CSCI361/MCS9361
Cryptography and Secure Applications
Autumn 2016

Assignment 2 (10 marks, worth 10%)

Due Week 7, Friday, 15 April 2016, 11:59pm AEST.

Aim: To become familiar with block ciphers, mode of operations, the importance of S-box and cryptanalyis of linear cipher. In this assignment, you will implement two "mini" versions of DES called LDES and MDES which work on 4-bit block message and 2-bit key. In the first algorithm LDES, the S-box is removed and replaced with a linear operation making LDES a linear cipher. You will perform cryptanalyis on LDES. The second algorithm MDES has a S-box and it is secure against linear cryptanalyis.

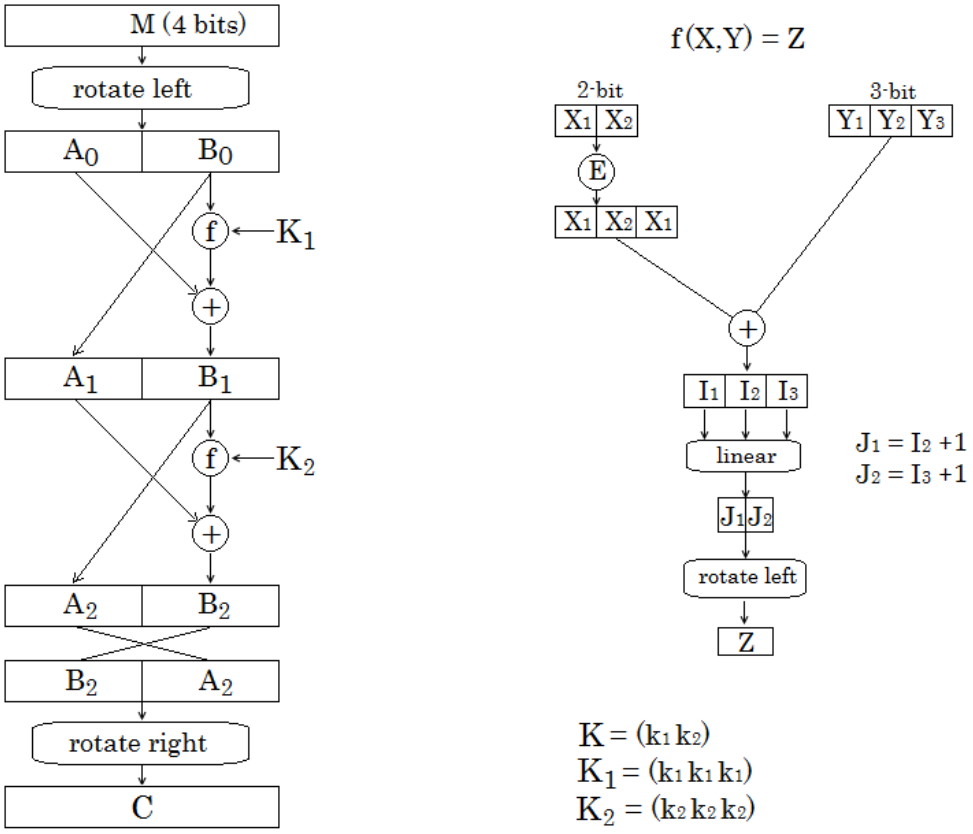
Notes

This assignment includes two parts: LDES and MDES.

- Your answers to non-programming questions should be typed up in a single report file. Do not use multiple files for the different parts of questions!
- Do not type out the questions in your report.
- Clearly label which questions your answers are associated with.
- There is no need to include a title page or cover sheet but please include your name, username and student number at the start of the document.
- Include a list of any reference material you have used at the end of the report. Note where you have used it at the point of use.
- For programming questions, you can use either C++ or Java to implement your solution.
- **Do not** copy work from others, give solutions to others or copy directly from websites. Plagiarism is taken seriously and is likely to result in a zero mark for the assignment for all the students involved.

Part 1: LDES 5 Marks

The figure below describes a block cipher LDES ("linear" DES).



LDES is a mini example of a block cipher that has 2 rounds in the Feistel network. It operates on 4-bit block and 2-bit key. The important feature of LDES is that the S-box has been replaced with a linear operation which makes the whole cipher linear. The purpose of this exercise is to help you understand the importance of S-box in DES. Without the S-box, the cipher becomes linear and can be totally broken. This exercise will guide you step by step to break the linear cipher LDES.

1. Based on the encryption in the figure above, draw a diagram for the corresponding decryption. **1 Marks**

2. Implement the encryption / decryption algorithms using C++ or Java. **2 Marks**

3. For each key = 00, 01, 10, 11, calculate $E(0000)$, $E(1000)$, $E(0100)$, $E(0010)$, $E(0001)$, $E(1100)$, $E(1010)$, $E(1001)$, $E(0110)$, $E(0101)$, $E(0011)$, $E(0111)$, $E(1011)$, $E(1101)$, $E(1110)$, $E(1111)$.

Verify that the values of $E(1100)$, $E(1010)$, $E(1001)$, $E(0110)$, $E(0101)$, $E(0011)$, $E(0111)$, $E(1011)$, $E(1101)$, $E(1110)$, $E(1111)$ can be determined from the values of $E(0000)$, $E(1000)$, $E(0100)$, $E(0010)$, $E(0001)$. Write down explicitly the relationship between these values. **2 Marks**

This demonstrates that if an adversary knows $E(0000)$, $E(1000)$, $E(0100)$, $E(0010)$, $E(0001)$, he can encrypt any message. In general, for a linear cipher with block size n, if an adversary knows $n+1$ pairs of plaintext/ciphertext, he can encrypt/decrypt any messages without even knowing the key.

Part 2: MDES 5 Marks

Consider a block cipher MDES ("mini" DES) which works exactly the same as LDES, except that the operation from (I_1, I_2, I_3) to (J_1, J_2) is based on the following S-box

$I_1I_2I_3$	000	001	010	011	100	101	110	111
J_1J_2	00	00	00	01	00	00	10	11

4. Implement the encryption / decryption algorithms of MDES using C++ or Java. **1 Marks**

5. Using MDES, for each key = 00, 01, 10, 11, calculate $E(0000)$, $E(1000)$, $E(0100)$, $E(0010)$, $E(0001)$, $E(1100)$, $E(1010)$, $E(1001)$, $E(0110)$, $E(0101)$, $E(0011)$, $E(0111)$, $E(1011)$, $E(1101)$, $E(1110)$, $E(1111)$.

Verify that the relationship that you observe in question 3 now is no longer valid **1 Marks**

6. Implement MDES in two modes: ECB and CBC using C++ or Java. Your program should accept a key as 2-bit binary string, a message as hex string and outputs a ciphertext as a hex string on the console. For example:

```
YourProgram -key 10 -mode ECB -encrypt f4a5a32
YourProgram -key 01 -mode ECB -decrypt b6f7a11
YourProgram -key 11 -mode CBC -iv a -encrypt 2a45def
YourProgram -key 00 -mode CBC -iv 4 -decrypt b412ab2
```

Note that each hex character is a 4-bit block message. **3 Marks**

Notes on submission:

You should zip your solutions into a file `<student-number>.zip` and submit it via Turnitin on Moodle site.

Your code must compile on Banshee with the instructions you provide. If it doesn't you will likely be given zero for the programming part of this assignment.

1. Late submissions will be marked with a 25% deduction for each day, including days over the weekend.
2. Submissions more than three days late will not be marked, unless an extension has been granted.
3. If you need an extension apply through SOLS, if possible **before** the assignment deadline.
4. Plagiarism is treated seriously. Students involved will likely receive zero.