

Ministerul Educației și Cercetării al Republicii Moldova
Universitatea Tehnică a Moldovei
Facultatea Calculatoare, Informatică și Microelectronică



Laboratory work 4
Subject: Block ciphers. The DES algorithm

Done by:

Gitlan Gabriel
st. gr. FAF-213

Verified by:

Mîțu Cătălin
asist. univ.

Chișinău - 2023

Description:

Data Encryption Standard (DES) is a block cipher with a 56-bit key length that has played a significant role in data security. Data encryption standard (DES) has been found vulnerable to very powerful attacks therefore, the popularity of DES has been found slightly on the decline. DES is a block cipher and encrypts data in blocks of size of 64 bits each, which means 64 bits of plain text go as the input to DES, which produces 64 bits of ciphertext. The same algorithm and key are used for encryption and decryption, with minor differences. The key length is 56 bits.

Task:

To develop a program in one of the preferred programming languages to implement an element of the DES algorithm. The task will be chosen according to the order number n of the student from the group list, according to the formula: $\text{no_task} = n \bmod 11$. For each task, the tables used and all intermediate steps should be displayed on the screen. Input data may be user-entered or randomly generated. Careful! When supporting the paper, questions will be asked about the work of the entire algorithm!!!

Variant 18 (2.7)

```
int32_t sbox[8][4][16] = {
    {{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}},
    {{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}},
    {{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, 0, 11, 1, 2, 12, 5, 10, 14, 7},
     {1, 10, 13, 0, 6, 9, 8, 7, 4, 15, 14, 3, 11, 5, 2, 12}},
    {{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}},
    {{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}},
    {{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}},
    {{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},
     {1, 4, 11, 13, 12, 3, 7, 14, 10, 15, 6, 8, 0, 5, 9, 2},
     {6, 11, 13, 8, 1, 4, 10, 7, 9, 5, 0, 15, 14, 2, 3, 12}},
    {{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}}};
```

Figure 1: S-box variable

The sbox array represents an 8x4x16 S-box. It's a key component of the substitution cipher, where each 4-bit input is substituted with a corresponding 4-bit output according to this S-box.

```
int32_t substitute(int32_t byte, int32_t index) {  
    int32_t row = ((byte & 0x20) >> 4) | (byte & 0x01);  
    int32_t col = (byte & 0x1E) >> 1;  
    return sbox[index][row][col];  
}
```

Figure 2: Substitution function

The substitute function takes a 4-bit byte and an index (representing which S-box to use) and performs a substitution based on the S-box. It calculates the row and column in the S-box and returns the corresponding substitution.

```
int32_t row = ((byte & 0x20) >> 4) | (byte & 0x01);
```

- `byte & 0x20`: This bitwise AND operation extracts the sixth bit (counting from the right, starting at 0) from the byte.
- `>> 4`: This right-shifts the result by 4 positions, effectively moving the extracted bit to the rightmost position.
- `byte & 0x01`: This bitwise AND operation extracts the least significant bit (LSB) from the byte.
- The two results are then combined using bitwise OR (`|`). This operation effectively puts the extracted bit from the sixth position in the leftmost position and retains the LSB in the rightmost position. This forms the 'row' value.

```
int32_t col = (byte & 0x1E) >> 1;
```

- `byte & 0x1E`: This bitwise AND operation extracts bits 1 through 4 from the byte
- `>> 1`: This right-shifts the result by 1 position, effectively moving the extracted bits one position to the right. This forms the 'col' value.

```

int main() {
    std::string input = "B1B2B3B4B5B6B7B8";

    assert(input.size() == 16); // Input must be exactly 16 chars lenght

    std::string output;
    for (uint8_t i = 0; i < 8; i++) {
        int32_t byte = stoi(input.substr(2 * i, 2), nullptr, 16); // to_hex

        output += std::to_string(substitute(byte, i));
    }

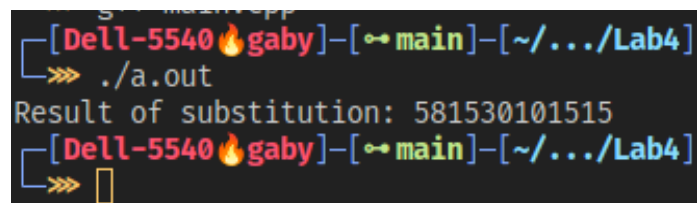
    std::cout << "Result of substitution: " << output << std::endl;

    return 0;
}

```

Figure 3: main function

- The main function initializes a string input with a hexadecimal value.
- It asserts that the input size is exactly 16 characters.
- It then processes the input string in 2-character chunks, converts them to integers, and applies the substitution using the substitute function.
- The results are concatenated into the output string.
- Finally, the result of the substitution is printed to the console.



```

[Dell-5540🔥gaby]-[↔main]-[~/.../Lab4]
>>> ./a.out
Result of substitution: 581530101515
[Dell-5540🔥gaby]-[↔main]-[~/.../Lab4]
>>> 

```

Figure 4: Output

Conclusion:

This code snippet implements a basic substitution cipher using a predefined S-box. In cryptography, substitution ciphers involve replacing elements of plaintext with other elements, often based on a fixed set of rules. While this code focuses on encryption, the principles underlying substitution ciphers are foundational to understanding cryptographic algorithms. Decoding such ciphers typically involves analyzing patterns, exploiting weaknesses, or utilizing statistical methods to uncover the original information. Advanced encryption algorithms like AES are preferred in real-world applications due to their robustness and resistance to cryptanalysis. Link to repo: https://github.com/5anji/CS_Labs