Laboratory session 1

Implementation and linear cryptanalysis of a Feistel cipher

Nicola Laurenti, Francesco Ardizzon

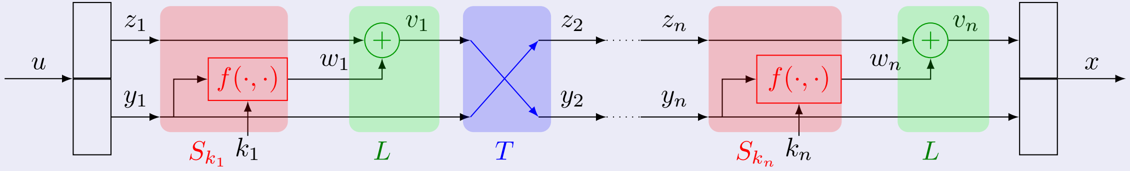
November 30, 2020

Laboratory session 1— Contents

Feistel ciphers

A Feistel cipher is a binary block cipher with $\mathcal{M} = \mathcal{X} = \mathbb{B}^{2\ell}$ that is based on the following n-round (S, T, L) iterated structure $E_k = LS_{k_n} \cdots TLS_{k_2}TLS_{k_1}$

Encryption



1. First the plaintext u is split into two ℓ-bit blocks y_1 and z_1
2. Then at each round i the following transformation are applied

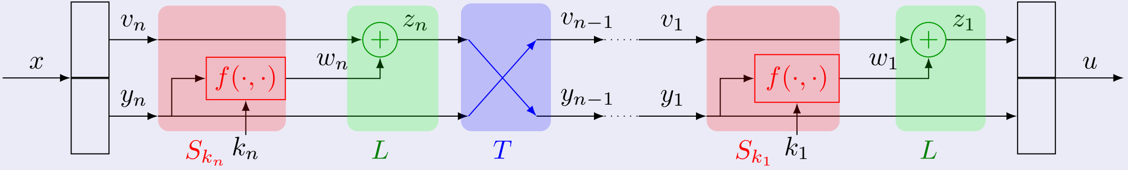
$$\begin{array}{lll} \text{substitution} & S : \mathcal{K}' \times \mathbb{B}^{2\ell} \mapsto \mathbb{B}^{3\ell} , & S_{k_i}(y_i, z_i) = [y_i, w_i, z_i] , \quad w_i = f(k_i, y_i) \\ \text{linear tf} & L : \mathbb{B}^{3\ell} \mapsto \mathbb{B}^{2\ell} , & L(y_i, w_i, z_i) = [y_i, v_i] , \quad \text{with } v_i = w_i + z_i \\ \text{transposition} & T : \mathbb{B}^{2\ell} \mapsto \mathbb{B}^{2\ell} , & T(y_i, v_i) = [v_i, y_i] = [y_{i+1}, z_{i+1}] , \quad i \neq n \end{array}$$

3. Last, y_n and v_n are concatenated to make the ciphertext x

Feistel ciphers

A Feistel cipher can be decrypted by using the same operations and in the same order, except for the inversion of the key sequence, i.e.: $D_k = LS_{k_1} TLS_{k_2} \cdots TLS_{k_n}$

Decryption



1. Split x into y_n and v_n
2. Then at each round i running backwards (from n to 1)

$$\begin{aligned} S_{k_i}(y_i, v_i) &= [y_i, w_i, v_i] \ , \quad \text{with } w_i = f(k_i, y_i) \\ L(y_i, w_i, v_i) &= [y_i, z_i] \ , \quad \text{with } z_i = w_i + v_i \\ T(y_i, z_i) &= [z_i, y_i] = [y_{i-1}, v_{i-1}] \ , \quad i \neq 1 \end{aligned}$$

3. Last, y_1 and z_1 are concatenated to make the plaintext u

Example: Data Encryption Standard (DES)

- A Feistel cipher with binary keys and lengths $\ell_k = 56$, $\ell_u = \ell_x = 64$, $\ell = 32$, using $n = 16$ rounds
- Designed by IBM in 1977 for the US NSA
- Efficient hardware implementation

Security features

- Moderately secure against brute force (key too short even then)
- Careful design of the round function $f(\cdot, \cdot)$ avoiding linear and differential cryptanalysis (only discovered in the 90's)

Implement a simple Feistel encryptor

Task 1

Using a programming language of your choice, implement the encryptor for a Feistel cipher with the following parameters:

message length $\ell_u = \ell_x = 2\ell = 32$, key length $\ell_k = 32$, nr. of rounds $n = 17$

round function the j-th bit of the output block w_i in the i-th round, denoted $w_i(j)$ is

$$f \quad : \quad w_i(j) = \begin{cases} y_i(j) \oplus k_i(4j-3) & , \quad 1 \leq j \leq \ell/2 \\ y_i(j) \oplus k_i(4j-2\ell) & , \quad \ell/2 < j \leq \ell \end{cases}$$

subkey generation the j-th bit of the subkey k_i for the i-th round, denoted $k_i(j)$ is

$$g_i \quad : \quad k_i(j) = k(((5i+j-1) \bmod \ell_k) + 1) \quad , \quad i = 1, \ldots, n$$

Check that your implementation is correct by verifying that the encryption of $u = \texttt{0x80000000} = [1, 0, \ldots, 0]$ with the key $k = \texttt{0x80000000} = [1, 0, \ldots, 0]$ is $x = \texttt{0xD80B1A63} = [1101\,1000\,0000\,1011\,0001\,1010\,0110\,0011]$

Task 2

Implement the decryptor for this Feistel cipher

Check that your implementation is correct by verifying that by concatenating encryption and decryption with the same key k you retrieve the original plaintext u. Experiment with different (u, k) pairs

Identify the cipher vulnerability

Observe that

- the round function $f(\cdot,\cdot)$ is linear in both the message block and the subkey
- the subkey generation function $g_i(\cdot)$ is linear in the key

and conclude that the cipher is linear

Task 3

Identify the overall linear relationship for this Feistel cipher, that is find the binary matrices $A \in \mathbb{B}^{\ell_x \times \ell_k}$ and $B \in \mathbb{B}^{\ell_x \times \ell_u}$ such that

$$x = E(k, u) = Ak + Bu$$

with all operations in the binary field $(\mathbb{B}, \oplus, \odot) = (\{0, 1\}, \text{XOR}, \text{AND})$
(if you do not know how to identify a linear system in a black box model, see Appendix 1)

Carry out linear cryptanalysis

Task 4

From a known plaintext/ciphertext pair (u, x), implement a linear cryptanalysis KPA against this cipher by computing

$$k = A^{-1}(x + Bu)$$

(if you do not know how to compute A^{-1}, the binary inverse of A, ▸ see Appendix 2)

You will find a few plaintext/ciphertext pairs, all encrypted with the same key k in a file labeled `KPApairsXxxxxx_linear.txt` in the folder `KPAdataXxxxxx`, where `Xxxxxx` is your team's city name. Find the key k

"Nearly linear" Feistel cipher

Task 5

Implement the encryptor and decryptor for a Feistel cipher with the following parameters:

message length $\ell_u = \ell_x = 2\ell = 32$, key length $\ell_k = 32$, nr. of rounds $n = 5$

round function with the notation from Task 1, and $\vee$ = bitwise OR, $\wedge$ = bitwise AND

$$w_i(j) = \begin{cases} y_i(j) \oplus \{k_i(4j-3) \wedge [y_i(2j-1) \vee k_i(2j-1) \vee k_i(2j) \vee k_i(4j-2)]\} & , 1 \leq j \leq \ell/2 \\ y_i(j) \oplus \{k_i(4j-2\ell) \wedge [k_i(4j-2\ell-1) \vee k_i(2j-1) \vee k_i(2j) \vee y_i(2j-\ell)]\} & , \ell/2 < j \leq \ell \end{cases}$$

for $i = 1, \ldots, n$

subkey generation is the same as in Task 1

Check that your implementation is correct by verifying that the encryption of
$u =$ `0x12345678` $= [0001\,0010\,0011\,0100\,0101\,0110\,0111\,1000]$ with the key
$k =$ `0x87654321` $= [1000\,0111\,0110\,0101\,0100\,0011\,0010\,0001]$ is
$x =$ `0x2E823D53` $= [0010\,1110\,1000\,0010\,0011\,1101\,0101\,0011]$

Linear cryptanalysis of a “nearly linear” cipher

Task 6

Find a linear approximation of the cipher in Task 5, that is, find matrices A, B, C such that (it might also be $C = I$)

$$\mathrm{P}\left[Ak \oplus Bu \oplus Cx = 0\right] \gg \frac{1}{2^{2\ell}}$$

From a few known plaintext/ciphertext pair (u, x), implement a linear cryptanalysis KPA against this cipher by computing

$$k = A^{-1}(Cx \oplus Bu)$$

and then explore “close” key values to find the key that encrypts u to x exactly see Appendix 2)

You will find a few plaintext/ciphertext pairs, all encrypted with the same key k in a file labeled `KPApairsXxxxxx_nearly_linear.txt` in the folder `KPAdataXxxxxx`, where `Xxxxxx` is your team's city name

Non linear Feistel cipher

Task 7

Implement the encryptor and decryptor for a Feistel cipher with the following parameters:

message length $\ell_u = \ell_x = 2\ell = 16$, key length $\ell_k = 16$, nr. of rounds $n = 13$

round function with the notation from Tasks 1 and 5

$$w_i(j) = \begin{cases} [y_i(j) \wedge k_i(2j-1)] \vee [y_i(2j-1) \wedge k_i(2j)] \vee k_i(4j) & , \quad 1 \leq j \leq \ell/2 \\ [y_i(j) \wedge k_i(2j-1)] \vee [k_i(4j-2\ell-1) \wedge k_i(2j)] \vee y_i(2j-\ell) & , \quad \ell/2 < j \leq \ell \end{cases}$$

subkey generation is the same as in Tasks 1 and 5

Check that your implementation is correct by verifying that the encryption of $u =$ `0x0000` $= [0, 0, \ldots, 0]$ with the key $k =$ `0x369C` $= [0011\,0110\,1001\,1100]$ is $x =$ `0x6A9B` $= [0110\,1010\,1001\,1011]$

Meet in the middle attack

Task 8

Implement a "meet-in-the-middle" attack see Appendix 3 against the concatenation of two instances of the non linear Feistel cipher defined in Task 7, with different keys k_1, k_2, respectively.

You will find a few plaintext/ciphertext pairs, all encrypted with the same pair of keys k, k' in a file labeled `KPApairsXxxxxx_non_linear.txt` in the folder `KPAdataXxxxxx`, where `Xxxxxx` is your team's city name

What you need to turn in

Each team must turn in, through the Moodle assignment submission procedure:

1. the code for your implementation (either as a single file, many separate files, or a compressed folder)
2. a short report (1-3 pages) in a graphics format (PDF, DJVU or PostScript are ok; Word, TEX or LATEX source are not), including:
   - 2.1 a brief description of your implementations for Tasks 1-8, explaining your choices;
   - 2.2 the results of your cryptanalysis effort:
     - 2.2.1 the matrices A and B that you used in Task 3;
     - 2.2.2 your guess $\hat{k}$ for the key we used to encrypt the KPA pairs in Task 4
     - 2.2.3 the matrices A, B and C that you used in Task 5;
     - 2.2.4 your guess $\hat{k}$ for the key we used to encrypt the KPA pairs in Task 6
     - 2.2.5 your guesses $\hat{k}, \hat{k}'$ for the keys we used to encrypt the KPA pairs in Task 8

Appendix 1: identifying a linear system

A general linear system, $y = Au$, with input u and output y can always be identified in a black box approach, by feeding it as inputs the vectors of the standard orthonormal basis

$$e_1 = [100\ldots0] \quad , \quad e_2 = [010\ldots0] \quad , \quad \cdots \quad , \quad e_\ell = [000\ldots01]$$

and observing the corresponding outputs.

In fact, by choosing a sequence of inputs $u_1, \ldots, u_\ell$ such that $u_j = e_j$, and observing the corresponding outputs y_j we obtain that $y_j = Ae_j$ is the j-th column of matrix A.

In our case there are two inputs, the plaintext and the key. By encrypting $(e_1, \ldots, e_\ell)$ and the all-zero vector 0 you can obtain each column a_j of the matrix A and each column b_j of matrix B, as

$$\begin{aligned} k = e_j, u = 0 \quad &\Rightarrow \quad x = E(e_j, 0) = Ae_j + B0 = a_j \quad , \quad j = 1, \ldots, \ell_k \\ k = 0, u = e_j \quad &\Rightarrow \quad x = E(0, e_j) = A0 + Be_j = b_j \quad , \quad j = 1, \ldots, \ell_u \end{aligned}$$

Appendix 2: computing the inverse of a binary matrix

The inverse of a square matrix A in the binary field $\mathbb{B}$ is the matrix A^{-1} is given by

$$A^{-1} = A^* \cdot \det(A) \bmod 2$$

where A^* and $\det(A)$ are the inverse and the determinant of A in the real field $\mathbb{R}$, so $A^* \cdot \det(A)$ is an integer matrix. In fact

$$A \odot A^{-1} = (A \cdot A^* \cdot \det(A)) \bmod 2 = (I \cdot \det(A)) \bmod 2 = I$$

where $\odot$ and $\cdot$ denote the product between binary and between real matrices, respectively

Example

$$A = \begin{bmatrix} 0&0&0&1&1 \\ 1&0&1&0&1 \\ 1&1&1&1&0 \\ 1&1&0&0&1 \\ 0&0&1&0&1 \end{bmatrix}, \quad A^* \cdot \det(A) = \begin{bmatrix} 0&3&0&0&-3 \\ -1&-3&1&2&2 \\ -1&0&1&-1&2 \\ 2&0&1&-1&-1 \\ 1&0&-1&1&1 \end{bmatrix}, \quad A^{-1} = \begin{bmatrix} 0&1&0&0&1 \\ 1&1&1&0&0 \\ 1&0&1&1&0 \\ 0&0&1&1&1 \\ 1&0&1&1&1 \end{bmatrix}$$

Appendix 3: "meet in the middle" attack

This is a KPA against a concatenated cipher (see slides), where $x = E''_{k''}(E'_{k'}(u))$ It consists in trying N' distinct guesses for $k' \in \mathcal{K}'$, and N'' distinct guesses for $k'' \in \mathcal{K}''$, with a complexity significantly lower than the product $N'N''$. Given a known plaintext/ciphertext pair (u, x)

1. Generate $N' \leq |\mathcal{K}'|$ random guesses of k', $\hat{k}'_1, \ldots \hat{k}'_{N'}$
2. For each guess $\hat{k}'_i$ compute the corresponding cipher guess $\hat{x}'_i = E'_{\hat{k}'_i}(u)$
3. Sort the table with key and cipher guesses, according to $\hat{x}'_i$
4. Generate $N'' \leq |\mathcal{K}''|$ random guesses of k'', $\hat{k}''_1, \ldots \hat{k}''_{N''}$
5. For each guess $\hat{k}''_i$ compute the corresponding plaintext guess $\hat{u}''_i = D''_{\hat{k}''_i}(x)$
6. Sort the table with key and cipher guesses, according to $\hat{u}''_i$
7. Search for a match between the two sorted tables, that is a pair of guesses $(\hat{k}'_i, \hat{k}''_j)$ such that $x'_i = u'_j$. Then, $\hat{k}' = \hat{k}'_i$ and $\hat{k}'' = \hat{k}''_j$ will be your final guess

If you get several matches you can increase the attack success probability with more KPA pairs