Signature from One-way Functions

Seminar: Digitale Signaturen

Cassius Puodzius

Winter term 2012/2013
March 21, 2013

Contents

1. Introduction

The security of cryptographic schemes relies on some security assumptions that are made. For instance, Rabin signature scheme has as security assumption that factorization is hard, or also GPV's security depends on the hardness of SVP. Some schemes have quite strong security assumptions, what means that a very specific problem has to be difficult in order to the scheme be secure. The goal of this seminar is to show how secure digital signature schemes can be obtained from one-way functions, which configures a very week assumption. Once accomplished to show that such scheme indeed can be built, we obtain secure digital signatures solely based on the existence of one-way functions.

2. One-way functions

A one-way function (OWF) can be efficiently computed on every input, however it is not feasible to invert given an image. Not feasible means that no probabilistic algorithm is able to find, given y, any x' such that $y = f(x')$ in polynomial-time.

It is not known yet, whether OWFs exist, therefore even with this weak security assumption, it is not possible to obtain any signature scheme fully (theoretically) secure. If this kind of function indeed exists, then $\mathcal{P} \neq \mathcal{NP}$, which is a famous open-problem yet. Nevertheless, some candidates endure many years of cryptanalysis and still remain unbroken, what allows digital-signatures from one-way functions in practice.

Part I.
One-time signatures

3. Lamport's scheme

3.1. Key Generation

In the key generation, a signing-key is randomly chosen in the domain of a OWF $f : \mathcal{D} \to \mathcal{I}$. Than the verification-key is obtained by calculating the image of each coefficient of the signing-key such that $\forall i \in \{1, ..., \ell(n)\}$ and $\forall b \in \{0, 1\}$: $s_i^b \xleftarrow{\$} \mathcal{D}$ and $v_i^b = f(s_i^b)$.

$$s_k = \begin{pmatrix} s_1^0 & \dots & s_{\ell(n)}^0 \\ s_1^1 & \dots & s_{\ell(n)}^1 \end{pmatrix} \rightsquigarrow v_k = \begin{pmatrix} v_1^0 & \dots & v_{\ell(n)}^0 \\ v_1^1 & \dots & v_{\ell(n)}^1 \end{pmatrix}$$

3.2. Signature

On the input of a length-restrict message $m \in \{0, 1\}^{\ell(n)}$, the i-th component of the signature is obtained by the i-th m_i of m as follows:

$$\sigma_i = \begin{cases} s_i^0 & \text{if } m_i = 0 \\ s_i^1 & \text{if } m_i = 1 \end{cases}$$

3.3. Verification

The verification of a component of the signature works as follows:

$$v_i^{m_i} = f(\sigma_i)$$

If the signature is valid for all $i \in \{1, ..., \ell(n)\}$, then the signature is accepted, otherwise it is rejected.

3.4. Security proof

Theorem: Lamport's OTS is unforgeable under a *chosen one-message attack* assuming that f is a OWF. *If there exists a forger against Lamport's scheme, then it is possible to construct an adversary $\mathcal{A}$, which inverts a OWF f with non-negligible probabilty.*

Proof. The proof is carried on in three parts: the setup of a signing oracle, the forgery and the final step.

3.4.1. Setup of a signing oracle

For a OWF $f : \mathcal{D} \to \mathcal{I}$, on input of $y \in \mathcal{I}$, an adversary $\mathcal{A}$ randomly chooses $k \xleftarrow{\$} \{0, \ldots, \ell(n)\}$ and $b \xleftarrow{\$} \{0, 1\}$. $\mathcal{A}$ generates a key-pair of Lamports scheme, such that the entry s_b^k is left empty and the entry p_b^k is placed with y.

3.5. Forgery

The signing oracle is not able to sign any message from the forger, because the oracle does not hold an entry on the signing-key for the k-th bit of messages which have this bit equal b. For any other message, the oracle is able to sign, since it has all left entries. Note that the forger can make at most one query, therefore the probabilty of a forgery depends on the sucess probabilty ϵ (non-negligible) of the forger itself and the sucess probability of the signature of the unique query, which happens with probabilty $1/2$, since the k-th bit of the query can be b or $1 - b$.

Hence, this step achieves successfully its end with probability $\epsilon/2$. Otherwise the attack interrupted and ends without inverting y.

3.6. Last step

The adversary $\mathcal{A}$ gets a forgery (m', σ') and check whether it holds a pre-image of y. If the k-th bit of m' is equal b, then σ'_k is a pre-image of y. Otherwise the attack interrupted and ends without inverting y.

A forgery, such that $m'_k = b$, happens with probabilty $1/2\ell(n)$, since the k-th bit of m' can be b or $1-b$ and the forgery differenciate from the query at least at one bit. Hence, a pre-image of y can be obtained with probability $\epsilon/4\ell(n)$, which is non-negligible and contradicts the statement that f is a OWF. □

Corollary 1: If there exists any one-way function then there exists a secure length-restricted one-time signature scheme as well.

Part II.
Full-fledged one-time signature from length-restricted one-time signature

Lamport's scheme is not only one-time but also a length-restricted signature. There are, however, ways to obtain a full-fledged signature from a length-restricted signature. In [1] two techniques are presented: augumenting blocks and hash-and-sign paradigm.

In hash-and-sign paradigm, a collision-free hashing function is used to obtain a digest of a message, and this digest is signed. Since the output size of the collision-free hashing function is fixed, the size of the signature depends only on the size of the signing-key. In [1] it is also shown how to replace the use of collision-free hashing with universal one-way hash functions, which can be constructed from one-way functions, thus it does not add any further requirement to the security assumption made of signatures from one-way functions.

Hence, *Corollary 1* can be extended to *Corollary 2*, which says:

Corollary 2: If there exists any one-way function then there exists a secure full-fledged one-time signature scheme as well.

Part III.
General signatures scheme from one-time signature

4. Authentication-trees

Authentication-trees aim to extend one-time signatures to gereral signatures, which have no restriction in relation to the number of the signatures. Firstly the idea of Tree-based signatures is introduced, although they are not sufficient to fulfill the requirements of general signatures, and then it is showed how to improve these trees to achieve general

signatures.

4.1. Tree-based signatures

A tree-based signature $\Pi(G, S, V)$ is built from a one-time signatures scheme $\Pi'(G', S', V')$, to allow a larger number of signatures. In this construction, the public key is assigned to the root node of the tree and the messages are signed at the leaves, such that each internal node of the tree has some part of the authentication path, in such manner that from the public key, a verifier can verify the path to the signature (leaf) of a message, and use the verification-key assigned to this node in order to verify the signature of the message itself.

4.1.1. Key Generation

A key-pair $(s_k, p_k) \leftarrow G'(1^n)$ is generated and assigned to the root node. Then the key p_k is published.

4.1.2. Signature

On input of $m \in \{0,1\}^{\ell}(n)$, let μ_i be the first i bits (prefix) of m. For all $i \in \{1, ..., \ell(n)\}$, if the node corresponding to μ_i still does not have a pair of children, then two key-pairs are generated under $G'(1^n)$, such that one of them is assigned to the left-child, the other one is assigned to the right-child, and both verification-keys are signed using the signing-key of the current node and assigned to the current node. This extension of the authentication path, described above, is as follows:

$$
\begin{aligned}
(s_{\mu_i|0}, v_{\mu_i|0}) &\leftarrow G(1^k) \\
(s_{\mu_i|1}, v_{\mu_i|1}) &\leftarrow G(1^k) \\
\sigma_i &\leftarrow S_{s_i}(v_{\mu_i|0} || v_{\mu_i|1}) \\
\boxed{\mu_i|0} &\leftarrow \{s_{\mu_i|0}, v_{\mu_i|0}\} \\
\boxed{\mu_i|1} &\leftarrow \{s_{\mu_i|1}, v_{\mu_i|1}\} \\
\boxed{\mu_i} &\leftarrow S_{s_{\mu_i}}(v_{\mu_i|0} || v_{\mu_i|1})
\end{aligned}
$$

When a leaf is reached, the signature σ_m of m is calculated using the signing-key of its node, i.e., $\sigma_m \leftarrow S_{s_m}(m)$. At the end of the process of extending the authentication path and signing the message, the final signature is output as the authentication path and the signature of the message, i.e., $\Sigma = (\underbrace{\{\sigma_j, v_{\mu_j|0}, v_{\mu_j|1}\}_{j=0}}_{auth_{\mu_j}}, \sigma_m)$.

4.2. Verification

To verify the signature Σ of a message $m \in \{0,1\}^{\ell}(n)$, the signature σ_m of m is checked and the authentication path as well. The verification of σ_m is straightforward, whereas

the verification of the authentication path is carried out by verifying the signature of the children's verification-key as follows:

$$\mathrm{V'}_{v_{\mu_j}}(auth_{\mu_{j+1}}) \stackrel{?}{=} ACCEPTED \ \forall j \in \{0, \ell(n)\}$$

where $auth_{\mu_{j+1}}$ is the combination of $auth_{\mu_j}$ and $S_{s_{\mu_i}}(v_{\mu_i|0}||v_{\mu_i|1}$. For $j = 0$, just $S_{s_k}(v_0||v_1$ is considered, where s_k is the signing-key generated in the key generation step.

4.3. Security proof

Theorem: If Π' is a unforgeable signature scheme under a one-time chosen-message attack, then a tree-based scheme Π under Π' is an unforgeable signature under an adaptive chosen-message attack. *If there exists a forger against the tree-based scheme Π', then it is possible to construct an adversary $\mathcal{A}$, which is able to forge the siganture scheme Π with non-negligible probabilty.*

Proof. The proof is carried on in three parts: the setup of a signing oracle, the forgery(divided into two cases) and the final step.

4.3.1. Setup of a signing oracle

The oracle must be able to sign messages of size $\ell(n)$, therefore the number of keys involved in a signature is upper-bounded by $2\ell(n) + 1$. Since the forger must execute in polynomial-time, it is allowed to make a polynomial number of queries at most (say t). Thus $\Omega(n) = (2\ell(n)+1)\cdot t$ is a upper-bound of the number of nodes that the oracle's tree has, furthermore $\Omega(n)$ is polynomially large.Than $\mathcal{A}$ randomly chooses $j \stackrel{\$}{\leftarrow} \{0, \ldots, \Omega(n)\}$, i.e., a node which might be requested during the forgery step.

4.4. Forgery

On input of j and a one-time verification-key v_f, to whom the forgery has to be done, the forgery step is divided into two cases depending on the type of node referenced by j:

1. The node is the root or an internal node
2. The node is a leaf

Case 1

A request in this node happens due to the authentication path. If a request has never been made, then two key-pairs must be generated using G', and both verification-key must be signed (together) by v_f, which is accomplished making a single query to the signing oracle of Π'. Otherwise, nothing has to be done, because the authentication path for this node is already built, what means that there is no need for another query to the signing oracle of Π'.

Case 2

A request in this node happens due to the signature of a query m_q. If m_q has never appeared, its signature under v_f can be obtained forwarding this query to the signing oracle of Π' and than the signature σ_q replied by the oracle is stored. Otherwise, the stored σ_q is output, because each leaf has to sign always the same message (another message leads to another path, and thefore another leaf), what means that there is no need for another query to the signing oracle of Π'

4.5. Last step

If a forgery of Π is output, what happens with non-negligible probability ϵ, then it is checked whether the forgery happend in the chosen node j, what happens with probability $1/\Omega(n)$ since it must have a forgery at least in one of the nodes of the tree. Following the two cases of the forgery step, the output depends whether the node j was a leaf or not.

Case 1: The forgery (m_f, σ_f) is the pair $(v_{\mu_j|0}||v_{\mu_j|1}, \sigma_j)$

Case 2: The forgery (m_f, σ_f) is the pair (m_q, σ_q)

Hence, a forgery of Π' can be obtained with probability $\epsilon/\Omega(n)$, which is non-negligible and contradicts the statement that Π' is a unforgeable signature scheme under a one-time chosen-message attack. □

Corollary 3: If there exists any one-way function then there exists a secure (stateful) general signature scheme as well.

4.6. Stateless tree-based signatures

The idea behind stateless *tree-based* signature is that messages are signed at the leaves, therefore if the tree is large enough, leaves can be randomly chosen to sign messages and moreover they are no likely to be picked twice. Since the number of leaves grows exponetialy with the height of the tree, the probability of picking the same leaf twice is exponetially-vanishing. Hence, in order to prove the security of stateless tree-based signatures, the case in which the same leaf is picked twice to sign a message is disregarded, what leads to a prove very similar to the stateful case.

A stateless tree-based signature $\Pi(G, S, V)$ is built from a one-time signatures scheme $\Pi'(G', S', V')$ and a pseudorandom function ensemble $\{f_r : \{0,1\}^* \to \{0,1\}^{|r|}\}_{r\in\{0,1\}^*}$[1]. In this construction, the public key is also assigned to the root node of the tree and the messages are signed at a leaf depending on f_r. The internal nodes, which composes the authentication path, are build on the fly, what relax the condition to store the state of the tree.

[1]

4.6.1. Key Generation

A key-pair $(s_k, p_k) \leftarrow G'(1^n)$ is generated and r is randomly chosen. Then the key p_k is published and (s_k, r) is stored as the private key.

4.6.2. Signature

On input of $m \in \{0,1\}^*$, a path $\rho = \rho_1 \dots \rho_h$ is chosen, where h is the height of the tree. This path can be obtained by:

$$\rho_1 \dots \rho_h \leftarrow f_r(\texttt{select-leaf}, m)$$

where `select-leaf` is a special character.

Than the key-pairs of the authentication path are generated on the fly as follows:

$$(s_{\mu_i|\tau}, v_{\mu_i|\tau}) \leftarrow G(1^n, f_r(\texttt{key-gen}, \mu_i|\tau))$$

for all $i \in \{0, \dots, |r| - 1\}$ and all $\tau \in \{0,1\}$, such that `key-gen` is also a special character. Note that the key-pair generation is not completely random anymore, but depends on the tree path that leads to this node. Therefore, in another signature, the same key-pair is going to be generated in each node.

After, the authentication path is buit in a similar way it is done in the stateful tree, i.e.:

$$\begin{aligned} \sigma_i &\leftarrow S_{s_i}(v_{\mu_i|0}||v_{\mu_i|1}, f_r(\texttt{sign}, \mu_i)) \\ \boxed{\mu_i} &\leftarrow S_{s_{\mu_i}}(v_{\mu_i|0}||v_{\mu_i|1}, f_r(\texttt{sign}, \mu_i)) \end{aligned}$$

such that `sign` is also a special character. Finally, the leaf ρ is used to calculate the signature σ_m of m as follows:

$$\sigma_m \leftarrow S_{s_{\mu_i}}(m, f_r(\texttt{sign}, \rho)$$

At the end of the process of creating the authentication path (generation + authentication), and signing the message, the final signature is output as the authentication path, the authentication of this path and the signature of the message, i.e., $\Sigma = (\rho, \{\underbrace{\sigma_j, v_{\mu_j|0}, v_{\mu_j|1}}_{auth_{\mu_j}}\}_{j=0}, \sigma_m)$.

4.7. Verification

To verify the signature Σ of a message $m \in \{0,1\}^*$, the signature σ_m of m is checked and the authentication path as well. The verification of σ_m is straightforward, whereas the verification of the authentication path is carried out by verifying the signature of the children's verification-key as follows:

$$\text{V'}_{v_{\mu_j}}(auth_{\mu_{j+1}}) \stackrel{?}{=} ACCEPTED \ \forall j \in \{0, \ell(n)\}$$

where $auth_{\mu_{j+1}}$ is the combination of $auth_{\mu_j}$ and $S_{s_{\mu_i}}(v_{\mu_i|0}||v_{\mu_i|1}$. For $j = 0$, just $S_{s_k}(v_0||v_1$ is considered, where s_k is the signing-key generated in the key generation step.

Corollary 4: If there exists any one-way function then there exists a secure general signature scheme as well.

5. Conclusions

Security of digital signatures relies on security assumptions. It has been shown that a secure digital signature scheme can be solely based on the existence of a OWF. It is still not enough to state that such scheme is secure due to the lack of OWF (currently there are only candidates to OWF), however it is a quite weak assumption.

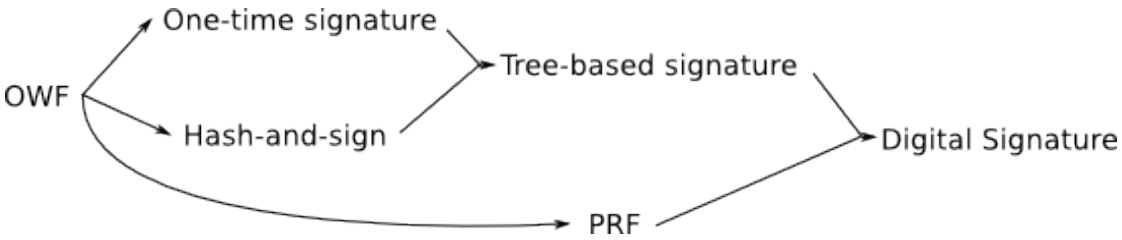

Figure 5 depicts the construction of a general signature scheme from a OWF. Firstly we have seen the Lamport's scheme, which is a length-restricted one-time signature scheme, and proved its security. Than the idea of how to adapt hash-and-sign paradigm was given (further details in [1]), and it was presented how to obtain a stateful general signature scheme using authentication-trees, and its security was proved. Later it was shown how to get rid of the stateful feature of the previous authentication-tree, and the idea of the security proof was also given.

As a final theoretical result, *corollary 4* states that if there exists any one-way function then there exists a secure general signature scheme as well.

References

[1] Oded Goldreich. Draft of chapter on signature schemes, 2003.