedure consists of the following steps:

Example: Let the plaintext of 8 bits is "10010111".

Step 1 (Initial Permutation (IP)): Shuffle the bits of plaintext in accordance with the IP function that has been provided to us.

Input (plaintext)	10010111
IP	26314857 Description: 2nd bit of input becomes 1st bit of output, 6th bit of input becomes 2nd bit of output, and so on.
Output	01011101

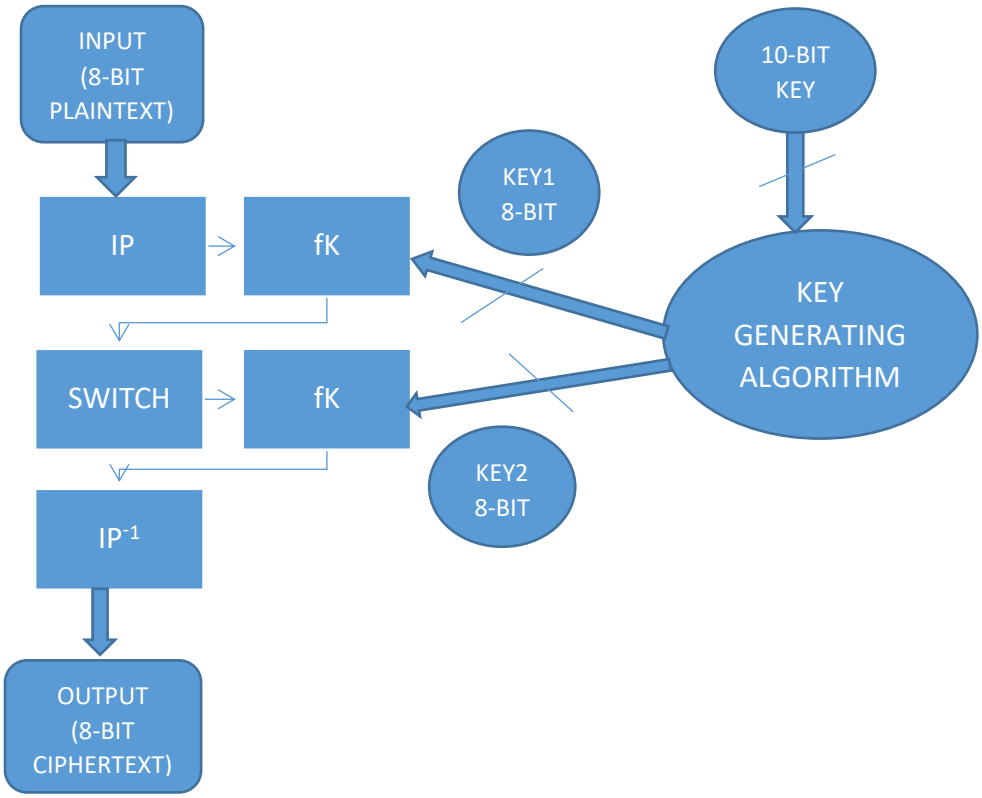

Figure 2: Encryption in S-DES

(IP: Initial Permutation Function, fk: Round Function Key, IP^{-1}: Inverse Permutation Function)

Step 2 (Round Function – Round 1): In this step, we pass the permutated plaintext and the generated round key (KEY1) in the function fk (see Fig. 3). The working of function fk is as follows:

i) Splitting: The output of Step 1 will be split into left and right halves (each having four bits). That is, Left – 0101 and Right – 1101.

ii) Expansion (EP): The right half (four bits) would now be taken and expanded to eight bits in accordance with the EP Function.

Input	1101
EP	41232341
Output	11101011

iii) XOR: Perform XOR operation between the output of Step 2-ii and KEY1.

KEY1	10101101

Output of Step 2-ii	11101011
Output (10101101 ⊕ 11101011)	01000110

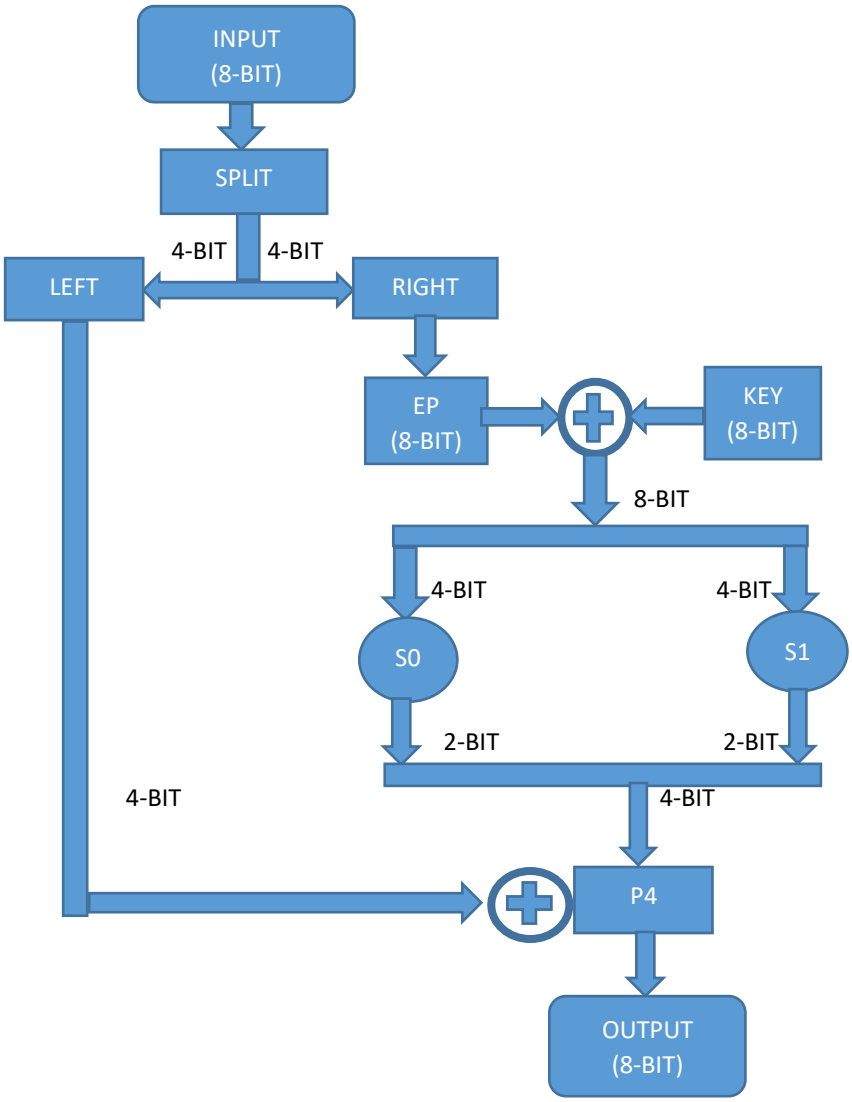

Figure 3: Round Function (fk) in S-DES

(EP: Expansion Box, SO and S1: Substitution Box, P4: Permutation Box-4)

iv) Substitution (S0 and S1): Divide the output obtained in Step 2-iii in two halves (four bits each) and apply S0 to the left half and S1 to the right half. Following Substitution-boxes (S-boxes) have been used for substitution of the bits.

S0 box

Row/Col.	00	01	10	11
00	01	00	11	10
01	11	10	01	00
10	00	10	01	11
11	11	01	11	10

S1 box

Row/Col.	00	01	10	11
00	00	01	10	11
01	10	00	01	11
10	11	00	01	00
11	10	01	00	11

To determine the outcome of S-boxes, the row is represented by the first and fourth bits, while the column is represented by the second and third bits. The output obtained in Step 2-iii is "01000110". The 4-bit left half is 0100, using these 4 bits, an outcome of two bits is obtained using S0, i.e., we will look row "00" and column "10" in S0 that will give "11". The 4-bit right half is 0110, using these 4 bits, an outcome of two bits is obtained using S1, i.e., we will look row "00" and column "11" in S1 that will give "11". After combining the 2-bit binary outputs obtained from left half and right half we get the following 4-bit output: 1111.

v) Permutation (P4): Pass the output obtained in above step (i.e., 1111) to P4 function that will shuffle these 4 bits.

Input	1111
P4	2431
Output	1111

vi) XOR: Perform XOR between above obtained output and output of left half of Step 1.

Left half of output of Step 1	0101
Output of Step 2-v	1111
Output (0101 $\oplus$ 1111)	1010

vii) Combine the above obtained output with the output of right half of Step 1. That is, combine 1010 and 1101, we get "10101101".

Step 3 (Switch): Break the output obtained in Step 2-vii in two parts again and swap the left and right halves, we get "11011010".

Step 4 (Round Function – Round 2): In this step, we pass the output obtained in Step 3 and the generated round key (KEY2) in the function fk (see Fig. 3). The working of function fk is same as we have done in Step 2.

i) Splitting: The output of Step 3 will be split into left and right halves (each having four bits). That is, Left – 1101 and Right – 1010.

ii) Expansion (EP): The right half (four bits) would now be taken and expanded to eight bits in accordance with the EP Function.

Input	1010
EP	41232341
Output	01010101

iii) XOR: Perform XOR operation between the output of Step 4-ii and KEY2.

KEY2	11100011
Output of Step 4-ii	01010101
Output (11100011 $\oplus$ 01010101)	10110110

iv) Substitution (S0 and S1): Divide the output obtained in Step 4-iii in two halves (four bits each) and apply S0 to the left half and S1 to the right half. The output obtained in Step 4-iii is "10110110". The 4-bit left half is 1011, using these 4 bits, an outcome of two bits is obtained using S0, i.e., we will look row "11" and column "01" in S0 that will give "01". The 4-bit right half is 0110, using these 4 bits, an outcome of two bits is obtained using S1, i.e., we will look row "00" and column "11" in S1 that will give "11". After combining the 2-bit binary outputs obtained from left half and right half we get the following 4-bit output: 0111.

v) Permutation (P4): Pass the output obtained in above step (i.e., 0111) to P4 function that will shuffle these 4 bits.

Input	0111
P4	2431
Output	1110

vi) XOR: Perform XOR between above obtained output and output of left half of Step 3.

Left half of output of Step 3	1101
Output of Step 2-v	1110
Output (1101 ⊕ 1110)	0011

vii) Combine the above obtained output with the output of right half of Step 3. That is, combine 0011 and 1010, we get "00111010".

Step 5 (Inverse Permutation (IP^{-1})): The output obtained in Step 4 put into the IP^{-1} table to obtain the ciphertext.

Input	00111010
IP^{-1}	41357286 Description: 4th bit of input becomes 1st bit of output, 1st bit of input becomes 2nd bit of output, and so on.
Output (ciphertext)	10111000

2.3 Decryption in S-DES

The decryption method of S-DES is shown in Fig. 4. This involves the conversion of the ciphertext into the plaintext. The procedure is like encryption method [14].

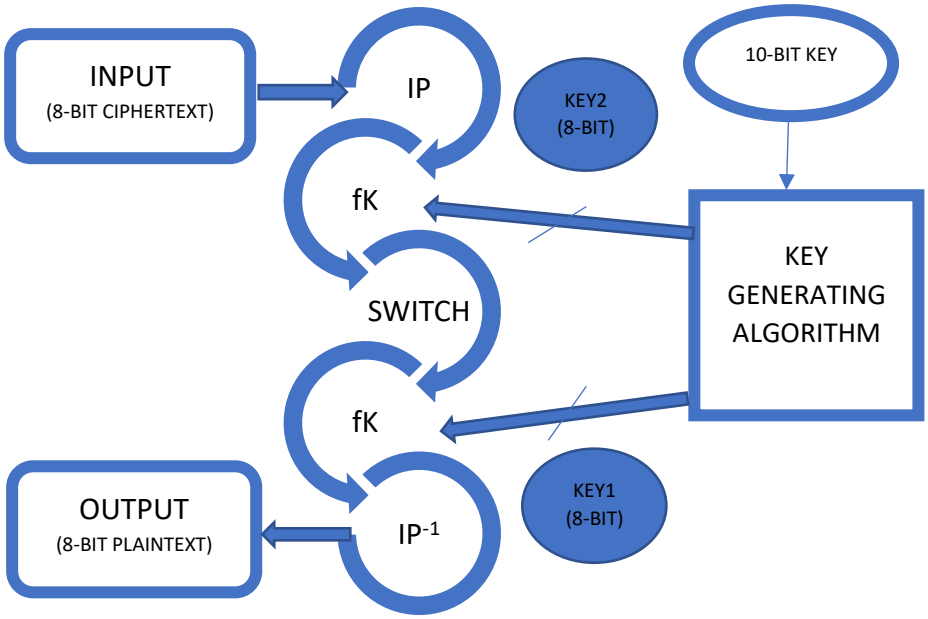

Figure 4: Decryption in S-DES

(IP: Initial Permutation Function, fk: Round Function Key, IP^{-1}: Inverse Permutation Function)

Example: The ciphertext of 8 bits is "10111000". The procedure of converting the ciphertext into the plaintext consists of the following steps:

Step 1 (Initial Permutation (IP)): Shuffle the bits of ciphertext in accordance with the IP function that has been provided to us.

Input (ciphertext)	10111000
IP	26314857
Output	00111010

Step 2 (Round Function – Round 1): In this step, we pass the permutated ciphertext and the second-round key (KEY2) in the function fk (see Fig. 4). The working of function fk is as follows:

i) Splitting: The output of Step 1 will be split into left and right halves (each having four bits). That is, Left – 0011 and Right – 1010.

ii) Expansion (EP): The right half (four bits) would now be taken and expanded to eight bits in accordance with the EP Function.

Input	1010
EP	41232341
Output	01010101

iii) XOR: Perform XOR operation between the output of Step 2-ii and KEY2.

KEY2	11100011
Output of Step 2-ii	01010101
Output (11100011 $\oplus$ 01010101)	10110110

iv) Substitution (S0 and S1): Divide the output obtained in Step 2-iii in two halves (four bits each) and apply S0 to the left half and S1 to the right half. The output obtained in Step 2-iii is "10110110". The 4-bit left half is 1011, using these 4 bits, an outcome of two bits

is obtained using S0, i.e., we will look row "11" and column "01" in S0 that will give "01". The 4-bit right half is 0110, using these 4 bits, an outcome of two bits is obtained using S1, i.e., we will look row "00" and column "11" in S1 that will give "11". After combining the 2-bit binary outputs obtained from left half and right half we get the following 4-bit output: 0111.

v) Permutation (P4): Pass the output obtained in above step (i.e., 0111) to P4 function that will shuffle these 4 bits.

Input	0111
P4	2431
Output	1110

vi) XOR: Perform XOR between above obtained output and output of left half of Step 1.

Left half of output of Step 1	0011
Output of Step 2-v	1110
Output (0011 $\oplus$ 1110)	1101

vii) Combine the above obtained output with the output of right half of Step 1. That is, combine 1101 and 1010, we get "11011010".

Step 3 (Switch): Break the output obtained in Step 2-vii in two parts again and swap the left and right halves, we get "10101101".

Step 4 (Round Function – Round 2): In this step, we pass the output obtained in Step 3 and the first-round key (KEY1) in the function fk (see Fig. 3). The working of function fk is same as we have done in Step 2.

i) Splitting: The output of Step 3 will be split into left and right halves (each having four bits). That is, Left – 1010 and Right – 1101.

ii) Expansion (EP): The right half (four bits) would now be taken and expanded to eight bits in accordance with the EP Function.

Input	1101
EP	41232341

Output	11101011

iii) XOR: Perform XOR operation between the output of Step 4-ii and KEY1.

KEY1	10101101
Output of Step 4-ii	11101011
Output (10101101⊕ 11101011)	01000110

iv) Substitution (S0 and S1): Divide the output obtained in Step 4-iii in two halves (four bits each) and apply S0 to the left half and S1 to the right half. The output obtained in Step 4-iii is "01000110". The 4-bit left half is 0100, using these 4 bits, an outcome of two bits is obtained using S0, i.e., we will look row "00" and column "10" in S0 that will give "11". The 4-bit right half is 0110, using these 4 bits, an outcome of two bits is obtained using S1, i.e., we will look row "00" and column "11" in S1 that will give "11". After combining the 2-bit binary outputs obtained from left half and right half we get the following 4-bit output: 1111.

v) Permutation (P4): Pass the output obtained in above step (i.e., 1111) to P4 function that will shuffle these 4 bits.

Input	1111
P4	2431
Output	1111

vi) XOR: Perform XOR between above obtained output and output of left half of Step 3.

Left half of output of Step 3	1010
Output of Step 2-v	1111
Output (1010 ⊕ 1111)	0101

vii) Combine the above obtained output with the output of right half of Step 3. That is, combine 0101 and 1101, we get "01011101".

Step 5 (Inverse Permutation (IP^{-1})): The output obtained in Step 4 put into the IP^{-1} table to obtain the plaintext.

Input	01011101
IP^{-1}	41357286
Output (plaintext)	10010111

Summary

The chapter begins by providing an overview of symmetric cryptography and DES before moving on to S-DES and providing a detailed explanation of it with an example.S-DES generates 8-bit cipher-text using a 10-bit key and 8-bit plaintext. Two 8-bit keys are generated from the 10-bit secret key. Further these keys and 5 functions are used to obtain the cipher-text.

References

1. Martin K. Cryptography: the key to digital security, how it works, and why it matters. WW Norton Company, 2020.

2. Stinson DR, Paterson M. Cryptography: theory and practice. CRC Press, 2018.

3. Dooley JF. History of cryptography and cryptanalysis: codes, ciphers, and their algorithms. Springer, 2018.

4. Bishop M. Introduction to computer security. Pearson Education India, 2005.

5. Rubinstein-Salzedo S. Cryptography. Springer, 2018

6. Pachghare VK. Cryptography and information security. PHI Learning Pvt. Ltd., 2019.

7. Stallings W. Cryptography and network security, 4/E. Pearson Education India, 2006.

8. Paar C, Pelzl J. Understanding cryptography: a textbook for students and practitioners. Springer Science & Business Media, 2009.

9. Delfs H, Knebl H. Introduction to cryptography principles and applications, 2007.

10. Bagad IDV. Cryptography and network security. Technical Publications, 2008.

11. Welsh D. Codes and cryptography. Oxford University Press, 1988.

12. Mitani M, Sato S, Hinoki I. The manga guide to cryptography. No Starch Press, 2018.

13. Pub FIPS. Data encryption standard (DES). Federal Information Processing Standards Publication, 1999, 1-24.

14. Schaefer EF. A simplified data encryption standard algorithm. Cryptologia, 1996, 77-84.