CS 600.442 – Modern Cryptography October 12, 2015

Lecture 12: Authentication

Instructor: Abhishek Jain *Scribe: Naomi Ephraim*

1 The Setting

Alice wants to send a message m to Bob. Bob wants to know that the message has not been tampered with.

- Alice(signer) signs m to produce σ
- Bob(verifier) verifies that σ is generated for m.
- We want the pair (m, σ) to be unforgable.

2 Message Authentication Code (MAC)

This is the analogue of secret-key encryption for the authentication problem. We allow the signer and verifier to share a secret. Intuitively, we want a MAC scheme to consist of the following algorithms:

- A key generation algorithm $\text{Gen}(1^n)$ that takes as input the security parameter and outputs secret key k.
- A signing algorithm $\text{Tag}_k(m)$ outputs tag σ.
- A verification algorithm $\text{Ver}_k(m, \sigma)$ that outputs 1 iff σ is a valid tag on m under secret key k.

From a security viewpoint, we want that an adversary who can observe multiple (m, σ) pairs, should not be able to forge a tag on a new message.

We now formally define a MAC.

Syntax. A Message Authentication Code consists of the following three algorithms:

- **Key Generation:** $k \leftarrow \text{Gen}(1^n)$
- **Sign:** $\sigma \leftarrow \text{Tag}_k(m)$
- **Verify:** $\text{Ver}_k : \mathcal{M} \times \mathcal{T} \rightarrow \{0, 1\}$

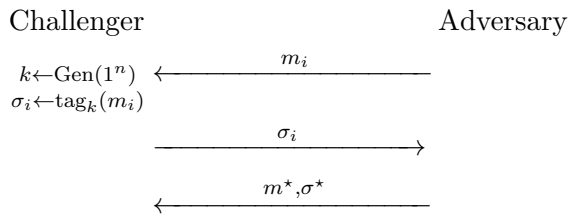
We require a MAC to satisfy the following two properties:

- **Correctness:** $\text{P}\left[k \leftarrow \text{Gen}(1^n), \sigma \leftarrow \text{Tag}_k(m) : \text{Ver}_k(m, \sigma) = 1\right] = 1$.

- **Security (UF-CMA):** For all n.u. PPT $\mathcal{A}$ there exists a negligible function ν such that :

$$\mathrm{P}\left[\begin{smallmatrix} k\leftarrow\mathrm{Gen}(1^n) \\ (m,\sigma)\leftarrow\mathcal{A}^{\mathrm{tag}_k(\cdot)}(1^n) \end{smallmatrix} : \mathcal{A} \text{ didn't query } m \wedge \mathrm{Ver}_k(m,\sigma)=1\right] \leq \nu(n).$$

  Here, UF-CMA refers to unforgeability under chosen message attacks. We can view this notion as the following game between a challenger and an adversary:



  where the first two exchanges are repeated polynomial number of times. Then, the adversary wins if and only if $m^\star \neq m_i$ for any i and $\mathrm{Ver}_k(m,\sigma)=1$. We say that the MAC scheme is UF-CMA secure if for PPT adversaries, Pr[Adversary wins] $\leq \nu(n)$.

3 MAC: Construction

Intuition. To construct a MAC we want something that uses a secret key and has unpredictable output. Therefore, it cannot be constructed from a one-way function or a PRG. Hence, to construct it we choose a pseudo-random function (PRF), since it has a secret key and the output looks random to an adversary.

Here is the construction. Let $f_k(\cdot)$ be a PRF with secret key k. We define the MAC by

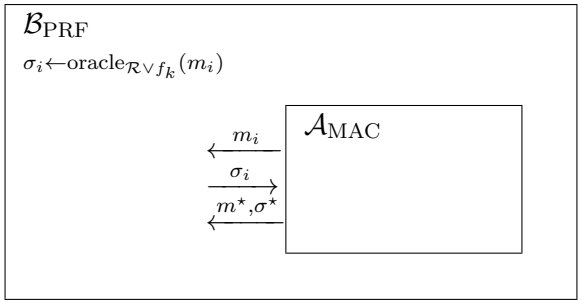
$\mathrm{Gen}(1^n)$: Output $k \xleftarrow{\$} \{0,1\}^n$.

$\mathrm{Tag}_k(m)$: Output $f_k(m)$.

$\mathrm{Ver}_k(m,\sigma)$: Output $f_k(m) \stackrel{?}{=} \sigma$.

Theorem 1 *PRF $\Rightarrow$ MAC.*

Proof. Suppose that there exists an adversary $\mathcal{A}_{\mathrm{MAC}}$ that can break the security for MAC. We want to construct $\mathcal{B}_{\mathrm{PRF}}$ that distinguishes between a PRF and a random function with non-negligible probability, thereby breaking the security for a PRF. Let f_k be sampled from a PRF family. $\mathcal{B}_{\mathrm{PRF}}$ has access to $\mathcal{A}_{\mathrm{MAC}}$ and an oracle that is either f_k or a truly random function. The reduction is as follows.



Here, $\mathcal{A}_{\text{MAC}}$ queries $\mathcal{B}_{\text{PRF}}$. On receiving a signature query m_i from $\mathcal{A}_{\text{MAC}}$, $\mathcal{B}_{\text{PRF}}$ forwards it as its own query to its oracle. Let y_i be the output of the oracle. $\mathcal{B}_{\text{PRF}}$ sets $\sigma_i = y_i$ and sends it to $\mathcal{A}_{\text{MAC}}$. After a number of these queries, $\mathcal{A}_{\text{MAC}}$ outputs $(m^\star, \sigma^\star)$. Then, if $\sigma^\star$ is the correct signature of $m^\star$, $\mathcal{B}_{\text{PRF}}$ guesses its oracle as f_k. Otherwise, he guesses its oracle to be $\mathcal{R}$.

Now, observe that when the oracle of $\mathcal{B}_{\text{PRF}}$ is a truly random function, then the tags received by $\mathcal{A}_{\text{MAC}}$ correspond to truly random values. In this case, $\mathcal{A}_{\text{MAC}}$ can output a valid forgery with probability only $\frac{1}{2^n}$ where n corresponds to the tag length. Now consider the case when the oracle of $\mathcal{B}_{\text{PRF}}$ is f_k. By our assumption on $\mathcal{A}_{\text{MAC}}$, we have that $\mathcal{A}_{\text{MAC}}$ outputs a valid forgery in this case with non-negligible probability $\varepsilon(n)$.

It follows from the above that $\mathcal{B}_{\text{PRF}}$ distinguishes between its oracles with non-negligible probability $\varepsilon(n) - \frac{1}{2^n}$. This is a contradiction to the security of the PRF family. ■

4 One-Time MAC

The one-time MAC is an analog of the one-time pad for authentication. Here, there is weaker security because the adversary is only allowed 1 query. The advantage to this is that it in fact gives unconditional security. It can be constructed from pairwise independent hash functions and the students are encouraged to read about this from the literature.

5 Digital Signatures

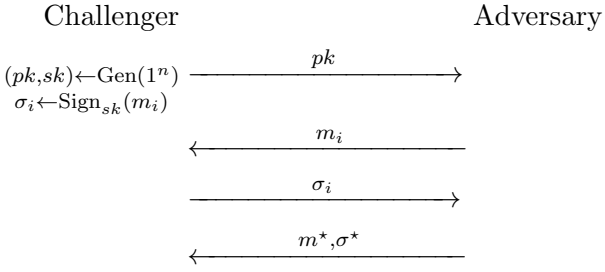
Digital signatures are the analogue of public-key encryption for authentication. In a digital signature scheme, there is a signer and a verifier. Unlike MACs where the signer and the verifier share a key (and therefore both can sign and verify), in digital signatures, the signer is the only one who can sign a message, but everyone can verify the signature.

Syntax. A digital signature scheme consists of three algorithms:

- **Key Generation:** $(sk, pk) \leftarrow \text{Gen}(1^n)$.
- **Sign:** $\sigma \leftarrow \text{Sign}_{sk}(m)$.
- **Verify:** $\text{Ver}_{pk}(m, \sigma) : \mathcal{M} \times \mathcal{S} \to \{0, 1\}$.

A digital signature scheme must satisfy the following two properties:

- **Correctness:** $\Pr[(sk, pk) \leftarrow \text{Gen}(1^n), \sigma \leftarrow \text{Sign}_{sk}(m) : \text{Ver}_{pk}(m, \sigma) = 1] = 1$
- **Security (UF-CMA):** The security definition can be formalized similarly to MACs. We simply describe the UF-CMA security game here:



Here the adversary can make multiple signatures queries m_i to receive the corresponding signatures σ_i. The adversary wins if $m^\star \neq m_i$ for any i and $\sigma^\star$ is the signature of message $m^\star$. We say that the signature scheme is UF-CMA secure if for all PPT adversaries, Pr[Adversary wins] $\leq \nu(n)$ where $\nu(\cdot)$ is some negligible function.

6 One-Time Signature (OTS)

We will now look at one-time signatures, a type of digital signature. Here, the adversary only gets 1 query before has to guess a correct message-signature pair. We will look at the construction of Lamport.

Lamport's OTS. Let f be a one-way function. We construct a one-time signature scheme as follows:

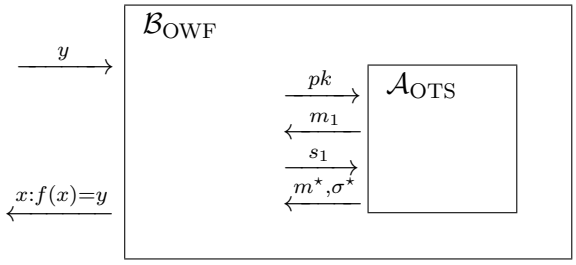
- **Key Generation:** Choose random values $x_i^b \xleftarrow{\$} \{0,1\}^n \ \forall i \in [n]$ and $b \in \{0,1\}$.
  Output secret key $sk = \begin{bmatrix} x_1^0 & x_2^0 & \cdots & x_n^0 \\ x_1^1 & x_2^1 & \cdots & x_n^1 \end{bmatrix}$
  Output public key $pk = \begin{bmatrix} y_1^0 & y_2^0 & \cdots & y_n^0 \\ y_1^1 & y_2^1 & \cdots & y_n^1 \end{bmatrix}$ where $y_i^b = f(x_i)^b \ \forall i \in [n]$ and $b \in \{0,1\}$
- **Sign:** $\text{Sign}_{sk}(m) : \sigma = (x_1^{m_1}, x_2^{m_2}, \ldots, x_n^{m_n})$.
- **Verification:** $\text{Ver}_{pk}(m, \sigma) : \bigwedge_{i \in [n]} f(\sigma_i) \stackrel{?}{=} y_i^{m_i}$.

Intuitively, this signature scheme works by looking at each bit m_i of the message m, and choosing the corresponding string $x_i^{m_i}$ from the secret key as a component in the signature. For example, if $m_i = 1$, then the ith component of the signature will be x_i^1. To verify, this signature scheme checks that f applied to each $x_i^{m_i}$ in the signature returns the corresponding $y_i^{m_i}$ in the public key.

A key point is that it is a one-time signature scheme. Indeed, we can easily mount an attack by querying twice, once on m and once on $\overline{m}$ and learn the entire secret key which will then allow us to sign arbitrary messages.

We will now argue its security for one query? The intuition is that if you can forge a signature after seeing only one message-signature pair, then you can invert f.

Proof. Suppose there is an adversary $\mathcal{A}_{\text{OTS}}$ that can break this one-time signature scheme. We want to construct $\mathcal{B}_{\text{OWF}}$ that can invert a one-way function.



We will first show this for a weaker notion of security. Since the signature corresponds to the pre-image of the one-way function, assume that adversary declares $m_1, m^\star$, and then after seeing

these we give him pk, σ_1. Then, he sends us his forgert, $\sigma^\star$. Clearly, this is a weaker adversary. Suppose $\mathcal{A}_{\text{OTS}}$ win this game.

$\mathcal{B}_{\text{OWF}}$ needs to somehow embed y in the challenge to $\mathcal{A}_{\text{OTS}}$. Therefore, $\mathcal{B}_{\text{OWF}}$ looks at $m_1, m^\star$. Since $m_1 \neq m^\star$, let $m_{1i} \neq m_i^\star$ and assume $m_{1j} = m_j^\star \; \forall j \neq i$. Then, $\mathcal{B}_{\text{OWF}}$ takes the challenge y and embeds it at the $(m_i^\star, i)$th coordinate of the public key, i.e., it sets $y_i^{m_i^\star} = y$. More specifically, $\mathcal{B}_{\text{OWF}}$ first chooses random x_j^0, x_j^1 for every $j \in [n]$. It then computes $y_j^b = f(x_j^b)$ for every $j \in [n], b \in \{0,1\}$. It then discards $x_i^{m_i^\star}$ and replaces $y_i^{m_i^\star}$ with its challenge y. He now sends this public key to $\mathcal{A}_{\text{OTS}}$.

Now,when $\mathcal{A}_{\text{OTS}}$ sends its forgery $\sigma^\star$, $\mathcal{B}_{\text{OWF}}$ simply sets $x = \sigma_i^\star$ as its inversion of y and stops. To see why this is a good inverter, note that if $\sigma^\star$ is a valid forgery, then it must contain inversions of $y_1^{m_1^\star}, \ldots, y_n^{m_n^\star}$. Since $y_i^{m_i^\star} = y$, we have that $\sigma_i^\star$ is a valid inversion of y, as required. Thus, if $\mathcal{A}_{\text{OTS}}$ outputs a valid forgery with non-negligible probability $\varepsilon(n)$, then $\mathcal{B}_{\text{OWF}}$ can invert f with the same non-negligible probability $\varepsilon(n)$, which is a contradiction.

Note that the above proof required that $\mathcal{B}_{\text{OWF}}$ receives both m_1 and $m^\star$ at the beginning from $\mathcal{A}_{\text{OTS}}$. How can we go from this case to the stronger case where $\mathcal{A}_{\text{OTS}}$ asks to see the public key before making his query? We will have a little bit of loss in the reduction, but the probability of inverting the one-way function will still be non-negligible, which will still be enough to get us a contradiction. To do this, we just guess location i and the value of $m_i^\star \in \{0,1\}$. If we guess correctly, then we do the same as in the weaker case. Otherwise, we abort and try again. We guess i correctly with probability $\frac{1}{n}$ and we guess $m_i^\star$ correctly with probability $\frac{1}{2}$. Therefore, if $\mathcal{A}_{\text{OTS}}$ outputs a valid forgery with non-negligible probability $\varepsilon(n)$, then $\mathcal{B}_{\text{OWF}}$ can invert f with probability $\frac{\varepsilon(n)}{2n}$. ■

Here, if the adversary sees the public key before or after choosing messages, it makes a big difference. This will usually be the case with signature schemes. Also, note that in the proof of signature schemes, we have to surmount the seeming paradox that we should be able to sign message queries of the adversaries and still use his forgery to get a contradiction.

How can we send arbitrarily long messages? Consider a signature scheme that signs n bit messages. We want to sign l bits. Therefore, we need to compress l bits to n bits. Take some compression function $h : l$ bits $\to n$ bits. Then, to sign a message m, compute the signature on $h(m)$. We say that h is a good compression function if the probability of a collision is small.

7 Collision-resistant Hash Functions (CRHF)

Intuition. Find a compression function h for which it is hard to find x, x' such that $x \neq x'$ but $h(x) = h(x')$. This is impossible with the notion of a non-uniform adversary, because there must exist collisions, so any non-uniform adversary could have a hard-coded circuit of collisions for h. Therefore, we consider a family of CRHFs.

Definition 1 (Collision Resistant Hash Function Family) *A family $H = \{h_i : D_i \to R_i\}_{i \in I}$ is a Collision Resistant Hash Function Family if*

1. ***Easy to sample**: there exists a PPT generation function such that $i \leftarrow Gen(1^n)$, with $i \in I$.*
2. ***Compression**: $|R_i| < |D_i|$*

3. ***Easy to evaluate****: There exists a polynomial time evaluation function such that given $x \in D_i$, $Eval(x, i) = h_i(x)$.*

4. ***Collision resistant****: For all n.u. PPT A, there exists a negligible function $\mu(\cdot)$ such that*

$$P\left[\begin{smallmatrix} i \leftarrow Gen(1^n) \\ (x,x') \leftarrow A(1^n,i) \end{smallmatrix} : x \neq x' \wedge h(x) = h_i(x')\right] \leq \mu(n).$$

We conclude with a few remarks about CHRFs.

- One-bit compression implies arbitrary bit compression. Think about a proof that simply iterates the one-bit compression. This is possible because the domain is arbitrary. Another way to do this is via Merkle trees. These are very useful and worthwhile to read about.
- The range can't be too small. Otherwise, finding collisions will become easy, and will be subject to the enumeration attack and the birthday attack.
- CHRFs are unlikely to be constructed from one-way functions or one-way permutations. They can however be constructed from number-theoretic assumptions, like factoring and discrete log.
- Often, we don't need collision resistance, and we can use a weaker notion. One such notion is that of a Universal One-Way Hash Function (UOWHF), where the security is defined similarly to CRHFs, except that we now require that for all n.u. PPT A, there exists a negligible function $\mu(n)$, s.t.

$$\mathrm{P}\left[\begin{smallmatrix} (x,\mathrm{state}) \leftarrow A(1^n) \\ i \leftarrow \mathrm{Gen}(1^n) \\ x' \leftarrow A(i,\mathrm{state}) \end{smallmatrix} : x \neq x' \wedge h_i(x) = h_i(x')\right] \leq \mu(n).$$

  These can be constructed from one-way functions and suffice for digital signatures.