Assumptions for Public Key Signature Schemes

In the last two lectures we showed how to build secure public key signature schemes based on RSA and discrete logarithms. One question that remains to be investigated is the following: What is the weakest cryptographic primitive the guarantees the existence of public key signature schemes? Surprisingly, it turns out that to obtain superpolynomially secure public key signature schemes, it is sufficient to assume the existence of superpolynomially secure one-way functions. In contrast, it is commonly believed that secure public key encryption requires the existence of much stronger primitives, such as trapdoor permutations.

In the following two lectures we sketch a somewhat weaker version of this result: A construction of secure public key signature schemes from one-way *permutations*. The construction consists of two stages: First, we use one-way permutations to build intermediate objects called *universal one-way hashes*. Then, using one-way functions and universal one-way hashes, we obtain secure public key signature schemes. We note that universal one-way hashes may also be obtained from one-way functions, but that construction is more difficult.

The construction presented here is not particularly efficient, and therefore not very useful in practice. However, it is interesting because it helps explain the inherent asymmetry between public key encryption and public key signatures.

1 One way functions imply secure one-time signatures

Recall that a public key signature scheme (G, S, V), where G is a key generator, S is a signing algorithm and V is a verifying algorithm is (t, q, ϵ)-secure (against existential forgery attacks) if for all adversaries $A^?$ running in time t and making at most q oracle queries,

$$\Pr_{(pk,sk)\sim G}[V_{pk}(A^{S_{sk}}(pk)) = 1] \leq \epsilon$$

where the message in the output $A^{S_{sk}}(pk)$ must differ from all the oracle queries of $A^?$.

Secure one-bit signatures. We first show how to construct a secure one-bit signature scheme using a one-way function. Note that for one-bit signatures, (t, ∞, ϵ) security is the same as $(t, 1, \epsilon)$ security, as any adversary that makes two distinct queries is left with no signatures to forge. Let $f : \{0,1\}^n \to \{0,1\}^n$ be a one-way function and consider the following signature scheme:

1. Key generation $(pk, sk) \sim G$: Pick x, y uniformly and independently from $\{0,1\}^n$. Let $sk = (x, y)$, $pk = (f(x), f(y))$.
2. Signature S: Let $S_{(x,y)}(0) = x, S_{(x,y)}(1) = y$.
3. Verification V: On input (c, M), accept iff $c = f(M)$.

We show that if f is a (t, ϵ)-secure one-way function, then (G, S, V) is a $(t - O(1), 1, 2\epsilon)$ secure public key signature scheme. For suppose that $A^?$ is an adversary that $(t-O(1), 1, 2\epsilon)$-breaks (G, S, V). Consider the following adversary A' that tries to invert f: On input u,

1. Pick $v \in \{0,1\}^n$ uniformly at random.
2. Pick an input pk for $A^?$ and a signing oracle simulator S as follows:
   - With probability 1/2, let $pk = (f(v), u)$, $S(0) = v$, $S(1) = error$.
   - Otherwise, let $pk = (u, f(v))$, $S(0) = error$, $S(1) = v$.
3. Return the signature part of $A^S(pk)$.

If u is a random output of f, then pk follows the distribution of the public key generated by G. Let q denote the answer to $A^?$'s (unique) oracle query. No matter what $A^?$'s strategy is, $q = v$ with probability exactly 1/2. Now given that $q = v$, $A^?$ succeeds in breaking (G, S, V) only if it outputs a preimage for u. By construction, the event $q = v$ is independent from the event "$A^S(pk)$ succeeds", so $Pr[f(A'(u)) = u] = \frac{1}{2} 2\epsilon = \epsilon$.

One-time secure m-bit signatures. It is natural to attempt an extension of the above construction to m-bit inputs by signing each bit separately and independently. Formally, given a one-way function $f : \{0,1\}^n \to \{0,1\}^n$, we consider the following signature scheme:

1. Key generation $(pk, sk) \sim G$: Pick $x_{ij}, i \in B, 1 \leq j \leq m$ uniformly and independently from $\{0,1\}^n$. Let $sk = (x_{01}, \ldots, x_{1m})$, $pk = (f(x_{01}), \ldots, f(x_{1m}))$.
2. Signature S: For $M \in \{0,1\}^m$, Let $S_{sk}(M) = (x_{M[1]1}, \ldots, x_{M[m]m})$.

Unfortunately, this scheme is not secure against two-query attackers for $m \geq 2$. If a two-query attacker queries the signature oracle on messages 0^m and 1^m, she discovers all of the x_{ij}s, or the whole secret key. Using this information, she can produce a correct signature for any message. However, the scheme is secure against one-query attackers, as the following lemma shows.

LEMMA 1
If f is a (t, ϵ)-secure one-way function, then (G, S, V) is a $(t - O(1), 1, 2m\epsilon)$-secure public key signature scheme.

PROOF: Suppose that $A^?$ is a $(t-O(1), 1, 2m\epsilon)$ adversary for (G, S, V). Here is an adversary A' that attempts to invert f: On input u,

1. Pick indices $i \in B, j \in \{1, \ldots, m\}$ uniformly at random.
2. Pick $x_{01}, \ldots, \hat{x_{ij}}, \ldots, x_{1m} \in \{0,1\}^n$ uniformly and independently.
3. Let $pk = (f(x_{01}), \ldots, f(x_{1m}))$, with $f(x_{ij})$ replaced by u. Define a signing oracle simulator S by $S(M) = (x_{M[1]1}, \ldots, x_{M[m]m})$ if $M[j] \neq i$, and $S(M) = error$, otherwise.
4. Return a_j, the j-th component of the signature in $A^S(pk)$.

Again, if u is a random output of f, then pk follows the distribution of the public key generated by G. Suppose that $A^S(pk)$ produces a successful forgery $(M', S_{pk}(M'))$. Let M denote the message that $A^?$ queries from its oracle. Then there exists an index j' such that $M'[j'] \neq M[j']$. Given that $j' = j$ and $M[j'] \neq i$, the forgery is successful only if $f(a_j) = u$, i.e., only if $A^?$ has produced a pre-image of u.

By our choice of distributions on i and j, the event $j' = j \wedge M[j'] \neq i$ is independent from the event that $A^S(pk)$ produces a successful forgery, so their probabilities multiply and:

$$\Pr_{u \sim f(\{0,1\}^n)}[f(A'(u)) = u] = \Pr_{i,j}[j' = j \wedge M[j'] \neq i] \Pr[A^S(pk) \text{ succeeds}] = \frac{1}{2m} 2m\epsilon = \epsilon,$$

contradicting the (t, ϵ)-security of f. □

2 One-time signatures with shorter keys

Apart from the lack of security against multiple oracle queries, the signature scheme (G, S, V) has another serious drawback: The signature and verification key of a message are many times longer than the message itself. We are interested in modifying the scheme so as to shorten these keys. It is these modified schemes that will be used to construct a signature scheme that is secure against multiple query attackers.

Universal one way hashes. The main ingredient in this construction is a family of functions $\{h_r : \{0,1\}^m \to \{0,1\}^k\}$ which hash m-bit messages to shorter, k-bit strings. Here, r is a random string of length $|r|$ and m is assumed to be much larger than $k + |r|$. The idea is to map the original m-bit message M into a short k-bit hash $h_r(M)$, and apply our one-time signature to the hash instead of the message. Since the hash is much shorter than the message, the key will be shorter as well.

A family of functions $\{h_r : \{0,1\}^l \to \{0,1\}^k : r \in \{0,1\}^{|r|}\}$ is a (t, ϵ)-*universal one way hash* if for every $x \in \{0,1\}^m$ and every circuit A of size less than t:

$$\Pr_r[A(r) \neq x \wedge h_r(A(r)) = h_r(x)] \leq \epsilon.$$

Note that if $m \leq k$, then any family consisting of a single injective function is a $(\infty, 0)$-universal one way hash. We will be interested in values of m for which m is much bigger than k and $|r|$.

Equivalently, we can characterize universal one way hashes as follows: The family $\{h_r\}$ is a universal one way hash if for every adversary A running in time $t + O(1)$ that interacts with an $|r|$-bit uniform random number generator R using the following protocol,

1. A picks a critical input $x \in \{0,1\}^m$.
2. R outputs r chosen uniformly at random.
3. A outputs $x' \in B^m$, where $x' \neq x$.

$h_r(x) = h_r(x')$ with probability at most ϵ.

Shortening the keys. We now construct a signature scheme (G', S', V') that has similar security guarantees as the one-time scheme (G, S, V), but works with shorter keys. If (G, S, V) is a scheme for k-bit messages based on an k-bit one way function, then its keys are of length $2k^2$. Let $\{h_r : \{0,1\}^m \to \{0,1\}^k\}$ be a universal one way hash with $m > 2(k + |r|)^2$, and consider the following security scheme (G', S', V') for m-bit messages:

1. Key generation: $G' = G$
2. Signature S': Pick a random $r \in R$. Let $S'_{sk}(M) = (S_{sk}(r, h_r(M)), r)$.
3. Verification V': Let $V'_{pk}(c, M) = V_{pk}(c, h_r(M))$.

Our choice of m ensures that the keys of (G', S', V') are no longer than the messages.

Lemma 2
If $\{h_r\}$ is a (t, ϵ)-secure universal one way hash and (G, S, V) is $(t, 1, \epsilon')$-secure then (G', S', V') is $(t - O(m), 1, \epsilon + \epsilon')$-secure.

Proof: Let $A^?$ be an adversary that $(t - O(m), 1, \epsilon + \epsilon')$- breaks (G', S', V'). Let (pk, sk) denote the key associated to the attack, M be the query $A^?$ makes to the signing oracle, M' denote the message forged by $A^?$, r denote the random string chosen by the signing oracle on $A^?$'s query and $(S_{sk}(r', h'_r(M')), r')$ denote the signature of the forgery produced by $A^?$.

If $r' \neq r$ or $h_r(M) = h'_r(M')$, then $A^?$ can be used to break the signature scheme (G, S, V). Otherwise, $A^?$ produces an $M' \neq M$ such that $h_r(M') = h_r(M)$, thereby breaking the one way hash. □

3 Security against multiple query attackers

We now show how the last construction can be used to build signature schemes that are secure against attackers using any polynomial number of queries. To illustrate the idea, we start with a scheme that is secure against two-query attacks.

Security against two-query attackers. Let (G, S, V) be a one-time signature scheme. Consider the following scheme (G', S', V'):

1. Key generation: $G' = G$.
2. Signature S': Generate $(pk_0, sk_0), (pk_1, sk_1) \sim G$ and
   - On the first query M_0, output $(pk_0, pk_1, S_{sk}(pk_0, pk_1), S_{sk_0}(M_0))$.
   - On the second query M_1, output $(pk_0, pk_1, S_{sk}(pk_0, pk_1), S_{sk_1}(M_1))$.
3. Verification V': On input $(pk_0, pk_1, c_k, c_M, M)$, accept if and only if $V_{pk}(c_k, pk_0, pk_1)$ accepts and either $V_{pk_0}(c_M) = M$ or $V_{pk_1}(c_M) = M$.

This is not a signature scheme according to our definition, as the signing algorithm maintains state between queries. Ignoring this problem for now, we show that this scheme is secure against two-query attackers.

LEMMA 3
If (G, S, V) *is* $(t, 1, \epsilon)$*-secure, then* (G', S', V') *is* $(t - O(m), 2, 3\epsilon)$*-secure.*

PROOF: Suppose that $A^?$ is an adversary that $(t - O(m), 2, 3\epsilon)$-breaks (G', S', V'). Let M_0 and M_1 denote the two queries that $A^?$ makes to the signing oracle. To produce a forgery $(pk'_0, pk'_1, S_sk(pk'_0, pk'_1), S_{sk'_b}(M))$ with $b \in \{1, 2\}$ and $M \neq M_0, M_1$, $A^?$ can pick between two attacks. If $pk'_0 = pk_0$ and $pk'_1 = pk_1$, $A^?$ must produce either (1) $S_{sk_0}(M)$ or (2) $S_{sk_1}(M)$. Otherwise, $A^?$ must come up with a forgery of (3) (pk_0, pk_1) (signed with sk). For $(pk, sk) \sim G$, at least one of these three events happens with probability ϵ. Any one of the events implies the existence of an adversary that $(t, 1, \epsilon)$-breaks (G, S, V). □

Security against multiple query attackers. We now describe an extension of the above scheme that ensures security against attackers of any polynomial number of queries. The construction is somewhat notationally unpleasant to describe, even though the idea behind the extension is conceptually simple. Again, we start with a one-time scheme (G, S, V) and describe a new scheme $(G' = G, S', V')$. We won't do a formal proof, but an argument along the lines of the previous proof can be used.

To explain how S' works, we introduce some notation. Let T be a complete binary tree of depth $d = \log^2 m$. Each node in T at distance l from the root can be uniquely identified by a string in $\{0, 1\}^t$ in such a manner that u is an ancestor of v if and only if the string corresponding to u is a prefix of the string corresponding to v. Thus the root is the empty string and its two children are the strings 0 and 1.

To each internal node $v \in \{0, 1\}^0 \cup \ldots \cup \{0, 1\}^{d-1}$ in T, we associate the following information:

1. Private information: Private keys sk_{v0} and sk_{v1}
2. Public information: Public keys pk_{v0} and pk_{v_1}, and the signature $S_{sk_v}(pk_{v0}, pk_{v1})$.

The keys are generated according to G' and independently of each other. The signature of the $(k+1)$-th message M_k consists of all the public information leading from the root to the leaf labeled by the binary expansion $\bar{k}$ of k (padded with enough zeros) together with the signature $S_{\bar{k}}(M_k)$. For example, the signature of M_0 is the string:

$$pk_0, pk_1, S_{sk}(pk_0, pk_1); pk_{00}, pk_{01}, S_{sk_0}(pk_{00}, pk_{01}); \ldots;$$
$$pk_{0\ldots00}, pk_{0\ldots01}, S_{sk_{0\ldots0}}(pk_{0\ldots00}, pk_{0\ldots01}); S_{sk_{0\ldots00}}(M_0).$$

Since T has $\Theta(2^{\log^2 m})$ nodes it appears that S' cannot run in polynomial time. However, we can amortize the running time of S' by generating T in an online fashion.

A stateless signature scheme. We now present a final modification to our scheme (G', S', V') that eliminates the need for S' to maintain state between queries. Since there are many more leaves in T than message queries, it is safe to sign a message with public information corresponding to a path between the root and a *random* leaf. This eliminates the need for a message counter.

The other part of the state of S' consists of key pairs (pk_v, sk_v). If we had access to a random oracle R, these could be regenerated by applying the key generator G' to the random seed $R(v)$. However, instead of a random oracle we can use a pseudorandom function, in which case the seed for the pseudorandom function becomes part of the secret key. If any adversary breaks the resulting signature scheme, such an adversary can be used to distinguish a random function from a pseudorandom function.