

# Exercise Sheet #10

Advanced Cryptography 2023

## Exercise 1 Linear Cryptanalysis of a Dummy Block Cipher

Consider the following (completely broken) block cipher with 32-bit inputs and 32-bit keys. Let the bits of the message be denoted by  $m_0, \dots, m_{31}$  and the bits of the key  $k_0, \dots, k_{31}$ . To encrypt, we do the following three operations for 5 rounds:

1. For all rounds, do:
2. We first XOR each bit of the message  $m_i$  with  $k_i$ . We call the result  $m'_i$ .
3. Then, we pass the message into the following  $4 \times 4$  Sbox. More precisely, for every  $i \in \{0, \dots, 7\}$ , we take the bits  $m'_{4i}, m'_{4i+1}, m'_{4i+2}, m'_{4i+3}$  and pass them through the following  $4 \times 4$  Sbox (where  $m'_{4i}$  is the most significant bit):

input	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
output	E	4	D	1	2	F	B	8	3	A	6	C	5	9	0	7

The output message is denoted by  $m''$ .

4. **Except in the last round**, the message bits are permuted using the following simple scheme (all the indices are modulo 32), for  $i \in \{0, \dots, 8\}$ :  $m'''_{4i} \leftarrow m''_{4i-3}, m'''_{4i+1} \leftarrow m''_{4i-2}, m'''_{4i+2} \leftarrow m''_{4i-4}, m'''_{4i+3} \leftarrow m''_{4i-1}$ .

**In the last round**, we apply another layer of XOR with the key (i.e., step 1) instead.

1. The equivalent for the DDT in linear cryptanalysis is the Linear Approximation Table (LAT). For an  $n \times n$  Sbox  $S$ , the LAT will be a  $2^n \times 2^n$  table. Each line will represent the possible input masks and each column the possible output masks. The value of the table at position  $(i, j)$  is the number of messages  $x$  such that  $x \cdot i = S(x) \cdot j$  **minus**  $2^{n-1}$ , i.e.,  $|\{x : x \cdot i = S(x) \cdot j\}| - 2^{n-1}$ .<sup>1</sup> An interesting property of the LAT is that the sum of any row or of any column is either  $-2^{n-1}$  or  $2^{n-1}$ .

As an exercise, compute for the value of the LAT of the SBox for input mask 3 and output mask 9, i.e., compute  $\text{LAT}(3, 9)$ .

2. We give now the complete LAT for our Sbox.

<sup>1</sup>Note that here, the  $\cdot$  notation is not multiplication but a dot product of two vectors. I.e.  $x \cdot i = x_1 \times i_1 \oplus x_2 \times i_2 \oplus x_3 \times i_3 \oplus x_4 \times i_4$

		Output mask															
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Input mask	0	+8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	1	0	0	-2	-2	0	0	-2	+6	+2	+2	0	0	+2	+2	0	0
	2	0	0	-2	-2	0	0	-2	-2	0	0	+2	+2	0	0	-6	+2
	3	0	0	0	0	0	0	0	0	+2	-6	-2	-2	+2	+2	-2	-2
	4	0	+2	0	-2	-2	-4	-2	0	0	-2	0	+2	+2	-4	+2	0
	5	0	-2	-2	0	-2	0	+4	+2	-2	0	-4	+2	0	-2	-2	0
	6	0	+2	-2	+4	+2	0	0	+2	0	-2	+2	+4	-2	0	0	-2
	7	0	-2	0	+2	+2	-4	+2	0	-2	0	+2	0	+4	+2	0	+2
	8	0	0	0	0	0	0	0	0	-2	+2	+2	-2	+2	-2	-2	-6
	9	0	0	-2	-2	0	0	-2	-2	-4	0	-2	+2	0	+4	+2	-2
	A	0	+4	-2	+2	-4	0	+2	-2	+2	+2	0	0	+2	+2	0	0
	B	0	+4	0	-4	+4	0	+4	0	0	0	0	0	0	0	0	0
	C	0	-2	+4	-2	-2	0	+2	0	+2	0	+2	+4	0	+2	0	-2
	D	0	+2	+2	0	-2	+4	0	+2	-4	-2	+2	0	+2	0	0	+2
	E	0	+2	+2	0	-2	-4	0	+2	-2	0	0	-2	-4	+2	-2	0
	F	0	-2	-4	-2	-2	0	+2	0	0	-2	+4	-2	-2	0	+2	0

Given the table, how do you obtain the probability that one linear approximation occurs? What is the probability bias (i.e, the difference to  $1/2$ )? What is its Linear Probability (LP)? What are the most interesting linear characteristics?

- Find a deviant property on four rounds that occurs with LP  $(9/16)^4 \approx 1/10$ . What is the probability that this linear approximation occurs (note that this depends on the key bits which shows why it is more convenient to use LPs)?
- Explain how you recover four bits of the last subkey using this deviant property. How many samples do you need approximatively?
- Show that it is possible to extend the previous attack to recover all the key with a simple shift of your previous attack.

## Exercise 2 Feistel Schemes

We consider a Feistel scheme of one round with 64-bit blocks (see Figure 1(a)).

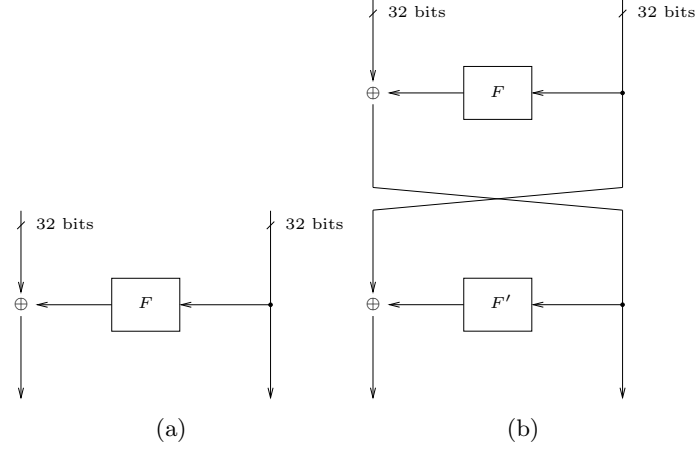


Figure 1: Feistel schemes with one or two rounds

Algorithm 1 is a distinguisher  $\mathcal{D}$  which tries to predict whether an oracle  $\mathcal{O}$  is a 1-round Feistel scheme or a uniformly random permutation.

---

**Algorithm 1** 1-round Feistel distinguisher  $\mathcal{D}$

---

**Input:** an oracle  $\mathcal{O}$  implementing either a 1-round Feistel scheme  $\Psi^{(1)}$  or a random permutation  $C^*$

**Output:** 0 (if the guess is that  $\mathcal{O}$  implements  $C^*$ ) or 1 (if the guess is that  $\mathcal{O}$  implements  $\Psi^{(1)}$ )

**Processing:**

- 1: let  $P = (x_\ell, x_r)$  be the input plaintext
  - 2: submit  $P$  to  $\mathcal{O}$  and get  $C = (y_\ell, y_r)$
  - 3: **if**  $x_r = y_r$  **then**
  - 4:   output 1
  - 5: **else**
  - 6:   output 0
  - 7: **end if**
- 

1. What is the probability that this distinguisher outputs “1” with a 1-round Feistel scheme, denoted  $\Psi^{(1)}$ ?
2. Same question with the uniformly random permutation over  $\{0, 1\}^{64}$ , denoted  $C^*$ . What is the advantage of  $\mathcal{D}$ ?

**Reminder:** The advantage of a distinguisher is defined as

$$\text{Adv}^{\mathcal{D}} = \left| \Pr[\mathcal{D}^{C^*} \rightarrow 1] - \Pr[\mathcal{D}^{\Psi^{(1)}} \rightarrow 1] \right|.$$

We consider now a 2-round Feistel scheme (see Figure 1(b)).

3. Propose a distinguisher for a 2-round Feistel scheme with a non-negligible advantage.
4. What is the probability that your distinguisher outputs “1” in both cases? What is the advantage of your distinguisher?