1 Attacking a Block Cipher by Introducing Faults

The aim of this problem is to show how introducing some faults in a block cipher can have a dramatic effect on its security. Throughout this exercise, we will consider a block cipher denoted $E$ with $\ell$ rounds, a block size and a key size of $n$ bits. This block cipher simply consists of an iteration of functions $T_i$ and subkey additions (see Figure 1). The subkeys $k_i$, $0 \leq i \leq \ell$ are all derived from the secret key $k$ associated to $E$. The $i$-th round is denoted as $R_i$ and the intermediate state of the plaintext $p$ after the $i$-th round is denoted $p_i$. So, we have $R_0(p) = k_0 \oplus p = p_0$, $R_i(p_{i-1}) = T_i(p_{i-1}) \oplus k_i = p_i$ for $1 \leq i \leq \ell$, and the ciphertext $c = p_\ell$.

![Diagram of the block cipher $E$]

**Figure 1:** The block cipher $E$
1. Show how the decryption algorithm works. Under which conditions can we decrypt the ciphertexts encrypted by \( E \)?

From now on, we will assume we have a device at our disposal which allows to produce some faults in a given implementation of \( E \) (in a smartcard, for example). Usually, one fault will correspond to flipping one chosen bit of an intermediate state \( p_i \). We will also assume that \( k_\ell \) is uniformly distributed in \( \{0, 1\}^n \) and that \( T_1 = T_2 = \ldots = T_\ell = T \).

2. Here, we will produce some faults on \( p_{\ell-1} \), i.e., we modify \( p_{\ell-1} \) to \( p'_{\ell-1} := p_{\ell-1} \oplus \delta \), where \( \delta \) is a bitstring of length \( n \), with a 1 at the position of the bit we aim at modifying in the ciphertext, and 0’s everywhere else. Let \( c' \) be the ciphertext obtained when introducing the faults \( \delta \). Find a relation between \( \delta, p_{\ell-1}, c, \) and \( c' \).

3. Suppose here that our device only allows us to produce some faults in the subkeys. Can we get the same \( c' \) as above with such a device? Justify your answer.

4. Assume here, that \( n = 12 \) and that \( T \) is defined as follows

\[
T : (x_1, x_2, x_3, x_4) \mapsto (f(x_1), f(x_2), f(x_3), f(x_4)),
\]

where the function \( f : \{0, 1\}^3 \rightarrow \{0, 1\}^3 \) is defined by the following table

<table>
<thead>
<tr>
<th>( x )</th>
<th>000</th>
<th>001</th>
<th>010</th>
<th>011</th>
<th>100</th>
<th>101</th>
<th>110</th>
<th>111</th>
</tr>
</thead>
<tbody>
<tr>
<td>( f(x) )</td>
<td>101</td>
<td>100</td>
<td>010</td>
<td>111</td>
<td>000</td>
<td>001</td>
<td>110</td>
<td>011</td>
</tr>
</tbody>
</table>

Now, we will try to obtain some information about one subkey. For this, we first encrypt a plaintext \( p \) chosen randomly with uniform distribution using the target implementation of \( E \). Later, we encrypt again the same plaintext but we introduce some faults in \( p_{\ell-1} \) such that this one is transformed in \( p_{\ell-1} \oplus \delta \), with \( \delta = (001, 000, 000, 000) \), i.e., we flip the last bit of \( x_1 \). Let \( c \) be the ciphertext \( E(p) \) and \( c' \) be the ciphertext obtained with the introduced fault. Show that we can deduce some information on \( p_{\ell-1} \) when \( c = (101, 111, 010, 100) \) and \( c' = (100, 110, 010, 011) \). How many candidate values for \( p_{\ell-1} \) does this leave?

5. How many candidates for the subkey \( k_\ell \) does this leave?

6. Let \( c, c' \) and \( \delta \) be as above. Set \( \delta' = c \oplus c' \). Compute \( \text{DP}^T(\delta, \delta') \) for the above defined transformation \( T \).

7. Now, we consider that \( n, T, \) and \( \delta \) are arbitrary again. We repeat the above experiment. Let \( \mathcal{N}_\ell \) be the number of possible remaining candidates for \( k_\ell \) after the experiment. Give an expression of \( \mathcal{N}_\ell \) depending on \( \delta, \delta'(= c \oplus c'), n, \) and \( T \). Justify your answer.

8. Show that \( \mathcal{N}_\ell \geq 2 \).

9. In practice, it is very difficult to produce some fault at a chosen bit position. We consider again the experiment of question 4. except that the we produce a fault for which the bit position is uniformly distributed at random, i.e., \( \delta \) is picked uniformly at random among the bitstrings of size \( n \) with Hamming weight 1. We also assume that \( n = 12 \) and \( T \) is the one defined in question 4. Results of the experiment provides \( c = (101, 111, 010, 100) \) and \( c' = (101, 111, 110, 100) \). How many candidate values for \( k_\ell \) does this leave?
2 Attacks on Yi-Lam Hash Function

(Disclaimer: the first inventor happens to have the same name as one assistant at LASEC!)

We use the following notations in this exercise:

- \( m \): a constant equal to 64
- \( \| \): concatenation of two blocks
- \( \oplus \): bitwise XOR
- \( + \): addition modulo \( 2^m \)
- \( E_K(\cdot) \): a perfectly secure block cipher to encrypt \( m \)-bit plaintext under \( 2m \)-bit key \( K \).

The Yi-Lam hash function can be described as follows: let \( H_1^i \)'s and \( H_2^i \)'s be \( m \)-bit blocks for \( i = 0, 1, \ldots, n \). Assume for simplicity that each message can be divided into blocks of \( m \) bits before we hash it. Given the message \( M = M_1 \| M_2 \| \ldots \| M_n \) (\( M_i \) is the \( i \)-th \( m \)-bit block of \( M \)) and the initial value \( IV = (H_1^0, H_2^0) \), we compute

\[
\begin{align*}
H_1^1 &= (E_{H_2^{i-1}}(M_i \| H_1^{i-1}) \oplus M_i) + H_2^{i-1} \quad (1) \\
H_2^1 &= E_{H_1^{i-1}}(M_i \| H_1^{i-1}) \oplus H_1^{i-1} \quad (2)
\end{align*}
\]

for \( i = 1, 2, \ldots, n \). The final hash of \( M \) is the \( 2m \)-bit \((H_1^n, H_2^n)\).

1. Give the complexity of a preimage attack (\( IV \) is fixed) on Yi-Lam hash function in terms of \( m \), supposing that it is an ideal hash scheme.

2. A faster preimage attack on Yi-Lam hash is shown in Algorithm 1. Read it carefully and find a necessary and sufficient termination condition of the loop in Line 8.

**Algorithm 1** A preimage attack on Yi-Lam hash

**Inputs:**
1: \( IV, H_1^n, H_2^n \) \((n \) is unknown)  

**Output:**
2: \( M \) such that the Yi-Lam hash of \( M \) equals \((H_1^n, H_2^n)\)

**Processing:**
3: repeat  
4: choose a random \( n \)  
5: choose \( M_1, M_2, \ldots, M_{n-1} \) at random  
6: compute \( H_1^{n-1}, H_2^{n-1} \)  
7: Find \( M_n \) such that \( H_1^n = (H_2^n \oplus H_1^{n-1} \oplus M_n) + H_2^{n-1} \)  
8: until a certain condition is met  
9: output \( M = M_1, M_2, \ldots, M_n \)

3. Compute the average number of rounds for the loop in Algorithm 1.

4. A free start collision attack on the hash function \( \text{hash}(IV, M) \) consists in finding \( IV, IV', M, M' \) with \( M \neq M' \) such that

\[
\text{hash}(IV, M) = \text{hash}(IV', M'), \tag{3}
\]
where $IV, IV'$ can be freely and independently chosen. Give the complexity of a free start collision attack on the Yi-Lam hash in terms of $m$, supposing that it is an ideal hash scheme.

5. Find a sufficient condition(s) to hold on $H_0^1, H_0^2$ and the one-block message $M = M_1$, such that $H_1^1 = H_1^2$ always holds.

6. Using the solution to the previous question, deduce a free start collision attack on Yi-Lam hash. Estimate the attack complexity.