Cryptanalysis 2021 Homework 1

Ling Song, Hosein Hadipour

March 8, 2021

Question 1.

Consider the following structure where ENC and DEC represent the encryption and decryption via the `DES` algorithm with 56-bit key respectively. Note that, the first and last encryption blocks use the same 56-bit key k_1, whereas the middle one utilizes k_2 which is not necessarily the same as k_1. Does it provide a 112 bits security level? If not so, provide a cryptographic attack with time complexity of strictly less than 2^{112} `DES` encryptions. Please explain what model is your attack classified in (known-plaintext, chosen-plaintext, ...). Besides, the amount of **time** and **memory** in your attack should be specified in detail.

A known plaintext attack has more points in comparison to a chosen plaintext attack.

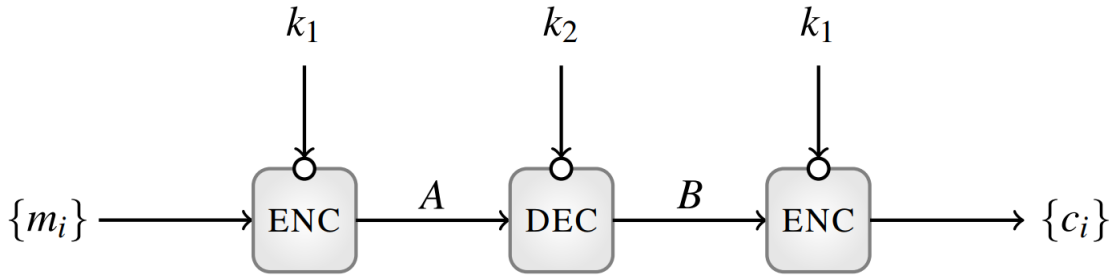

Figure 1: Two-Key Triple Encryption

Question 2.

This concerns using memory in favor of speed in implementing the encryption algorithms. Let S be the AES S-box. Let MC be the mix-column, i.e., it takes as input a column consisting of 4 bytes and outputs such a column. Note that MC is linear, i.e., we have for any two columns C_1, C_2 that:

$$MC(C_1 \oplus C_2) = MC(C_1) \oplus MC(C_2).$$

Define a function T_0 as follows: it takes as input a byte b, and the output $T_0(b)$ is a 4-byte column computed as follows: you first form a column $C(b)$ by placing $S(b)$ in the top byte and all-0 bytes in the lower three positions. Then set $T_0(b) = MC(C(b))$. We also define functions T_1, T_2, T_3. They are similar to T_0, except that when we form $C(b)$, we place $S(b)$ in the second, third and fourth entry from the top respectively, and put in 0's elsewhere.

Now consider the state of AES encryption algorithm at the start of some round. Name the bytes in this state a_{ij} as in ? and let R be the state after we have done `SubBytes`, `ShiftRow` and `MixColumn`. So R is a 4 by 4 matrix of bytes.

Show that the first column of R is

$$T_0(a_{00}) \oplus T_1(a_{11}) \oplus T_2(a_{22}) \oplus T_3(a_{33}).$$

Give similar expressions for the other 3 columns of R.

Sketch how this result can be used to implement AES based only on table look-up and XOR, instead of explicitly computing the operations. How much memory would you need for this?

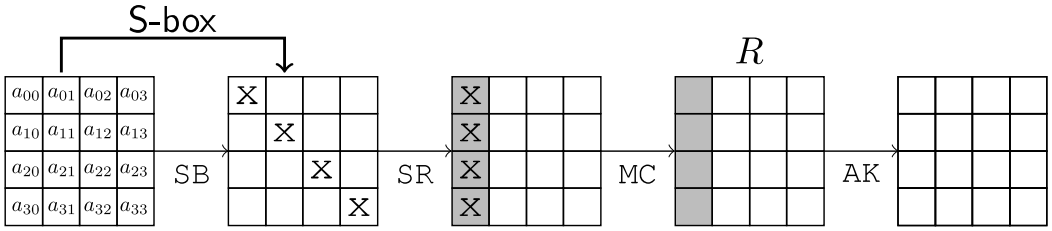

Figure 2: A Round of AES

Question 3.

Let $S = \{1, 2, \ldots, d\}$, where $d \in \mathbb{N}$. Besides, let $A = \{a_1, \ldots, a_m\}$ and $B = \{b_1, \ldots, b_n\}$ be two subsets of S.

(a) Prove that if $m \times n \geq d$, then $\Pr\{A \cap B \neq \emptyset\} \geq 0.5$.

(b) Assume that f is a function from $S = \{1, \ldots, N\}$ to itself where $N \in \mathbb{N}$. Besides, let $x_1^0, \ldots, x_m^0$ be some elements of S. Build the following array of chains:

$$\begin{array}{ll}
x_1^0 \longrightarrow x_1^1 = f(x_1^0) \longrightarrow & \cdots \longrightarrow x_1^{t-1} = f(x_1^{t-2}) \\
x_2^0 \longrightarrow x_2^1 = f(x_1^0) \longrightarrow & \cdots \longrightarrow x_1^{t-1} = f(x_2^{t-2}) \\
\cdots & \\
x_m^0 \longrightarrow x_m^1 = f(x_1^0) \longrightarrow & \cdots \longrightarrow x_m^{t-1} = f(x_m^{t-2}),
\end{array}$$

where $t \in \mathbb{N}$. Prove that if $mt^2 \leq N$, then all entries in the above array will be different with high probability ($\Pr \geq 0.5$).

(c) To do

Question 4.

This concerns a trick that is very useful to find a cycle in a sequence of iterated function values. Let S be any finite set, f be any function from S to itself, and x_0 be any element of S. For any $i > 0$, let $x_i = f(x_{i-1})$. Let μ be the smallest index such that the value x_μ reappears infinitely often within the sequence of values x_i, and let λ be the smallest positive integer such that $x_\mu = x_{\lambda+\mu}$.

(a) Prove that $i = k\lambda \geq \mu$ for some k if and only if $x_i = x_{2i}$.

(b) Based on the above fact, propose an algorithm to find μ and λ, given f and x_0.

(c) Using the proposed algorithm in the previous part, find a 64-bit collision for `SHA3-512`.

Hint: Study about the cycle detection algorithms in `https://en.wikipedia.org/wiki/Cycle_detection`.

To compute the `SHA3-512` using the `Python` language you can use the following commands:

```
In [1]: import hashlib
In [2]: st = "Hello World!"
In [3]: digest = hashlib.sha3_512(st.encode())
In [4]: digest.hexdigest()
Out[4]: '32400b5e89822de254e8d5d94252c52bdcb27a3562ca593e980364d9848b8041
b98eabe16c1a6797484941d2376864a1b0e248b0f7af8b1555a778c336a5bf48'
```