Indian Institute of Technology Delhi

COL 759 - Cryptography

Assignment 2

Security of PRFs, MACs and Padding Oracle Attack

Garv Nagori, Naman Agarwal
2021CS10549, 2021CS10555

Contents

§1 Part A

§1.1 Question 1.

To define F' which is secure PRF but not 1-leakage resilient, we define the key space, $\mathcal{K}'$ as $\{0,1\}^{n+1}$, the input space $\mathcal{X}'$ and output space $\mathcal{Y}'$ as $\{0,1\}^n$.
The PRF F' with key k' is constructed as follows:

$$F'(k',x) = \begin{cases} F(k,x) & x \neq 0^n \\ F(k,0^n), \text{remove last bit and adding the last bit of k'} & x = 0^n \end{cases}$$

where k is k' without the last bit.

§1.1.1 Subpart a)

To prove that F' is a secure PRF.
The proof is based on the fact that the last bit of $F'(k',0^n)$ can be 0 or 1 with equal probability.
For the proof, we prove the contrapositive, that is if there exists an Adversary $\mathcal{A}$ that breaks F', we can use it to create a reduction $\mathcal{B}$ that can break F. We create $\mathcal{B}$ as follows:

Reduction $\mathcal{B}$

Challenger $\mathcal{C}$		**Reduction $\mathcal{B}$**		**Adversary $\mathcal{A}$**
$b \leftarrow\$ \{0,1\}$		$k[n+1] \leftarrow\$ \{0,1\}$		
$k \leftarrow\$ \{0,1\}^n$				
if b = 0 then$y = F(k,x)$	$\xleftarrow{x}$	if $x \neq 0^n$ then $y' = y$	$\xleftarrow{x}$	
else $y \leftarrow\$ \{0,1\}^n$	$\xrightarrow{y}$	else y' = make y'[n] = k[n+1]	$\xrightarrow{y'}$	
	$\xleftarrow{b'}$		$\xleftarrow{b'}$	
		Adversary wins if $b = b'$		

Changing the last bit provides no additional information about k or $F(\text{k}, \cdot)$. So we can conclude that $\mathsf{PRFAdv}_{\mathcal{B}} = \mathsf{PRFAdv}_{\mathcal{A}}$, which is non-negligible.
Hence, F' is secure.

§1.1.2 Subpart b)

To show F' is not 1-leakage resilient, we can show an Adversary $\mathcal{A}$ which can win the security game with a non-negligible advantage.
The adversary $\mathcal{A}$ is defined as follows:

Breaking 1-leakage resilience

Challenger $\mathcal{C}$		**Adversary** $\mathcal{A}$
$b \leftarrow\$ \{0,1\}$		
$k \leftarrow\$ \{0,1\}^{n+1}$		
if $b = 0$ then $y = F'(k', x)$	$\xleftarrow{\text{PRF Query: } x}$	$x = 0^n$
else $y \leftarrow\$ \{0,1\}^n$	$\xrightarrow{y}$	
	$\xleftarrow{\text{Leakage Query: } n+1}$	if $k[n+1] = y[n]$ then $b' = 0$
	$\xrightarrow{k[n+1]}$	else $b' = 1$
	$\xleftarrow{b'}$	

Advantage is given as,
$\Pr[\mathcal{A} \leftarrow 0 \mid \mathcal{C} \leftarrow 0] = 1$ since k[n+1] would always be y[n]
$\Pr[\mathcal{A} \leftarrow 0 \mid \mathcal{C} \leftarrow 1] = \frac{1}{2}$ since there is equal probability of $k[n+1] = y[n]$ and $k[n+1] \neq y[n]$
Therefore $\mathsf{PRFAdv}_{\mathcal{A}} = \frac{1}{2}$, which is non-negligible.
Hence, F' is not 1-leakage resilient

§1.2 Question 2.

§1.2.1 Subpart a)

For $(Sign_{uq}, Verify_{uq})$, first we define key space $\mathcal{K}$ as $\{0,1\}^{4n}$, message space $\mathcal{M}$ as $\{0,1\}^n$, randomness space $\mathcal{R}$ as $\{0,1\}^{2n-8}$ and output space $\mathcal{C}$ as $\{0,1\}^{6n-8}$
We define $Sign_{uq}(k, m)$ as:
First, we break k into 4 equal parts, say k_1, k_2, k_3 and k_4 and also message into 2 equal parts m_0 and m_1. If n is odd we break into 2 parts with length $\lfloor \frac{n}{2} \rfloor$ and $\lceil \frac{n}{2} \rceil$ and adjust the randomness accordingly. Choose a randomness r and break into 4 equal parts r_1, r_2, r_3 and r_4. If n is odd we break appropriately to make the input to F of size n.
Now,
$Sign_{uq}(k, m) = r_0, F(k_0, r_0||00||m_0), r_1, F(k_1, r_1||01||m_1), r_2, F(k_2, r_2||10||m_0), r_3, F(k_3, r_3||11||m_1)$
We can define $Verify_{uq}(k, m, \sigma)$ as,
Split σ and get r_0, r_1, r_2 and r_3. Now compute and check $F(k_0, r_0||00||m_0), F(k_1, r_1||01||m_1), F(k_2, r_2||10||m_0)$ and $F(k_3, r_3||11||m_1)$.
If all four match output 1 else output 0.

§1.2.2 Subpart b)

To prove: I_{uq} is a secure MAC, given that F is a secure PRF, or in other words, the probability of adversary winning is negligible.
We can define Hybrid Worlds to prove the security.
We define World 0, which is the MAC scheme and then 4 Hybrid Worlds.

World 0

Challenger $\mathcal{C}$		**Adversary** $\mathcal{A}$
$k \leftarrow\$ \{0,1\}^{4n}$		
	$\xleftarrow{m}$	Polynomially
$\sigma = Sign(k, m)$	$\xrightarrow{\sigma}$	many queries
	$\xleftarrow{\text{Forgery: } (m'\sigma')}$	

Adversary wins if $Verify(k, m', \sigma') = 1$

We define the hybrid worlds similarly as above, but replacing one Pseudo-Random $F(k_i, \cdot)$ with a Truly Random Function in each.
In the **First Hybrid World**, instead of $F(k_0, \cdot)$, we use f_0 which is a truly random function chosen from set of all possible functions.
In the **Second Hybrid World**, we replace $F(k_0, \cdot)$ and $F(k_1, \cdot)$ with f_0 and f_1 respectively, both of which are chosen truly at random.
Similarly, in the third hybrid world, we replace $F(k_0, \cdot)$, $F(k_1, \cdot)$ and $F(k_2, \cdot)$ with f_0, f_1 and f_2 respectively, all three of which are chosen truly at random.
Finally, in the **Fourth Hybrid World**, we replace all $F(k_0, \cdot)$, $F(k_1, \cdot)$, $F(k_2, \cdot)$ and $F(k_3, \cdot)$ with f_0, f_1, f_2 and f_3 respectively, all of which chosen truly at random.

Claim 1.1 — The probability of an Adversary $\mathcal{A}$ winning in the **Fourth Hybrid World** is negligible

Proof. Since f_0, f_1, f_2, and f_3 are chosen truly at random, the adversary has to calculate f_i for a new value for at least one value of $i \in \{1, 2, 3, 4\}$.
This is because if he returns a forgery on a message that is never queried, all four values are new and if the forgery is on a message previously queried, he has to change the randomness, otherwise the signing will be the same.
Hence, at least one value would have to be new and so $\Pr[\mathcal{A}\, winning] \leq \frac{1}{2^n}$. □

Claim 1.2 — The probabilities of an adversary winning in **Third Hybrid World** and **Fourth Hybrid World** are negligibly close.

Proof. If the probabilities are not negligibly close, the probability of the adversary winning in **Hybrid World 3** is non-negligible but the probability of winning in **Hybrid World 4** is negligible (proved above).
Hence, we can create a reduction $\mathcal{B}_3$, which can break the PRF security of F.

Difference in Third and Fourth Hybrid Worlds

Challenger $\mathcal{C}$		**Reduction** $\mathcal{B}_3$		**Adversary** $\mathcal{A}$
$k_3 \leftarrow\$ \{0,1\}^n$		$f_0, f_1, f_2 \leftarrow\$ Func(\mathcal{X} \to \mathcal{Y})$		
$b \leftarrow\$ \{0,1\}$				
if $b = 0$ then	$\xleftarrow{r_3\|\|11\|\|m_1}$	$\sigma_0 = f_0(r_0\|\|00\|\|m_0)$	$\xleftarrow{m}$	
$y = F(k_3, x)$		$\sigma_1 = f_1(r_1\|\|01\|\|m_1)$ $\sigma_2 = f_2(r_2\|\|10\|\|m_0)$		
else $y \leftarrow\$ \{0,1\}^n$	$\xrightarrow{y}$	$\sigma = r_0, \sigma_0, r_1, \sigma_1, r_2, \sigma_2, r_3, y$	$\xrightarrow{\sigma}$	
	$\xleftarrow{x'}$	$x' = r'_3\|\|11\|\|m'_1$	$\xleftarrow{(m'\sigma')}$	
	$\xrightarrow{y'}$	if $y' = \sigma'_2$ and others match $b' = 0$		
	$\xleftarrow{b'}$	else $b' = 1$		

In this case, the adversary receives **Hybrid World 3** if $b = 0$, else he receives **Hybrid World 4**. $\mathcal{A}$ can break Hybrid World 3 with non-negligible probability, so if the verification passes we assume it is Hybrid World 3 else it is Hybrid World 4.
Therefore $\mathsf{PRFAdv}_{\mathcal{B}_3} = \mathsf{MACAdv}_{\mathcal{B}}$.
Hence, the probability of adversary winning in **Hybrid World 3** is negligible. □

Claim 1.3 — The probabilities of an adversary winning in **Second Hybrid World** and **Third Hybrid World** are negligibly close.

If the probabilities are not negligibly close, the probability of the adversary winning in **Hybrid World 2** is non-negligible but the probability of winning in **Hybrid World 3** is negligible (proved above).
Hence, we can create a reduction $\mathcal{B}_2$, which can break the PRF security of F.

Difference in Second and Third Hybrid Worlds

Challenger $\mathcal{C}$		**Reduction** $\mathcal{B}_2$		**Adversary** $\mathcal{A}$
$k_2 \leftarrow\$ \{0,1\}^n$		$f_0, f_1 \leftarrow\$ Func(\mathcal{X} \to \mathcal{Y})$		
$b \leftarrow\$ \{0,1\}$		$k_3 \leftarrow\$ \{0,1\}^n$		
if $b = 0$ then	$\xleftarrow{r_2\|\|10\|\|m_0}$	$\sigma_0 = f_0(r_0\|\|00\|\|m_0)$	$\xleftarrow{m}$	
$y = F(k_2, x)$		$\sigma_1 = f_1(r_1\|\|01\|\|m_1)$ $\sigma_3 = F(k_3, r_3\|\|11\|\|m_1)$		
else $y \leftarrow\$ \{0,1\}^n$	$\xrightarrow{y}$	$\sigma = r_0, \sigma_0, r_1, \sigma_1, r_2, y, r_3$	$\xrightarrow{\sigma}$	
	$\xleftarrow{x'}$	$x' = r'_2\|\|10\|\|m'_0$	$\xleftarrow{(m'\sigma')}$	
	$\xrightarrow{y'}$	if $y' = \sigma'_2$ and others match $b' = 0$		
	$\xleftarrow{b'}$	else $b' = 1$		

Similar to the proof above, in this case, the adversary receives **Hybrid World 2** if $b = 0$, else he receives **Hybrid World 3**. $\mathcal{A}$ can break Hybrid World 2 with non-negligible probability, so if the verification passes we assume it is Hybrid World 2 else it is Hybrid World 3.
Therefore $\mathsf{PRFAdv}_{\mathcal{B}_2} = \mathsf{MACAdv}_{\mathcal{A}}$.
Hence, the probability of adversary winning in **Hybrid World 2** is negligible.

Claim 1.4 — The probabilities of an adversary winning in **First Hybrid World** and **Second Hybrid World** are negligibly close.

If the probabilities are not negligibly close, the probability of the adversary winning in **Hybrid World 1** is non-negligible but the probability of winning in **Hybrid World 2** is negligible (proved above).
Hence, we can create a reduction $\mathcal{B}_1$, which can break the PRF security of F.

Difference in First and Second Hybrid Worlds

Challenger $\mathcal{C}$		**Reduction** $\mathcal{B}_1$		**Adversary** $\mathcal{A}$
$k_1 \leftarrow\$ \{0,1\}^n$		$f_0 \leftarrow\$ Func(\mathcal{X} \to \mathcal{Y})$		
$b \leftarrow\$ \{0,1\}$		$k_2, k_3 \leftarrow\$ \{0,1\}^n$		
if $b = 0$ then	$\xleftarrow{r_1\|\|01\|\|m_1}$	$\sigma_0 = f_0(r_0\|\|00\|\|m_0)$	$\xleftarrow{m}$	
$y = F(k_1, x)$		$\sigma_2 = F(k_2, r_2\|\|10\|\|m_0)$ $\sigma_3 = F(k_3, r_3\|\|11\|\|m_1)$		
else $y \leftarrow\$ \{0,1\}^n$	$\xrightarrow{y}$	$\sigma = r_0, \sigma_0, r_1, y, r_2, \sigma_2, r_3, \sigma_3$	$\xrightarrow{\sigma}$	
	$\xleftarrow{x'}$	$x' = r_1'\|\|01\|\|m_1'$	$\xleftarrow{(m'\sigma')}$	
	$\xrightarrow{y'}$	if $y' = \sigma_2'$ and others match $b' = 0$		
	$\xleftarrow{b'}$	else $b' = 1$		

Similar to the proof above, in this case, the adversary receives **Hybrid World 1** if $b = 0$, else he receives **Hybrid World 2**. $\mathcal{A}$ can break Hybrid World 1 with non-negligible probability, so if the verification passes we assume it is Hybrid World 1 else it is Hybrid World 2.
Therefore $\mathsf{PRFAdv}_{\mathcal{B}_1} = \mathsf{MACAdv}_{\mathcal{A}}$.
Hence, the probability of adversary winning in **Hybrid World 1** is negligible.

Claim 1.5 — The probabilities of an adversary winning in **World 0** and **First Hybrid World** are negligibly close.

If the probabilities are not negligibly close, the probability of the adversary winning in **World 0** is non-negligible but the probability of winning in **Hybrid World 1** is negligible (proved above).
Hence, we can create a reduction $\mathcal{B}_0$, which can break the PRF security of F.

Difference in World 0 and First Hybrid World

Challenger $\mathcal{C}$		**Reduction $\mathcal{B}_0$**		**Adversary $\mathcal{A}$**
$k_0 \leftarrow\$ \{0,1\}^n$				
$b \leftarrow\$ \{0,1\}$		$k_1, k_2, k_3 \leftarrow\$ \{0,1\}^n$		
if $b = 0$ then	$\xleftarrow{r_0 \| 00 \| m_0}$	$\sigma_1 = F(k_1, r_1 \| 01 \| m_1)$	$\xleftarrow{m}$	
$y = F(k_0, x)$		$\sigma_2 = F(k_2, r_2 \| 10 \| m_0)$ $\sigma_3 = F(k_3, r_3 \| 11 \| m_1)$		
else $y \leftarrow\$ \{0,1\}^n$	$\xrightarrow{y}$	$\sigma = r_0, y, r_1, \sigma_1, r_2, \sigma_2, r_3, \sigma_3$	$\xrightarrow{\sigma}$	
	$\xleftarrow{x'}$	$x' = r'_0 \| 00 \| m'_0$	$\xleftarrow{(m'\sigma')}$	
	$\xrightarrow{y'}$	if $y' = \sigma'_2$ and others match $b' = 0$		
	$\xleftarrow{b'}$	else $b' = 1$		

Similar to the proof above, in this case, the adversary receives **World 0** if $b = 0$, else he receives **Hybrid World 1**. $\mathcal{A}$ can break World 0 with non-negligible probability, so if the verification passes we assume it is World 0 else it is Hybrid World 1.
Therefore $\mathsf{PRFAdv}_{\mathcal{B}_0} = \mathsf{MACAdv}_{\mathcal{A}}$.
Hence, the probability of adversary winning in **World 0** is negligible.

This completes the proof that I_{uq} is secure, assuming F is secure.

§1.2.3 Subpart c)

We will do a simple mix and match attack where the adversary will send the same message query twice.

Attack to show not secure in UFCMA

Challenger $\mathcal{C}$		**Adversary $\mathcal{A}$**
$k \leftarrow\$ \{0,1\}^{4n}$		
	$\xleftarrow{m}$	
$r_1, r_2, r_3, r_4 \leftarrow\$ \mathrm{R}$		
	$\xrightarrow{\textbf{Sign(k,m)=:}\ (\mathrm{r}_1\, \sigma_1\, r_2\, \sigma_2\, r_3\, \sigma_3\, r_4\, \sigma_4)}$	
	$\xleftarrow{m}$	
$r'_1, r'_2, r'_3, r'_4 \leftarrow\$ \mathrm{R}$		
	$\xrightarrow{\textbf{Sign(k,m):}\ (\mathrm{r'}_1\, \sigma'_1\, r'_2\, \sigma'_2\, r'_3\, \sigma'_3\, r'_4\, \sigma'_4)}$	
		$\sigma = (\mathrm{r}_1\, \sigma_1\, r'_2\, \sigma'_2\, r_3\, \sigma_3\, r'_4\, \sigma'_4)$
	$\xleftarrow{\textbf{Forgery:}\ (\mathrm{m}\ ;\ \sigma\ )}$	

Probability of adversary winning is 1.

Adversary sends 2 queries on m $= (m_0, m_1)$ and the challenger will send
$Sign_{uq}(k, m) = r_1, F(k_0, r_1||00||m_0), r_2, F(k_1, r_2||01||m_1), r_3, F(k_2, r_3||10||m_0), r_4, F(k_3, r_4||11||m_1)$
and,https://www.overleaf.com/project/6513020f8ebbbfffa89d805f
$Sign_{uq}(k, m) = r'_1, F(k_0, r'_1||00||m_0), r'_2, F(k_1, r'_2||01||m_1), r'_3, F(k_2, r'_3||10||m_0), r'_4, F(k_3, r'_4||11||m_1)$
respectively.
Doing a mix and match attack we will have successfully forged a new signature for m,
$Forgery = r_1, F(k_0, r_1||00||m_0), r'_2, F(k_1, r'_2||01||m_1), r_3, F(k_2, r_3||10||m_0), r'_4, F(k_3, r'_4||11||m_1)$
Hence Pr[$\mathcal{A}$ wins] $= 1$.

§1.3 Question 3.

We can define an adversary $\mathcal{A}$ for the MAC game with verification queries such that he queries all possible signings σ for a message m. When he gets a 1 for a verification query he can just send it as a forgery.
The reduction $\mathcal{B}$ would not work because he sends the verification queries as signing queries and when he receives the forgery, he will have already sent it as a signing query, so now he cannot send the same forgery.
Here, $\mathcal{A}$ is not a polynomial time adversary because the key space is exponential in the security parameter, and finding the specific key will take exponential queries.

Algorithm 1.6 Adversary $\mathcal{A}$

Require: $Sign, \mathcal{K}, \mathcal{M}$
  $m \leftarrow\$ \mathcal{M}$
  for $k \leftarrow \mathcal{K}$ **do**
    $\sigma \leftarrow Sign(k, m)$
    $check \leftarrow VerifyQuery(m, \sigma)$
    if $check = 1$ **then**
      break
    end if
  end for
  Send Forgery(m, σ)

Pr[$\mathcal{A}$ *wins*] $= 1$ since $\mathcal{A}$ will eventually find the key k, which the challenger was using.

§1.4 Question 4.

Here, we exploit the fact that we know $xx^{-1} = 1 \mod p$
We query for three values of x, x_1, x_2 and x_3 and get three values of y, y_1, y_2 and y_3.
Now we know that if $b = 0$ then,

$$\begin{aligned} y &= (x + k_1)^{-1} + k_2 \mod p \\ (x + k1)(y - k_2) &= 1 \mod p \\ xy + yk_1 - xk_2 - k_1k_2 &= 1 \mod p \end{aligned}$$

Since we have queried for three values of x, and gotten three values of y, we get three equations. Subtracting any two gets us two equations linear in k_1 and k_2, by getting rid of that pesky k_1k_2 term.

$$\begin{aligned} x_2y_2 - x_1y_1 + (y_2 - y_1)k_1 - (x_2 - x_1)k_2 &= 0 \mod p \\ x_3y_3 - x_1y_1 + (y_3 - y_1)k_1 - (x_3 - x_1)k_2 &= 0 \mod p \end{aligned}$$

To make equations simpler, we replace them with better variables,

$$\begin{aligned} a_1 + a_2k_1 + a_3k_2 = 0 \mod p \\ b_1 + b_2k_1 + b_3k_2 = 0 \mod p \end{aligned}$$

Solving this system of equations we get,

$$\begin{aligned} k_1 = (a_3b_1 - a_1b_3)(a_2b_3 - a_3b_2)^{-1} \mod p \\ k_2 = (a_2b_1 - a_1b_2)(a_3b_2 - a_2b_3)^{-1} \mod p \end{aligned}$$

These values of k_1 and k_2 are correct if Pseudo-Random Permutation is used, but will be incorrect if a Truly Random Function is used. Hence, we just check if $(x + k_1)(y - k_2) = 1 \mod p$ for all three pairs of (x, y).

§1.5 Question 5

Attack for showing 3-Round Luby Rackoff is not a Strong Pseudorandom Function
This attack is based on the fact that we can reach $F(k_2, \cdot)$ from both sides and show they are equal
First, we rewrite $F(k_1, \cdot)$, $F(k_2, \cdot)$ and $F(k_3, \cdot)$ as F_1, F_2 and F_3 to make the notation simpler.

For input (x, y) the output of P is (u, v) where

$$\begin{aligned} u = x \oplus F_2(y \oplus F_1(x)) \\ v = y \oplus F_1(x) \oplus F_3(u) \end{aligned}$$

For input (x, y) the output of P^{-1} is (w, z) where

$$\begin{aligned} w = x \oplus F_2(y \oplus F_3(x)) \\ z = y \oplus F_3(x) \oplus F_1(w) \end{aligned}$$

We start by querying for $P^{-1}(0, 0)$ and we get (a, b)

$$\begin{aligned} a = F_2(F_3(0)) \\ b = F_3(0) \oplus F_1(a) \end{aligned}$$

Then we query for $P(a, 0)$ and get (c, d)

$$\begin{aligned} c = a \oplus F_2(F_1(a)) \\ d = F_1(a) \oplus F_3(c) \end{aligned}$$

Finally we query for $P^{-1}(c, b \oplus d)$ and we get (e, f)

$$\begin{aligned} b \oplus d &= F_3(0) \oplus F_3(c) \\ e &= c \oplus F_2(b \oplus d \oplus F_3(c)) \\ &= c \oplus F_2(F_3(0) \oplus F_3(c) \oplus F_3(c)) \\ &= c \oplus F_2(F_3(0)) \\ e &= c \oplus a \end{aligned}$$

Here, we reached a once from $F_2(F_3(0))$ and once by reversing. Now, we can just check if $e = c \oplus a$, because this will only be true if the 3-round Luby-Rackoff Permutation is used.
Hence, the 3-Round Luby-Rackoff Permutation is not a Strong Pseudo-Random Permutation

§1.6 Question 6

The attack depends on two important pieces of information,
1. The Key is used for both the Initialization Vector and in the PRP
2. CBC Mode is used for encrypting the block of ciphertext.

The Encryption Algorithm works as follows:

1. Break the message into blocks of 16 bytes. Here there will be three blocks, say m_0, m_1 and m_2.
2. Encrypt the blocks using CBC Mode and Initialization Vector as the Key. Let's say the ciphertext obtained is ct_0, ct_1 and ct_2.
3. $$\begin{aligned} ct_0 &= P_k(m_0 \oplus k) \\ ct_1 &= P_k(m_1 \oplus ct_0) \\ ct_2 &= P_k(m_2 \oplus ct_1) \end{aligned}$$

The Decryption Algorithm works as follows:

1. Break the cipher-text into blocks of 16 bytes. Here there will be three blocks, say ct_0, ct_1 and ct_2.
2. Decrypt the blocks using CBC Mode and Initialization Vector as the Key. Let's say the message obtained is m_0, m_1 and m_2.
3. $$\begin{aligned} m_2 &= P_k^{-1}(ct_2) \oplus ct_1 \\ m_1 &= P_k^{-1}(ct_1) \oplus ct_0 \\ m_0 &= P_k^{-1}(ct_0) \oplus k \end{aligned}$$

We can manipulate this to out advantage because we are allowed to tamper with the cipher-text. We send $ct_0||ct_1||ct_0$ as the cipher-text and get back $m_0'||m_1'||m_2'$.

$$\begin{aligned} m_0' &= P_k^{-1}(ct_0) \oplus k \\ m_1' &= P_k^{-1}(ct_1) \oplus ct_0 \\ m_2' &= P_k^{-1}(ct_0) \oplus ct_1 \end{aligned}$$

Now, we can compute k by doing the following

$$m_2' \oplus m_0' \oplus ct_1 = P_k^{-1}(ct_0) \oplus ct_1 \oplus P_k^{-1}(ct_0) \oplus k \oplus ct_1$$
$$m_2' \oplus m_0' \oplus ct_1 = k$$

Hence, this scheme is insecure and shows that the initialization vector should be chosen truly at random.

§2 Part B

§2.1 Padding Oracle Attack

There are two broad subtasks in this attack,

1. Find out the padding applied to the input message.
2. Using the padding figure out the bytes in the message one by one.

§2.1.1 The padding attack :

We manipulate a byte in the cipher text starting from the second byte till we get a valid padding response or we reach the 16th byte.
We change the respective byte by XORing it with 1.
The logic behind this is :-
Let the first 16 bytes be ct_0 and the next 16 bytes be ct_1 .
In decryption, the first block of message is calculated as

$$m_1 = P_k^{-1}(ct_1) \oplus ct_0$$

We are manipulating ct_0 to find the first byte in m_1 which on modification does not give padding error.
This byte will tell us the padding of m_1 .
So suppose the padding done to m_1 is 4 then first 4 bytes of m1 will be 4 followed by the data in m_1 .
When we change the fifth byte in ct_0 we will get a valid padding response which will let us know that the padding in ct_0 is 4.
If suppose we don't get a valid padding response till the 16^{th} byte we will know that the padding is 16.

§2.1.2 The message recovery attack:

Using the padding of the message we will figure out the byte on the right of the last padding byte.
Let the given padding be p .
We will change the padding of original message to $p+1$ by XORing first p bytes of ct_0 with $p+1 \oplus p$.
This will change the first p bytes of the corresponding decrypted message block from p to $p+1$.
Next we will try all possible values from 255 to 0 for the $p+1^{th}$ byte of this message block.
Suppose n is the number which gives a valid padding response.
So we know that the $p+1^{th}$ byte of decrypted message block is $p+1$.
Hence we know the corresponding byte in corresponding message block of original m is $p+1 \oplus n$.
Now using the updated cipher text which gives the valid padding response
i.e. we now treat m_1 as a message with $p+1$ padding , and repeating the above steps byte by byte we will have recovered the first block of the message.
After this, we discard the first block of cipher text and apply the same algorithm all over again just this time we know the padding of the second block of the message is trivially 0.
So for example, consider that the original message had two 16-byte blocks m_0 and m_1 and the corresponding cipher text had three 16-byte blocks ct_0 , ct_1 and ct_2 ;
We first use the padding attack to find out the padding done to m_1.
Then we use ct_0 as the cipher text we are manipulating to find out the bytes in m_1.
After that, we discard ct_0 and start manipulating ct_1 according to the message recovery attack to decipher m_2.
To do that we start with $p = 0$ and follow the above-mentioned till we get the 16-byte block.
Do this till no more blocks are left in the cipher text that is we have recovered the whole message.
Hence we have deciphered the original message! :)

§3 Acknowledgements

We have used the style file from here[1] to typeset and the style file from here[2] for cryptographic games and protocols to produce this document.

[1] https://github.com/vEnhance/dotfiles/blob/main/texmf/tex/latex/evan/evan.sty
[2] https://github.com/arnomi/cryptocode