! **Try again once you are ready**

TO PASS 80% or higher

Try again

GRADE
70%

Week 4 - Problem Set

LATEST SUBMISSION GRADE

70%

1. **An attacker intercepts the following ciphertext (hex encoded):**

1 / 1 point

20814804c1767293b99f1d9cab3bc3e7 ac1e37bfb15599e5f40eef805488281d

He knows that the plaintext is the ASCII encoding of the message "Pay Bob 100$" (excluding the quotes). He also knows that the cipher used is CBC encryption with a random IV using AES as the underlying block cipher.

Show that the attacker can change the ciphertext so that it will decrypt to "Pay Bob 500$". What is the resulting ciphertext (hex encoded)?

This shows that CBC provides no integrity.

20814804c1767293bd9f1d9cab3bc3e7 ac1e37bfb15599e5f40eef805488281d

✓ Correct
You got it!

2. **Let (E, D) be an encryption system with key space K, message space $\{0,1\}^n$ and ciphertext space $\{0,1\}^s$. Suppose (E, D) provides authenticated encryption. Which of the following systems provide authenticated encryption: (as usual, we use $\|$ to denote string concatenation)**

0 / 1 point

- [x] $E'(k,m) = E(k,m) \bigoplus 1^s$ and

  $D'(k,c) = D(k, c \bigoplus 1^s)$

✓ Correct
(E', D') provides authenticated encryption because an attack on (E', D') directly gives an attack on (E, D).

- [x] $E'(k,m) = E(k, m \bigoplus 1^n)$ and

  $$D'(k,c) = \begin{cases} D(k,c) \bigoplus 1^n & \text{if } D(k,c) \neq \perp \\ \perp & \text{otherwise} \end{cases}$$

✓ Correct
(E', D') provides authenticated encryption because an attack on (E', D') directly gives an attack on (E, D).

- [ ] $E'(k,m) = (E(k,m),\ 0)$ and

  $D'(k,\ (c,b)\ ) = D(k,c)$

- [x] $E'(k,m) = E(k,m)$ and

  $$D'(k,c) = \begin{cases} D(k,c) & \text{if } D(k,c) \neq \perp \\ 0^n & \text{otherwise} \end{cases}$$

! **This should not be selected**
This system does not provide ciphertext integrity since an attacker can simply output the ciphertext 0^s and win the ciphertext integrity game.

3. **If you need to build an application that needs to encrypt multiple messages using a single key, what encryption method should you use? (for now, we ignore the question of key generation and management)**

1 / 1 point

- ○ implement MAC-then-Encrypt yourself
- ◉ use a standard implementation of one of the authenticated encryption modes GCM, CCM, EAX or OCB.
- ○ implement Encrypt-and-MAC yourself
- ○ use a standard implementation of CBC encryption with a random IV.

✓ **Correct**

4. **Let (E, D) be a symmetric encryption system with message space M (think of M as only consisting for short messages, say 32 bytes).**

1 / 1 point

Define the following MAC (S, V) for messages in M:

$$S(k,m) := E(k,m) \quad ; \quad V(k,m,t) := \begin{cases} 1 & \text{if } D(k,t) = m \\ 0 & \text{otherwise} \end{cases}$$

What is the property that the encryption system (E, D) needs to satisfy for this MAC system to be secure?

- ◉ authenticated encryption
- ○ semantic security under a chosen plaintext attack
- ○ semantic security
- ○ chosen ciphertext security

✓ **Correct**
Indeed, authenticated encryption implies ciphertext integrity which prevents existential forgery under a chosen message attack.

5. **In [Key Derivation] we discussed how to derive session keys from a shared secret. The problem is what to do when the shared secret is non-uniform. In this question we show that using a PRF with a *non-uniform* key may result in non-uniform values. This shows that session keys cannot be derived by directly using a *non-uniform* secret as a key in a PRF. Instead, one has to use a key derivation function like HKDF.**

0 / 1 point

Suppose k is a *non-uniform* secret key sampled from the key space $\{0,1\}^{256}$.

In particular, k is sampled uniformly from the set of all keys whose most significant

128 bits are all 0. In other words, k is chosen uniformly from a small subset of the key space. More precisely,

for all $c \in \{0,1\}^{256}$: $\quad \Pr[k = c] = \begin{cases} 1/2^{128} & \text{if } \mathrm{MSB}_{128}(c) = 0^{128} \\ 0 & \text{otherwise} \end{cases}$

Let $F(k, x)$ be a secure PRF with input space $\{0,1\}^{256}$. Which

of the following is a secure PRF when the key k is uniform in the

key space $\{0,1\}^{256}$, but is insecure when the key is sampled from the *non-uniform*

distribution described above?

- ○ $F'(k,x) = \begin{cases} F(k,x) & \text{if } \mathrm{MSB}_{128}(k) \neq 0^{128} \\ 0^{256} & \text{otherwise} \end{cases}$
- ◉ $F'(k,x) = \begin{cases} F(k,x) & \text{if } \mathrm{MSB}_{128}(k) = 0^{128} \\ 0^{256} & \text{otherwise} \end{cases}$
- ○ $F'(k,x) = \begin{cases} F(k,x) & \text{if } \mathrm{MSB}_{128}(k) \neq 1^{128} \\ 0^{256} & \text{otherwise} \end{cases}$
- ○ $F'(k,x) = \begin{cases} F(k,x) & \text{if } \mathrm{MSB}_{128}(k) = 0^{128} \\ 1^{256} & \text{otherwise} \end{cases}$

> ! **Incorrect**
> This F' is trivially insecure as a PRF.

6. **In what settings is it acceptable to use *deterministic* authenticated** **1 / 1 point**

encryption (DAE) like SIV?

- ◉ when the encryption key is used to encrypt only one message.
- ○ when a fixed message is repeatedly encrypted using a single key.
- ○ to individually encrypt many packets in a voice conversation with a single key.
- ○ to encrypt many records in a database with a single key when the same record may repeat multiple times.

> ✓ **Correct**
> Deterministic encryption is safe to use when the message/key pair
>
> is never used more than once.

7. **Let $E(k, x)$ be a secure block cipher. Consider the following** **0 / 1 point**

tweakable block cipher:

$$E'\big((k_1,k_2),t,x\big) = \;\; E(k_1,x) \bigoplus E(k_2,t).$$

Is this tweakable block cipher secure?

- ○ no because for $t \neq t'$ we have

  $E'((k_1,k_2),t,0) \bigoplus E'((k_1,k_2),t,1) = E'((k_1,k_2),t',0) \bigoplus E'((k_1,k_2),t',1)$
- ◉ no because for $t \neq t'$ we have

  $E'((k_1,k_2),t,0) \bigoplus E'((k_1,k_2),t',1) = E'((k_1,k_2),t',1) \bigoplus E'((k_1,k_2),t',0)$
- ○ no because for $x \neq x'$ and $t \neq t'$ we have

  $E'((k_1,k_2),t,x) \bigoplus E'((k_1,k_2),t',x) = E'((k_1,k_2),t,x') \bigoplus E'((k_1,k_2),t',x)$
- ○ no because for $x \neq x'$ we have

  $E'((k_1,k_2),0,x) \bigoplus E'((k_1,k_2),0,x) = E'((k_1,k_2),0,x') \bigoplus E'((k_1,k_2),0,x')$
- ○ yes, it is secure assuming E is a secure block cipher.

> ! **Incorrect**
> This relation doesn't hold for E'

This relation doesn't hold for [illegible].

8. In [Format Preserving Encryption]() we discussed format preserving encryption which is a PRP on a domain $\{0, \ldots, s-1\}$ for some pre-specified value of s.

Recall that the construction we presented worked in two steps, where the second step worked by iterating the PRP until the output fell into the set $\{0, \ldots, s-1\}$.

Suppose we try to build a format preserving credit card encryption system from AES using *only* the second step. That is, we start with a PRP with domain $\{0,1\}^{128}$ from which we want to build a PRP with domain 10^{16}. If we only used step (2), how many iterations of AES would be needed in expectation for each evaluation of the PRP with domain 10^{16}?

1 / 1 point

- (•) $2^{128}/10^{16} \approx 3.4 \times 10^{22}$
- ( ) 2^{128}
- ( ) 4
- ( ) $10^{16}/2^{128}$

Correct

On every iteration we have a probability of $10^{16}/2^{128}$ of falling into the set $\{0, \ldots, 10^{16}\}$ and therefore in expectation we will need $2^{128}/10^{16}$ iterations. This should explain why step (1) is needed.

9. Let (E, D) be a secure tweakable block cipher.

1 / 1 point

Define the following MAC (S, V):

$$S(k,m) := E(k,m,0) \quad ; \quad V(k,m,\text{tag}) := \begin{cases} 1 & \text{if } E(k,m,0) = \text{tag} \\ 0 & \text{otherwise} \end{cases}$$

In other words, the message m is used as the tweak and the plaintext given to E is always set to 0.

Is this MAC secure?

- ( ) it depends on the tweakable block cipher.
- (•) yes
- ( ) no

Correct

A tweakable block cipher is indistinguishable from a collection of random permutations. The chosen message attack on the MAC gives the attacker the image of 0 under a number of the permutations in the family. But that tells the attacker nothing about the image of 0 under some other member of the family.

10. In [CBC Padding Attacks]() we discussed padding oracle attacks. These chosen-ciphertext attacks can break poor implementations of MAC-then-encrypt.

1 / 1 point

Consider a system that implements MAC-then-encrypt where encryption is done using CBC with a random IV using AES as the block cipher. Suppose the system is vulnerable to a padding oracle

attack. An attacker intercepts a 64-byte ciphertext c (the first 16 bytes of c are the IV and the remaining 48 bytes are the encrypted payload). How many chosen ciphertext queries would the attacker need *in the worst case* in order to decrypt the entire 48 byte payload? Recall that padding oracle attacks decrypt the payload one byte at a time.

- ○ 48
- ○ 1024
- ○ 256
- ◉ 12288
- ○ 16384

Correct

Correct. Padding oracle attacks decrypt the payload one byte at a time. For each byte the attacker needs no more than 256 guesses in the worst case. Since there are 48 bytes total, the number queries needed is $256 \times 48 = 12288$.