Institute of Computer Science, University of Bern
Cryptography, HS 2019

Prof. Christian Cachin
Luca Zanolini

Exercise 12

12.1 Encoding for Textbook-RSA signatures (5 pts)

An approach that has been tried to construct secure textbook RSA-based signatures is to *encode* the message before applying the inverse of the RSA function. Here the signer fixes a public encoding function *encode*:$\{0, 1\}^l \to \mathbb{Z}_N^*$ as part of its public key, and the signature on a message m is $\sigma := \mathit{encode}(m)^d \textbf{ mod } N$.

a) Suppose the encoding function uses $l = c|N|$ for a constant $c \in (0, 1)$. (Here, $|N|$ denotes the bit length of N, that is, $|N| = \lceil \log(N) \rceil$.) Why does this make the Textbook-RSA signature scheme secure against the simple signature forgery attack for an arbitrary message, where any $\sigma^* \in \mathbb{Z}_N^*$ is chosen first?

b) Show that encoded Textbook-RSA is insecure if $\mathit{encode}(m) = 0^8||m||0^{\frac{k}{2}-8}$, for parameters $k = |N|$ and $l = |m| = \frac{k}{2}$, and where m is not the all-0 message).

   Hint. Find three messages m_0, m_1, m_2 such that $\mathit{encode}(m_2) = \mathit{encode}(m_0) \cdot \mathit{encode}(m_1)$.

12.2 One-time signatures (5 pts)

This question discusses the *one-time signature scheme* introduced by Lamport in 1979, a digital signature scheme that is not based on public-key cryptographic methods but rather on any, generic *one-way function*. (For a longer description, see also [KL15; Chap. 12.6].) A one-way function $F : \{0, 1\}^k \to \{0, 1\}^k$ is efficiently computable but hard to invert on average. More precisely, this means for $x \leftarrow \{0, 1\}^k$, i.e., a random, uniformly chosen k-bit string, and $y = F(x)$, and any efficient adversary $\mathcal{A}$ that is given y, the probability that $\mathcal{A}(y)$ outputs x is negligible. In practice a one-way function can be implemented by a collision-free hash function, such as SHA-2 or constructed from a block cipher like AES.

The key-generation function *KeyGen* for a Lamport signature on ℓ-bit messages first selects 2ℓ bit strings of length k each at random, which together form the private key

$$\mathit{sk} = (x_1[0], x_1[1], \ldots, x_\ell[0], x_\ell[1]).$$

Then the one-way function F is applied to each $x_j[b]$ to compute

$$y_j[b] = F(x_j[b]), \quad \text{for } j = 1, \ldots, \ell \text{ and } b \in \{0, 1\}.$$

The public key is the list of 2ℓ such bit strings of length k each,

$$\mathit{pk} = (y_1[0], y_1[1], \ldots, y_\ell[0], y_\ell[1]).$$

To sign an ℓ-bit message $m = (m_1, \ldots, m_\ell)$, algorithm $\mathit{Sign}(\mathit{sk}, m)$ returns

$$\sigma = (x_1[m_1], \ldots, x_\ell[m_\ell]),$$

i.e., one preimage under F as prepared during key generation for each message position, selected by the bit value in that position.

For validating a signature $\sigma = (\sigma_1, \ldots, \sigma_\ell)$ on m, algorithm $\mathit{Ver}(\mathit{pk}, m, \sigma)$ applies F itself to the bit strings in the signature and checks if all of them match *pk*, as determined by the bits in the message, according to

$$y_j[m_j] \overset{?}{=} F(\sigma_j), \quad \text{for } j = 1, \ldots, \ell.$$

This signature scheme is fast because it does not rely on number theoretic constructions, but impractical because it is *one-time*. Why must it be used only once for each public key? Describe an adversary that obtains signatures on *two* messages of its choice and can then forge a signature on any message it likes.