Formal and Strong Security Definitions:
Digital Signatures

*We know everything about nothing
and nothing about everything ...*

Sven Laur
swen@math.ut.ee

Helsinki University of Technology

Basic theoretical notions

Formal syntax of a signature scheme I

Various domains associated with the signature scheme:

$\mathcal{M}$ – a set of plausible messages;

$\mathcal{S}$ – a set of possible signatures;

$\mathcal{R}$ – random coins used by the signing algorithm.

Parameters used by the signing and verification algorithms:

pk – a public key (public knowledge needed to verify signatures);

sk – a secret key (knowledge that allows efficient creation of signatures).

Formal syntax of a signature scheme II

Algorithms that define a signature scheme:

$\mathcal{G}$ – a randomised key generation algorithm;

$\mathcal{S}_{\mathsf{sk}}$ – a randomised signing algorithm;

$\mathcal{V}_{\mathsf{pk}}$ – a deterministic verification algorithm.

The key generation algorithm $\mathcal{G}$ outputs a key pair $(\mathsf{pk}, \mathsf{sk})$.

The signing algorithm is an efficient mapping $\mathcal{S}_{\mathsf{sk}} : \mathcal{M} \times \mathcal{R} \to \mathcal{S}$.

The verification algorithm is an efficient predicate $\mathcal{V}_{\mathsf{pk}} : \mathcal{M} \times \mathcal{S} \to \{0, 1\}$.

A signature scheme must be functional

$$\forall (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathcal{G},\ \forall m \in \mathcal{M},\ \forall r \in \mathcal{R} : \quad \mathcal{V}_{\mathsf{pk}}(m, \mathcal{S}_{\mathsf{sk}}(m; r)) = 1 \ .$$

Example. RSA-1024 signature scheme

Key generation $\mathcal{G}$:

1. Choose uniformly 512-bit prime numbers p and q.
2. Compute $N = p \cdot q$ and $\phi(N) = (p-1)(q-1)$.
3. Choose uniformly $e \leftarrow \mathbb{Z}^*_{\phi(N)}$ and set $d = e^{-1} \mod \phi(N)$.
4. Output $\mathsf{sk} = (p, q, e, d)$ and $\mathsf{pk} = (N, e)$.

Signing and verification:

$$\mathcal{M} = \mathbb{Z}_N, \quad \mathcal{S} = \mathbb{Z}_N, \quad \mathcal{R} = \emptyset$$

$$\mathcal{S}_{\mathsf{sk}}(m) = m^d \mod N$$

$$\mathcal{V}_{\mathsf{pk}}(m, s) = 1 \quad \Leftrightarrow \quad m = s^e \mod N \ .$$

When is a signature scheme secure?

Signature schemes like cryptosystems have many applications and thus the corresponding security requirements are quite diverse.

- **Key only attack.** Given pk, the adversary creates a valid signature (m, s) in a *feasible* time with a *reasonable* probability.
- **One more signature attack.** Given pk and a list of valid signatures $(m_1, s_1), \ldots, (m_n, s_n)$, the adversary creates a new valid signature (m_{n+1}, s_{n+1}) in a *feasible* time with a *reasonable* probability.
- **Universal forgery.** The adversary must create a valid signature for a message m that is chosen from some prescribed distribution $\mathcal{M}_0$.
- **Existential forgery.** The adversary must create a valid signature for any message m, i.e., there are no limitations on the message.

Standard attack model

Normally a signature scheme must be secure against existential forgeries and against chosen message attack:

1. Challenger generates $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathcal{G}$ and sends pk to Malice.
2. Malice adaptively queries signatures for messages $m_1, \ldots, m_n$.
3. Using pk and a list of queried signatures $(m_1, s_1), \ldots, (m_n, s_n)$ Malice creates and sends a candidate signature (m_{n+1}, s_{n+1}) to Challenger.
4. Challenger outputs 1 only if $\mathcal{V}_{\mathsf{pk}}(m_{n+1}, s_{n+1}) = 1$ and the candidate signature (m_{n+1}, s_{n+1}) is not in the list $(m_1, s_1), \ldots, (m_n, s_n)$.

Success probability

$$\mathsf{Adv}^{\mathsf{forge}}(\mathsf{Malice}) = \Pr\left[\mathsf{Challenger} = 1\right]$$

Show the RSA signature scheme is insecure
What does it mean in practise?

Digital Signatures. Conceptual description

Digital signature is a non-interactive version of the following protocol:

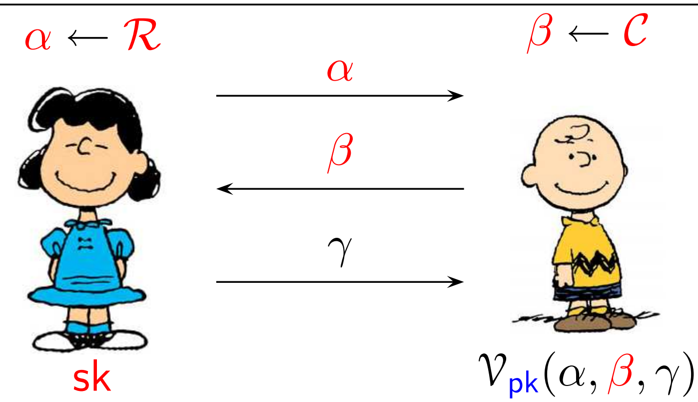
1. Charlie sends a message m to Alice.
2. Alice authenticates herself by proving that
   - she knows the secret key sk,
   - she agrees with the message m.

Differently from the protocol the digital signature must be transferable:

$\Rightarrow$ The signature must be verifiable by other persons.

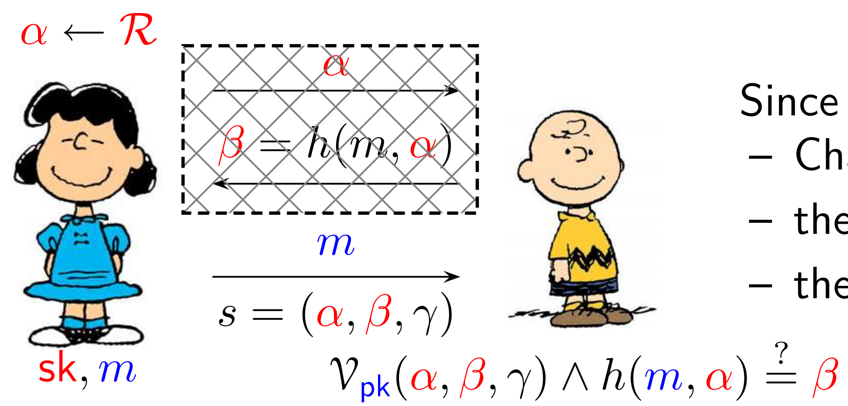
Fiat-Shamir heuristics converts any sigma-protocol to a signature scheme by replacing the second message with a cleverly chosen hash value.

Fiat-Shamir heuristics



If $\mathcal{V}_{pk}(\alpha, \beta, \gamma) = 1$ then
– Alice passes the test.

Bob can efficiently create the protocol transcript by himself.



Since $\beta = h(m, \alpha)$ then
– Charlie cannot cheat,
– the protocol is non-interactive,
– the protocol is transferable.

What are the main differences between
these scenarios?

How to achieve equivalence between
these different scenarios?

Sigma protocols. Zero-knowledge property

Schnorr identification protocol

Alice		Charlie
$x \in \mathbb{Z}_q$	$\alpha = g^k$ $\longrightarrow$	$y = g^x$
$k \leftarrow \mathbb{Z}_q$	$\longleftarrow$ β	$\beta \leftarrow \mathbb{Z}_q$
	$\gamma = k + \beta x$ $\longrightarrow$	

$$g^\gamma = g^k g^{\beta x} \stackrel{?}{=} \alpha y^\beta$$

Simulation Lemma

To generate a transcript (α, β, γ):

1. Choose $\beta \leftarrow \mathbb{Z}_q$ and $\gamma \leftarrow \mathbb{Z}_q$.
2. Compute $\alpha = g^\gamma \cdot y^{-\beta}$.
3. Output (α, β, γ).

} Simulation is perfect.

Sigma protocols. Special Soundness

Schnorr identification protocol

Alice		Charlie
$x \in \mathbb{Z}_q$	$\alpha = g^k$ $\longrightarrow$	$y = g^x$
$k \leftarrow \mathbb{Z}_q$	β $\longleftarrow$	$\beta \leftarrow \mathbb{Z}_q$
	$\gamma = k + \beta x$ $\longrightarrow$	

$$g^\gamma = g^k g^{\beta x} \stackrel{?}{=} \alpha y^\beta$$

Knowledge-extraction Lemma

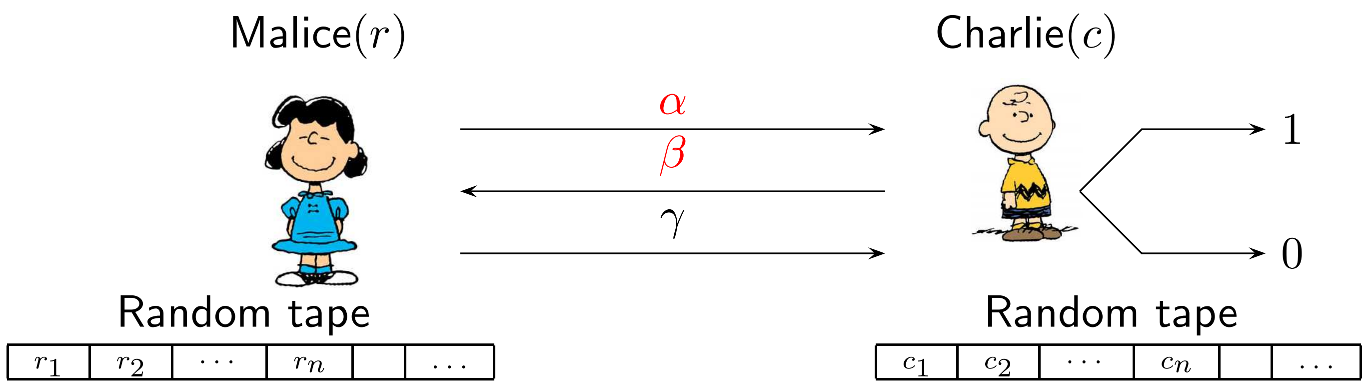
$$\alpha = g^k$$

β ↙ ↘ β'

$$\gamma = k + \beta x \qquad \gamma' = k + \beta' x$$

We can extract the secret key $x = \frac{\gamma - \gamma'}{\beta - \beta'}$.

Knowledge extraction task



Let $A(r, c)$ be the output of Charlie(c) that interacts with Malice(r).

- ▷ Then all matrix elements in the same row $A(r, \cdot)$ lead to same α value.
- ▷ To extract the secret key sk, we must find two ones in the same row.
- ▷ We can compute the entries of the matrix on the fly.

Propose a randomised algorithm for this task!

Estimate the approximate complexity.

Classical algorithm

Rewind:

1. Probe random entries $A(r, c)$ until $A(r, c) = 1$.
2. Store the matrix location (r, c).
3. Probe random entries $A(r, \overline{c})$ in the same row until $A(r, \overline{c}) = 1$.
4. Output the location triple $(r, c, \overline{c})$.

Rewind-Exp:

1. Repeat the procedure Rewind until $c \neq \overline{c}$.
2. Use the Knowledge extraction lemma to extract sk.

Average case complexity I

Assume that the matrix contains ε-fraction of nonzero elements, i.e., Malice convinces Charlie with probability ε. Then on average we make

$$\mathbf{E}[\mathsf{probes}_1] = \varepsilon + 2(1-\varepsilon)\varepsilon + 3(1-\varepsilon)^2\varepsilon + \cdots = \tfrac{1}{\varepsilon}$$

matrix probes to find the first non-zero entry. Analogously, we make

$$\mathbf{E}[\mathsf{probes}_2|r] = \tfrac{1}{\varepsilon_r}$$

probes to find the second non-zero entry. Also, note that

$$\mathbf{E}[\mathsf{probes}_2] = \sum_r \Pr[r] \cdot \mathbf{E}[\mathsf{probes}_2|r] = \sum_r \frac{\varepsilon_r}{\sum_{r'} \varepsilon_{r'}} \cdot \frac{1}{\varepsilon_r} = \frac{1}{\varepsilon} \ ,$$

where ε_r is the fraction of non-zero entries in the r^{th} row.

Average case complexity II

As a result we obtain that the Rewind algorithm does on average

$$\mathbf{E}[\text{probes}] = \tfrac{2}{\varepsilon}$$

probes. Since the Rewind algorithm fails with probability

$$\Pr[\text{failure}] = \frac{\Pr[\text{halting} \wedge c = \overline{c}]}{\Pr[\text{halting}]} \leq \frac{\kappa}{\varepsilon} \qquad \text{where} \qquad \kappa = \frac{1}{q} \ .$$

we make on average

$$\mathbf{E}[\text{probes}^*] = \frac{1}{\Pr[\text{success}]} \cdot \mathbf{E}[\text{probes}] \leq \frac{\varepsilon}{\varepsilon - \kappa} \cdot \frac{2}{\varepsilon} = \frac{2}{\varepsilon - \kappa} \ .$$

Formal security guarantees

Theorem. If Malice manages to convince Charlie with a probability ε over all possible runs of the Schnorr identification scheme, then there exist an extraction algorithm $\mathcal{K}$ that runs in expected time

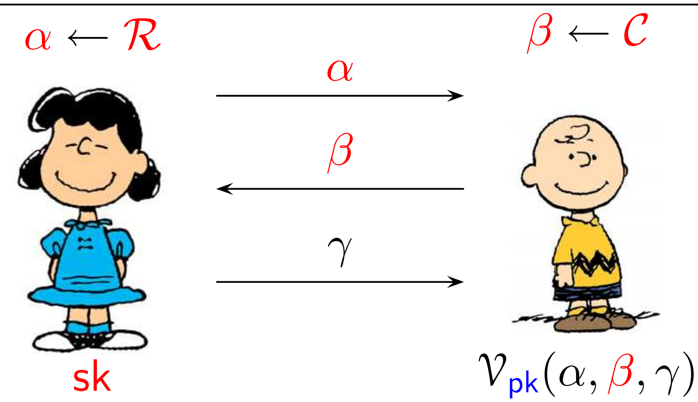
$$\mathbf{E}[t_{\mathcal{K}}] = \Theta\left(\frac{2 \cdot t_{\mathsf{Malice}}}{\varepsilon - \kappa}\right) \qquad \text{where} \qquad \kappa = \frac{1}{q}$$

and extracts the corresponding secret key.

Subjective security guarantee. If I *believe* that finding a particular discrete logarithm $\log(\mathsf{pk})$ is hard then Malice cannot succeed against pk.

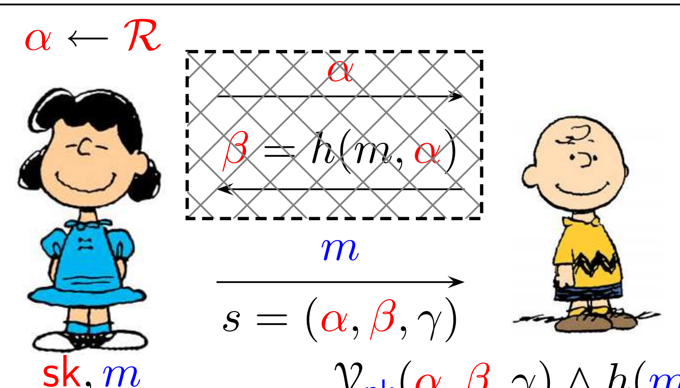
Objective security guarantee. If computing discrete logarithm is hard in the group $\langle g \rangle$ then the Malice success probability over all possible public keys must be small or otherwise Theorem leads to a contradiction.

Fiat-Shamir heuristics



If $\mathcal{V}_{\mathsf{pk}}(\alpha, \beta, \gamma) = 1$ then
– Alice passes the test.

Bob can efficiently create the protocol transcript by himself.



$\mathcal{V}_{\mathsf{pk}}(\alpha, \beta, \gamma) \wedge h(m, \alpha) \stackrel{?}{=} \beta$

Since $\beta = h(m, \alpha)$ then
– Charlie cannot cheat,
– the protocol is non-interactive,
– the protocol is transferable.

What are the main differences between these scenarios?

How to achieve equivalence between these different scenarios?

An obvious choice of the function family

Let $\mathcal{H}_{\mathrm{all}}$ of all functions $\{h : \mathcal{M} \times \mathcal{R} \to \mathbb{Z}_q\}$.

- ▷ If h is chosen uniformly from the function family $\mathcal{H}_{\mathrm{all}}$ then β has the same distribution as in the Schnorr identification protocol.
- ▷ The value $h(m, \alpha)$ is independent form other values $h(m_i, \alpha_i)$.
- ▷ If Malice has only a black-box access to h and must make oracle queries to evaluate $h(m, \alpha)$ then Malice cannot know β before choosing α.

The corresponding model is known as random oracle model.

- ▷ We can always assume that Malice computes β as $h(m, \alpha)$.
- ▷ If Malice makes a single hashing query then Malice succeeds with the same probability as in the Schnorr identification protocol.

General knowledge extraction task

Assume that Malice never queries the same value $h(m_i, \alpha_i)$ twice and that Malice herself verifies the validity of the candidate signature (m_{n+1}, s_{n+1}).

Let ω_0 denote the randomness used by Malice and let $\omega_1, \ldots \omega_{n+1}$ be the replies for the hash queries $h(m_i, \alpha_i)$. Now define

$$A(\omega_0, \omega_1, \ldots, \omega_{n+1}) = \begin{cases} i, & \text{if the } i^{\text{th}} \text{ reply } \omega_i \text{ is used in forgery} \ , \\ 0, & \text{if Malice fails} \ . \end{cases}$$

- ▷ For any $\overline{\boldsymbol{\omega}} = (\omega_0, \ldots, \omega_{i-1}, \overline{\omega}_i, \ldots, \overline{\omega}_{n+1})$, Malice behaves identically up to the i^{th} query as with the randomness $\boldsymbol{\omega}$.
- ▷ To extract the secret key sk, we must find $\boldsymbol{\omega}$ and $\overline{\boldsymbol{\omega}}$ such that $A(\boldsymbol{\omega}) = i$ and $A(\overline{\boldsymbol{\omega}}) = i$ and $\omega_i \neq \overline{\omega}_i$.

Extended classical algorithm

Rewind:

1. Probe random entries $A(\boldsymbol{\omega})$ until $A(r, c) \neq 0$.
2. Store the matrix location $\boldsymbol{\omega}$ and the rewinding point $i \leftarrow A(\boldsymbol{\omega})$.
3. Probe random entries $A(\overline{\boldsymbol{\omega}})$ until $A(\overline{\boldsymbol{\omega}}) = i$.
4. Output the location tuple $(\boldsymbol{\omega}, \overline{\boldsymbol{\omega}})$.

Rewind-Exp:

1. Repeat the procedure Rewind until $\omega_i \neq \overline{\omega}_i$.
2. Use the Knowledge extraction lemma to extract sk.

Average case complexity I

Assume that Malice convinces Charlie with probability ε. Then the results proved for the simplified case imply

$$\mathbf{E}[\mathsf{probes}_1] = \tfrac{1}{\varepsilon} \qquad \text{and} \qquad \mathbf{E}[\mathsf{probes}_2|A(\boldsymbol{\omega}) = i] = \tfrac{1}{\varepsilon_i}$$

where ε_i is the fraction of entries labelled with i. Thus

$$\mathbf{E}[\mathsf{probes}_2] = \sum_{i=1}^{n+1} \Pr\left[A(\boldsymbol{\omega}) = i\right] \cdot \mathbf{E}[\mathsf{probes}_2|A(\boldsymbol{\omega}) = i]$$

$$\mathbf{E}[\mathsf{probes}_2] = \sum_{i=1}^{n+1} \frac{\varepsilon_i}{\varepsilon} \cdot \frac{1}{\varepsilon_i} = \frac{n+1}{\varepsilon} \quad .$$

Average case complexity II

As a result we obtain that the Rewind algorithm does on average

$$\mathbf{E}[\text{probes}] = \tfrac{n+2}{\varepsilon}$$

probes. Since the Rewind algorithm fails with probability

$$\Pr[\text{failure}] = \frac{\Pr[\text{halting} \wedge \omega_i = \overline{\omega}_i]}{\Pr[\text{halting}]} \leq \frac{\kappa}{\varepsilon} \qquad \text{where} \qquad \kappa = \frac{1}{q} \ .$$

we make on average

$$\mathbf{E}[\text{probes}^*] = \frac{1}{\Pr[\text{success}]} \cdot \mathbf{E}[\text{probes}] \leq \frac{\varepsilon}{\varepsilon - \kappa} \cdot \frac{n+2}{\varepsilon} = \frac{n+2}{\varepsilon - \kappa} \ .$$

Formal security guarantees

Theorem. If Malice manages to output valid signature by making at most n queries to the random oracle, then there exist an extraction algorithm $\mathcal{K}$ that runs in expected time

$$\mathbf{E}[t_{\mathcal{K}}] = \Theta\left(\frac{(n+2)\cdot t_{\mathsf{Malice}}}{\varepsilon - \kappa}\right) \qquad \text{where} \qquad \kappa = \frac{1}{q}$$

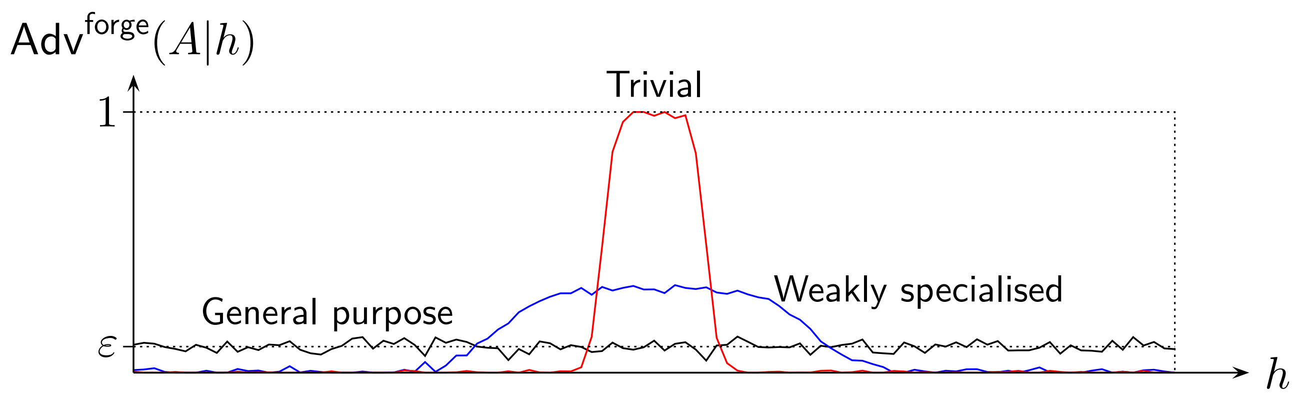
and extracts the corresponding secret key.

Subjective security guarantee. If I *believe* that finding a particular discrete logarithm $\log(\mathsf{pk})$ is hard then Malice cannot succeed against pk.

Objective security guarantee. If computing discrete logarithm is hard in the group $\langle g \rangle$ then the Malice success probability over all possible public keys must be small or otherwise Theorem leads to a contradiction.

What do these security guarantees mean in practise?

Average case nature of advantages



The limit on the average advantage over all functions means:

- An attack algorithm A can be successful on few functions
- For randomly chosen function family $\mathcal{H}$ the corresponding average advantage is comparable with high probability over the choice of $\mathcal{H}$.

Such argumentation does not rule out possibility that Malice can choose adaptively a specialised attack algorithm A based on the description of h.

Security against generic attacks

An adaptive choice of a specialised attack algorithm implies that the attack depends on the description of the hash function and not the family $\mathcal{H}$.

Often, it is advantageous to consider only generic attacks that depend on the description of function family $\mathcal{H}$ and use only black-box access to the function h. Therefore, we can consider two oracles $\mathcal{O}_{\mathcal{H}_{\mathrm{all}}}$ and $\mathcal{O}_{\mathcal{H}}$.

If $\mathcal{H}$ is pseudorandom function family then for any generic attack, we can substitute $\mathcal{H}$ with the $\mathcal{H}_{\mathrm{all}}$ and the success decreases marginally.

Theorem. Security in the random oracle model implies security against generic attacks if $\mathcal{H}$ is a pseudorandom function family.

- ▷ The assumption that Malice uses only generic attacks is subjective.
- ▷ Such an assumption are not universal, i.e., there are settings where this assumption is clearly irrational (various non-instantiability results).

Literature

- E. Käsper and S. Laur and H. Lipmaa. *Black-Box Knowledge Extraction Revisited: Universal Approach with Precise Bounds.* Cryptology ePrint Archive, Report 2006/356.
- M. Bellare and G. Neven. *Multi-Signatures in the Plain Public-Key Modeland a General Forking Lemma.* ACM CCS 2006.
- D. Pointcheval and Jacques Stern. *Security Arguments for Digital Signatures and Blind Signatures.* J. Cryptology 13(3): 361-396, 2000.