Digital Signature

Yu Zhang

Harbin Institute of Technology

Cryptography, Autumn, 2018

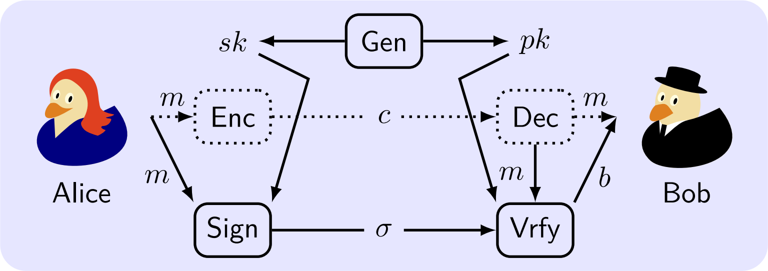
Outline

Content

Digital Signatures – An Overview

- **Digital signature scheme** is a mathematical scheme for demonstrating the authenticity/integrity of a digital message
- allow a **signer** S to "**sign**" a message with its own sk, anyone who knows S's pk can **verify** the authenticity/integrity
- (Comparing to MAC) digital signature is:
  - publicly verifiable
  - transferable
  - non-repudiation
  - but slow
- Q: What are the differences between digital signatures and handwritten signatures?
- Digital signature is NOT the "inverse" of public-key encryption

The Syntax of Digital Signature Scheme



- signature σ, a bit b means valid if $b = 1$; invalid if $b = 0$.
- **Key-generation** algorithm $(pk, sk) \leftarrow \mathsf{Gen}(1^n), |pk|, |sk| \geq n$.
- **Signing** algorithm $\sigma \leftarrow \mathsf{Sign}_{sk}(m)$.
- **Verification** algorithm $b := \mathsf{Vrfy}_{pk}(m, \sigma)$.
- **Basic correctness requirement**: $\mathsf{Vrfy}_{pk}(m, \mathsf{Sign}_{sk}(m)) = 1$.

Defining of Signature Security

The signature experiment $\mathsf{Sigforge}_{\mathcal{A},\Pi}(n)$:

1. $(pk, sk) \leftarrow \mathsf{Gen}(1^n)$.
2. $\mathcal{A}$ is given input 1^n and oracle access to $\mathsf{Sign}_{sk}(\cdot)$, and outputs (m, σ). $\mathcal{Q}$ is the set of queries to its oracle.
3. $\mathsf{Sigforge}_{\mathcal{A},\Pi}(n) = 1 \iff \mathsf{Vrfy}_{pk}(m, \sigma) = 1 \wedge m \notin \mathcal{Q}$.

Definition 1

A signature scheme Π is **existentially unforgeable under an adaptive CMA** if $\forall$ PPT $\mathcal{A}$, $\exists$ negl such that:

$$\Pr[\mathsf{Sigforge}_{\mathcal{A},\Pi}(n) = 1] \leq \mathsf{negl}(n).$$

Q: What's the difference on the ability of adversary between MAC and digital signature? What if an adversary is not limited to PPT?

Content

1 Definitions of Digital Signatures

2 RSA Signatures

3 Digital Signature from the Discrete-Log Problem

4 One-Time Signature Scheme

5 Certificates and Public-Key Infrastructures

Insecurity of "Textbook RSA"

Construction 2

- Gen*: on input* 1^n *run* GenRSA(1^n) *to obtain* N, e, d*.* $pk = \langle N, e \rangle$ *and* $sk = \langle N, d \rangle$*.*
- Sign*: on input* sk *and* $m \in \mathbb{Z}_N^*$, $\sigma := [m^d \bmod N]$*.*
- Vrfy*: on input* pk *and* $m \in \mathbb{Z}_N^*$, $m \stackrel{?}{=} [\sigma^e \bmod N]$*.*

- **A no-message attack**: choose an arbitrary $\sigma \in \mathbb{Z}_N^*$ and compute $m := [\sigma^e \bmod N]$. Output the forgery (m, σ).

$pk = \langle 15, 3 \rangle$, $\sigma = 2$, $m =?$ $m^d =?$

- **Forging a signature on an arbitrary message**: To forge a signature on m, choose a random m_1, set $m_2 := [m/m_1 \bmod N]$, obtain signatures σ_1, σ_2 on m_1, m_2. Q: $\sigma :=$ [____ $\bmod N$] is a valid signature on m.

Hashed RSA

- Gen: a hash function $H : \{0,1\}^* \to \mathbb{Z}_N^*$ is part of public key.
- Sign: $\sigma := [H(m)^d \bmod N]$.
- Vrfy: $\sigma^e \stackrel{?}{=} H(m) \bmod N$.

If H is not efficiently invertible, then the no-message attack and forging a signature on an arbitrary message is difficult.

Insecurity

There is NO known function H for which hashed RSA signatures are secure.

RSA-FDH Signature Scheme: Random Oracle as a **Full Domain Hash (FDH)** whose image size $=$ the RSA modulus $N-1$.

The "Hash-and-Sign" Paradigm

Construction 3

$\Pi = (\mathsf{Gen}_S, \mathsf{Sign}, \mathsf{Vrfy})$, $\Pi_H = (\mathsf{Gen}_H, H)$. *A signature scheme* Π'*:*

- Gen'*: on input* 1^n *run* $\mathsf{Gen}_S(1^n)$ *to obtain* (pk, sk)*, and run* $\mathsf{Gen}_H(1^n)$ *to obtain* s*. The public key is* $pk' = \langle pk, s\rangle$ *and the private key is* $sk' = \langle sk, s\rangle$*.*
- Sign'*: on input* sk' *and* $m \in \{0,1\}^*$*,* $\sigma \leftarrow \mathsf{Sign}_{sk}(H^s(m))$*.*
- Vrfy'*: on input* pk'*,* $m \in \{0,1\}^*$ *and* σ*, output 1* $\iff$ $\mathsf{Vrfy}_{pk}(H^s(m), \sigma) = 1$*.*

Theorem 4

If Π *is existentially unforgeable under an adaptive CMA and* Π_H *is collision resistant, then Construction is existentially unforgeable under an adaptive CMA.*

Proof of Security of “Hash-and-Sign” Paradigm

Idea: a forgery must involve either finding a collision in H or forging a signature with respect to Π.

Proof.

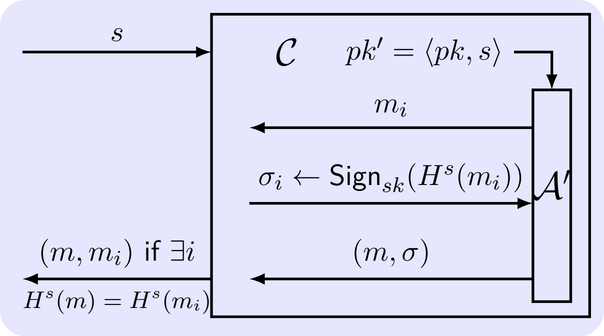
$\mathcal{A}'$ attacks Π' and output (m, σ), $m \notin \mathcal{Q}$.
SF: $\mathsf{Sigforge}_{\mathcal{A}',\Pi'}(n) = 1$.
coll: $\exists m' \in \mathcal{Q}$, $H^s(m') = H^s(m)$.

$$\Pr[\mathsf{SF}] = \Pr[\mathsf{SF} \wedge \mathsf{coll}] + \Pr[\mathsf{SF} \wedge \overline{\mathsf{coll}}] \leq \Pr[\mathsf{coll}] + \Pr[\mathsf{SF} \wedge \overline{\mathsf{coll}}].$$

Reduce $\mathcal{C}$ for Π_H to $\mathcal{A}'$. $\Pr[\mathsf{coll}] = \Pr[\mathsf{Hashcoll}_{\mathcal{C},\Pi_H}(n) = 1]$.
Reduce $\mathcal{A}$ for Π to $\mathcal{A}'$. $\Pr[\mathsf{SF} \wedge \overline{\mathsf{coll}}] = \Pr[\mathsf{Sigforge}_{\mathcal{A},\Pi}(n) = 1]$.
So both $\Pr[\mathsf{coll}]$ and $\Pr[\mathsf{SF} \wedge \overline{\mathsf{coll}}]$ are negligible. $\square$

Proof (Cont.)

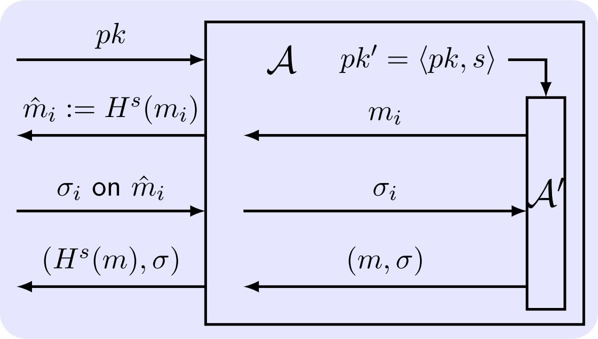
Reduce $\mathcal{C}$ for Π_H to $\mathcal{A}'$. $\mathcal{A}'$ queries the signature σ_i of i-th message m_i, $i = 1, \ldots, |\mathcal{Q}|$.



$\Pr[\mathsf{coll}] = \Pr[\mathsf{Hashcoll}_{\mathcal{C},\Pi_H}(n) = 1]$.

Proof (Cont.)

Reduce $\mathcal{A}$ for Π to $\mathcal{A}'$.

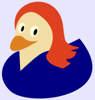


$\Pr[\mathsf{SF} \wedge \overline{\mathsf{coll}}] = \Pr[\mathsf{Sigforge}_{\mathcal{A},\Pi}(n) = 1]$.

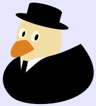
Content

Identification Schemes

An identification scheme $\Pi = (\mathsf{Gen}, \mathcal{P}_1, \mathcal{P}_2, \mathcal{V})$ is a 3-round protocol between the prover and the verifier. The attacker can do eavesdropping and has an access to an oracle Trans_{sk} to learn (I, r, s) by executing the protocol as a verifier.



Prover (sk)



Verifier (pk)

$(I, \mathsf{st}) \leftarrow \mathcal{P}_1(sk) \xrightarrow{\quad I \quad}$

$\xleftarrow{\quad r \quad} \; r \leftarrow \Omega_{pk}$

$s := \mathcal{P}_2(sk, \mathsf{st}, r) \xrightarrow{\quad s \quad}$

$\mathcal{V}(pk, r, s) \stackrel{?}{=} I$

Identification Schemes: Definition

The identification experiment $\mathsf{Ident}_{\mathcal{A},\Pi}(n)$:

1. $(pk, sk) \leftarrow \mathsf{Gen}(1^n)$.
2. $\mathcal{A}$ is given input 1^n and oracle access to $\mathsf{Trans}_{sk}(\cdot)$, and outputs a message I.
3. A uniform challenge r is chosen and given to $\mathcal{A}$, and $\mathcal{A}$ outpus s. ($\mathcal{A}$ may continue to query the oracle.)
4. $\mathsf{Ident}_{\mathcal{A},\Pi}(n) = 1 \iff \mathcal{V}(pk, r, s) \stackrel{?}{=} I$.

Definition 5

An identification scheme $\Pi = (\mathsf{Gen}, \mathcal{P}_1, \mathcal{P}_2, \mathcal{V})$ is **secure** if $\forall$ PPT $\mathcal{A}$, $\exists$ negl such that:

$$\Pr[\mathsf{Ident}_{\mathcal{A},\Pi}(n) = 1] \leq \mathsf{negl}(n).$$

The Fiat-Shamir Transform

The Fiat-Shamir transform constructs a (non-interactive) signature scheme by letting the signer run the protocol by itself.

Construction 6

Let $\Pi = (\mathsf{Gen}_{\mathsf{id}}, \mathcal{P}_1, \mathcal{P}_2, \mathcal{V})$ *be an identification scheme.*

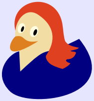
- Gen: $(pk, sk) \leftarrow \mathsf{Gen}_{\mathsf{id}}$. *A function* $H : \{0,1\}^* \to \Omega_{pk}$ *(a set of challenges).*
- Sign: *On input* sk *and* $m \in \{0,1\}^*$, *do*
  1. *Compute* $(I, \mathsf{st}) \leftarrow \mathcal{P}_1(sk)$
  2. *Compute* $r := H(I, m)$
  3. *Compute* $s := \mathcal{P}_2(sk, \mathsf{st}, r)$

  Outpus the signature r, s.
- Vrfy: $I := \mathcal{V}(pk, r, s)$. *Output* $1 \iff H(I, m) \stackrel{?}{=} r$.

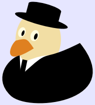
Theorem 7

If Π *is a secure identification scheme and* H *is a random oracle, then the Fiat-Shamir transform results a secure signature scheme.*

The Schnorr Identification Scheme



Prover (x)



Verifier ($\mathbb{G}, q, g, y$)

$$k \leftarrow \mathbb{Z}_q;\ I := g^k \xrightarrow{\quad I \quad}$$

$$\xleftarrow{\quad r \quad} \quad r \leftarrow \mathbb{Z}_q$$

$$s := [rx + k \mod q] \xrightarrow{\quad s \quad}$$

$$g^s \cdot y^{-r} \stackrel{?}{=} I$$

Theorem 8

If the discrete-log problem is hard, then the Schnorr identification scheme is secure.

Proof of the Schnorr Identification Scheme

Idea: If the attacker can let $g^s \cdot y^{-r} = I$, then the attacker can compute x.

Proof.

Reduce $\mathcal{A}'$ inverting y to $\mathcal{A}$ attacking the Schnorr scheme:

1. $\mathcal{A}'$ as a verifier, answering all queries, runs $\mathcal{A}$ as a prover.
2. When $\mathcal{A}$ outputs I, $\mathcal{A}'$ choose $r_1 \in \mathbb{Z}_q$ and give it to $\mathcal{A}$, who responds with s_1.
3. Run $\mathcal{A}$ a second time, send $r_2 \in \mathbb{Z}_q$ to $\mathcal{A}$ who responds with s_2.
4. If $g^{s_1} \cdot h^{-r_1} = I$ and $g^{s_2} \cdot h^{-r_2} = I$ and $r_1 \neq r_2$ then output $x = [(s_1 - s_2) \cdot (r_1 - r_2)^{-1} \mod q]$. Else, output nothing.

$\square$

The Schnorr Signature Scheme

Construction 9

- Gen*: $(\mathcal{G}, q, g) \leftarrow \mathcal{G}(1^n)$. Choose $x \in \mathbb{Z}_q$ and set $y := g^x$. The private key is x and the public key is $(\mathcal{G}, q, g, y)$. A function $H : \{0,1\}^* \to \mathbb{Z}_q$.*
- Sign*: On input x and $m \in \{0,1\}^*$, do*
  1. *Compute $I := g^k$, where a uniform $k \in \mathbb{Z}_q$*
  2. *Compute $r := H(I, m)$*
  3. *Compute $s := [rx + k \mod q]$*

  Outpus the signature (r, s).
- Vrfy*: Compute $I := g^s \cdot y^{-r}$ and output $1 \iff H(I, m) \stackrel{?}{=} r$.*

DSS/DSA

DSS (Digital Signature Standard) uses Digital Signature Algorithm (DSA, a variant of ElGamal signature scheme). [FIPS 186]

Construction 10

$\mathcal{G}$ outputs (p, q, g): (1) p and q are primes with $\|q\| = n$;
(2) $q|(p-1)$ but $q^2 \nmid (p-1)$;
(3) g is a generator of the subgroup of $\mathbb{Z}_p^*$ of order q.

- Gen: $(p, q, g) \leftarrow \mathcal{G}$. *hash function* $H : \{0,1\}^* \rightarrow \mathbb{Z}_q$.
  $x \leftarrow \mathbb{Z}_q$ *and* $y := [g^x \bmod p]$.
  $pk = \langle H, p, q, g, y \rangle$. $sk = \langle H, p, q, g, x \rangle$.
- Sign: $k \leftarrow \mathbb{Z}_q^*$ *and* $r := [[g^k \bmod p] \bmod q]$,
  $s := [(H(m) + xr) \cdot k^{-1} \bmod q]$. *Output* (r, s).
- Vrfy: $u_1 := [H(m) \cdot s^{-1} \bmod q]$, $u_2 := [r \cdot s^{-1} \bmod q]$.
  Output $1 \iff r \stackrel{?}{=} [[g^{u_1} y^{u_2} \bmod p] \bmod q]$.

Correctness and Security of DSS/DSA

$r = [[g^k \bmod p] \bmod q]$ and $s = [(\hat{m}+xr)\cdot k^{-1} \bmod q],\ \hat{m} = H(m).$

$$\begin{aligned} g^{\hat{m}s^{-1}} y^{rs^{-1}} &= g^{\hat{m}\cdot(\hat{m}+xr)^{-1}k} g^{xr\cdot(\hat{m}+xr)^{-1}k} \pmod p \\ &= g^{(\hat{m}+xr)\cdot(\hat{m}+xr)^{-1}k} \pmod p \\ &= g^k \pmod p. \end{aligned}$$

$$[[g^k \bmod p] \bmod q] = r.$$

Security of DSS relies on the hardness of discrete log problem.
The entropy, secrecy and uniqueness of k is critical.

Insecurity

No proof of security for DSS based on discrete log assumption.

Content

One-Time Signature (OTS)

One-Time Signature (OTS): Under a weaker attack scenario, sign only one message with one secret.

The OTS experiment $\mathsf{Sigforge}^{\text{1-time}}_{\mathcal{A},\Pi}(n)$:

1. $(pk, sk) \leftarrow \mathsf{Gen}(1^n)$.
2. $\mathcal{A}$ is given input 1^n and a single query m' to $\mathsf{Sign}_{sk}(\cdot)$, and outputs (m, σ), $m \neq m'$.
3. $\mathsf{Sigforge}^{\text{1-time}}_{\mathcal{A},\Pi}(n) = 1 \iff \mathsf{Vrfy}_{pk}(m, \sigma) = 1$.

Definition 11

A signature scheme Π is **existentially unforgeable under a single-message attack** if $\forall$ PPT $\mathcal{A}$, $\exists$ negl such that:

$$\Pr[\mathsf{Sigforge}^{\text{1-time}}_{\mathcal{A},\Pi}(n) = 1] \leq \mathsf{negl}(n).$$

Lamport's OTS

Idea: OTS from OWF; one mapping per bit.

Construction 12

f is a one-way function.

- Gen*: on input 1^n, for $i \in \{1, \ldots, \ell\}$:*
  1. *choose random $x_{i,0}, x_{i,1} \leftarrow \{0,1\}^n$.*
  2. *compute $y_{i,0} := f(x_{i,0})$ and $y_{i,1} := f(x_{i,1})$.*

$$pk = \begin{pmatrix} y_{1,0} & y_{2,0} & \cdots & y_{\ell,0} \\ y_{1,1} & y_{2,1} & \cdots & y_{\ell,1} \end{pmatrix} \quad sk = \begin{pmatrix} x_{1,0} & x_{2,0} & \cdots & x_{\ell,0} \\ x_{1,1} & x_{2,1} & \cdots & x_{\ell,1} \end{pmatrix}.$$

- Sign*: $m = m_1 \cdots m_\ell$, output $\sigma = (x_{1,m_1}, \ldots, x_{\ell,m_\ell})$.*
- Vrfy*: $\sigma = (x_1, \ldots, x_\ell)$, output $1 \iff f(x_i) = y_{i,m_i}$, for all i.*

Theorem 13

If f is OWF, Π is OTS for messages of length polynomial ℓ.

Example of Lamport's OTS

Signing $m = 011$

$$sk = \begin{pmatrix} x_{1,0} & x_{2,0} & x_{3,0} \\ x_{1,1} & x_{2,1} & x_{3,1} \end{pmatrix} \implies \sigma = \underline{\qquad}$$

$\sigma = (x_1, x_2, x_3)$:

$$pk = \begin{pmatrix} y_{1,0} & y_{2,0} & y_{3,0} \\ y_{1,1} & y_{2,1} & y_{3,1} \end{pmatrix} \implies \begin{array}{l} f(x_1) \stackrel{?}{=} \underline{\qquad} \\ f(x_2) \stackrel{?}{=} \underline{\qquad} \\ f(x_3) \stackrel{?}{=} \underline{\qquad} \end{array}$$

Proof of Lamport's OTS Security

Idea: If $m \neq m'$, then $\exists i^*, m_{i*} = b^* \neq m'_{i*}$. So to forge a signature on m can invert a single y_{i^*,b^*} at least.

Proof.

Reduce $\mathcal{I}$ inverting y to $\mathcal{A}$ attacking Π:

1. Construct pk: Choose $i^* \leftarrow \{1, \ldots, \ell\}$ and $b^* \leftarrow \{0, 1\}$, set $y_{i^*,b^*} := y$. For $i \neq i^*$, $y_{i,b} := f(x_{i,b})$.
2. $\mathcal{A}$ queries m': If $m'_{i_*} = b^*$, stop. Otherwise, return $\sigma = (x_{1,m'_1}, \ldots, x_{\ell,m'_\ell})$.
3. When $\mathcal{A}$ outputs (m, σ), $\sigma = (x_1, \ldots, x_\ell)$, if $\mathcal{A}$ output a forgery at (i^*, b^*): $\mathsf{Vrfy}_{pk}(m, \sigma) = 1$ and $m_{i^*} = b^* \neq m'_{i^*}$, then output x_{i^*,b^*}.

$$\Pr[\mathcal{I} \text{ succeeds}] \geq \frac{1}{2\ell} \Pr[\mathcal{A} \text{ succeeds}]$$

□

Stateful Signature Scheme

Idea: OTS by signing with **"new" key** derived from **"old" state**.

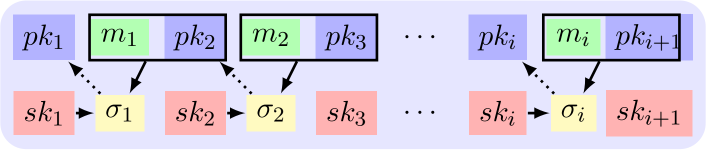
Definition 14 (Stateful signature scheme)

- **Key-generation** algorithm $(pk, sk, s_0) \leftarrow \mathsf{Gen}(1^n)$. s_0 is **initial state**.
- **Signing** algorithm $(\sigma, s_i) \leftarrow \mathsf{Sign}_{sk,s_{i-1}}(m)$.
- **Verification** algorithm $b := \mathsf{Vrfy}_{pk}(m, \sigma)$.

A simple stateful signature scheme for OTS:
Generate (pk_i, sk_i) independently, set $pk := (pk_1, \ldots, pk_\ell)$ and $sk := (sk_1, \ldots, sk_\ell)$.
Start from the state 1, sign the s-th message with sk_s, verify with pk_s, and update the state to $s + 1$.
Weakness: the upper bound ℓ must be fixed in advance.

"Chain-Based" Signatures

Idea: generate keys "on-the-fly" and sign the key chain.



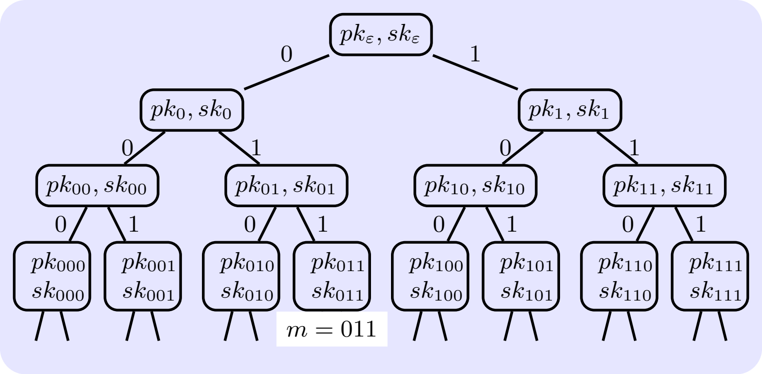
Use a single public key pk_1, sign each m_i and pk_{i+1} with sk_i:

$$\sigma_i \leftarrow \mathsf{Sign}_{sk_i}(m_i \| pk_{i+1}),$$

output $\langle pk_{i+1}, \sigma_i \rangle$, and verify σ_i with pk_i.
The signature is $(pk_{i+1}, \sigma_i, \{m_j, pk_{j+1}, \sigma_j\}_{j=1}^{i-1})$.
Weakness: stateful, not efficient, revealing all previous messages.

"Tree-Based" Signatures

Idea: generate a chain of keys for each message and sign the key chain.



- root is ε (empty string), leaf is a message m, and internal nodes (pk_w, sk_w), where w is the prefix of m.
- each node pk_w "certifies" its children $pk_{w0} \| pk_{w1}$ or w.

A Stateless Solution

Idea: use deterministic randomness to emulate the state of tree.

Use PRF F and two keys k, k' (secrets) to generate pk_w, sk_w:

1. compute $r_w := F_k(w)$.
2. compute $(pk_w, sk_w) := \mathsf{Gen}(1^n; r_w)$, using r_w as random coins.

k' is used to generate r'_w that is used to compute σ_w.

Lemma 15

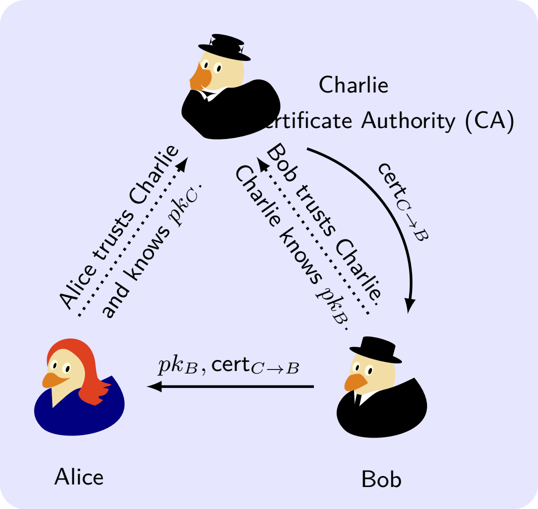
If OWF exist, then $\exists$ OTS (for messages of arbitrary length).

Theorem 16

If OWF exists, then $\exists$ (stateless) secure signature scheme.

Content

Certificates



Certificates $\mathsf{cert}_{C\to B} \stackrel{\text{def}}{=} \mathsf{Sign}_{sk_C}(\text{`Bob's key is } pk_B\text{'})$.

Public-Key Infrastructure (PKI)

- **A single CA**: is trusted by everybody.
  - Strength: simple
  - Weakness: single-point-of-failure
- **Multiple CAs**: are trusted by everybody.
  - Strength: robust
  - Weakness: cannikin law
- **Delegation and certificate chains**: The trust is transitive.
  - Strength: ease the burden on the root CA.
  - Weakness: difficult for management, cannikin law.
- **“Web of trust”**: No central points of trust, e.g., PGP.
  - Strength: robust, work at “grass-roots” level.
  - Weakness: difficult to manage/give a guarantee on trust.

Invalidating Certificates

- **Expiration**: include an *expiry date* in the certificate.

$$\mathsf{cert}_{C\to B} \stackrel{\mathsf{def}}{=} \mathsf{Sign}_{sk_C}(\text{‘bob's key is } pk_B\text{'},\ \text{date}).$$

- **Revocation**: explicitly revoke the certificate.

$$\mathsf{cert}_{C\to B} \stackrel{\mathsf{def}}{=} \mathsf{Sign}_{sk_C}(\text{‘bob's key is } pk_B\text{'},\ \#\#\#).$$

  "###" represents the serial number of this certificate.
  Cumulated Revocation: CA generates *certificate revocation list* (CRL) containing the serial numbers of all revoked certificates, signs CRL with the current date.

Summary

- Textbook RSA, Hashed RSA, Hash-and-Sign
- Identification, Fiat-Shamir Transform, Schnorr Signature, DSS/DSA
- Lamport's OTS, Stateful/Chain-based/Tree-based/Stateless Signature
- Certificates, PKI, CA, Web-of-trust, Invalidation