

Lab 1: Implementation of Data Encryption Standard (DES)

Network Security Course

Hoang Le

The University Aizu, Japan

September 19, 2025



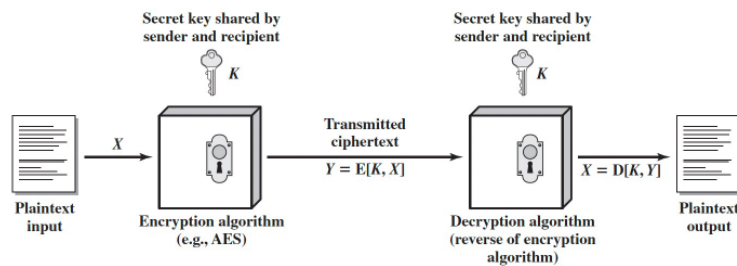
Outline

- Review of Data Encryption Standard (DES)
- Lab 1: Overview and Requirements

September 19, 2025

2

Symmetric Encryption

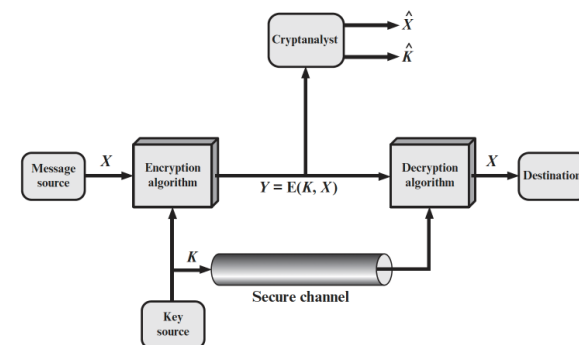


- Mathematically we have: $Y = E(K, X)$ and $X = D(K, Y)$
 - $E(K, X)$ = encryption of X using the key K
 - $D(K, Y)$ = decryption of Y using the key K

September 19, 2025

3

Symmetric Encryption



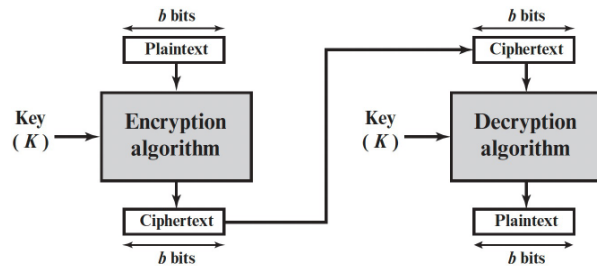
- An opponent, observing Y but not having access to K or X , may attempt to recover X or K or both X and K

September 19, 2025

4

Block Cipher

- A 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.



September 19, 2025

5

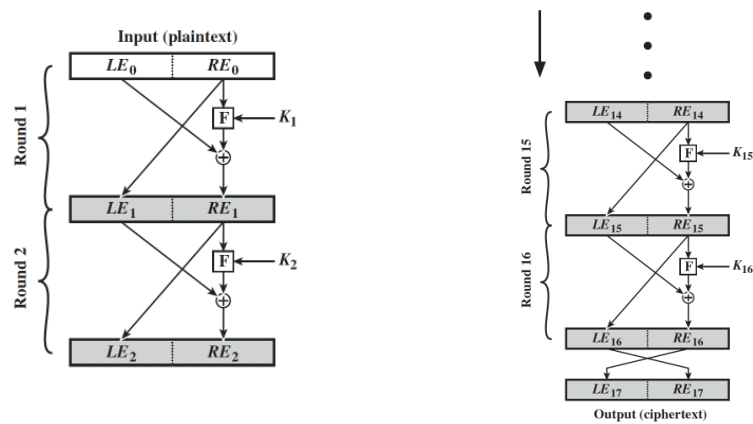
Feistel Cipher Structure

- Horst Feistel (of IBM in 1973) proposed the use of a cipher that alternates substitutions and permutations
 - Substitution:** Each plaintext element or group of elements is uniquely replaced by a corresponding ciphertext element or group of elements.
 - Permutation:** A sequence of plaintext elements is replaced by a permutation of that sequence (no elements are added or deleted or replaced in the sequence, rather the order in which the elements appear in the sequence is changed)
- Many symmetric block encryption algorithms, including DES, have a structure following this principle

September 19, 2025

6

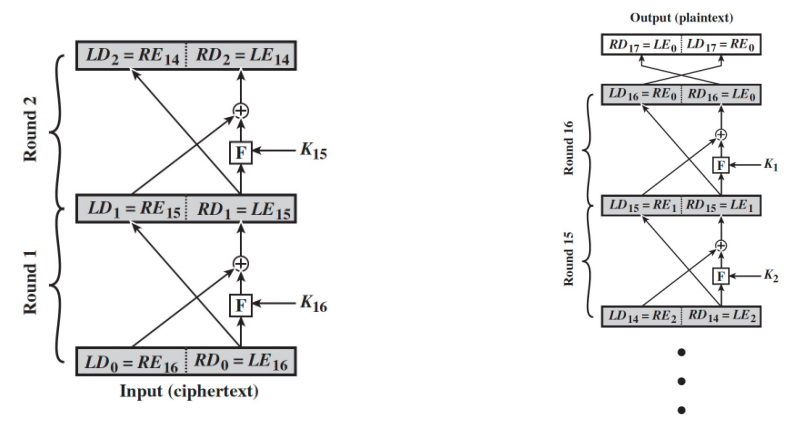
Feistel Cipher Structure: Encryption



September 19, 2025

7

Feistel Cipher Structure: Decryption



September 19, 2025

8

Data Encryption Standard (DES)

- Until the introduction of the Advanced Encryption Standard (AES) in 2001, *the Data Encryption Standard (DES) was the most widely used encryption scheme*
- DES was issued in 1977 by the National Bureau of Standards, now the National Institute of Standards and Technology (NIST)
 - Plaintext: 64 bits
 - Key size: 56 bits
 - Number of rounds: 16

General Depiction of DES Encryption Algorithm

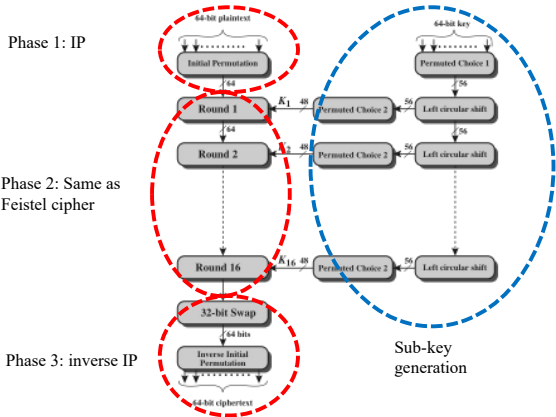


Figure 4.5 General Depiction of DES Encryption Algorithm

An Example of DES (1)

- For this example, the plaintext is a hexadecimal palindrome. The plaintext, key, and resulting ciphertext are as follows:

Plaintext:	02468aceeca86420
Key:	0f1571c947d9e859
Ciphertext:	da02ce3a89ecac3b

An Example of DES (2)

Round	K_i	L_i	R_i
IP		5a005a00	3cf03c0f
1	1e030f03080d2930	3cf03c0f	bad22845
2	0a31293432242318	bad22845	99e9b723
3	23072318201d0c1d	99e9b723	0bae3b9e
4	05261d3824311a20	0bae3b9e	42415649
5	3325340136002c25	42415649	18b3fa41
6	123a2d0d04262a1c	18b3fa41	9616fe23
7	021f120b1c130611	9616fe23	67117cf2
8	1c10372a2832002b	67117cf2	c11bfc09
9	04292a380c341f03	c11bfc09	887fbc6c
10	2703212607280403	887fbc6c	600f7e8b
11	2826390c31261504	600f7e8b	f596506e
12	12071c241a0a0f08	f596506e	738538b8
13	300935393c0d100b	738538b8	c6a62c4e
14	311e09231321182a	c6a62c4e	56b0bd75
15	283d3e0227072528	56b0bd75	75e8fd8f
16	2921080b13143025	75e8fd8f	25896490
IP ⁻¹		da02ce3a	89ecac3b

Outline

- Review of Data Encryption Standard (DES)
- Lab I: Overview and Requirements

September 19, 2025

13

Overview of Lab I

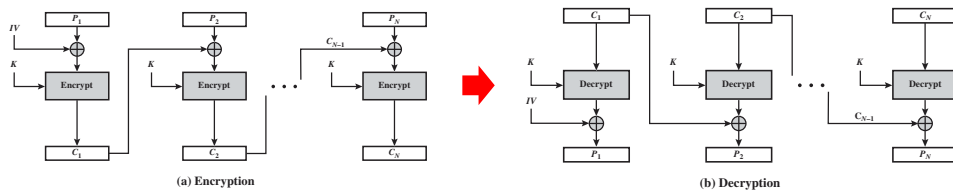
- In this lab, we will implement the Data Encryption Standard (DES)
- The lab has two main problems, and you should follow the order to solve these problems:
 - Problem 1: Implementation of DES encryption/decryption
 - Problem 2: Benchmark DES

September 19, 2025

14

DES in Cipher Block Chaining Mode

- In the first part, we will reimplement the Data Encryption Standard (DES) algorithm in Cipher Block Chaining (CBC) mode
- Review of CBC mode:** The input to the encryption algorithm is the XOR of the next block of plaintext and the preceding block of ciphertext
 - The message is broken into blocks
 - Linked together in the encryption operation
 - Each previous cipher block is chained with the current plaintext block



September 19, 2025

15

Provided Materials

For this part, we provides three files as follows:

- tempdes_simple.c:** An example of the implementation of DES encryption and decryption on a fixed 64-bit (8 bytes) block
- tempdes_cbc.c:** The skeleton code, which is the main file that you will work on
- test.txt:** The 'plain text' that you will use later to verify your code

September 19, 2025

16

OpenSSL

- **A very important note:** In this lab, we will use a lot of structures and functions from the library OpenSSL => Make sure that OpenSSL is already on your computer!
 - The OpenSSL project was founded in 1998 to provide a free set of encryption tools for Internet protocols (including SSL and TLS)
- To check the SSL version, run the following command

```
$ openssl version
```
- To install OpenSSL
 - **Windows:** Please check this link for the installations <https://wiki.nhanhoa.com/kb/huong-dan-cai-dat-openssl-tren-windows-cuc-de-ai-cung-lam-duoc/>
 - **MacOS:** `$ brew install openssl`
 - **Ubuntu:** `$ sudo apt-get install libssl-dev`

September 19, 2025

17

Problem 0: Run the example code

- Let's first run the example code: `tempdes_simple.c`
- Run the following command:
 - For Windows/Ubuntu:

```
$ gcc -o tempdes_simple tempdes_simple.c -lssl -lcrypto
$ ./tempdes_simple
```
 - For MacOS:

```
$ gcc -o tempdes_simple tempdes_simple.c -lssl -lcrypto -L
/opt/homebrew/opt/openssl@3/lib -I /opt/homebrew/opt/openssl@3/include
$ ./tempdes_simple
```

September 19, 2025

18

Problem 0 (cont.)

```
gcc -o tempdes_simple tempdes_simple.c -lssl -lcrypto
tempdes_simple.c: In function 'main':
tempdes_simple.c:34:19: warning: 'DES_set_key_checked' is deprecated: Since OpenSSL 3.0 [-Wdeprecated-declarations]
34 |     if ((k = DES_set_key_checked(&des_key, &key)) != 0) //Generate the actual key from des_key for encryption
    |                   ^
In file included from tempdes_simple.c:1:
/usr/include/openssl/des.h:198:15: note: declared here
198 | int DES_set_key_checked(const_DES_cblock *key, DES_key_schedule *schedule);
    | ^~~~~~
tempdes_simple.c:42:19: warning: 'DES_encrypt1' is deprecated: Since OpenSSL 3.0 [-Wdeprecated-declarations]
42 |     DES_encrypt1(in, &key, ENC);
    |     ^~~~~~
In file included from tempdes_simple.c:1:
/usr/include/openssl/des.h:121:16: note: declared here
121 | void DES_encrypt1(DES_LONG *data, DES_key_schedule *ks, int enc);
    | ^~~~~~
tempdes_simple.c:48:19: warning: 'DES_encrypt1' is deprecated: Since OpenSSL 3.0 [-Wdeprecated-declarations]
48 |     DES_encrypt1(in, &key, DEC); // DES decryption
    |     ^~~~~~
In file included from tempdes_simple.c:1:
/usr/include/openssl/des.h:121:16: note: declared here
121 | void DES_encrypt1(DES_LONG *data, DES_key_schedule *ks, int enc);
    | ^~~~~~
DES Plaintext: 12345678
DES Ciphertext: 11317051683322844863
DES Decrypted Plaintext: 12345678
```

The results

Some warnings when compiling the file. We can ignore them.

September 19, 2025

19

Problem 1: DES/CBC Implementation

- In Problem 1, we will implement the DES in CBC mode using the skeleton code `tempdes_cbc.c`.
- You are not allowed to use any built-in function besides what is present in `tempdes_simple.c`
- `my_des_cbc_encrypt(unsigned char *input, unsigned char *output, long length, DES_key_schedule ks, DES_cblock *ivec, int env):`
 - `unsigned char *input`: a string (plaintext) for the input of the encryption
 - `unsigned char *output`: the output of the encryption (ciphertext)
 - `long length`: size of the input (in bytes)
 - `DES_key_schedule ks`: key for encryption (and decryption)
 - `DES_cblock *ivec`: initialization vector
 - `int env`: encryption/decryption switches (1 for Encryption, 0 for Decryption)

September 19, 2025

20

Problem 1: DES/CBC Implementation (cont.)

- **Compile your code:**

- Ubuntu/Windows:

```
$ gcc -o tempdes_cbc tempdes_cbc.c -lssl -lcrypto
```

- MacOS:

```
$ gcc -o tempdes_cbc tempdes_cbc.c -lssl -lcrypto -L/opt/homebrew/opt/openssl@3/lib -I/opt/homebrew/opt/openssl@3/include
```

- **Run your code:**

```
$ ./tempdes_cbc fedcba9876543210 40fedf386da13d57 test.txt my_test.des
```

↓ ↓ ↓ ↓

The IV Key Input file Output file

Problem 1: DES/CBC Implementation (cont.)

- **Compare the result of your function with that of the built-in function `DES_cbc_encrypt()`**

- Details on how to use the function:

<http://web.mit.edu/macdev/Development/MITKerberos/MITKerberosLib/DESLib/Documentation/api.html>

- **To do that, please print out the ciphertexts from `my_des_cbc_encrypt()` and `DES_cbc_encrypt()` to compare**

Problem 2: Benchmark DES

- **In Problem 2, we will**

1. Implement the DES using the built-in functions in the OpenSSL library
2. Measure the processing time of DES on different file sizes (from 8 bytes to 2 Mbytes).

- **For this part, we provided five files as follows**

- **data:** The folder contains files of different sizes
- **tempdes.c :** code skeleton for DES
- **func_desc.txt:** general descriptions of the built-in functions that you are going to use

Problem 2: Benchmark DES (cont.)

- **Please check the file “Lab I_Problems_Description.pdf” for a more detailed description on how to measure the processing time**

- **After that, please make a graph of the time measurements (in micro seconds - μs) as a function of file sizes [bytes] that show DES encryption and decryption times**

Submission Policies

- You should submit the following items to “Classroom”:
 1. A report in PDF format
 2. All source code file(s)
- The report:
 - Please follow the sample report that we uploaded to “Classroom ” (Word)
 - Language: English
 - After finishing, please convert it into PDF format for submission
 - Naming convention: `<Student_ID>_Lab_<X>_Report.pdf`
 - Example: B22DCAT184_Lab_1_Report.pdf
- Source code file(s)
 - If you have multiple files, please compress them into one zip file.
 - Naming convention: `<Student_ID>_Lab_<X>_Codes.zip`
 - Example: B22DCAT184_Lab_1_Codes.zip
- **Deadline for Lab 1 submission: 23:59, September 23 (next Monday)**

Grading Policies

- Please submit everything before the deadline
 - *Late submissions are subject to penalties*
- Please pay attention to the presentation of your report
 - Following the report sample
 - Consistent word fonts and sizes throughout the report
 - Clear images
 - Proper English
- While we encourage group discussions, each student must complete the exercises individually
- If any incidents of plagiarism or abuse of generative AI are detected, the submission will immediately receive zero