

Problem Set 4 — Solutions

1. The basic idea in attacking this scheme is that given an encryption $\langle r, c \rangle$ of a known message m , we can compute the key k as $k = F_r^{-1}(c \oplus m)$. (You should convince yourself that this works.) This computation is possible because: (1) r is known, and (2) as always, the adversary knows the code for any underlying primitives being used (in this case, F and F^{-1}). Once we have the key, breaking the scheme is easy.

Formally, we have the following adversary \mathcal{A} :

$\mathcal{A}(1^n)$
submit $m = 0^n$ to the encryption oracle
obtain in response the ciphertext $\langle r, c \rangle$
compute $k := F_r^{-1}(c)$
output $m_0 = 0^n$ and $m_1 = 1^n$
receive in response a ciphertext $\langle r', c' \rangle$
decrypt this ciphertext using k ; i.e., compute $m' := F_{r'}(k) \oplus c'$
if $m' = m_0$, output 0; if $m' = m_1$, output 1

It should be clear that the above adversary always outputs a correct guess of b , and so if we let Π denote this scheme then we have $\Pr[\text{PrivK}_{\mathcal{A}, \Pi}^{\text{cpa}}(n) = 1] = 1$.

2. Given a ciphertext $\langle c_0, c_1 \rangle$ that is the encryption of an unknown, 1-block message m , we know that $c_1 = F_k(c_0 \oplus m)$, or $m = F_k^{-1}(c_1) \oplus c_0$ (of course, k is unknown). Now, if we request a decryption of the ciphertext $\langle c'_0, c'_1 \rangle$, we get back $m' = F_k^{-1}(c'_1) \oplus c'_0$. So, we set $c'_1 = c_1$ and $c'_0 = c_0 \oplus 1^n$. Then we get back

$$m' = F_k^{-1}(c_1) \oplus c_0 \oplus 1^n = m \oplus 1^n.$$

We thus conclude that $m = m' \oplus 1^n$. Note that the ciphertext $\langle c'_0, c'_1 \rangle$ whose decryption we request is different from $\langle c_0, c_1 \rangle$.

It is easy to turn this into a formal attack:

$\mathcal{A}(1^n)$
output $m_0 = 0^n$ and $m_1 = 1^n$
receive in response a ciphertext $\langle c_0, c_1 \rangle$
set $c'_0 := c_0 \oplus 1^n$ and $c'_1 := c_1$
submit $\langle c'_0, c'_1 \rangle$ to the decryption oracle; receive in response m'
if $m' \oplus 1^n = m_0$, output 0; if $m' \oplus 1^n = m_1$, output 1

It should be clear that the above adversary always outputs a correct guess of b , and so if we let Π denote CBC-mode encryption then we have $\Pr[\text{PrivK}_{\mathcal{A}, \Pi}^{\text{cca}}(n) = 1] = 1$.

3. (a) This scheme is secure. Here, I will prove that any forgery output by an adversary would imply that (except with negligible probability) the adversary was able to predict the value of $F_k(\cdot)$ on a “new” input value. You should then be able to turn this into a formal proof by reduction (based on the security of F as a pseudorandom function) as in the proof of Theorem 4.4.

Let M_1, \dots, M_q be the messages for which the adversary has requested tags, and let r_i denote the r -value used when the sender authenticates M_i . Let m_j^i denote the j th block of message M_i , and let ℓ_i denote the number of blocks in M_i .

Say the adversary outputs a valid tag $\langle \hat{r}, \hat{t}_1, \dots, \hat{t}_{\hat{\ell}} \rangle$ on a message $\hat{M} = \hat{m}_1, \dots, \hat{m}_{\hat{\ell}}$, where $\hat{M} \notin \{M_1, \dots, M_q\}$. (I.e., this is a forgery.) When this tag is verified, the receiver will evaluate F_k on the blocks $\hat{r}||0||i||\hat{m}_i$ for $1 \leq i < \hat{\ell}$, and also on the block $\hat{r}||1||\hat{\ell}||\hat{m}_{\hat{\ell}}$. We want to show that at least one of these blocks is new. (Since the tag output by the adversary is valid, this implies that the adversary was able to predict the value of F_k on a new input.) We do this by case-by-case analysis:

Case 1: $\hat{r} \notin \{r_1, \dots, r_q\}$. In this case, *all* of the adversary’s blocks are new. This is because each of the adversary’s blocks begins with \hat{r} , and none of the blocks used by the sender begin with this value.

Case 2: $\hat{r} = r_i$ **for a unique value i** . Here, we can restrict our attention to the blocks used by the sender when authenticating M_i (since all blocks used when authenticating other messages will begin with a prefix that is different from \hat{r}). Consider two sub-cases.

- First, say $\ell_i \neq \hat{\ell}$, where ℓ_i is the length of M_i . We claim that the block $\hat{r}||1||\hat{\ell}||\hat{m}_{\hat{\ell}}$ is new. This is because the only block authenticated by the sender that has prefix $\hat{r}||1$ is the block $r_i||1||\ell_i||m_{\ell_i}^i$. But since $\ell_i \neq \hat{\ell}$, these blocks are not the same.
- Otherwise, say $\ell_i = \hat{\ell}$. Since $\hat{M} \neq M_i$ but their lengths are the same, there must be some block j with $\hat{m}_j \neq m_j^i$. But then the block $\hat{r}||\star||j||\hat{m}_j$ (where \star indicates that we do not care what the value is) is new: the only block it could possibly be equal to is $r_i||\star||j||m_j^i$, but we know that $\hat{m}_j \neq m_j^i$.

Case 3: $\hat{r} = r_i = r_j$ **for two distinct values i, j** . This, in particular, means that $r_i = r_j$. But this means that the sender uses the same r -value when authenticating two different messages, something that (as in the proof of Theorem 4.6) occurs with only negligible probability.

- (b) This scheme is insecure. The attack is simple: the adversary requests a tag for the 1-block message m_1 , and receives in return $\langle r, t_1 \rangle$. It then outputs the forged tag $\langle r \oplus 1^n, t_1 \rangle$ on the message $m_1 \oplus 1^n$. It is easy to check that this is a valid forgery with probability 1.

4. (a) The adversary proceeds as follows:
- i. Request a tag on message m , receive in return the tag t .
 - ii. Request a tag on the message m, m' , and receive in return the tag t' .

iii. Output the tag t' on the message $m' \oplus t$.

In step 1, m is arbitrary. In step 2, pick any $m' \neq t \oplus m$. This ensures that $m' \oplus t \neq m$, and so the message $m' \oplus t$ that the adversary outputs is new.

This attack succeeds with probability 1. To see this, observe that $t = F_k(m)$ and $t' = F_k(m' \oplus F_k(m)) = F_k(m' \oplus t)$. But then t' is a valid tag on $m' \oplus t$.

- (b) Assume we are authenticating messages of length 1 (though the attack works for any length). The adversary requests a tag on the message m_1 and receives in return the tag $\langle t_0, t_1 \rangle$ where t_0 was chosen at random and $t_1 = F_k(t_0 \oplus m_1)$. Set $t'_0 = t_0 \oplus 1^n$, and output the tag $\langle t'_0, t_1 \rangle$ on the message $m_1 \oplus 1^n$. This is always a valid forgery, since $m_1 \oplus 1^n \neq m_1$ and

$$F_k(t'_0 \oplus (m_1 \oplus 1^n)) = F_k(t_0 \oplus m_1) = t_1.$$

- (c) Assume we are authenticating messages of length 2 (though the attack works for any length greater than 1). The adversary requests a tag on the message m_1, m_2 and receives in return the tag $\langle t_1, t_2 \rangle$ where $t_1 = F_k(m_1)$ and $t_2 = F_k(m_2 \oplus t_1)$. Output the tag $\langle t_1, t_1 \rangle$ on the message $m_1, m_1 \oplus t_1$. This tag is always valid because $F_k(m_1) = t_1$ and

$$F_k((m_1 \oplus t_1) \oplus t_1) = F_k(m_1) = t_1.$$

The only hitch is that we might have $t_1 = m_1 \oplus m_2$ in which case the adversary has not output a new message. But it is not hard to see that this can only occur with negligible probability.

- (d) I describe one attack; for another attack see reference [12] from the book.

Let $\langle i \rangle$ denote the encoding of i (the method used for this encoding is, as always, assumed to be known to the adversary), and use \parallel for concatenation. The first thing the adversary does is request a tag on the 1-block message m . (This means that basic CBC-MAC is applied to the 2-block message $m \parallel \langle 1 \rangle$.) Call the result t_2 . Let $t_1 = F_k(m)$ (note the adversary does not know t_1); then $t_2 = F_k(t_1 \oplus \langle 1 \rangle)$.

Next, the adversary requests a tag on the message $m \parallel \langle 1 \rangle \parallel (m \oplus t_2)$. Call the result t_3 . Following the computation of basic CBC-MAC on the 4-block message $m \parallel \langle 1 \rangle \parallel (m \oplus t_2) \parallel \langle 3 \rangle$, we see that:

- The output of the first call to F_k is $F_k(m)$, which is t_1 .
- The output of the second call to F_k is $F_k(t_1 \oplus \langle 1 \rangle)$, which is t_2 .
- The output of the third call to F_k is $F_k(t_2 \oplus (m \oplus t_2)) = F_k(m)$, which is again t_1 .
- The output of the fourth call to F_k is t_3 (which the adversary is given). We know that $t_3 = F_k(t_1 \oplus \langle 3 \rangle)$.

Finally, the adversary outputs the forgery t_3 on the message $m \parallel \langle 3 \rangle \parallel (m \oplus t_3)$. This is clearly a new message. To see that it is a valid forgery, we again follow the computation of basic CBC-MAC on the 4-block message $m \parallel \langle 3 \rangle \parallel (m \oplus t_3) \parallel \langle 3 \rangle$:

- The output of the first call to F_k is $F_k(m)$, which is t_1 .
- The output of the second call to F_k is $F_k(t_1 \oplus \langle 3 \rangle)$, which is t_3 .
- The output of the third call to F_k is $F_k(t_3 \oplus (m \oplus t_3)) = F_k(m)$, which is again t_1 .
- The output of the fourth call to F_k is $F_k(t_1 \oplus \langle 3 \rangle)$, which is again t_3 .