ETH Zürich, D-INFK
Spring 2018

Prof. Ueli Maurer
Christian Badertscher
Fabio Banfi

Cryptography Foundations
Solution Exercise 5

5.1 The Lamport One-Time Signature Scheme

Let the key-pair distribution be the distribution induced by choosing $x_0^1, x_1^1, x_0^2, x_1^2, \dots x_0^n, x_1^n \in \mathcal{X}$ uniformly at random, setting $y_b^i = f(x_b^i)$ for $b \in \{0,1\}$, $i \in \{1,\dots,n\}$, and letting the verification key and signing key be $v = (y_0^1, y_1^1, \dots y_0^n, y_1^n) \in \mathcal{V}$ and $z = (x_0^1, x_1^1, \dots x_0^n, x_1^n) \in \mathcal{Z}$, respectively. Further let for $m = (m_1, \dots, m_n) \in \mathcal{M}$,

$$\sigma(m, z) = (x_{m_1}^1, x_{m_2}^2, \dots, x_{m_n}^n) \in \mathcal{S}$$

and for $m \in \mathcal{M}$ and $(s_1, \dots, s_n) \in \mathcal{S}$,

$$\tau\big(m, (s_1, \dots, s_n), v\big) = \begin{cases} 1, & \forall i \in \{1,\dots,n\} \quad f(s_i) = y_{m_i}^i \\ 0, & \text{otherwise.} \end{cases}$$

Note that the scheme is correct, i.e., $\tau\big(m, \sigma(m,z), v\big) = 1$ for all $m \in \mathcal{M}$ and all v, z generated according to the key-pair distribution, because $f(x_{m_i}^i) = y_{m_i}^i$ for all $i \in \{1,\dots,n\}$.

Now let $\mathcal{A}$ be an adversary for the signature forgery game for 1 message with success probability α. We can assume without loss of generality that $\mathcal{A}$ asks for exactly one message to be signed: If $\mathcal{A}$ asks for no signature, consider the adversary $\mathcal{A}'$ that does the same as $\mathcal{A}$, but when $\mathcal{A}$ wants to submit a forgery attempt (m', s'), $\mathcal{A}'$ before asks for a signature for an arbitrary $m \neq m'$, ignores the result, and then submits the forgery attempt (m', s'). We then clearly have that $\mathcal{A}'$ also has success probability α.

We define an algorithm for the inversion game for f as follows:

- Take as input a value $y \in \mathcal{Y}$.
- Choose $b' \in \{0,1\}$ and $i' \in \{1,\dots,n\}$ uniformly at random.
- For all $(i,b) \in \big(\{1,\dots,n\} \times \{0,1\}\big) \setminus \{(i',b')\}$, choose $x_b^i \in \mathcal{X}$ uniformly at random, and set $y_b^i \coloneqq f(x_b^i)$.
- Let $y_{b'}^{i'} \coloneqq y$.
- Give $v \coloneqq (y_0^1, y_1^1, \dots y_0^n, y_1^n)$ as verification key to the adversary $\mathcal{A}$.
- When $\mathcal{A}$ asks for a signature for $m \in \mathcal{M}$:
  - If $m_{i'} \neq b'$, give $(x_{m_1}^1, x_{m_2}^2, \dots, x_{m_n}^n)$ as a signature to $\mathcal{A}$.
  - Otherwise, give up.
- When $\mathcal{A}$ submits a forgery attempt $(m', s') \in \mathcal{M} \times \mathcal{S}$, return $s'_{i'}$.

Note that the distribution of v matches the distribution of verification keys in the signature forgery game. Furthermore, the distribution of v does not depend on b' and thus, b' and the query m from $\mathcal{A}$ are independent. This implies that $m_{i'} \neq b'$ with probability $\frac{1}{2}$. In this case, $(x_{m_1}^1, x_{m_2}^2, \dots, x_{m_n}^n)$ is a valid signature for m.

If $\mathcal{A}$ wins the game, s' is a valid signature for m' and $m' \neq m$. Hence, the messages m and m' differ in at least one position. Since i' is uniform and independent from m and m', we have $m_{i'} \neq m'_{i'}$ with probability at least $\frac{1}{n}$. In this case and if $m_{i'} \neq b'$, we have $m'_{i'} = b'$. Since s' is a valid signature for m', we further have $f(s'_{i'}) = y_{b'}^{i'} = y$. Therefore, $s'_{i'}$ is a preimage of y under f. We conclude that the winning probability of our algorithm is at least $\frac{\alpha}{2n}$.

5.2 Signature Schemes from Trapdoor One-Way Permutations

a) Only knowing the RSA public key (n, e) one could select an arbitrary $s \in \mathbb{Z}_n^*$ and compute $m := s^e \mod n$. (m, s) is a valid message-signature pair.

b) Given a concrete message m, the above attack does not work any more. But we can still find a forgery as follows: Choose $r \in \mathbb{Z}_n$ uniformly at random. If $\gcd(r, n) \neq 1$, then the scheme is broken (since we factor n). Otherwise, obtain the signatures s_1,s_2 on messages $m_1 := r$ and $m_2 := m \cdot r^{-1}$, respectively. Due to the homomorphic property of the RSA function, $s := s_2 \cdot s_1 = (m \cdot r^{-1})^d \cdot r^d = m^d$ is a signature on m.

c) The inversion winner W' expects the problem instance parameter p and the problem instance $y = f(x, p)$ for a uniformly chosen value x. Note that y is also uniformly random over the set $\mathcal{X}$ since $f(\cdot, p)$ is a permutation on $\mathcal{X}$. The task of the winner W' is to find x with the help of an assumed winner W of the signature forgery game (in the random oracle model).

To this end, it emulates towards W an execution of the signature forgery game. W' defines p as the public (verification) key and provides it to W. W' also chooses a uniformly random (and independent) integer $j \in \{1, \ldots, t+1\}$. It also manages an internal counter i initially set to 1, an array of messages $(m_1, \ldots, m_{t+1})$, an array of hashes $(y_1, \ldots, y_{t+1})$, and an array of signatures $(s_1, \ldots, s_{t+1})$, all three initially set to $(\perp, \ldots, \perp)$.

The winner W will make two kinds of queries: signing queries and hash queries.

- Upon the ith hash-oracle query m by winner W, define $m_i := m$.
  1. If $i = j$ then do the following: if for all $k < i$, $m_k \neq m_i$, then set $y_i := y$ and return y_i (i.e., the problem instance is embedded into the output of the random oracle). Otherwise, abort the execution (we will say that event $\mathsf{Fail}_{\text{embed}}$ occurs).
  2. Otherwise, if there is an index $k < i$ s.t. $m_k = m_i$, then set $y_i := y_k$, $s_i := s_k$, and return y_i.
  3. In any other case (i.e., $i \neq j$ and m has not been queried before), choose a uniformly random element $s \in \mathcal{X}$, set $s_i := s$ and define (and return) $y_i := f(s_i, p)$. (Note that with this "trick" the reduction knows the pre-image s_i of y_i—i.e., the signature of m_i—without the trapdoor).
  4. Increment i.
- Upon a signing query m by winner W, W' defines $m_i := m$ and proceeds as above (items 1. to 4.) to obtain the corresponding (emulated) random oracle output y_i.
  1. If $y_i = y$ (i.e., winner W request a signature on the problem instance y) then abort the execution. (We will say that event $\mathsf{Fail}_{\text{sign}}$ occurs).
  2. Otherwise, W' has already generated internally an element s_i such that $y_i = f(s_i, p)$ and W' simply outputs this value s_i as the corresponding signature.

Finally, upon the forgery output $(\hat{m}, \hat{s})$ by winner W, W' defines $m_i := \hat{m}$ and proceeds as above to obtain the corresponding (emulated) random oracle output y_i. Now, if $j < i$ then W' outputs $\hat{s}$ and halts. If $j \geq i$, then W' aborts the execution (we will say that event $\mathsf{Fail}_{\text{embed}}$ occurs).

This completes the description of the game winner W'. The analysis follows.

We define the event $\mathsf{Fail}_{\text{embed}}$ in the above random experiment as the event that W' halts and this is due to aborting upon a hash query or upon the forgery query as indicated in the description above. A failure in this case means that the challenge y could not be embedded into the emulated output of the random oracle. We also define the event $\mathsf{Fail}_{\text{sign}}$ as the event that W' halts and this is due to a failure upon a signing-query. This occurs when W requests the pre-image of the challenge y.

Let A be the event that winner W' outputs the pre-image of $y = f(\cdot, p)$ for values p and y defined in the random experiment of W' in the interaction with game G_f^{TOWP}. Let further $\mathcal{Q}$ be the set of messages queried by W to the random oracle (either directly or indirectly upon a signing query).

In the following, we say that $(\hat{m}, \hat{s})$ is a *valid forgery* if it is a valid message-signature pair and W has not requested a signature for $\hat{m}$. For clarity, we explicitly condition on the event that $\mathcal{Q} = Q$.

$$
\begin{aligned}
&\mathsf{Pr}^{W'G_f^{\mathsf{TOWP}}}[A \mid \neg\mathsf{Fail}_{\text{embed}}, \mathcal{Q} = Q] \\
&= \mathsf{Pr}^{W'G_f^{\mathsf{TOWP}}}[\neg\mathsf{Fail}_{\text{sign}} \wedge f(\hat{s}, p) = y \mid \neg\mathsf{Fail}_{\text{embed}}, \mathcal{Q} = Q] \\
&\geq \mathsf{Pr}^{W'G_f^{\mathsf{TOWP}}}[\neg\mathsf{Fail}_{\text{sign}} \wedge (\hat{m}, \hat{s}) \text{ valid forgery} \wedge \hat{m} = m_j \mid \neg\mathsf{Fail}_{\text{embed}}, \mathcal{Q} = Q] \\
&\overset{(1)}{=} \mathsf{Pr}^{W'G_f^{\mathsf{TOWP}}}[(\hat{m}, \hat{s}) \text{ valid forgery} \wedge \hat{m} = m_j \mid \neg\mathsf{Fail}_{\text{embed}}, \mathcal{Q} = Q] \\
&\overset{(2)}{=} \frac{1}{|Q|} \cdot \mathsf{Pr}^{W'G_f^{\mathsf{TOWP}}}[(\hat{m}, \hat{s}) \text{ valid forgery} \mid \neg\mathsf{Fail}_{\text{embed}}, \mathcal{Q} = Q] = \frac{\alpha}{|Q|}.
\end{aligned}
$$

Equality (1) follows since the event that a valid forgery $(\hat{m}, \hat{s})$ occurs and that $\hat{m} = m_j$ includes in particular $\neg\mathsf{Fail}_{\text{sign}}$. This is because a forgery is only valid if no signature for $\hat{m}$ — and thus for m_j (and all m_k equal to $\hat{m}$, $k > j$) — has been requested by W and hence there is no reason for W' to abort upon a signature query by W.

The final step (2) follows from the following observation: By construction, we have $\hat{m} \in Q$ (and hence has at least one element). Given $\neg\mathsf{Fail}_{\text{embed}}$ (i.e., that the challenge y was successfully embedded as an output of the random oracle), the probability that m_j equals a particular message $m \in Q$ is $\frac{1}{|Q|}$, since j is chosen independently and uniformly at random from $\{1, \ldots, t+1\}$ and for each message $m \in Q$ there is exactly one smallest index k s.t. $m_k = m$ (and hence, conditioned on a successful embedding, there are $|Q|$ possible indexes for j). Finally, this equals the advantage of the assumed game winner W since the interaction in this case is indistinguishable from an interaction with the actual forgery game.

Finally, the probability $\mathsf{Pr}^{W'G_f^{\mathsf{TOWP}}}[\neg\mathsf{Fail}_{\text{embed}} \mid \mathcal{Q} = Q]$ (i.e., that indeed the challenge y can be embedded as a random oracle output), is lower bounded by observing that choosing j uniformly at random, there are $|Q|$ places in the array of messages that allow to embed y, hence $\mathsf{Pr}^{W'G_f^{\mathsf{TOWP}}}[\neg\mathsf{Fail}_{\text{embed}} \mid \mathcal{Q} = Q] = \frac{|Q|}{t+1}$.

Since if we fail to embed, the probability of W' winning is zero, we get

$$
\mathsf{Pr}^{W'G_f^{\mathsf{TOWP}}}[A \mid \mathcal{Q} = Q] = \mathsf{Pr}^{W'G_f^{\mathsf{TOWP}}}[A \mid \neg\mathsf{Fail}_{\text{embed}}, \mathcal{Q} = Q] \cdot \frac{|Q|}{t+1} \geq \frac{\alpha}{t+1}.
$$

Since the above calculations hold conditioned on any set Q (with $|Q| > 1$) it holds also for distributions over such sets as induced by a game winner W.

5.3 The Merkle-Damgård Construction

a) An easy collision is given by $x = 0$ and $y = (0, 0)$. To see this note that $\hat{x} = \hat{y} = (0, \ldots, 0) \in \{0,1\}^m$ and thus $h(x) = f(0, \ldots, 0) = h(y)$.

b) The winner of the collision-finding game for h outputs two messages $x \neq y$ such that $h(x) = h(y)$. From this collision of h we need to compute a collision of f. Let d_x and d_y be the numbers of 0's that have to be appended to x and y, respectively, in order that we get strings that are multiples of m bits long. So, $d_x = -|x| \bmod m$ and $d_y = -|y| \bmod m$. This allows us to write

$$\hat{x} = x \parallel \underbrace{(0, \ldots, 0)}_{d_x \text{ times}} \parallel \langle d_x \rangle \quad \text{and} \quad \hat{y} = y \parallel \underbrace{(0, \ldots, 0)}_{d_y \text{ times}} \parallel \langle d_y \rangle.$$

Moreover, let h_k^x, $1 \leq k \leq s = \frac{|x|+d_x}{m} + 1$, and h_k^y, $1 \leq k \leq t = \frac{|y|+d_y}{m} + 1$, be the outputs of f in the iterative evaluation of $h(x)$ and $h(y)$. We can assume without loss of generality that $t \geq s$. Note that, by definition,

$$h_s^x = h(x) = h(y) = h_t^y.$$

If there exists a $k \in \{1, \ldots, s-1\}$ with $h_{s-k}^x \neq h_{t-k}^y$ and k is the smallest such number, then

$$f(h_{s-k}^x \parallel 1 \parallel \hat{x}_{s-(k-1)}) = h_{s-(k-1)}^x = h_{t-(k-1)}^y = f(h_{t-k}^y \parallel 1 \parallel \hat{y}_{t-(k-1)}),$$

which gives a collision of f. Therefore we can assume in the remainder of the proof that $h_{s-k}^x = h_{t-k}^y$ for all $0 \leq k \leq s-1$. We proceed by considering three cases. First suppose that $|x| \not\equiv |y| \pmod{m} \Leftrightarrow d_x \neq d_y$. Then the last compression stages in the evaluations of $h(x)$ and $h(y)$ already give a collision of f. Concretely,

$$f(\underbrace{h_{s-1}^x \parallel 1 \parallel \langle d_x \rangle}_{=x'}) = h_s^x = h(x) = h(y) = h_t^y = f(\underbrace{h_{t-1}^y \parallel 1 \parallel \langle d_y \rangle}_{=y'})$$

with $x' \neq y'$ as $d_x \neq d_y$. Next we turn to the case where $|x| \equiv |y| \pmod{m}$ but $|x| \neq |y|$. Here it follows that $t > s$ and, with $k = s-1$,

$$f((0, \ldots, 0) \parallel 0 \parallel \hat{x}_1) = h_1^x = h_{s-k}^x = h_{t-k}^y = h_{t-(s-1)}^y = f(h_{t-s}^y \parallel 1 \parallel \hat{y}_{t-(s-1)}),$$

which again gives a collision of f. Finally, suppose $|x| = |y|$. In this case there is a $1 \leq k \leq t = s$ such that $\hat{x}_k \neq \hat{y}_k$. From this we get the collision

$$f((0, \ldots, 0) \parallel \hat{x}_1) = h_1^x = h_1^y = f((0, \ldots, 0) \parallel \hat{y}_1)$$

if $k = 1$ or else the collision

$$f(h_{k-1}^x \parallel 1 \parallel \hat{x}_k) = h_k^x = h_k^y = f(h_{k-1}^y \parallel 1 \parallel \hat{y}_k).$$