Solution Sheet #10

Advanced Cryptography 2021

Solution 1 Linear Cryptanalysis of a Dummy Block Cipher

The Sbox used in this exercise is taken from H. M. Hays' paper "A Tutorial on Linear and Differential Cryptanalysis".

1. We give here an algorithm to compute the value of the entry (i, j) of the LAT.

   Require: An $n \times n$ Sbox S, an input mask i, an output mask j

   ```
   Count ← 0
   for x = 0 to 2^n − 1 do                    ▷ For all possible inputs
       if x · i == S(x) · j then
           Count ← Count + 1
       end if
   end for
   return Count − 2^(n−1).
   ```

   If we apply this algorithm to the input mask 3 and the output mask 9, we obtain -6.

2. Let v be the value of $\mathsf{LAT}(i, j)$. The probability bias is simply $\mathsf{bias} := v/2^n$. The probability that the linear equation holds is then $1/2 + \mathsf{bias} = 1/2 + v/2^n$. The LP is $(2 \cdot \mathsf{bias})^2 = (v/2^{n-1})^2$. If we apply it to the entry $(3, 9)$, we get that $\mathsf{bias} = -3/8$, the probability that the equation holds is $1/8$ (note that this is smaller than $1/2$) and the LP is $9/16$.

   The most interesting entries of the table are $(1, 7)$, $(2, E)$, $(3, 9)$, and $(8, F)$. All these entries have an LP of $9/16$.

3. One of the best linear characteristics of the Sbox has input mask 3 and output mask 9. Notice that during the permutation layer (step 4), this mask 9 is transformed into the mask 3 (but shifted by one Sbox to the right). We will use these masks for one Sbox at each round (and a zero mask for the other). Let m be the initial message, let u_i be the message *before* the Sbox at round $i \in [1, 5]$, and let v_i be the message *after* the Sbox at round i. If we take as input mask 0x30000000, we end up with the mask 0x90000000 after the Sbox layer with $\mathsf{LP} = 9/16$. More precisely, this means that $\texttt{0x30000000} \cdot m \oplus \texttt{0x90000000} \cdot v_1 = 0$ holds with LP $9/16$. After the permutation layer, the mask becomes 0x03000000 as mentionned above, i.e, $\texttt{0x30000000} \cdot m \oplus \texttt{0x03000000} \cdot u_2 = 0$ holds with LP $9/16$.

We can iterate the same reasoning to obtain a characteristic on four rounds. At each Sbox, we get the wanted output mask with LP 9/16. Hence, for four rounds, by the piling-up lemma, we have $\mathsf{LP}(\texttt{0x30000000}, \texttt{0x00003000}) = (9/16)^4$, which implies that $\texttt{0x30000000} \cdot m \oplus \texttt{0x00003000} \cdot u_5 = 0$ holds with LP $(9/16)^4$.

If we want to express the corresponding probability, we have also to take care of the key bits which are involved during the XORing phase (step 1). The property we found means that $\texttt{0x30000000} \cdot m \oplus \texttt{0x00003000} \cdot u_5 = f(K)$ with probability $1/2 - (9/16)^2/2$, for some deterministic function $f : \{0,1\}^{32} \to \{0,1\}$ of the secret key. Hence, depending on the secret key, $\texttt{0x30000000} \cdot m \oplus \texttt{0x00003000} \cdot u_5 = 0$ holds either with probability $1/2 - (9/16)^2/2$ or $1/2 + (9/16)^2/2$. Note however that this dependency with the key doesn't matter when performing our statistics since we care only on *how far our probability is from* 1/2.

4. Let m be the initial message and z the value at the entrance of the fifth layer of Sboxes. Using the previous linear characteristic, we know that $\texttt{0x30000000} \cdot m \oplus \texttt{0x00003000} \cdot z = 0$ holds with LP $(9/16)^4$. We can rewrite this equation as

$$m_2 \oplus m_3 \oplus z_{18} \oplus z_{19} = 0 \; . \tag{1}$$

Our goal will be to guess four key bits from the last XORing operation (step 4 of the last round). Note that our linear characteristic gives us information only about the input of the fifth Sbox. The output of this Sbox is then XORed with the key bits k_{16}, k_{17}, k_{18}, and k_{19}. Hence, these are the four bits we are trying to recover. For each plaintext/ciphertext sample (recall that we are doing a known plaintext attack), we try all possible values κ for the partial key $k_{16}, k_{17}, k_{18}, k_{19}$ and use it to decrypt one round of the fifth group of four bits (hence obtaining $z_{16}, z_{17}, z_{18}, z_{19}$). Whenever, for a guess κ, Equation (1) holds, we increase a counter n_κ.

Let n be the number of samples. From the course, we know that we have to use about $1/\mathsf{LP} = (16/9)^4 \approx 10$ samples. Once we have processed all the samples, we sort the candidates for κ according to $|n_\kappa - n/2|$. The biggest value is the most likely one.

5. We can easily do the same attack with input masks 0x03000000, 0x00300000, ..., 0x00000003. This will shift by one Sbox the key bits we can recover and allow us to recover the whole key.

Solution 2 Feistel Schemes

1. The probability is equal to 1 as we always have $x_r = y_r$ for $\Psi^{(1)}$.

2. We have

$$\Pr[\mathcal{D}^{C^*} \to 1] = \Pr[C^*(x_r) = x_r].$$

We recall that for the random permutation C^* uniformly distributed over all possible permutations of $\{0,1\}^n$, we have for any $x, y \in \{0,1\}^n$

$$\Pr[C^*(x) = y] = \Pr_{Y \in \{0,1\}^n}[Y = y] = 2^{-n},$$

Therefore

$$\Pr[\mathcal{D}^{C^*} \to 1] = 2^{-32}.$$

Finally, the advantage of the distinguisher $\mathcal{D}$ is $\mathrm{Adv}^{\mathcal{D}} = 1 - 2^{-32}$.

3. We consider the distinguisher described in Algorithm 1.

Algorithm 1 2-round Feistel distinguisher $\mathcal{D}$

Input: an oracle $\mathcal{O}$ implementing either a 2-round Feistel scheme $\Psi^{(2)}$ or a uniformly random permutation C^*

Output: 0 (if the guess is that $\mathcal{O}$ implements C^*) or 1 (if the guess is that $\mathcal{O}$ implements $\Psi^{(2)}$)

Processing:

1: let $P = (x_\ell, x_r)$ and $P' = (x'_\ell, x_r)$ with $x_\ell \neq x'_\ell$ be two input plaintexts
2: submit P and P' to the oracle and get $C = (y_\ell, y_r)$ and $C' = (y'_\ell, y'_r)$
3: if $x_\ell \oplus x'_\ell = y_r \oplus y'_r$, then output "1", otherwise, output "0"

4. If the oracle $\mathcal{O}$ implements a 2-round Feistel scheme $\Psi^{(2)}$, we always have $x_\ell \oplus x'_\ell = y_r \oplus y'_r$, so that

$$\Pr[\mathcal{D}^{\Psi^{(2)}} \to 1] = 1.$$

Consider now the case where $\mathcal{O}$ implements C^* and denote $x = (x_\ell, x_r)$, $x' = (x'_\ell, x_r)$, $y = (y_\ell, y_r)$, and $y' = (y'_\ell, y'_r)$ such that

$$C^*(x) = y \qquad C^*(x') = y'.$$

As already mentioned, one can consider $C^*(x)$ and $C^*(x')$ as two random variables, that we will respectively denote $Y = (Y_\ell, Y_r)$ and $Y' = (Y'_\ell, Y'_r)$, uniformly distributed over $\{0,1\}^{64}$. But as we know that $x \neq x'$ and as C^* is a permutation, Y and Y' are different (which are therefore not independent). Consequently, if we denote $\alpha = x_\ell \oplus x'_\ell \neq 0$, we obtain

$$\begin{aligned}
\Pr[\mathcal{D}^{C^*} \to 1] &= \Pr[Y_r \oplus Y'_r = \alpha \mid Y \neq Y'] \\
&= \frac{\Pr[Y_r \oplus Y'_r = \alpha, Y \neq Y']}{\Pr[Y \neq Y']} \\
&= \frac{\Pr[Y \neq Y' \mid Y_r \oplus Y'_r = \alpha] \Pr[Y_r \oplus Y'_r = \alpha]}{\Pr[Y \neq Y']} \\
&= \frac{1 \times 2^{-32}}{1 - 2^{-64}} \\
&\approx 2^{-32}.
\end{aligned}$$

Consequently, the distinguisher $\mathcal{D}$ defined by Algorithm 1 has the following advantage

$$\mathrm{Adv}^{\mathcal{D}} \approx 1 - 2^{-32}.$$