CMSC 858K — Advanced Topics in Cryptography March 18, 2004

Lecture 16

Lecturer: Jonathan Katz *Scribe(s): Chiu Yuen Koo, Nikolai Yakovenko, Jeffrey Blank*

1 Digital Signature Schemes

In this lecture, we introduce the notion of *digital signature schemes*, show a construction of a *one-time* signature scheme based on one-way functions in the standard model [4], and then cover the full-domain-hash (FDH) signature scheme based on trapdoor permutations in the random oracle model [1, 2]. We first define the semantics of a digital signature scheme.

Definition 1 A digital signature scheme consists of a triple of PPT algorithms $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Vrfy})$ such that:

- Gen is a randomized algorithm which, on input security parameter 1^k, generates a pair of keys: a public (verification) key pk, and a secret (signing) key sk.
- Sign, which may be randomized, takes as input a secret key sk and a message m, and generates a signature σ. We write this as $\sigma \leftarrow \mathsf{Sign}_{sk}(m)$.
- Vrfy takes as input a public key, a message, and a (purported) signature; it outputs a single bit b with $b = 1$ indicating acceptance and $b = 0$ indicating rejection. (We assume for simplicity that Vrfy is deterministic.) We write this as $b = \mathsf{Vrfy}_{pk}(m, \sigma)$.

For correctness, we require that for all (pk, sk) output by $\mathsf{Gen}(1^k)$, for all messages m, and for all σ output by $\mathsf{Sign}_{sk}(m)$ we have $\mathsf{Vrfy}_{pk}(m, \sigma) = 1$. $\diamondsuit$

Technically, one also has to specify a message space but this will be implicit in all the schemes we discuss. We sometimes say a signature σ is *valid* (for a particular message m and with respect to a particular public key pk) if $\mathsf{Vrfy}_{pk}(m, \sigma) = 1$.

We now give a notion of security for digital signatures, following the definition first given by [3]. The definition is a rather strong one: we allow an adversary (who is given the public key) to repeatedly ask for signatures on multiple messages of his choice (this is referred to as an "adaptive chosen-message attack"); the adversary succeeds if it can output a valid signature on any message of its choice which was not signed previously (this is called "existential forgery"). Security requires that the success probability of any polynomial-time adversary is negligible. This notion corresponds to *security against existential forgery under adaptive chosen-message attacks*, and is essentially the strongest considered in the literature.[1] One can imagine weakening this definition in several ways; except for introducing the notion of one-time signatures (below) we will not pursue this further here.

[1] Actually, a slightly stronger definition has recently been considered whereby the adversary succeeds even if it outputs a signature σ on a *previously-signed* message, such that σ is valid but not identical to one produced previously by the signer. Many schemes achieve this definition without further modification.

Definition 2 A signature scheme $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Vrfy})$ is *existentially unforgeable under an adaptive chosen-message attack* if for all PPT adversaries A, the following is negligible:

$$\Pr[(pk, sk) \leftarrow \mathsf{Gen}(1^k); (m, \sigma) \leftarrow A^{\mathsf{Sign}_{sk}(\cdot)}(pk) : \mathsf{Vrfy}(m, \sigma) = 1 \wedge m \notin M],$$

where M is the set of messages submitted by A to the Sign oracle. $\diamondsuit$

In other words, A is allowed to submit a polynomial number of message for signing. Even based on these signatures, A should not be able to generate a signature on any message not submitted to the oracle. We will also consider the following weaker definition of security whereby A is only allowed to submit a *single* message to its signing oracle. Formally:

Definition 3 A signature scheme $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Vrfy})$ is a secure *one-time signature scheme* if for all PPT adversaries A, the following is negligible:

$$\Pr[(pk, sk) \leftarrow \mathsf{Gen}(1^k); (m, \sigma) \leftarrow A^{\mathsf{Sign}_{sk}(\cdot)}(pk) : \mathsf{Vrfy}(m, \sigma) = 1 \wedge m \neq m'],$$

where m' is the single message that A submitted to its signing oracle. $\diamondsuit$

2 One-Way Functions

One-way functions are simply functions that are efficient to compute but hard to invert. They form the minimal "hardness" assumption necessary for most of cryptography, including symmetric-key encryption and digital signatures. We give a definition here tailored to a "concrete" security analysis rather than an asymptotic one.

Definition 4 A polynomial-time-computable function f over domain D_f is said to be (t, ε)-one-way if for all adversaries A running it time t we have:

$$\Pr[x \leftarrow D_f; y = f(x); x' \leftarrow A(y) : f(x') = y] \leq \varepsilon.$$

$\diamondsuit$

As an informal example, one may conjecture that 2048-bit RSA is currently (5 years, 2^{-60})-one-way (note that "RSA" does not quite fit into the above framework, but the definition can be easily modified to accommodate it). This means that no adversary running in five years can invert 2048-bit RSA (on a randomly-chosen challenge point, and for randomly generated modulus) with probability greater than 2^{-60}.

3 The Lamport One-Time Signature Scheme

We now show a one-time signature scheme based on one-way functions. Although not very practical, this scheme is important for several reasons: (1) it illustrates that signature schemes (at least weak ones) can be constructed from *one-way functions* in the *standard model* and do not require any sort of "trapdoor" as was initially believed; (2) the scheme is used as a building block in the construction of many other schemes, including "full-fledged" signature schemes secure against existential forgery against adaptive chosen-message attacks. The scheme shown here was suggested by Lamport [4], and we describe it for messages of length ℓ and using a one-way function f defined over domain $\{0, 1\}^k$:

$\underline{\mathsf{Gen}(1^k)}$
for $i = 1$ to ℓ and $b \in \{0,1\}$:
  $x_{i,b} \leftarrow \{0,1\}^k$
  $y_{i,b} = f(x_{i,b})$
$pk \stackrel{\text{def}}{=} \begin{pmatrix} y_{1,0} \cdots y_{\ell,0} \\ y_{1,1} \cdots y_{\ell,1} \end{pmatrix}$
$sk \stackrel{\text{def}}{=} \begin{pmatrix} x_{1,0} \cdots x_{\ell,0} \\ x_{1,1} \cdots x_{\ell,1} \end{pmatrix}$
output (pk, sk)

$\underline{\mathsf{Sign}_{sk}(m)}$
let $m = m_1 \cdots m_\ell$
 with $m_i \in \{0,1\}$
let sk be as before
output $(x_{1,m_1}, \ldots, x_{\ell,m_\ell})$

$\underline{\mathsf{Vrfy}_{pk}(m,\sigma)}$
parse σ as $(x_1, \ldots, x_\ell)$
if $f(x_i) \stackrel{?}{=} y_{i,m_i}$ for $1 \le i \le \ell$
 output 1
else output 0

Theorem 1 *If f is (t,ε)-one-way (for some particular value of k) and requires time t_f to evaluate (in the forward direction), then no adversary running in time $O(t - 2\ell t_f)$ can "break" the one-time security of the scheme with probability better than $2\ell\varepsilon$.*

In particular, since ℓ and t_f are polynomial in k, this means that if f is (asymptotically) one-way then the Lamport scheme is (asymptotically) secure.

Proof Assume to the contrary that there exists an adversary A' running in time $t' = O(t - 2\ell t_f)$ and forging a signature with probability $\varepsilon' > 2\ell\varepsilon$. We construct an adversary A running in time t and inverting f with probability better than ε, a contradiction.

Define algorithm A (which gets a value y and tries to find an $x \in \{0,1\}^k$ such that $f(x) = y$) as follows:

$\underline{A(y)}$
$i^* \leftarrow \{1, \ldots, \ell\}$; $b^* \leftarrow \{0,1\}$
$y_{i^*,b^*} = y$
for all i, b with $1 \le i \le \ell$ and $b \in \{0,1\}$ and $(i,b) \neq (i^*, b^*)$:
  $x_{i,b} \leftarrow \{0,1\}^k$; $y_{i,b} = f(x_{i,b})$
$pk = \begin{pmatrix} y_{1,0} \cdots y_{\ell,0} \\ y_{1,1} \cdots y_{\ell,1} \end{pmatrix}$
run $A'(pk)$ until it requests a signature on $m = m_1 \cdots m_\ell$
if $m_{i^*} = b^*$, abort; otherwise, return the correct signature to A'
eventually, A' outputs a forged signature $(x_1, \ldots, x_\ell)$ on $m'_1 \cdots m'_\ell$
if $m'_{i^*} \neq b^*$ abort; otherwise, output x_{b^*}

In words, A does the following: it first chooses a random index i^* and a random bit b^*. This defines a position in the public key at which A will place the value y that it wants to invert. (Namely, A' sets $y_{i^*,b^*} = y$.) The remainder of the public key is generated honestly. This means that A *can output a correct signature for any message m such that $m_{i^*} \neq b^*$.* Then, A runs A' (giving A' the public key that A prepared) until A' requests a signature on message m. As noted, A can generate a perfectly valid signature as long as $m_{i^*} \neq b^*$. Otherwise, A simply aborts and gives up.

Assuming A has not aborted, it gives the signature thus computed to A' and continues running A' until A' returns a (supposed) forgery $(x_1, \ldots, x_\ell)$ on message $m'_1 \cdots m'_\ell$. Conditioned on the fact that A' has not aborted, this is a valid forgery with probability ε'. A valid forgery in particular means that $f(x_{i^*}) = y_{i^*,m'_{i^*}}$ and also that $m' \neq m$. In this case, if we also have $m'_{i^*} = b^*$ then A has found an inverse for y (since $y_{i^*,b^*} = y$).

We now analyze the probability that A finds a correct inverse. This occurs as long as the following three things happen: (1) A is able to return a correct signature to A' (i.e., $m_{i^*} \neq b^*$); (2) A' outputs a valid forgery; and (3) the forgery satisfies $m'_{i^*} = b^*$. The probability of event (1) is 1/2 since b^* was chosen at random, and is independent of the view of A' up to and including the point when it requests a signature on message m (this follows since *all* entries $y_{i,b}$ [including y_{i^*,b^*}] of the public key are computed by choosing a random element $x_{i,b}$ from the domain of f and then setting $y_{i,b} = f(x_{i,b})$). Conditioned on the fact that event (1) occurs, the probability of event (2) is exactly ε', the assumed probability of forgery for A'. Finally, conditioned on the fact that events (1) and (2) both occur, we must have $m' \neq m$. So there exists at least one position i such that $m_i \neq m'_i$. If this i equals i^* then we are done (since event (1) occurred we know that $m_{i^*} \neq b^*$ so $m'_{i^*} = b^*$). Since m' is ℓ bits long and since the value of i^* is independent of the view of A' up to this point, i is equal to i^* with probability at least $1/\ell$.

Putting everything together, we see that A inverts f with probability $\frac{1}{2} \cdot \varepsilon' \cdot \frac{1}{\ell} = \varepsilon'/2\ell > \varepsilon$. Also, A runs in time (essentially) $t' + (2\ell - 1)t_f < t$. This contradicts the assumed security of f, proving that A' as described cannot exist. ∎

4 Full Domain Hash (FDH)

As we have mentioned in passing previously, "full-fledged" signature schemes can be constructed from the minimal assumption of one-way functions (if you think about it, this is quite an amazing result!). However, constructions based on general assumptions such as one-way functions are not very practical. In fact, there is essentially only one known construction of a secure signature scheme in the standard model which is practical. Thus, we turn to the random oracle model to help us design efficient and provably-secure (albeit with all the caveats of the random oracle model) scheme based on trapdoor permutations. Before presenting the scheme, we define a concrete notion of security for the latter.

Definition 5 Let Gen_{td} represent a generation algorithm for a trapdoor permutation family. We say this family is (t, ε)-secure if for all adversaries A running in time at most t we have:

$$\Pr[(f, f^{-1}) \leftarrow \mathsf{Gen}_{td}; y \leftarrow D_f; x \leftarrow A(f, y) : f(x) = y] \leq \varepsilon,$$

where D_f is the domain/range of f (implicit in the description of f). ♢

We now describe the *full-domain hash* (FDH) signature scheme [1].

$\underline{\mathsf{Gen}(1^k)}$
$(f, f^{-1}) \leftarrow \mathsf{Gen}_{td}$
let $H : \{0,1\}^* \rightarrow D_f$
$pk = (f, H); sk = f^{-1}$
output (pk, sk)

$\underline{\mathsf{Sign}_{sk}(m)}$
output $\sigma = f^{-1}(H(m))$

$\underline{\mathsf{Vrfy}_{pk}(m, \sigma)}$
output 1 iff $f(\sigma) \stackrel{?}{=} H(m)$

It is not hard to see that correctness is satisfied. We now show that the scheme is secure if H is modeled as a random oracle.

Theorem 2 *If Gen_{td} is (t, ε)-secure and f requires time t_f to evaluate, then no adversary running in time $O(t - q_h \cdot t_f)$ can "break" FDH in the sense of existential unforgeability under adaptive chosen-message attack with probability better than $q_h \cdot \varepsilon$ in the random oracle model. Here, q_h is a bound on the number of hash queries made by the adversary.*

Again, since q_h and t_f are polynomial in the security parameter, this means that FDH is asymptotically secure as well.

Proof Assume to the contrary that there exists an adversary A' running in time $t' = O(t - q_h \cdot t_f)$ that succeeds in breaking the scheme with probability $\varepsilon' > q_h \cdot \varepsilon$. We construct an adversary A running in time t and inverting f with probability better than ε, a contradiction. We assume without loss of generality that (1) whenever A' asks a query $\mathsf{Sign}_{sk}(m)$, it has previously asked query $H(m)$; (2) if A' outputs alleged forgery (m, σ), it has previously queried $H(m)$ and has not previously queried $\mathsf{Sign}_{sk}(m)$; and (3) A' never queries the same value twice to H. Construct A (who tries to invert f at point y) as follows:

$\underline{A(f, y)}$
choose $i^* \leftarrow \{1, \ldots, q_h\}$
run $A'(f)$, answering queries to H and Sign_{sk} as follows:
 $\underline{\text{on the } i^{\text{th}} \text{ query } m_i \text{ to } H\text{:}}$
  if $i = i^*$ return y
  else, $x_i \leftarrow D_f$ and return $y_i = f(x_i)$
 $\underline{\text{on query } \mathsf{Sign}_{sk}(m)\text{:}}$
  let i be such that $m = m_i$
   (i.e., m was the i^{th} query to H)
  if $i = i^*$ abort
  otherwise, return x_i
when A' outputs (m^*, σ), find i such that $m^* = m_i$
if $i \neq i^*$ abort
else output σ

We make a number of observations about A. First, if A does not abort before A' outputs its (supposed) forgery, then the simulation provided for A' is perfect: all queries to the random oracle are answered with a point in D_f chosen independently at random (where we use the fact that f is a permutation, and also the fact that y is chosen at random) and all signing queries are answered correctly (since $f(x_i) = H(m_i)$ be construction). Second, if A does not abort during the course of the entire experiment that means $m^* = m_{i^*}$ and hence if A' has output a valid forgery we have $f(\sigma) = H(m_{i^*}) = y$, and thus A succeeds in inverting f at y. Finally, the running time of A is (essentially) $t' + (q_h - 1)t_f \leq t$.

It remains to analyze the probability that A does not abort. Note that A does not abort whenever $m^* = m_{i^*}$ (since in this case A' has also not queried $\mathsf{Sign}_{sk}(m_{i^*})$). Furthermore, the value of i^* is information-theoretically hidden from A' until such time (if any) that A aborts. Since $m^* = m_i$ for *some* $i \in \{1, \ldots, q_h\}$, the probability that $m^* = m_{i^*}$ is exactly $1/q_h$. The probability that A outputs a correct inverse is therefore $\varepsilon'/q_h > \varepsilon$, giving the desired contradiction. ■

4.1 An Improved Security Reduction Using RSA [2]

The proof in the previous section shows that FDH is asymptotically secure. In practice, however, the concrete security bound derived may not be "good enough" (or, put another way, achieving a reasonable level of security may not be "efficient enough"). For example, say we set $q_h \approx 2^{50}$ which simply means that an adversary evaluates SHA-1 on their own computer 2^{50} times (this is a large, but perfectly reasonably, number). If we use a trapdoor permutation which is (5 years, 2^{-60})-secure for sake of argument, say 2048-bit RSA), then the proof given previously shows that an adversary running for 3 years... cannot forge a signature in FDH with probability better than $2^{50} \cdot 2^{-60} = 2^{-10}$, which is not such a great guarantee! Of course, we can always "fix" this by using larger moduli; for example, if we assume that 4096-bit RSA is (5 years, 2^{-120})-secure then we achieve the acceptable probability of forgery $2^{50} \cdot 2^{-120} = 2^{-70}$. In this case, however, our scheme will be less efficient (since we are using a large modulus).

A natural question is: can we design a signature scheme with a better security reduction (say, where the probability of forgery is roughly equal to the probability of inverting the trapdoor permutation)? Or, can we improve our proof of security for the case of FDH? Both of these problems have been considered, and we will focus on the second one here. In particular, we will show that *for the particular case when RSA is used* as the trapdoor permutation for FDH, a better security reduction can be obtained. (The technique relies on some specific algebraic properties of RSA, and extends to other trapdoor permutations, but not to all trapdoor permutations.)

Theorem 3 *If the RSA trapdoor permutation (for some particular choice of key length) is (t, ε)-secure and takes time t_f to evaluate, then no adversary running in time $O(t - q_h \cdot t_f)$ can "break" RSA-FDH with probability better than $O(q_s \cdot \varepsilon)$ in the random oracle model. Here, q_s is a bound on the number of* signature *queries made by the adversary.*

Note that this offers much better security since $q_s \ll q_h$ (it is much harder to get a signer to "obliviously" sign something for you than to evaluate a hash function repeatedly).

Proof Here, we let $f = (N, e)$ be the RSA function (where the modulus N is generated at random, and e is relatively prime to $\varphi(N)$). Assume to the contrary that there exists an adversary A' running in time $t' = O(t - q_h \cdot t_f)$ that succeeds in breaking the scheme with probability $\varepsilon' > 3q_s \cdot \varepsilon$. We construct an adversary A running in time t and inverting f with probability better than ε, a contradiction. We assume without loss of generality that (1) whenever A' asks a query $\mathsf{Sign}_{sk}(m)$, it has previously asked query $H(m)$; (2) if A' outputs alleged forgery (m, σ), it has previously queried $H(m)$ and has not previously queried $\mathsf{Sign}_{sk}(m)$; and (3) A' never queries the same value twice to H or Sign_{sk}. We now construct A (who tries to find $y^{1/e} \bmod N$) as follows:

$\underline{A(N, e, y)}$
run $A'(pk = (N, e))$, answering queries to H and Sign_{sk} as follows:
  $\underline{\text{on the } i^{\text{th}} \text{ query } m_i \text{ to } H:}$
    $x_i \leftarrow \mathbb{Z}_N^*$
    with probability γ set $b_i = 0$ and return $x_i^e \bmod N$
    otherwise (i.e., with probability $1 - \gamma$) set $b_i = 1$ and return $x_i^e \cdot y \bmod N$

on query $\mathsf{Sign}_{sk}(m)$:
let i be such that $m = m_i$
(i.e., m was the i^{th} query to H)
if $b_i = 1$ abort
otherwise, return x_i
when A' outputs (m^*, σ), find i such that $m^* = m_i$
if $b_i = 0$ abort
else output $\sigma/x_i \bmod N$

Here, γ is a parameter we will fix later. For future reference, note that the running time of A is (essentially) $t' + (q_h - 1)t_f \leq t$ (since an RSA exponentiation dominates multiplications and other operations modulo N).

The hash queries of A' can be divided into two classes: "class 0" (with $b = 0$) consists of messages m_i for which A knows $x_i \stackrel{\text{def}}{=} H(m_i)^{1/e}$; thus, A can answer signing queries for messages in this class, but if A' forges a signature for a message in this class it does not help A (since it already knows a "forged signature" for this message anyway). "Class 1" (with $b = 1$) consists of messages for which A knows an x_i such that $x_i^e y = H(m_i)$; now, A cannot answer signing queries for such messages, but if A' forges a signature for a message in this class then A can invert y as follows: if $\sigma = H(m_i)^{1/e}$ then:

$$\sigma/x_i \bmod N = H(m_i)^{1/e}/(x_i^e)^{1/e} = (H(m_i)/x_i^e)^{1/e} = y^{1/e};$$

i.e., σ/x_i is the desired inverse of y.

Now, until such time (if any) that A aborts, the simulation provided to A' is perfect (in particular, all queries to the random oracle are answered with an independent and uniformly-random point in $\mathbb{Z}_N^*$), and furthermore A' has no information about which messages are in class 0 and which are in class 1. Since the probability that A does not abort is given by the product of the probabilities that (1) all the signing queries of A' are for "class 0" messages, and (2) the purported forgery of A' is for a "class 1" message, the probability that A does not abort is $\Pr[\text{no abort}] = \gamma^{q_s}(1-\gamma)$. Choosing $\gamma = \frac{q_s}{q_s+1}$ maximizes this expression and gives $\Pr[\text{no abort}] = e^{-1}/q_s$ (where $e \approx 2.72$ is the base of natural logarithms). Putting everything together, the probability that A outputs a correct inverse is $\varepsilon'/(e \cdot q_s) > \varepsilon$, giving the desired contradiction. ■

References

[1] M. Bellare and P. Rogaway. Random Oracles are Practical: a Paradigm for Designing Efficient Protocols. ACM Conference on Computer and Communications Security, 1993.

[2] J.-S. Coron. On the Exact Security of Full Domain Hash. Crypto 2000.

[3] S. Goldwasser, S. Micali, and R. Rivest. A Digital Signature Scheme Secure Against Adaptive Chosen-Message Attacks. *SIAM J. Computing* 17(2): 281–308, 1988.

[4] L. Lamport. Constructing Digital Signatures from a One Way Function. SRI International Technical Report CSL-98 (October 1979).