Exercise sheet 6, 11 May 2023

1. Explain in your own words how the the Lamport one-time-signature scheme works.

2. Explain in your own words how the the Winternitz one-time-signature scheme works.

3. Consider the simple version of Lamport's one-time signature scheme where bits of the message (rather than the hash of the message) are signed. Let messages have n bits and assume that Alice has published $2n$ hash values as her public key and knows the matching $2n$ secret bit strings representing her private key. Alice uses this signature system multiple times with the same key. Analyze the following two scenarios for your chances of faking a signature on a message M:

   (a) You get to see signatures on random messages.

   (b) You get to specify messages that Alice signs. You may not ask Alice to sign M in this scenario.

   How many signatures do you need on average in order to construct a signature on M?

   How many signatures do you need on average to be able to sign any message?

   Answer these questions in both scenarios.

4. Consider the simple version of Winternitz' one-time signature scheme where bits of the message (rather than the hash of the message) are signed. A user accidentally uses his Winternitz signature key twice. Explain how an attacker can (typically) use these signatures to create a new signature.

5. Let $H : \{0,1\}^* \to \{0,1\}^{256}$ be a cryptographic hash function. We use H as the hash function inside the Lamport and the Winternitz scheme and also as the hash function to compress messages before signing. For the random elements in the secret key you should assume that they each need 256 bits.

(a) We use Lamport's one-time signature, using a Merkle tree to compress the public key (i.e., every public key element in the uncompressed public key becomes a leaf in a Merkle tree, the root is the compressed public key). To sign $H(m)$ of length 256 we need to have a tree with 512 leaves. Compute the size (in bits) of the public key, the private key, and the signature for this scheme.

How many hash function computations are needed in signing and how many in verifying?

(b) We use Winternitz' scheme with parameter $k = 5$, i.e., we process 5 bits at once to sign $H(m)$ of length 256. Compute the size (in bits) of the public key, the private key, and the signature for this scheme.

Hint: Remember that you also need to sign the checksum component.

How many hash function computations are needed in signing and how many in verifying?

(c) Compare the two answers above to using the Winternitz scheme with parameter $k = 8$ (also for $H(m)$ of 256 bits) in terms of the size of keys and signature and also in terms of how many times you need to evaluate H.

6. In the second video it is stated that c for Winternitz decreases if any m_i increases. Explain why this is the case.

7. The HORS (Hash to Obtain Random Subset) signature scheme is an example of a few-time signature scheme. It has integer parameters k, t, and ℓ, uses a hash function $H : \{0,1\}^* \to \{0,1\}^{k \cdot \log_2 t}$ and a one-way function $f : \{0,1\}^\ell \to \{0,1\}^\ell$. For simplicity assume that H is surjective.

   To generate the key pair, a user picks t strings $s_i \in \{0,1\}^\ell$ and computes $v_i = f(s_i)$ for $0 \leq i < t$. The public key is $P = (v_0, v_1, \ldots, v_{t-1})$; the secret key is $S = (s_0, s_1, \ldots, s_{t-1})$.

   To sign a message $m \in \{0,1\}^*$ compute $H(m) = (h_0, h_1, \ldots, h_{k-1})$, where each $h_i \in \{0, 1, 2, \ldots, t-1\}$. The signature on m is $\sigma = (s_{h_0}, s_{h_1}, s_{h_2}, \ldots, s_{h_{k-1}})$.

   To verify the signature, compute $H(m) = (h_0, h_1, \ldots, h_{k-1})$ and $(f(s_{h_0}), f(s_{h_1}), f(s_{h_2}), \ldots, f(s_{h_{k-1}}))$ and verify that $f(s_{h_i}) = v_{h_i}$ for

$0 \leq i < t$.

(a) Let $\ell = 80$, $t = 2^5$, and $k = 3$. How large (in bits) are the public and secret keys? How large is a signature? How many different signatures can the signer generate for a fixed key pair as $H(m)$ varies? Ignore that s-values could collide.

(b) The same public key can be used for $r + 1$ signatures if H is r-subset-resilient, meaning that given r signatures and thus r vectors $\sigma_j = (s_{h_{j,0}}, s_{h_{j,1}}, s_{h_{j,2}}, \ldots, s_{h_{j,k-1}})$, $1 \leq j \leq r$ the probability that $H(m')$ consists entirely of components in $\{h_{j,i} | 0 \leq i < k, 1 \leq j \leq r\}$ is negligible.

Even for $r = 1$, i.e. after seeing just one typical signature, an attacker has an advantage at creating a fake signature. What are the options beyond exact collisions in H?

(c) Let $\ell = 80$, $t = 2^5$, and $k = 3$. Let m be a message so that $H(m) = (h_0, h_1, h_2)$ satisfies that $h_i \neq h_j$ for $i \neq j$. You get to specify messages that Alice signs. You may not ask Alice to sign m.

State the smallest number of HORS signatures you need to request from Alice in order to construct a signature on m? How many calls to H does this require on average? You should assume that H and f do not have additional weaknesses beyond having too small parameters. Explain how you could use under 1000 evaluations of H if you are allowed to ask for two signatures.