1 Summary of last lecture, and plan for today's lecture

- Last lecture, we introduced hash functions (keyed functions that compress the input). There are two different security notions that can be considered here.
  - universal hash functions: here the adversary does not get the hash key. It must output two messages m_0, m_1 such that $H_k(m_0) = H_k(m_1)$
  - collision resistant hash functions: here the adversary gets the hash key, and must output two messages that map to the same output.
- Constructing UHFs is easy — these can be constructed without any cryptographic assumptions. Any PRF/MAC will also imply a UHF family.
- Constructing CRHFs is more challenging. We saw a few attempts that didn't work.

Plan for today:

- First, we will see an application of UHFs: for constructing a *specific* MAC scheme that can handle unbounded length messages.
- If we use a CRHF instead of a UHF, then *any* MAC scheme can be converted to one that handles unbounded length messages.
- We will then discuss how to extend the message space of a CRHF.
- Finally, we will talk about some practical CRHFs, and how to make sense of the 'keyless' hash functions

2 UHF + PRF $\implies$ MAC with unbounded message space

Last lecture, we saw that a UHF can be constructed unconditionally, and it handles unbounded length messages. The UHF family is parameterized by an n-bit prime p (we can assume this is known to everyone, including the adversary). The key space is $\mathbb{Z}_p^*$. The message is interpreted as a string of numbers in $\mathbb{Z}_p$ (that is, $m = (m_0, m_1, \ldots, m_{d-1})$ where each $m_i \in \mathbb{Z}_p$). The hash of m using a key $k \in \mathbb{Z}_p^*$ is

$$H_k(m) = (m_0 + m_1 \cdot k + \ldots + m_{d-1} \cdot k^{d-1} + k^d) \mod p.$$

Qn: What happens if we set $H'_k(m) = m_0 + m_1 \cdot k + \ldots + m_{d-1} \cdot k^{d-1}$?
Ans: Since the message space is $(\mathbb{Z}_p)^*$ (that is, any message is a string of numbers in $\mathbb{Z}_p$), the following is a collision for H'_k: $m = m_0$, $m' = (m_0, 0)$. However, if the message space was $(\mathbb{Z}_p^*)^*$ (that is, any message is a string of elements in $\mathbb{Z}_p^*$), then H'_k is also a secure UHF.

A few lectures back, we also discussed how to construct a strongly unforgeable MAC, using just a PRF $F : \mathcal{X} \times \mathcal{K} \to \mathcal{Y}$. The message space for the MAC was $\mathcal{X}$, the key space was $\mathcal{K}$ and the signature space was $\mathcal{Y}$. Recall, the signing algorithm simply outputs $F(m, k)$ as a signature on m.

A very natural idea therefore is to combine these two primitives, and check if this gives us a strongly unforgeable MAC with unbounded message space. Let $\mathcal{H} = \{H_k\}_{k \in \mathcal{K}_{\text{uhf}}}$ be a secure UHF with unbounded message space, key space $\mathcal{K}_{\text{uhf}}$ and output space $\mathcal{X}$, and $F : \mathcal{X} \times \mathcal{K}_{\text{prf}} \to \mathcal{Y}$ a secure PRF. The MAC scheme is described below.

A strongly unforgeable MAC with unbounded message space

The key space for the MAC scheme is $\mathcal{K}_{\text{uhf}} \times \mathcal{K}_{\text{prf}}$, and the signature space is $\mathcal{Y}$.

- $\mathsf{Sign}(m, (k_1, k_2))$: Compute $x = H_{k_1}(m)$, and output $\sigma = F(x, k_2)$.
- $\mathsf{Verify}(m, \sigma, (k_1, k_2))$: Compute $x = H_{k_1}(m)$, and output 1 iff $\sigma = F(x, k_2)$.

Theorem 22.01. Assuming $\mathcal{H}$ is a secure UHF and F is a secure PRF, the above construction is a secure MAC scheme.

Proof. The proof will go through a sequence of hybrid games, where we alter the winning condition slightly in each game, until we reach a point where the adversary has little/no chance of winning. Each modification will rely on one of the building blocks that we are using.

- Game 0: This corresponds to the strong unforgeability game.
  - (Setup) Challenger chooses $k_1 \leftarrow \mathcal{K}_{\text{uhf}}$ and $k_2 \leftarrow \mathcal{K}_{\text{prf}}$.
  - (Signing queries) The adversary queries for polynomially many signature queries. Since this is a MAC scheme with deterministic signing, we assume that all signing queries are distinct. For each query m_i, the challenger computes $x_i = H_{k_1}(x_i)$ and outputs $\sigma_i = F(x_i, k_2)$.
  - (Forgery) After polynomially many signature queries, the adversary sends (m^*, σ^*) and wins if $(m^*, \sigma^*) \neq (m_i, \sigma_i)$ for all i, and $\sigma^* = F(H_{k_1}(m^*), k_2)$. Since every message has a unique valid signature, the adversary wins only if $m^* \neq m_i$ for all i and $\sigma^* = F(H_{k_1}(m^*), k_2)$.
- Game 1: In this game, the challenger uses a truly random function instead of a PRF.
  - (Setup) Challenger chooses $k_1 \leftarrow \mathcal{K}_{\text{uhf}}$ and $f \leftarrow \mathsf{Func}[\mathcal{X}, \mathcal{Y}]$.
  - (Signing queries) The adversary queries for polynomially many signature queries. For each query m_i, the challenger computes $x_i = H_{k_1}(x_i)$ and outputs $\sigma_i = f(x_i)$.
  - (Forgery) After polynomially many signature queries, the adversary sends (m^*, σ^*) and wins if $m^* \neq m_i$ for all i, and $\sigma^* = f(H_{k_1}(m^*))$.
- Game 2: This game is identical to the previous one. Instead of choosing a uniformly random function, the challenger defines the random function 'on-the-fly'.
  - (Setup) Challenger chooses $k_1 \leftarrow \mathcal{K}_{\text{uhf}}$ and maintains a table $\mathcal{T}$, which is initially empty.
  - (Signing queries) The adversary queries for polynomially many signature queries. For each query m_i, the challenger computes $x_i = H_{k_1}(x_i)$.
    It checks if there exists a pair corresponding to x_i in $\mathcal{T}$. If $(x_i, y_i) \in \mathcal{T}$, it sends y_i to the adversary. Else it samples a fresh $y_i \leftarrow \mathcal{Y}$, adds (x_i, y_i) to $\mathcal{T}$ and sends y_i to $\mathcal{A}$.
  - (Forgery) After polynomially many signature queries, the adversary sends (m^*, σ^*) and wins if $m^* \neq m_i$ for all i, and
    - either $(x^*, y^*) \in \mathcal{T}$ where $x^* = H_{k_1}(m^*)$ and $\sigma^* = y^*$
    - or $\sigma^* = y^*$ for a uniformly random $y^* \leftarrow \mathcal{Y}$
- Game 3: In this game, the challenger does not maintain a table $\mathcal{T}$. Instead, it sends a uniformly random y_i for each query. In the final forgery check, it chooses a uniformly random y^*.
  - (Setup) Challenger chooses $k_1 \leftarrow \mathcal{K}_{\text{uhf}}$.
  - (Signing queries) The adversary queries for polynomially many signature queries. For each query m_i, the challenger computes $x_i = H_{k_1}(x_i)$.
    It samples a fresh $y_i \leftarrow \mathcal{Y}$, and sends y_i to $\mathcal{A}$.
  - (Forgery) After polynomially many signature queries, the adversary sends (m^*, σ^*) and wins if $m^* \neq m_i$ for all i, and $\sigma^* = y^*$ for a uniformly random $y^* \leftarrow \mathcal{Y}$.

Analysis: Let p_i denote the winning probability of $\mathcal{A}$ in Game i. From the description of the games, it is clear that the adversary's winning probability in Game 3 is $1/|\mathcal{Y}|$, which is negligible in the security parameter. However, we need to show that the probabilities in the other games are also negligible. First, note that $p_0 \approx p_1$, using the PRF security. Next, note that $p_1 = p_2$ (Game 1 and Game 2 are identical; in one case the entire random function is chosen upfront, while in the other case it is chosen 'on-the-fly'). Therefore, to complete the argument, we need to show that p_2 is negligible if p_3 is negligible.

Claim 22.02. For any p.p.t. adversary $\mathcal{A}$ that makes Q signing queries,

$$(p_2 - p_3) \leq (Q+1)^2 \cdot (\text{maximum winning probability in the UHF game}).$$

Proof. If we use a information-theoretically secure UHF, then this claim holds for all adversaries (not just computationally bounded ones). For notational convenience, we will set $m^* = m_{Q+1}$, $x_{Q+1} = H_{k_1}(m^*)$. Let Coll denote the event that there exist two distinct indices $i, j \in [Q+1]$ such that $x_i = x_j$.

$$p_2 = \Pr\left[\mathcal{A} \text{ wins in Game 2 } \wedge \neg\mathsf{Coll}\right] + \Pr\left[\mathcal{A} \text{ wins in Game 2 } \wedge \mathsf{Coll}\right]$$
$$p_3 = \Pr\left[\mathcal{A} \text{ wins in Game 3 } \wedge \neg\mathsf{Coll}\right] + \Pr\left[\mathcal{A} \text{ wins in Game 3 } \wedge \mathsf{Coll}\right]$$

Note that if there is no collision, then Game 2 and Game 3 are identical. Hence,

$$\begin{aligned} p_2 - p_3 &= \Pr\left[\mathcal{A} \text{ wins in Game 2 } \wedge \mathsf{Coll}\right] - \Pr\left[\mathcal{A} \text{ wins in Game 3 } \wedge \mathsf{Coll}\right] \\ &\leq \Pr\left[\mathsf{Coll}\right] \end{aligned}$$

Suppose $p_2 - p_3$ is non-negligible. Then $\mathcal{A}$ produces a collision with non-negligible probability, and we can construct a reduction algorithm $\mathcal{B}$ that breaks UHF security as follows. $\mathcal{B}$ simply interacts with $\mathcal{A}$ as in Game 3, collects all the $Q+1$ queries $\{m_1, \ldots, m_{Q+1}\}$. Next, it picks a uniformly random pair of messages from this set and sends them to the UHF challenger. The reduction algorithm wins the UHF game with probability $\Pr\left[\mathsf{Coll}\right] / \binom{Q+1}{2}$, which is non-negligible if $\Pr\left[\mathsf{Coll}\right]$ is non-negligible.

□

Putting everything together: suppose any p.p.t. adversary has advantage at most μ_{prf} in the PRF security game, and any p.p.t. adversary has advantage at most μ_{uhf} in the UHF security game. Then, any p.p.t. adversary making at most Q queries has at most $\mu_{\text{prf}} + O(Q^2) \cdot \mu_{\text{uhf}}$ winning probability in the MAC forgery game. □

2.1 Post-proof discussion

Implications of the above proof There are many real-world MAC schemes whose security can be proven using the above result. For instance, consider the 'encrypted CBC-MAC' scheme discussed in Lecture 20. The signing algorithm is described below.

Encrypted CBC-MAC

The scheme uses a PRF $F : \mathcal{X} \times \mathcal{K} \to \mathcal{X}$, and can handle unbounded length messages. Let $m = (m_1, \ldots, m_\ell)$ be the message to be signed. The secret key consists of two PRF keys k_1, k_2.

- $\mathsf{Sign}(m, (k_1, k_2))$: Let $y_1 = F(m_1, k_1)$. For $i \in [2, \ell]$, the signing algorithm computes $y_i = F(m_i \oplus y_{i-1}, k_1)$. Finally, it outputs $\sigma = F(y_\ell, k_2)$.

Proving this to be a secure MAC might look intimidating, but note that we can use Theorem 22.01 here. In the above construction, the computation using key k_1 can be seen as a UHF, and therefore ECBC-MAC is just a special case of the 'hash-and-sign' approach in disguise. In Assignment 4, you will prove that the following is a UHF:

UHF hiding inside ECBC-MAC construction

Let $F : \mathcal{X} \times \mathcal{K} \to \mathcal{X}$ be a secure PRF. Consider the hash function family $\mathcal{H} = \{H_k : \mathcal{X}^{\geq 1} \to \mathcal{X}\}_{k \in \mathcal{K}}$ defined as follows:

- $H_k(m = (m_1, \ldots, m_\ell))$: Set $y_1 = F(m_1, k)$. For $i \in [2, \ell]$, compute $y_i = F(m_i \oplus y_{i-1}, k)$. Output y_ℓ.

Another popular MAC scheme used in practice is the 'nested-MAC' scheme (NMAC). This also uses a PRF F whose key space is equal to the output space.[1] The scheme is defined as follows.

NMAC

The scheme uses a PRF $F : \mathcal{X} \times \mathcal{K} \to \mathcal{K}$, and can handle unbounded length messages. Let $m = (m_1, \ldots, m_\ell)$ be the message to be signed. The secret key consists of two PRF keys k_1, k_2.

- $\mathsf{Sign}(m, (k_1, k_2))$: Let $y_1 = F(m_1, k_1)$. For $i \in [2, \ell]$, the signing algorithm computes $y_i = F(m_i, y_{i-1})$. Finally, it outputs $\sigma = F(y_\ell, k_2)$.

This MAC scheme also looks hard to analyse at first. However, this is also an instantiation of the 'hash-and-sign' paradigm, and it suffices to show that the following construction is a UHF.

UHF hiding inside NMAC construction

Let $F : \mathcal{X} \times \mathcal{K} \to \mathcal{K}$ be a secure PRF. Consider the hash function family $\mathcal{H} = \{H_k : \mathcal{X}^{\geq 1} \to \mathcal{K}\}_{k \in \mathcal{K}}$ defined as follows:

- $H_k(m = (m_1, \ldots, m_\ell))$: Set $y_1 = F(m_1, k)$. For $i \in [2, \ell]$, compute $y_i = F(m_i, y_{i-1})$. Output y_ℓ.

Reusing the same key for UHF and PRF Suppose $\mathcal{K}_{\text{uhf}} = \mathcal{K}_{\text{prf}}$. Can we use the same key for both the UHF and PRF? This may not be a good idea. Consider, for instance, the ECBC-MAC construction. As noted above, this can be seen as a UHF + PRF construction. We know that if the same key is used in ECBC-MAC, then it is not a secure MAC scheme (see Question 22.01).

Does the above transformation work with any MAC scheme? A natural question to ask is whether the above transformation works with any MAC scheme that handles bounded length messages. That is, consider a scheme $\mathsf{MAC}_{\text{bdd}}$ that handles n-bit messages, and a UHF $\mathcal{H}$ that takes as input unbounded-length messages, and outputs n-bit messages. Does the 'hash-and-sign' approach combining $\mathcal{H}$ and $\mathsf{MAC}_{\text{bdd}}$ result in a secure MAC scheme?

No! There exist secure UHF schemes, and secure MAC schemes for bounded length messages, such that the resulting combination is not a secure MAC scheme. The problem is that the MAC scheme need not hide

[1] Strictly speaking, they don't need to be equal. We only require the output space to be a subset of the key space.

the message being signed, and UHF doesn't offer any security if the key is revealed. Consider a UHF scheme where the output also includes the UHF key, and consider a MAC scheme where the signing algorithm also includes the string being signed. When the adversary queries for a signature on some message m, it also receives the UHF key as part of the signature. As a result, given the key, the adversary can now find two messages m_1 and m_2 that map to the same digest. Therefore, the adversary can query for a signature on m_1, receive σ, and output (m_2, σ) as a forgery.

3 Constructing CRHFs

As mentioned in the last lecture, we don't have any CRHF constructions based on one way functions. We will see a few constructions based on the hardness of number-theoretic problems. In practice, hash functions such as SHA are used as collision resistant hash functions. Note that these are not keyed functions. There are two ways of making sense of this:

- we can assume that when SHA was designed, a key was chosen (which is embedded in the SHA circuit)
- the *random oracle heuristic* is often used to justify the use of a keyless hash function. Suppose the construction of a cryptographic primitive uses a keyless hash function H. In the security proof, the function H is replaced with a random function. Perhaps this heuristic doesn't make much sense at this point. Later in the course, we will prove security of a few constructions in the random oracle model; hopefully it'll make more sense then.

Suppose you are given a CRHF $\left\{H_k : \{0,1\}^{2n} \to \{0,1\}^n\right\}$. Can we construct a hash function that can compress $3n$ bits to n bits? Or more generally, can we construct a hash function that can compress any bit-string to an n bit string? We saw the following proposals in class:

1. On input $m = (m_1, m_2, m_3)$, compute $y_1 = H_k(m_1, m_2)$, $y_2 = H_k(m_2, m_3)$ and output $y_1 \oplus y_2$ as the hash. This is not collision-resistant. Consider the message (m, m, m). The new hash function always maps it to 0^n (independent of m).

2. On input $m = (m_1, m_2, m_3)$, compute $y_1 = H_k(m_1, m_2)$. Output $m_3 \oplus y_1$. This is also not collision resistant, as $(m, m, H_k(m, m))$ always outputs 0^n.

3. On input $m = (m_1, m_2, m_3)$, compute $y_2 = H_k(m_1, m_2)$, then output $y_3 = H_k(y_2, m_3)$. This is a secure CRHF, assuming $\{H_k\}$. This construction was first proposed by Ralph Merkle and Ivan Damgard, and hence is referred to as the Merkle-Damgard transform.

3.1 The Merkle-Damgard transform:

Let $\mathcal{H} = \left\{H_k : \{0,1\}^{2n} \to \{0,1\}^n\right\}_{k \in \mathcal{K}}$ be a collision resistant hash function family mapping $2n$ bits to n bits. Consider the function family $\mathcal{H}' = \left\{H'_k : \left(\{0,1\}^n\right)^{\geq 2} \to \{0,1\}^n\right\}_{k \in \mathcal{K}}$ defined as follows:

- on input $m = (m_1, m_2, m_3, \ldots, m_\ell)$, set $y_1 = m_1$. For $i > 1$, set $y_i = H_k(y_{i-1}, m_i)$. Output y_ℓ as the final digest.

Theorem 22.01. Assuming $\mathcal{H}$ is a secure CRHF, for any fixed input length $\ell \cdot n$, $\mathcal{H}'$ is collision resistant for input-space $\{0,1\}^{\ell \cdot n}$.

Proof. Suppose there exists a p.p.t. adversary $\mathcal{A}$ that, on input k, outputs a collision (m, m'). The message space is $\{0,1\}^{\ell \cdot n}$, therefore $m = (m_1, m_2, \ldots, m_\ell)$ and $m' = (m'_1, \ldots, m'_\ell)$. Since it is a nontrivial collision, $H'_k(m) = H'_k(m') = y_\ell$ and $m \neq m'$. If $\ell = 2$, then we already have a collision for $\mathcal{H}$, and we are done. If $\ell > 2$, we will either find a collision for $\mathcal{H}$, or find a collision for $\mathcal{H}'$ with shorter length messages.

Let $y_1 = m_1$, $y'_1 = m'_1$. For $i > 1$, set $y_i = H_k(y_{i-1}, m_i)$ and $y'_i = H_k(y'_{i-1}, m_i)$. Since $y_\ell = y'_\ell$, we know that $H_k(y_{\ell-1}, m_\ell) = H_k(y'_{\ell-1}, m'_\ell)$. We have three cases here:

- $m_\ell = m'_\ell$ but $y_{\ell-1} \neq y'_{\ell-1}$: in this case, we have a collision for H_k.

- $m_\ell \neq m'_\ell$ and $y_{\ell-1} \neq y'_{\ell-1}$: again, we have a collision for H_k.
- $m_\ell = m'_\ell$ and $y_{\ell-1} = y'_{\ell-1}$. In this case, consider the messages $\widetilde{m} = (m_1, \ldots, m_{\ell-1})$ and $\widetilde{m'} = (m'_1, \ldots, m'_{\ell-1})$. Since $m \neq m'$ but $m_\ell = m'_\ell$, this implies that $\widetilde{m} \neq \widetilde{m'}$. Moreover, since $y_{\ell-1} = y'_{\ell-1}$, we have that $H'_k(\widetilde{m}) = H'_k(\widetilde{m'})$. As a result, $(\widetilde{m}, \widetilde{m'})$ constitute a nontrivial collision for $\mathcal{H}'$ with shorter length than m and m'. Repeat the above process until you find a collision for H_k.

□

The Merkle-Damgard transformation we described above does not work for messages of different lengths. If we use the same transformation as above, an adversary, given the hash key k, can compute $m = H_k(m_1 \mid\mid m_2)$, and then output $(m_1 \mid\mid m_2 \mid\mid m_3, m \mid\mid m_3)$ as a collision. Note that both these strings will map to the same value. To handle arbitrary length messages, one needs to append the message length at the end, and compute the hash on this new string. The proof of this is left as an exercise (Question 22.05).

4 Lecture summary, plan for next lecture, additional resources

Summary: In this lecture, we discussed how to use UHFs + PRFs to construct MAC schemes with unbounded message space. The same construction and proof also works if we use a CRHF instead of a UHF. The benefit of using a CRHF is that it is more robust — we can use any MAC scheme for bounded-length messages. Next, we discussed how to improve the compression factor of a CRHF.

Next lecture: We will go back to our discussion of encryption security, and see how to define and construct encryption schemes that are secure against tampering attacks.

Relevant sections from textbook [Boneh-Shoup]: Section 7.3 discusses the UHF+PRF construction. Section 8.4 discusses the Merkle-Damgard transformation.

5 Questions

Question 22.01. Consider ECBC-MAC construction described in Section 2.1, but with same key. The scheme uses a PRF $F : \mathcal{X} \times \mathcal{K} \to \mathcal{X}$. Let $m = (m_1, \ldots, m_\ell)$ be the message to be signed. The secret key consists of a PRF keys k.

- $\mathsf{Sign}(m, k)$: Let $y_1 = F(m_1, k)$. For $i \in [2, \ell]$, the signing algorithm computes $y_i = F(m_i \oplus y_{i-1}, k)$. Finally, it outputs $\sigma = F(y_\ell, k)$.

Show that this is not a secure MAC scheme for unbounded-length messages.

Question 22.02. Consider a MAC scheme $\mathsf{MAC}_{\text{bdd}}$ that signs n-bit messages, and let $\mathcal{H}$ be a CRHF family that maps unbounded-length inputs to n-bit ouptut. Construct a MAC scheme that handles unbounded-length messages, and prove that it is strongly unforgeable.

Question 22.03. Let H and G be two keyed compressing functions that map ℓ bits to n bits. You are given that at least one of the two function families is collision resistant.

- Show that $F_{(k_1,k_2)}(x) = H_{k_1}(x) \mid\mid G_{k_2}(x)$ is a collision resistant hash function family.
- Can we say the same about $F_{(k_1,k_2)}(x) = H_{k_1}(x) \oplus G_{k_2}(x)$?

Question 22.04. Additive hash functions: Let $H : \{0,1\}^* \to \{0,1\}^n$ be a compressing function with the property that $H(x \oplus y) = H(x) \oplus H(y)$ for all x, y of the same length. Show that H is not collision resistant.

All are easy questions. Look in the notebook.

Question 22.05. Let $\mathcal{H} = \{H_k : \{0,1\}^{2n} \to \{0,1\}^n\}$ be a family of collision-resistant hash functions. Consider the function family $\mathcal{H}' = \left\{H'_k : (\{0,1\}^n)^{\geq 2} \to \{0,1\}^n\right\}$ defined as follows:
on input $m = (m_1, \ldots, m_\ell)$ where each $m_i \in \{0,1\}^n$, set $y_1 = m_1$, and for all $i \in [2, \ell]$, compute $y_i = H_k(y_{i-1} \mid\mid m_i)$. Finally, output $y_{\ell+1} = H(y_\ell \mid\mid [\ell]_2)$. Here, $[\ell]_2$ denotes the binary representation of ℓ using n bits.

Show that $\mathcal{H}'$ is a secure CRHF, assuming $\mathcal{H}$ is.

Hint: if the two messages are of different lengths, then we definitely have a collision in the last H_k computation. If the message lengths are same, then we can use the proof in Claim 22.01.