CS255: Intro. to Cryptography **Winter 2025**

Assignment #3

Due: 11:59pm on Thu., Feb. 20, 2025, on Gradescope (each answer on a new page).

Problem 1. (*One-time MAC*) Recall that the one-time pad (OTP) is a semantically secure cipher that is unconditionally secure (that is, we can prove it secure without making any assumptions). In this question we build a one-time MAC that is unconditionally secure. A *one-time MAC* is a MAC that is secure against an adversary that makes at most a *single* chosen message query. The adversary chooses a message $m \in \mathcal{M}$; issues a chosen message query for m and gets back a tag t for m; and then wins the MAC game if it can output a valid message-tag pair (m^*, t^*) where $(m^*, t^*) \neq (m, t)$. The MAC is one-time unconditionally secure if no adversary can win this game with probability better than $1/|\mathcal{T}|$.

Let p be a prime and let $\mathcal{M} := \mathbb{Z}_p$, $\mathcal{K} := (\mathbb{Z}_p)^2$, and $\mathcal{T} := \mathbb{Z}_p$. Consider the following MAC (S, V) defined over $(\mathcal{M}, \mathcal{K}, \mathcal{T})$:

$$S\big((k_1, k_2), m\big) := k_1 m + k_2 \quad \text{and} \quad V\big((k_1, k_2), m, t\big) := \Big\{ \text{ accept if } t = k_1 m + k_2 \Big\}$$

Here additions and multiplications are defined in $\mathbb{Z}_p$. It is not difficult to show that (S, V) is an unconditionally secure one-time MAC (while it is not part of the homework problem, you can try to prove this for yourself). Your goal for this problem is to show that (S, V) is not two-time secure. That is, describe an adversary that can forge the MAC on some third message after issuing two chosen message queries.

Answer: The adversary issues two different chosen messages, $m_1 \in \mathbb{Z}_p^*$ and $m_2 \in \mathbb{Z}_p^*$, then get $t_1 = k_1 m_1 + k_2$ and $t_2 = k_1 m_2 + k_2$, where $t_1, t_2 \in \mathbb{Z}_p$.
From here $t_1 m_2 - t_2 m_1 = k_1 m_1 m_2 + k_2 m_2 - k_1 m_1 m_2 - k_2 m_1 = (m_2 - m_1) k_2$
Given $m1 \neq m2$ and p is a prime, the $GCD((m_2 - m_1), p) = 1$, meaning $(m_2 - m_1)$ has an inverse $(m_2 - m_1)^{-1}$
So we can compute $k2 = (t_1 m_2 - t_2 m_1)((m_2 - m_1)^{-1})$. In this equation, t_1, m_2, t_2, m_1 are all known by adversary, so k_2 is known by adversary.
Moreover, from $t_1 = k_1 m_1 + k_2$ and $m_1 \in \mathbb{Z}_p^*$, adversary can compute $k_1 = (t_1 - k_2) m_1^{-1}$.
After getting k_1, k_2, adversary can construct an existential forgery $\forall m_3 \in \mathbb{Z}_p$, and $m_3 \neq m_1, m_3 \neq m_2$, compute its MAC $t_3 = k_1 m_3 + k_2$, and $V\big((k_1, k_2), m_3, t_3\big) = \text{accept}$, meaning the adversary has an advantage of 1 in the security game

Problem 2. (*Multicast MACs*) Suppose user A wants to broadcast a message to n recipients $B_1, \dots, B_n$. Privacy is not important but integrity is: each of $B_1, \dots, B_n$ should be assured that the message it received was sent by A. User A decides to use a MAC.

a. Suppose user A and $B_1, \dots, B_n$ all share a secret key k. User A computes the tag for every message she sends using k. Every user B_i verifies the tag using k. Using at most two sentences explain why this scheme is insecure, namely, show that user B_1 is not assured that the messages it received are from A.

> **Answer:** $B_2, \dots, B_n$ also knows the secret key k, thus can construct existential forgery of the MAC of any messages. B_1 always accepts those (msg, tag) pair even it is not from A

b. Suppose user A has a set $S = \{k_1, \dots, k_\ell\}$ of ℓ secret keys. Each user B_i has some subset $S_i \subseteq S$ of the keys. When A transmits a message she appends ℓ tags to it by MACing the message with each of her ℓ keys. When user B_i receives a message it accepts the message as valid only if all tags corresponding to keys in S_i are valid. Let us assume that the users $B_1, \dots, B_n$ do not collude with each other. What property should the sets $S_1, \dots, S_n$ satisfy so that the attack from part (a) does not apply?

> **Answer:** We want to achieve $\forall B_i$, it cannot construct existential forgery of A for any other user B_j. It requires two property
> 1) $\forall i, S_i \subsetneq S$
> 2)$\forall i, j (i \neq j), S_i \nsubseteq S_j$
> Proof:
> 1) user A generates all tags $\forall i = 1, 2, \cdots, l$, which is a super set of needed tags, meaning user $\forall i = 1, 2, \cdots, l$, user B_i will accepts the message.
> 2) For a certain user B_i, none of other user key set is a subset of S_i, meaning there is at least one tag missing if B_i wants to construct existential forgery of A for any other user B_j. So its message will be accepted with negligible probability
> 3) From 1) and 2), only A can broadcast messages, and messages from any user B_i will be accepted with negligible probability by any other users.

c. Show that when $n = 10$ (i.e. ten recipients) it suffices to take $\ell = 5$ in part (b). Describe the sets $S_1, \dots, S_{10} \subseteq \{k_1, \dots, k_5\}$ you would use.

> **Answer:** $s_1 = \{k_1, k_2\}$ $s_2 = \{k_1, k_3\}$ $s_3 = \{k_1, k_4\}$ $s_4 = \{k_1, k_5\}$ $s_5 = \{k_2, k_3\}$ $s_6 = \{k_2, k_4\}$ $s_7 = \{k_2, k_5\}$ $s_8 = \{k_3, k_4\}$ $s_9 = \{k_3, k_5\}$ $s_{10} = \{k_4, k_5\}$

d. Show that the scheme from part (c) is insecure if two users are allowed to collude.

Answer: I will show an example how B_1 and B_8 collude and then make message accepted by B_2.
Step1: B_1 with $s_1 = \{k_1, k_2\}$ sends message m with its tag $m||t_1||t_2$ to B_8
Step2: B_8 with $s_8 = \{k_3, k_4\}$ accept message from B_1 although t_3 and t_4 are missing. It then appends t_3 and t_4 to the original message $m||t_1||t_2||t_3||t_4$ and sends it to B_2
Step3: B_3 with $s_3 = \{k_1, k_4\}$ will check t_1 and t_4. t_1 is valid because B_1 has k_1, t_4 is valid because B_8 has k_4. Then B_3 accepted the message with probability of 1, although the message is not from A.

Problem 3. (*Parallel Merkle-Damgård*) Recall that the Merkle-Damgård construction gives a *sequential* method for extending the domain of a CRHF. The tree construction in the figure below is a parallelizable approach: all the hash functions h within a single level can be computed in parallel. Prove that the resulting hash function defined over $(\mathcal{X}^{\leq L},\ \mathcal{X})$ is collision resistant, assuming h is collision resistant. Here h is a compression function $h : \mathcal{X}^2 \to \mathcal{X}$, and we assume the message length can be encoded as an element of $\mathcal{X}$.

CS255_hw3_tree.png

More precisely, the hash function is defined as follows:

input: $m_1 \ldots m_s \in \mathcal{X}^s$ for some $1 \leq s \leq L$
output: $y \in \mathcal{X}$

let $t \in \mathbb{Z}$ be the smallest power of two such that $t \geq s$ (i.e., $t := 2^{\lceil \log_2 s \rceil}$)
for $i = s+1$ to t: $m_i \leftarrow \perp$
for $i = t+1$ to $2t-1$:
- $\ell \leftarrow 2(i-t)-1, \quad r \leftarrow \ell + 1$ // *indices of left and right children*
- if $m_\ell = \perp$ and $m_r = \perp$: $m_i \leftarrow \perp$ // *if node has no children, set node to null*
- else if $m_r = \perp$: $m_i \leftarrow m_\ell$ // *if one child, propagate child as is*
- else $m_i \leftarrow h(m_\ell, m_r)$ // *if two children, hash with h*

output $y \leftarrow h\big(m_{2t-1},\ s\big)$ // *hash final output and message length*

Answer: Basically we want to prove the two Merkle trees are identical given
1)h is collision resistant
2) root is identical.
Proof sketch: we can prove the level $l+1$ of merkle trees are identical if level l of merkle trees are. Given root is identical, by induction we can prove the whole trees are identical, using collision resistant property of the h.

A more detailed proof by contrapositive is as follows:
Assume the resulting hash function is not collision resistant while h is collision resistant, there exists an adversary A that can find messages $M \neq M'$ that hash to the same value y.
Step1: Given $y \leftarrow h\big(m_{2t-1},\ s\big) = h\big(m'_{2t-1},\ s'\big)$, if $m'_{2t-1} \neq m_{2t-1}$ or $s \neq s'$, then we find a collision of h and finish the proof. Otherwise $m'_{2t-1} = m_{2t-1}$ and $s = s'$
Step2: We can recursively apply the argument in step1 on node m_{2t-1} and all descendants. At each level, we either find a collision of h and finish the proof or descend the same argument to the next level
Step3: At leave level, here we can conclude all corresponding leaves are identical, and as a consequence, $M = M'$ which contradicts the assumption we made at the very beginning.

Here we finish the proof, h must be collision resistant

Problem 4. (*Davies-Meyer*) In the lecture we saw that Davies-Meyer is used to convert an ideal block cipher into a collision resistant compression function. Let $E(k, m)$ be a block cipher where the message space is the same as the key space (e.g. 128-bit AES). Show that the following methods do not work:

$$f_1(x, y) = E(y, x) \oplus y \quad \text{and} \quad f_2(x, y) = E(x,\ x \oplus y)$$

That is, show an efficient algorithm for constructing collisions for f_1 and f_2. Recall that the block cipher E and the corresponding decryption algorithm D are both known to you.

Answer: For f_1, choose a random $y' \neq y$ and construct $x' = D(y', y' \oplus E(y, x) \oplus y)$
From this construction, $E(y', x') = y' \oplus E(y, x) \oplus y$, then $E(y', x') \oplus y' = E(y, x) \oplus y$
Here we found a collision $f_1(x, y) = f_1(x', y')$ and $(x, y) \neq (x', y')$
Moreover, construction of x' only involves $\oplus$ and D, so this is an efficient algorithm.

For f_2, choose a random $x' \neq x$ and construct $y' = D(x', E(x,\ x \oplus y)) \oplus x'$
From this construction,$E(x,\ x \oplus y) = E(x',\ x' \oplus y')$
Here we found a collision $f_2(x, y) = f_2(x', y')$ and $(x, y) \neq (x', y')$
Moreover, construction of y' only involves $\oplus$ and D, so this is an efficient algorithm.

Problem 5. (*Authenticated encryption*) Let (E, D) be an encryption system that provides authenticated encryption. Here E does not take a nonce as input and therefore must be a randomized encryption algorithm. Which of the following systems provide authenticated encryption? For those that do, give a short proof. For those that do not, present an attack that either breaks CPA security or ciphertext integrity.

a. $E_1(k, m) = \big[c \leftarrow E(k, m),\ \text{output}\ \ (c, c)\ \big]$ and $D_1(k,\ (c_1, c_2)\ ) = D(k, c_1)$

Answer: This is not an authenticated encryption system. An attack can be described as follows:
step1: adversary chooses a random message m and send it to challenger
step2: challenger returns adversary (c, c)
step3: adversary chooses a $c' \neq c$, then sends (c, c') to challenger
step4: challenger decrypt $D_1(k,\ (c, c')\ ) = D(k, E(k, m))$, given (E, D) is authenticated encryption system, $D_1(k,\ (c, c')\ ) = m$

Here we describe an efficient way to break cipher text integrity since $(c, c') \neq (c, c)$ and it successfully decrypts.

b. $E_2(k, m) = \big[c \leftarrow E(k, m),\ \text{output}\ \ (c, c)\ \big]$ and $D_2(k,\ (c_1, c_2)\ ) = \begin{cases} D(k, c_1) & \text{if } c_1 = c_2 \\ \mathsf{fail} & \text{otherwise} \end{cases}$

Answer: This provides authenticated encryption. We need to prove (E_2, D_2) has ciphertext integrity and CPA security.

Proof of ciphertext integrity as follows:
Step1: Assume there exists an attacker that constructs $(c_1, c_2) \neq (c, c)$, and D_2 doesn't reject
Step2: If $c_1 \neq c_2$, then D_2 will fail, which contradicts with our assumption. So $c_1 = c_2$, $D_2 = D(k, c_1)$
Step3: If $D(k, c_1)$ rejects, then contradicts with our assumption. If $D(k, c_1)$ accepts, then attacker successfully constructs a valid cipher text, then this violates (E, D) is authenticated encryption
Here we prove that (E_2, D_2) has ciphertext integrity

Proof of CPA security as follows:
Assume (E_2, D_2) doesn't provide CPA security, then there exists an adversary A that has non negligible advantage when attacking. We can construct adversary B that attacks CPA security of (E, D)
Step1: B receive messages m_0, m_1 from A then submit query (m_0, m_1) to the challenge
Step2: B receive $c = E(k, m_b)$, then B supplies A with (c, c) and get back b^*
Step3: B output A's guess b^*. Since the process of A is equivalent, and A has non negligible advantage over (E_2, D_2), then B at least has non negligible advantage over (E, D). This contradict (E, D) has CPA security.
Here we prove that (E_2, D_2) also has CPA integrity

Base on the above, (E_2, D_2) has both CPA security and ciphertext integrity, so it is an authenticated encryption system.

c. $E_3(k, m) = \big(\ E(k, m),\ E(k, m)\ \big)$ and $D_3(k,\ (c_1, c_2)\ ) = \begin{cases} D(k, c_1) & \text{if } D(k, c_1) = D(k, c_2) \\ \text{fail} & \text{otherwise} \end{cases}$

To clarify: $E(k, m)$ is randomized so that running it twice on the same input will result in different outputs with high probability.

Answer: This is not an authenticated encryption system. An permutation attack can be described as follows:
step1: adversary chooses a random message m and send it to challenger
step2: challenger returns adversary $(c_1 = E(k,m), c_2 = E(k,m))$
step3: adversary sends $(c_2 = E(k,m), c_1 = E(k,m))$ to challenger
step4: challenger decrypt $D_3(k,\ (c_2, c_1)\ ) = D(k, E(k,m)) = m$, given (E, D) is authenticated encryption system so that $D(k, c_1) = D(k, c_2) = m$

Here we describe an efficient way to break cipher text integrity of (E_3, D_3)
This is just one example, actually either c_1, or c_2 can be any $E(k,m)$ and the attack still works.

d. $E_4(k,m) = \big(\ E(k,m),\ H(m)\ \big)$ and $D_4(k,\ (c_1,c_2)\ ) = \begin{cases} D(k,c_1) & \text{if } H(D(k,c_1)) = c_2 \\ \mathsf{fail} & \text{otherwise} \end{cases}$
where H is a collision resistant hash function.

Answer: This provides authenticated encryption. We need to prove (E_4, D_4) has ciphertext integrity and CPA security.

Proof of ciphertext integrity as follows:
Step1: Assume there exists an attacker that constructs $(c_1, c_2) \neq (c, c)$, and D_2 doesn't reject
Step2: If $D(k, c_1)$ rejects, then $H(D(k, c_1)) \neq c_2$, D_4 fails. This contradicts with our assumption. If $D(k, c_1)$ accepts, then attacker successfully constructs a valid cipher text, then this violates (E, D) is authenticated encryption
Here we prove that (E_4, D_4) has ciphertext integrity

Proof of CPA security as follows:
Assume (E_4, D_4) doesn't provide CPA security, then there exists an adversary A that has non negligible advantage when attacking. We can construct adversary B that attacks CPA security of (E, D)
Step1: B receive messages m_0, m_1 from A then submit query (m_0, m_1) to the challenge E
Step2: B receive $c = E(k, m_b)$, then B supplies A with 2 pairs $(c, H(m_0))$ and $(c, H(m_1))$
step3: A has non negligible advantage over one and only one query from $(c, H(m_0))$ and $(c, H(m_1))$ is meaningful to A. Suppose A output $b_{non-neg}$ and b_{random}
step4: B either output $b_{non-neg}$ or b_{random} from A with probability 50/50. Since the process of A of guessing $b_{non-neg}$ from one of the B's query is equivalent, and A has non negligible advantage over (E_4, D_4), then B also has non negligible advantage over (E, D), because half of non negligible advantage is still non negligible, this contradict (E, D) has CPA security.
Here we prove that (E_4, D_4) also has CPA integrity

Base on the above, (E_4, D_4) has both CPA security and ciphertext integrity, so it is an authenticated encryption system.
Base on the above, we conclude $(k', m'^{)} = (k, m)$. This proves that (E_4, D_4) provides authenticated encryption.

Problem 6. Alice and Bob run the Diffie-Hellman protocol in the cyclic group $\mathbb{G} = \mathbb{Z}^*_{101}$ with generator $g = 11$. What is the Diffie-Hellman secret $s = g^{ab} \in \mathbb{G}$ if Alice uses $a = 7$ and Bob uses $b = 43$? You do not need a calculator to solve this problem!

Answer: Given none of the 2,3,4,5,6,7,8,9,10,11 can divide 101, p = 101 is a prime number. So $g^{(101-1)} = g^{100} = 1$ mod p
$s = g^{ab} = g^{7 \cdot 43} = g^{301} = g \cdot (g^{100})^3 = g$ mod p, so the Diffi-Hellman secret s = g = 11