Foundation of Cryptography, Lecture 5
MACs and Signatures

Handout Mode

Iftach Haitner, Tel Aviv University

Tel Aviv University.

April 4, 2019
Part I

Message Authentication Codes (MACs)
Definition 1 (MAC)

A trippet of PPT’s (Gen, Mac, Vrfy) such that:

1. Gen(1^n) outputs a key $k \in \{0, 1\}^*$
2. Mac($k$, $m$) outputs a “tag” $t$
3. Vrfy($k$, $m$, $t$) output 1 (YES) or 0 (NO)

Consistency:

$\forall k \in \text{Supp}(\text{Gen}(1^n))$, $m \in \{0, 1\}^n$ and $t = \text{Mac}_k(m)$: $\text{Vrfy}_k(m, t) = 1$

Definition 2 (Existential unforgeability)

A MAC (Gen, Mac, Vrfy) is existential unforgeable (EU), if $\forall \text{ PPT } A$:

$$\Pr_{k \leftarrow \text{Gen}(1^n) \atop (m, t) \leftarrow A^{\text{Mac}_k, \text{Vrfy}_k(1^n)}} \left[ \text{Vrfy}_k(m, t) = 1 \land \text{Mac}_k \text{ was not asked on } m \right] = \text{neg}(n)$$

Remark: convention
Definition of MAC cont.

- “Private key" definition
- Security definition too strong? Any message? Use of Verifier?
- “Replay attacks"
- Strong existential unforgeable MACS (for short, strong MAC): infeasible to generate new valid tag (even for message for which a MAC was asked)
Restricted MACs

Definition 3 (Length-restricted MAC)
Same as in Definition 1, but for $k \in \text{Supp}(G(1^n))$, $\text{Mac}_k$ and $\text{Vrfy}_k$ only accept messages of length $n$.

Definition 4 ($\ell$-time MAC)
A MAC scheme is existential unforgeable against $\ell$ queries (for short, $\ell$-time MAC), if it is existential unforgeable as in Definition 2, but $A$ can only make $\ell$ queries.
Section 1

Constructions
One-time length-restricted MAC

Construction 5 (One-time MAC)

- **Gen**(1^n): output k ← {0, 1}^n.
- **Mac_k**(m): output h_k(m).
- Vrfy_k(m, t): output 1 iff t = h_k(m).

Claim 6
The scheme is one-time MAC if \( \{h_k\} \) is pairwise-independent.
Subsection 1

Restricted-Length MAC
$\ell$-wise independent functions

**Definition 7 ($\ell$-wise independent)**

A function family $\mathcal{H}$ from $\{0, 1\}^n$ to $\{0, 1\}^m$ is $\ell$-wise independent, if for every distinct $x_1, \ldots, x_\ell \in \{0, 1\}^n$ and every $y_1, \ldots, y_\ell \in \{0, 1\}^m$, it holds that

$$\Pr_{h \leftarrow \mathcal{H}} [h(x_1) = y_1 \land \ldots \land h(x_\ell) = y_\ell] = 2^{-\ell m}.$$
Construction 8 ($\ell$-time MAC)

Let $\mathcal{H} = \{\mathcal{H}_n: \{0, 1\}^n \mapsto \{0, 1\}^n\}$ be an efficient $(\ell + 1)$-wise independent function family.

- $\text{Gen}(1^n)$: output $h \leftarrow \mathcal{H}_n$.
- $\text{Mac}(h, m)$: output $h(m)$.
- $\text{Vrfy}(h, m, t)$: output 1 iff $t = h(m)$.

Claim 9

The above scheme is a length-restricted, $\ell$-time MAC

Proof: ?
**Construction 10**

Same as Construction 8, but uses function $\mathcal{F} = \{\mathcal{F}_n : \{0, 1\}^n \rightarrow \{0, 1\}^n\}$ instead of $\mathcal{H}$.

**Claim 11**

Assuming that $\mathcal{F}$ is a PRF, then Construction 10 is an existential unforgeable MAC.

Proof: Easy to prove if $\mathcal{F}$ is a family of random functions. Hence, also holds in case $\mathcal{F}$ is a PRF. □
Subsection 2

Any Length
Definition 12 (collision resistant hash family (CRH))

A function family \( H = \{ H_n : \{0, 1\}^* \rightarrow \{0, 1\}^n \} \) is collision resistant, if

\[
\Pr_{h \leftarrow H_n} [A(1^n, h) = (x, x') \text{ s.t. } x \neq x' \land h(x) = h(x')] = \neg(n)
\]

for any PPT \( A \).

- Not known to implied by OWFs.
Construction 13 (Length restricted MAC \(\Rightarrow\) MAC)

Let \((\text{Gen}, \text{Mac}, \text{Vrfy})\) be a length-restricted MAC, and let \(\mathcal{H} = \{\mathcal{H}_n: \{0, 1\}^* \mapsto \{0, 1\}^n\}\) be an efficient function family.

- Gen'(\(1^n\)): Sample \(k \leftarrow \text{Gen}(1^n)\) and \(h \leftarrow \mathcal{H}_n\). Output \(k' = (k, h)\)
- Mac'\(_k(h(m)) = \text{Mac}_k(h(m))\)
- Vrfy'\(_{k,h}(t, m) = \text{Vrfy}_k(t, h(m))\)

Claim 14

Assume \(\mathcal{H}\) is an efficient collision-resistant family and \((\text{Gen}, \text{Mac}, \text{Vrfy})\) is existential unforgeable, then \((\text{Gen}', \text{Mac}', \text{Vrfy}')\) is existential unforgeable MAC.

Proof: ?
Part II

Signature Schemes
Signature schemes

Definition 15 (Signature schemes)

A trippet of PPT’s $(\text{Gen, Sign, Vfy})$ such that

1. $\text{Gen}(1^n)$: output a pair of keys $(s, v) \in \{0, 1\}^* \times \{0, 1\}^*$
2. $\text{Sign}(s, m)$: output a “signature” $\sigma \in \{0, 1\}^*$
3. $\text{Vfy}(v, m, \sigma)$: output $1$ (YES) or $0$ (NO)

Consistency: for any $(s, v) \in \text{Supp}(\text{Gen}(1^n))$, $m \in \{0, 1\}^*$ and $\sigma \in \text{Supp}(\text{Sign}_s(m))$: $\text{Vfy}_v(m, \sigma) = 1$.

Definition 16 (Existential unforgability)

A signature scheme is existential unforgeable (EU), if $\forall$ PPT $A$

$$\Pr_{(s, v) \leftarrow \text{Gen}(1^n)} \left[ A^{\text{Sign}_s(1^n, v)} = (m, \sigma) \text{ s.t } \text{Vfy}_v(m, \sigma) = 1 \land \text{Sign}_s \text{ didn’t query } m \right]$$

is negligible in $n$. 
Signature schemes cont.

- Signature $\Rightarrow$ MAC
- “Harder" to construct than MACs: (even restricted forms) require OWF
- Oracle access to $\text{Vrfy}$ is not given
- **Strong existential unforgeable signatures** (for short, strong signatures): infeasible to generate new valid signatures (even for message for which a signature was asked)

**Theorem 17**

*OWFs imply strong existential unforgeable signatures.*
Section 2

OWFs $\implies$ Signatures
Subsection 1

One-time signatures
Definition 18 (length-restricted signatures)

Same as in Definition 15, but for \((s, v) \in \text{Supp}(G(1^n))\), \(\text{Sign}_s\) and \(\text{Vrfy}_v\) only accept messages of length \(n\).
Bounded-query signatures

**Definition 19 (ℓ-time signatures)**

A signature scheme is **existential unforgeable against ℓ-query** (for short, ℓ-time signature), if it is existential unforgeable as in **Definition 16**, but A can only ask for ℓ queries.

**Claim 20**

Assuming CRH exists, then length restricted k-time signatures can be used to construct k-time signatures.

Proof: ?

**Proposition 21**

Wlg, the signer of a k-time signature scheme, for fixed k, is deterministic

Proof: ?
Construction 22 (length-restricted, one-time signature)

Let $f : \{0, 1\}^n \mapsto \{0, 1\}^n$.

1. Gen($1^n$):
   1.1 $s_1^0, s_1^1, \ldots, s_n^0, s_n^1 \leftarrow \{0, 1\}^n$.
   1.2 Secret (signing) key is $s = \{s_i^0, s_i^1\}_{i=1}^n$.
   1.3 Public (verification) is $v = \{v_i^0, v_i^1\}_{i=1}^n$ for $v_i^b = f(s_i^b)$.

2. Sign($s, m$): $\sigma = (s_1^{m_1}, \ldots, s_n^{m_n})$

3. Vrfy($v, m, \sigma = (\sigma_1, \ldots, \sigma_n)$): check that $f(\sigma_i) = v_i^{m_i}$ for all $i \in [n]$

Lemma 23

If $f$ is a OWF, then Construction 22 is a length restricted one-time signature scheme.

Is this a strong signature scheme? With some additional work, it can be turned into a strong one.
Proving Lemma 23

Let a PPT $A$, $\mathcal{I} \subseteq \mathbb{N}$ and $p \in \text{poly}$ that break the security of Construction 22, we use $A$ to invert $f$.

Algorithm 24 ($\text{Inv}$)

**Input:** $y \in \{0, 1\}^n$

1. Choose $(s, v) \leftarrow \text{Gen}(1^n)$ and replace $v_{i^*}^{b^*}$ for a random $i^* \in [n]$ and $b^* \in \{0, 1\}$, with $y$.

2. Abort, if $A(1^n, v)$ asks to sign message $m \in \{0, 1\}^n$ with $m_{i^*} = b^*$. Otherwise, use $s$ to answer the query.

3. Let $(m', \sigma')$ be $A$’s output.
   Abort, if $\sigma'$ is not a valid signature for $m'$, or $m'_{i^*} \neq b^*$.
   Otherwise, return $\sigma_{i^*}$.

- $v$ is distributed as is in the real “signature game”
- $v$ is independent of $i^*$ and $b^*$.
- Therefore $\text{Inv}$ inverts $f$ w.p. $\frac{1}{2np(n)}$ for every $n \in \mathcal{I}$. 
Subsection 2

Stateful Schemes
Stateful signature schemes

Definition 25 (Stateful scheme)
Same as in Definition 15, but Sign might keep state which is updated every signature.

- Make sense in many applications (e.g., smartcards)
- We’ll later use it a building block for building stateless scheme

\(^1\) Also known as memory-dependant schemes
Stateful schemes — straight-line construction

Let \((\text{Gen}, \text{Sign}, \text{Vrfy})\) be a strong one-time signature scheme.

**Construction 26 (straight-line construction)**

- **\(\text{Gen}'(1^n)\):** Output \((s', v') = (s_1, v_1) \leftarrow \text{Gen}(1^n)\).

- **\(\text{Sign}'_{s_i}(m_i)\), where \(m_i\) is \(i\)’th message to sign:**
  1. Let \((s_{i+1}, v_{i+1}) \leftarrow \text{Gen}(1^n)\)
  2. Let \(\sigma_i = \text{Sign}_{s_i}(m_i, v_{i+1})\)
  3. Output \(\sigma'_i = (\sigma'_{i-1}, m_i, v_{i+1}, \sigma_i)\).\(^a\)

- **\(\text{Vrfy}'_{v_i}(m, \sigma' = (m_1, v_2, \sigma_1), \ldots, (m_i, v_{i+1}, \sigma_i))\):**
  Check that
  1. \(\text{Vrfy}_{v_j}((m_j, v_{j+1}), \sigma_j) = 1\) for every \(j \in [i]\)
  2. \(m_i = m\)

\(^a\sigma'_0\) is the empty string.
The state of Sign' is used for maintaining the most recent signing key (e.g., $s_i$), and the last published signature that connects $s_i$ to $v_1$.

While polynomial time, it is rather inefficient scheme: both running time and signature size are linear in number of published signatures.

That (Gen, Sign, Vrfy) works for any length (specifically, it is possible to sign message that is longer than the verification key), is critically used.

Lemma 27

(\text{Gen}', \text{Sign}', \text{Vrfy}') \text{ is a stateful, strong signature scheme.}

Proof: Assume $\exists \text{ PPT } A', \ p \in \text{ poly}$ and infinite set $\mathcal{I} \subseteq \mathbb{N}$, such that $A'$ breaks the strong security of (Gen', Sign', Vrfy') with probability $\frac{1}{p(n)}$ for all $n \in \mathcal{I}$. We present PPT $A$ that breaks the security of (Gen, Sign, Vrfy).

We assume for simplicity that $p$ also bounds the query complexity of $A'$
Proving Lemma 27 cont.

Let \((m_t, \sigma') = (m_1, v_2, \sigma_1), \ldots, (m_t, v_{t+1}, \sigma_t)\) be the pair output by \(A'\)

Claim 28

Whenever \(A'\) succeeds, \(\exists i \in [p]\) such that:

1. \(\text{Sign}'\) has output \(\sigma'_{i-1} = (m_1, v_2, \sigma_1), \ldots, (m_{i-1}, v_i, \sigma_{i-1})\)

2. \(\text{Sign}'\) has not output \(\sigma'_{i} = (m_1, v_2, \sigma_1), \ldots, (m_i, v_{i+1}, \sigma_i)\)

Proof: ?

It follows that

- \(v_i\) was sampled by \(\text{Sign}'\)
  
  Let \(s_i\) be the signing key generated by \(\text{Sign}'\) along with \(v_i\), and let \(\tilde{m} = (m_i, v_{i+1})\)

- \(\text{Vrfy}_{v_i}(\tilde{m}, \sigma_i) = 1\)

- \(\text{Sign}_{s_i}\) was not queried by \(\text{Sign}'\) on \(\tilde{m}\) and output \(\sigma_i\).

- \(\text{Sign}_{s_i}\) was queried at most once by \(\text{Sign}'\)
Definition of A

Algorithm 29 (A)

Input: $1^n$, $\nu$
Oracle: $\text{Sign}_s$

1. Choose $i^* \leftarrow [p = p(n)]$ and $(s', \nu') \leftarrow \text{Gen}'(1^n)$.

2. Emulate a random execution of $\text{A}'^{\text{Sign}_s'}$ with a single twist:
   ▶ On the $i^*$'th call to $\text{Sign}_s'$, set $\nu_{i^*} = \nu$ (rather than choosing it via $\text{Gen}$)
   ▶ When need to sign using $s_{i^*}$, use $\text{Sign}_s$.

3. Let $(m, \sigma = (m_1, \nu_1, \sigma_1), \ldots, (m_q, \nu_q, \sigma_q)) \leftarrow \text{A}'$

4. Output $((m_{i^*}, \nu_{i^*}), \sigma_{i^*})$ (abort if $i^* > q$)

- The emulated game $\text{A}'^{\text{Sign}_s'}$ has the same distribution as the real game.
- $\text{Sign}_s$ is called at most once
- $A$ breaks $(\text{Gen}, \text{Sign}, \text{Vrfy})$ whenever $i^* = \tilde{i}$. 
Subsection 3

Somewhat-Stateful Schemes
A somewhat-stateful scheme

Let \((Gen, Sign, Vrfy)\) be a strong one-time signature scheme.

**Construction 30 (A somewhat-stateful scheme)**

- **Gen'\((1^n)\):** Output \((s', v') = (s_\lambda, v_\lambda) \leftarrow Gen(1^n)\).
- **Sign'_{s_\lambda}(m):** choose an unused \(r \in \{0, 1\}^n\)
  
  1. For \(i = 0\) to \(n - 1\): if \(a_{r_1, \ldots, i}\) was not set before:
     1.1 For both \(j \in \{0, 1\}\), let \((s_{r_1, \ldots, i, j}, v_{r_1, \ldots, i, j}) \leftarrow Gen(1^n)\)
     1.2 Let \(a_{r_1, \ldots, i} = (v_{r_1, \ldots, i, 0}, v_{r_1, \ldots, i, 1})\).
     1.3 Let \(\sigma_{r_1, \ldots, i} = Sign_{s_{r_1, \ldots, i}}(a_{r_1, \ldots, i})\)
  
  2. Output \((r, a_\lambda, \sigma_\lambda, \ldots, a_{r_1, \ldots, n-1}, \sigma_{r_1, \ldots, n-1}, \sigma_r = Sign_{s_r}(m))\)

- **Vrfy'_{v_\lambda}(m, \sigma') = (r, a_\lambda, \sigma_\lambda, \ldots, a_{r-1}, \sigma_{r_1, \ldots, n-1}, \sigma_r)\)

Check that

1. \(Vrfy_{v_{r_1, \ldots, i}}(a_{r_1, \ldots, i}, \sigma_{r_1, \ldots, i}) = 1\) for every \(i \in \{0, \ldots, n - 1\}\)

2. \(Vrfy_{v_r}(m, \sigma_r) = 1\), for \(v_r = (a_{r_1, \ldots, n-1})_r\)
A somewhat-stateful Scheme, cont.

- Each one-time signature key is used at most once.

**Lemma 31**

(Gen', Sign', Vrfy') is a stateful strong signature scheme.

Proof: ?

- Note that Sign' does not keep track of the message history.
- More efficient scheme — Enough to construct tree of depth $\omega(\log n)$ (i.e., to choose $r \in \{0, 1\}^{\ell \in \omega(\log n)}$)
Subsection 4

Stateless Schemes
Stateless Scheme

Let $\tilde{\Pi}_k$ be the set of all functions from $\{0, 1\}^*$ to $\{0, 1\}^k$, let $q \in \text{poly}$ be “large enough”, and let $\mathcal{H} = \{\mathcal{H}_n: \{0, 1\}^* \rightarrow \{0, 1\}^n\}$ be a CRH.

Construction 32 (Inefficient stateless Scheme)

- $\text{Gen}'(1^n)$: Sample $(s_\lambda, v_\lambda) \leftarrow \text{Gen}(1^n)$, $\pi \leftarrow \tilde{\Pi}_{q(n)}$ and $h \leftarrow \mathcal{H}_n$.
  Output $(s' = (s_\lambda, \pi, h), v' = v_\lambda)$.

- $\text{Sign}'_s(m)$: Set $r = \pi(h(m))_{1,...,n}$.
  1. For $i = 0$ to $n - 1$:
     1.1 For both $j \in \{0, 1\}$, let $(s_{r_1,...,i,j}, v_{r_1,...,i,j}) \leftarrow \text{Gen}(1^n; \pi(r_1,...,i,j))$
     1.2 Let $\sigma_{r_1,...,i} = \text{Sign}_{s_{r_1,...,i}}(a_{r_1,...,i} = (v_{r_1,...,i,0}, v_{r_1,...,i,1}))$
  2. Output $(r, a_\lambda, \sigma_\lambda, \ldots, a_{r_1,...,n-1}, \sigma_{r_1,...,n-1}, \sigma_r = \text{Sign}_{s_r}(m))$

- $\text{Vrfy}'$: unchanged

- One one-time signature key might be used several times, but always on the same message.

- Efficient scheme: use PRF (?)
Subsection 5

“CRH free" Schemes
Target collision-resistant functions

Definition 33 (target collision-resistant functions (TCR))
A function family \( \mathcal{H} = \{ \mathcal{H}_n : \{0, 1\}^* \rightarrow \{0, 1\}^n \} \), if

\[
\Pr_{(x,a)\leftarrow A_1(1^n); h\leftarrow \mathcal{H}_n; x'\leftarrow A_2(a,h)} [x \neq x' \land h(x) = h(x')] = \text{neg}(n)
\]

for any pair of PPT’s \( A_1, A_2 \).

Theorem 34

OWFs imply efficient compressing TCRs.

Proof: not that trivial...
Target one-time signatures

For simplicity we will focus on non-strong schemes.

**Definition 35 (target one-time signatures)**

A signature scheme \((\text{Gen}, \text{Sign}, \text{Vrfy})\) is target one-time existential unforgeable (for short, target one-time signature), if

\[
\Pr_{m \leftarrow A(1^n), (s, v) \leftarrow \text{Gen}(1^n), (m', \sigma) \leftarrow A(\text{Sign}_S(m))} [m' \neq m \land \text{Vrfy}_v(m', \sigma) = 1] = \text{neg}(n)
\]

for any PPT \(A\)

**Claim 36**

OWFs imply target one-time signatures.
Definition 37 (random one-time signatures)

A signature scheme \((\text{Gen}, \text{Sign}, \text{Vrfy})\) is random one-time existential unforgeable (for short, random one-time signature), if

\[
\Pr_{m \leftarrow \mathcal{M}_n; (s, v) \leftarrow \text{Gen}(1^n); (m', \sigma) \leftarrow \mathcal{A}(m, \text{Sign}_s(m))} [m' \neq m \land \text{Vrfy}_v(m', \sigma) = 1] = \text{neg}(n)
\]

for any PPT \(\mathcal{A}\) and any efficiently samplable string ensemble \(\mathcal{M} = \{\mathcal{M}_n\}_{n \in \mathbb{N}}\).

Claim 38

Assume \((\text{Gen}, \text{Sign}, \text{Vrfy})\) is target one-time signature scheme, then it is random one-time signature scheme.
“CRH free” schemes

Lemma 39

If \((\text{Gen}, \text{Sign}, \text{Vrfy})\) and \(H\) in Construction 32 are target-one-time signature scheme and TCR respectively, then it is a signature scheme.

Proof:
Focus on the target-one-time signatures.
Show that

- An adversary cannot make the signer use the same \(r\) for signing two different messages.
- Random-one-time signatures suffice for the nodes signatures
- Target-one-time signatures suffice for the leaves signatures