On the Multi-User Security of Short Schnorr Signatures

Jeremiah Blocki and Seunghoon Lee

Department of Computer Science, Purdue University

October 10, 2019

Contents

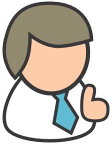
We are now at...

Motivation: Digital Signatures

Motivation: Digital Signatures

Motivation: Digital Signatures

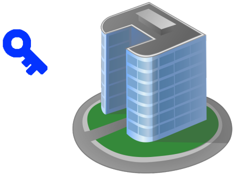
Software update m



Motivation: Digital Signatures

Software update m

pk

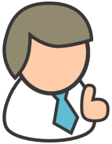


Motivation: Digital Signatures

Software update m

pk

sk

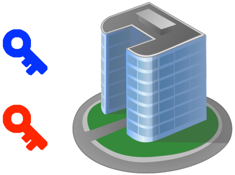


Motivation: Digital Signatures

Software update m

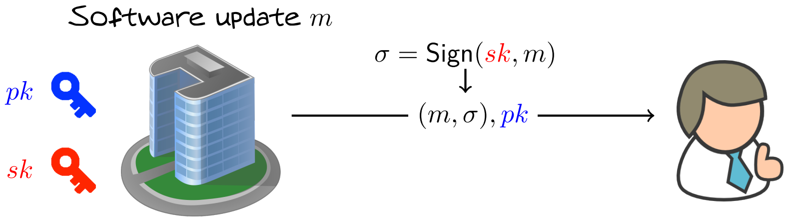
pk

sk

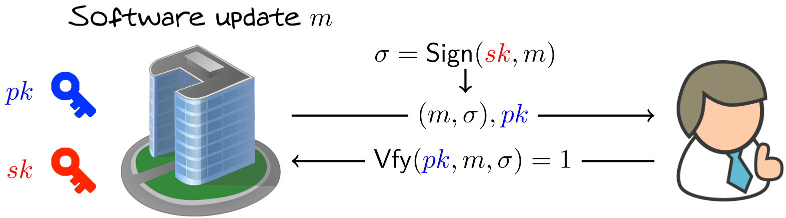


$\sigma = \mathsf{Sign}(sk, m)$

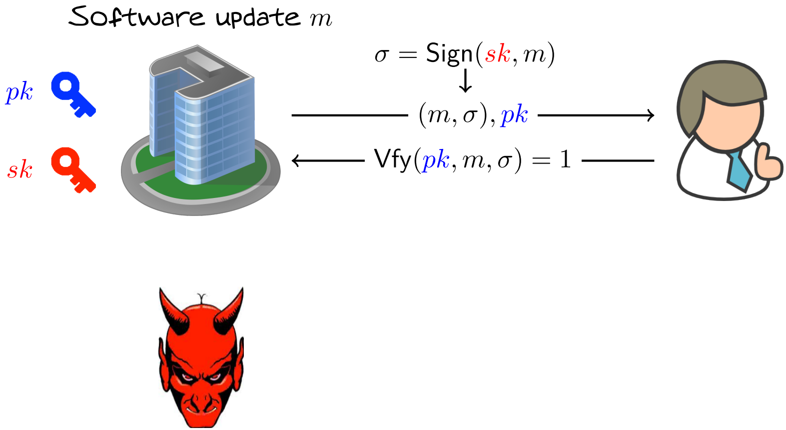
Motivation: Digital Signatures



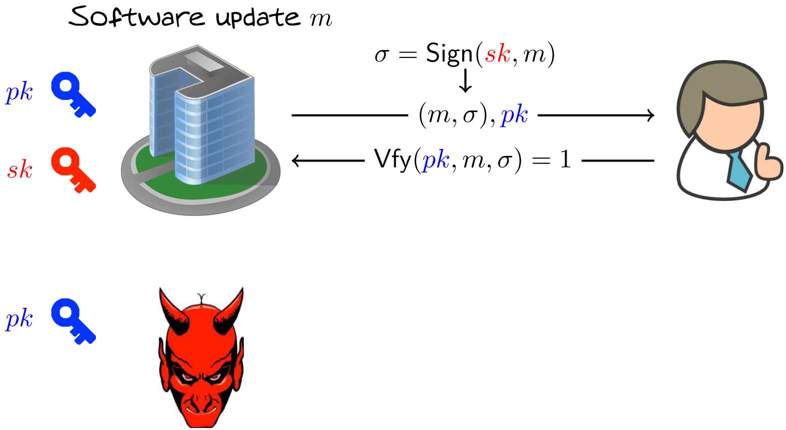
Motivation: Digital Signatures



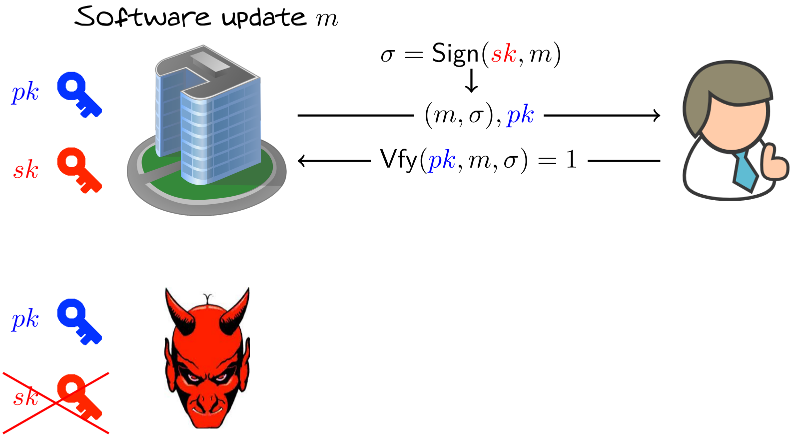
Motivation: Digital Signatures



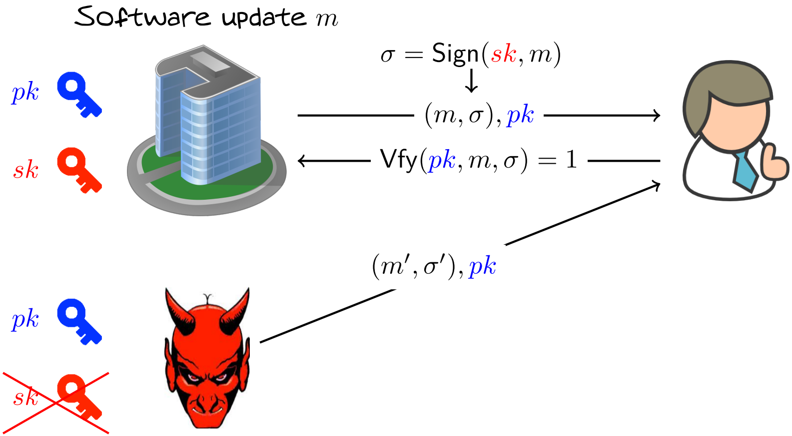
Motivation: Digital Signatures



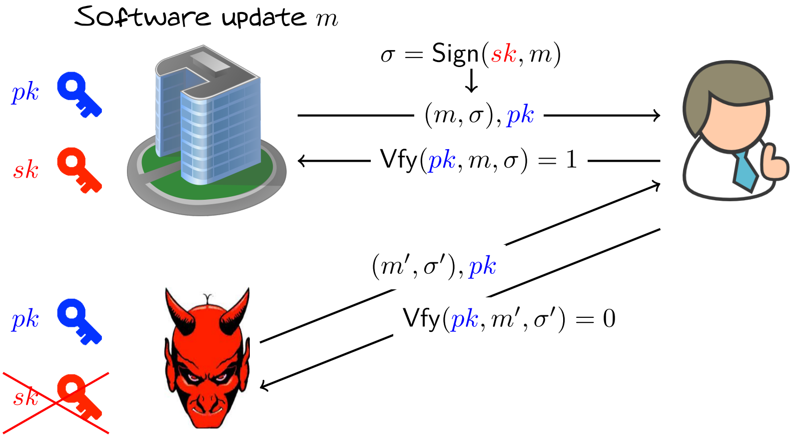
Motivation: Digital Signatures



Motivation: Digital Signatures



Motivation: Digital Signatures



The Schnorr Signature Scheme

- An efficient signature scheme based on discrete logarithms.
- Consider a $2k$-bit prime q, i.e., $q \approx 2^{2k}$.

$\mathsf{Kg}(1^k)$	$\mathsf{Sign}(sk, m)$	$\mathsf{Vfy}(pk, m, \sigma)$
1 : $sk \leftarrow \mathbb{Z}_q$	1 : $r \leftarrow \mathbb{Z}_q$; $I \leftarrow g^r$	1 : $R \leftarrow g^s \cdot pk^{-e}$
2 : $pk \leftarrow g^{sk}$	2 : $e \leftarrow \mathsf{H}(I\|\|m)$	2 : **if** $\mathsf{H}(R\|\|m) = e$ **then**
3 : **return** (pk, sk)	3 : $s \leftarrow r + sk \cdot e \mod q$	3 : **return** 1
	4 : **return** $\sigma = (s, e)$	4 : **else return** 0

- The verification works for a correct signature $\sigma = (s, e)$ because

$$R = g^s \cdot pk^{-e} = g^{s - sk \cdot e} = g^r = I.$$

- The length of the signature: $\underbrace{2k}_{\text{the length of } s} + \underbrace{2k}_{\text{the hash output}} = 4k.$

The "Short" Schnorr Signatures

$\mathsf{Kg}(1^k)$	$\mathsf{Sign}(sk, m)$	$\mathsf{Vfy}(pk, m, \sigma)$
1 : $sk \leftarrow \mathbb{Z}_q$	1 : $r \leftarrow \mathbb{Z}_q$; $I \leftarrow g^r$	1 : $R \leftarrow g^s \cdot pk^{-e}$
2 : $pk \leftarrow g^{sk}$	2 : $e \leftarrow \mathsf{H}(I\|\|m)$	2 : **if** $\mathsf{H}(R\|\|m) = e$ **then**
3 : **return** (pk, sk)	3 : $s \leftarrow r + sk \cdot e \mod q$	3 : **return** 1
	4 : **return** $\sigma = (s, e)$	4 : **else return** 0

$$\sigma = \underbrace{\boxed{\quad s \quad}}_{2k \text{ bits}}\underbrace{\boxed{e = \mathsf{H}(I||m)}}_{2k \text{ bits}}$$

$\downarrow$ ***truncating the hash output by half***

$$\sigma = \underbrace{\boxed{\quad s \quad}}_{2k \text{ bits}}\underbrace{\boxed{e}}_{k \text{ bits}}$$

Signature Length Comparison

Definition

A signature scheme $\Pi = (\mathsf{Kg}, \mathsf{Sign}, \mathsf{Vfy})$ yields k-***bits of security*** if any attacker running in time at most t can forge a signature with probability at most $\varepsilon_t = t/2^k$ and this should hold for all $t \le 2^k$.

Signatures	Signature Length[1]	Security Level	Notes
RSA-FDH	3072	128	NIST recommendation
Schnorr	512	128	
Short Schnorr	384	128?	Our result
BLS	256	128	Computationally expensive
iO	128	128	Completely impractical

[1] Signature lengths and security level are provided in bits

Multi-User Security Definition

- We consider the multi-user security in the "1-out-of-N" setting
- The probability that the attacker can forge ***any one*** of N signatures is negligible
- We define the ***1-out-of-N signature forgery game*** $\mathsf{SigForge}^N_{\mathcal{A},\Pi}(k)$ as follows:
  1. $\mathsf{Gen}(1^k)$ is run N times to obtain keys $(pk_i, sk_i), 1 \le i \le N$.
  2. Adversary $\mathcal{A}$ is given $pk_1, \cdots, pk_N$ and access to oracles $\mathsf{Sign}(sk_j, \cdot), 1 \le j \le N$. The adversary then outputs (m, σ). Let $\mathcal{Q}_j$ denote the set of all queries that $\mathcal{A}$ asked to oracle $\mathsf{Sign}(sk_j, \cdot)$.
  3. $\mathcal{A}$ succeeds if and only if there exists some j such that (1) $\mathsf{Vfy}(pk_j, m, \sigma) = 1$ and (2) $m \notin \mathcal{Q}_j$. In this case the output of the experiment is defined to be 1.

Definition

We say that a signature scheme $\Pi = (\mathsf{Kg}, \mathsf{Sign}, \mathsf{Vfy})$ is (t, N, ϵ)-***MU-UF-CMA secure (multi-user unforgeable against chosen message attack)*** if for every adversary $\mathcal{A}$ running in time at most t, the following bound holds:

$$\Pr\left[\mathsf{SigForge}^N_{\mathcal{A},\Pi}(k) = 1\right] \le \epsilon.$$

Security Proofs of the Schnorr Signatures

	Single-User Security	Multi-User Security
Original Schnorr Signatures	• [PS96] – in the ROM • [NPSW09] – in the GGM • [Seu12, FJS14] – loss of factor q_{RO} seems to be unavoidable	• [GMLS02] – flawed • [KMP16] – in the ROM + GGM
"Short" Schnorr Signatures	• [SJ00] – in the ROM + GGM • [NPSW09] – non-tight reduction	• Our result!

Security Proofs of the Schnorr Signatures

	Single-User Security	Multi-User Security
Original Schnorr Signatures	• [PS96] – in the ROM • [NPSW09] – in the GGM • [Seu12, FJS14] – loss of factor q_{RO} seems to be unavoidable	• [GMLS02] – flawed • [KMP16] – in the ROM + GGM
"Short" Schnorr Signatures	• [SJ00] – in the ROM + GGM • [NPSW09] – non-tight reduction	• Our result!

[Ber15] - "Key-Prefixed" Schnorr signatures

We are now at...

Our Result

We show that the "short" Schnorr signature scheme provides k-bits of security in ***both*** the single and multi-user versions of the signature forgery game.

Theorem (informal)

Any attacker running in time t against the short Schnorr signature scheme

1. ***wins the signature forgery game*** (UF-CMA) ***with probability at most $\mathcal{O}(t/2^k)$, and***
2. ***wins the multi-user signature forgery game*** (MU-UF-CMA) ***with probability at most $\mathcal{O}((t+N)/2^k)$ (where N denote the number of distinct users/public keys)***

in the generic group model (of order $q \approx 2^{2k}$) plus random oracle model.

Why is this important? We don't lose a factor of N in the security reduction!

Example

Suppose that $q \approx 2^{224}$ (i.e., $k = 112$), $N = 2^{32}$, and $t = 2^{80}$.

- Naïve approach: $\epsilon_{\text{MU}} \approx N \cdot t/2^k = 1$
- Our result: $\epsilon_{\text{MU}} \approx (t+N)/2^k = 2^{-32}$

We are now at...

The Generic Group Model [Sho97]

$$\tau : G \longrightarrow \mathbb{G}$$

(multiplicative) cyclic group of prime order $q \approx 2^{2k}$; $G = \langle g \rangle$

set of bit strings of length $\ell = 2k$

- no need to be a group homomorphism
- an adversary has no access to the concrete representation of the group elements/map τ

The Generic Group Model [Sho97]

$$\tau : G \longrightarrow \mathbb{G}$$

(multiplicative) cyclic group of prime order $q \approx 2^{2k}$; $G = \langle g \rangle$

set of bit strings of length $\ell = 2k$

- no need to be a group homomorphism
- an adversary has no access to the concrete representation of the group elements/map τ

Generic Oracles (GO) Initially, $\mathfrak{g} = \tau(g)$ is given.

$\mathfrak{a}, \mathfrak{b} \rightarrow$ `Mult`$(\cdot,\cdot) \longrightarrow \tau(\tau^{-1}(\mathfrak{a}) \cdot \tau^{-1}(\mathfrak{b}))$

$\mathfrak{a} \rightarrow$ `Inv`$(\cdot) \longrightarrow \tau(\tau^{-1}(\mathfrak{a})^{-1})$

Note. $\mathtt{Pow}(\mathfrak{a}, k) = \tau(\tau^{-1}(\mathfrak{a})^{k})$

The Generic Group Model: Justification

- For certain elliptic curve groups the best known attacks are all generic [JMV01, FST10].
- **Heuristic:** experience suggests that protocols with security proofs in the GGM doesn't have inherent structural weaknesses and will be secure as long as we instantiate with a reasonable elliptic curve group.
- Counterexamples are artificially crafted [Den02].

We are now at...

The Known/Partially Known Set in the Global List

We can keep track of group elements with (partially) known discrete-log solutions.

- $(\mathfrak{y}, a, b) \in \mathcal{L} \Leftrightarrow \mathfrak{y} = \tau(g^{a \cdot x + b})$

Global List $\mathcal{L}$	
Known Set $\mathcal{K}$	Partially Known Set $\mathcal{PK}_x$
$(\tau(g), 0, 1)$	$(\tau(h), 1, 0)$

Public parameters: $\tau(g), \tau(h) = \tau(g^x)$

The Known/Partially Known Set in the Global List

We can keep track of group elements with (partially) known discrete-log solutions.

- $(\mathfrak{y}, a, b) \in \mathcal{L} \Leftrightarrow \mathfrak{y} = \tau(g^{a \cdot x + b})$

Global List $\mathcal{L}$	
Known Set $\mathcal{K}$	Partially Known Set $\mathcal{PK}_x$
$(\tau(g), 0, 1)$	$(\tau(h), 1, 0)$
$(\tau(g^2), 0, 2)$	

Public parameters: $\tau(g), \tau(h) = \tau(g^x)$

- $\mathtt{Mult}(\tau(g), \tau(g)) = \tau(g^2)$

The Known/Partially Known Set in the Global List

We can keep track of group elements with (partially) known discrete-log solutions.

- $(\mathfrak{y}, a, b) \in \mathcal{L} \Leftrightarrow \mathfrak{y} = \tau(g^{a \cdot x + b})$

Global List $\mathcal{L}$	
Known Set $\mathcal{K}$	Partially Known Set $\mathcal{PK}_x$
$(\tau(g), 0, 1)$	$(\tau(h), 1, 0)$
$(\tau(g^2), 0, 2)$	$(\tau(g^{x+1}), 1, 1)$

Public parameters: $\tau(g), \tau(h) = \tau(g^x)$

- $\texttt{Mult}(\tau(g), \tau(g)) = \tau(g^2)$
- $\texttt{Mult}(\tau(g), \tau(h)) = \tau(g^{x+1})$

The Known/Partially Known Set in the Global List

We can keep track of group elements with (partially) known discrete-log solutions.

- $(\mathfrak{y}, a, b) \in \mathcal{L} \Leftrightarrow \mathfrak{y} = \tau(g^{a \cdot x + b})$

Global List $\mathcal{L}$	
Known Set $\mathcal{K}$	Partially Known Set $\mathcal{PK}_x$
$(\tau(g), 0, 1)$	$(\tau(h), 1, 0)$
$(\tau(g^2), 0, 2)$	$(\tau(g^{x+1}), 1, 1)$
$(\tau(g^{-1}), 0, -1)$	

Public parameters: $\tau(g), \tau(h) = \tau(g^x)$

- $\texttt{Mult}(\tau(g), \tau(g)) = \tau(g^2)$
- $\texttt{Mult}(\tau(g), \tau(h)) = \tau(g^{x+1})$
- $\texttt{Inv}(\tau(g)) = \tau(g^{-1})$

The Known/Partially Known Set in the Global List

We can keep track of group elements with (partially) known discrete-log solutions.

- $(\mathfrak{y}, a, b) \in \mathcal{L} \Leftrightarrow \mathfrak{y} = \tau(g^{a \cdot x + b})$

Global List $\mathcal{L}$	
Known Set $\mathcal{K}$	Partially Known Set $\mathcal{PK}_x$
$(\tau(g), 0, 1)$	$(\tau(h), 1, 0)$
$(\tau(g^2), 0, 2)$	$(\tau(g^{x+1}), 1, 1)$
$(\tau(g^{-1}), 0, -1)$	$(\tau(g^{-x}), -1, 0)$

Public parameters: $\tau(g), \tau(h) = \tau(g^x)$

- $\texttt{Mult}(\tau(g), \tau(g)) = \tau(g^2)$
- $\texttt{Mult}(\tau(g), \tau(h)) = \tau(g^{x+1})$
- $\texttt{Inv}(\tau(g)) = \tau(g^{-1})$
- $\texttt{Inv}(\tau(h)) = \tau(g^{-x})$

The Known/Partially Known Set in the Global List

We can keep track of group elements with (partially) known discrete-log solutions.

- $(\mathfrak{y}, a, b) \in \mathcal{L} \Leftrightarrow \mathfrak{y} = \tau(g^{a\cdot x+b})$

Global List $\mathcal{L}$	
Known Set $\mathcal{K}$	Partially Known Set $\mathcal{PK}_x$
$(\tau(g), 0, 1)$	$(\tau(h), 1, 0)$
$(\tau(g^2), 0, 2)$	$(\tau(g^{x+1}), 1, 1)$
$(\tau(g^{-1}), 0, -1)$	$(\tau(g^{-x}), -1, 0)$

Public parameters: $\tau(g), \tau(h) = \tau(g^x)$

- $\texttt{Mult}(\tau(g), \tau(g)) = \tau(g^2)$
- $\texttt{Mult}(\tau(g), \tau(h)) = \tau(g^{x+1})$
- $\texttt{Inv}(\tau(g)) = \tau(g^{-1})$
- $\texttt{Inv}(\tau(h)) = \tau(g^{-x})$
- $\texttt{Mult}(\tau(g^{x+1}), \tau(g^{-x})) = \tau(g)$

The Known/Partially Known Set in the Global List

We can keep track of group elements with (partially) known discrete-log solutions.

- $(\mathfrak{y}, a, b) \in \mathcal{L} \Leftrightarrow \mathfrak{y} = \tau(g^{a \cdot x + b})$

Global List $\mathcal{L}$	
Known Set $\mathcal{K}$	Partially Known Set $\mathcal{PK}_x$
$(\tau(g), 0, 1)$	$(\tau(h), 1, 0)$
$(\tau(g^2), 0, 2)$	$(\tau(g^{x+1}), 1, 1)$
$(\tau(g^{-1}), 0, -1)$	$(\tau(g^{-x}), -1, 0)$
$\vdots$	$\vdots$

Public parameters: $\tau(g), \tau(h) = \tau(g^x)$

- $\texttt{Mult}(\tau(g), \tau(g)) = \tau(g^2)$
- $\texttt{Mult}(\tau(g), \tau(h)) = \tau(g^{x+1})$
- $\texttt{Inv}(\tau(g)) = \tau(g^{-1})$
- $\texttt{Inv}(\tau(h)) = \tau(g^{-x})$
- $\texttt{Mult}(\tau(g^{x+1}), \tau(g^{-x})) = \tau(g)$

$\vdots$

The Known/Partially Known Set in the Global List

We can keep track of group elements with (partially) known discrete-log solutions.

- $(\mathfrak{y}, a, b) \in \mathcal{L} \Leftrightarrow \mathfrak{y} = \tau(g^{a\cdot x+b})$

Global List $\mathcal{L}$	
Known Set $\mathcal{K}$	Partially Known Set $\mathcal{PK}_x$
$(\tau(g), 0, 1)$	$(\tau(h), 1, 0)$
$(\tau(g^2), 0, 2)$	$(\tau(g^{x+1}), 1, 1)$
$(\tau(g^{-1}), 0, -1)$	$(\tau(g^{-x}), -1, 0)$
$\vdots$	$\vdots$

Public parameters: $\tau(g), \tau(h) = \tau(g^x)$

- $\texttt{Mult}(\tau(g), \tau(g)) = \tau(g^2)$
- $\texttt{Mult}(\tau(g), \tau(h)) = \tau(g^{x+1})$
- $\texttt{Inv}(\tau(g)) = \tau(g^{-1})$
- $\texttt{Inv}(\tau(h)) = \tau(g^{-x})$
- $\texttt{Mult}(\tau(g^{x+1}), \tau(g^{-x})) = \tau(g)$

$\vdots$

Event "BRIDGE":

$(\mathfrak{y}, a, b), (\mathfrak{y}, a', b') \in \mathcal{L}$ with $(a, b) \neq (a', b')$ $\Rightarrow$ $ax + b = a'x + b'$, $\therefore x = (a - a')^{-1}(b' - b)$.

The Known/Partially Known Set in the Global List

We can extend this to the multi-user case.

- Public parameters: $\tau(g), (\tau(h_1), \cdots, \tau(h_N)) = (\tau(g^{x_1}), \cdots, \tau(g^{x_N}))$
- Instead of scalar a, we will have an N-dimensional vector $\vec{a}$ such that the list $\mathcal{L}$ contains a tuple $(\mathfrak{y}, \vec{a}, b)$ such that
$$\mathfrak{y} = \tau(g^{\vec{a}\cdot\vec{x}+b})$$
where $\vec{x} = (x_1, \cdots, x_N)$.
- The known set $\mathcal{K}^N$ contains tuples $(\mathfrak{y}, \vec{0}, b)$, and
- The partially known set $\mathcal{PK}^N_{\{x_i\}_{i=1}^N}$ contains tuples $(\mathfrak{y}, \vec{a} \neq \vec{0}, b)$.
- The event "BRIDGE^N" occurs if $(\mathfrak{y}, \vec{a}, b), (\mathfrak{y}, \vec{a}', b') \in \mathcal{L}$ with $(\vec{a}, b) \neq (\vec{a}', b')$.

Claim

$$\Pr\left[\text{BRIDGE}^N\right] = \mathcal{O}\left(\frac{(t+N)^2}{q}\right).$$

The Known/Partially Known Set in the Global List

We can extend this to the multi-user case.

- Public parameters: $\tau(g), (\tau(h_1), \cdots, \tau(h_N)) = (\tau(g^{x_1}), \cdots, \tau(g^{x_N}))$
- Instead of scalar a, we will have an N-dimensional vector $\vec{a}$ such that the list $\mathcal{L}$ contains a tuple $(\mathfrak{y}, \vec{a}, b)$ such that

$$\mathfrak{y} = \tau(g^{\vec{a}\cdot\vec{x}+b})$$

  where $\vec{x} = (x_1, \cdots, x_N)$.
- The known set $\mathcal{K}^N$ contains tuples $(\mathfrak{y}, \vec{0}, b)$, and
- The partially known set $\mathcal{PK}^N_{\{x_i\}_{i=1}^N}$ contains tuples $(\mathfrak{y}, \vec{a} \neq \vec{0}, b)$.
- The event "BRIDGE^N" occurs if $(\mathfrak{y}, \vec{a}, b), (\mathfrak{y}, \vec{a}', b') \in \mathcal{L}$ with $(\vec{a}, b) \neq (\vec{a}', b')$.

Claim

$$\Pr\left[\text{BRIDGE}^N\right] = \mathcal{O}\left(\frac{(t+N)^2}{q}\right).$$

- But what if $\mathfrak{y} \notin \mathcal{L}$, i.e., "fresh"?

We are now at...

Restricted Discrete-Log Oracle in the GGM

Consider the generic group model for a cyclic group $(G = \langle g \rangle, \cdot)$ of prime order q with random injective encoding map $\tau : G \to \mathbb{G}$.

Public parameters:

$$g, (h_1, \cdots, h_N) = (g^{x_1}, \cdots, g^{x_N})$$

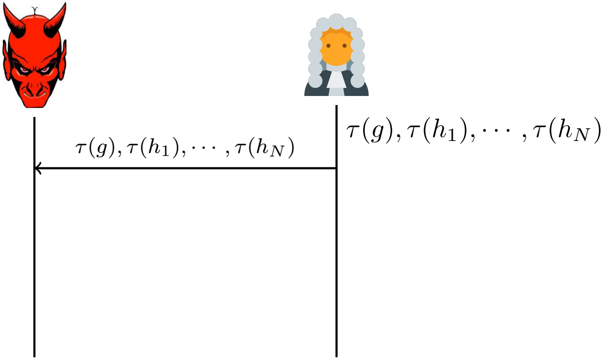
$\tau(g), \tau(h_1), \cdots, \tau(h_N)$

Restricted Discrete-Log Oracle in the GGM

Consider the generic group model for a cyclic group $(G = \langle g \rangle, \cdot)$ of prime order q with random injective encoding map $\tau : G \to \mathbb{G}$.

Public parameters:

$$g, (h_1, \cdots, h_N) = (g^{x_1}, \cdots, g^{x_N})$$



Restricted Discrete-Log Oracle in the GGM

Consider the generic group model for a cyclic group $(G = \langle g \rangle, \cdot)$ of prime order q with random injective encoding map $\tau : G \to \mathbb{G}$.

Public parameters:

$$g, (h_1, \cdots, h_N) = (g^{x_1}, \cdots, g^{x_N})$$

$\tau(g), \tau(h_1), \cdots, \tau(h_N)$

$\xleftarrow{\tau(g), \tau(h_1), \cdots, \tau(h_N)}$

Pick $\mathfrak{y} = \tau(y)$

Restricted Discrete-Log Oracle in the GGM

Consider the generic group model for a cyclic group $(G = \langle g \rangle, \cdot)$ of prime order q with random injective encoding map $\tau : G \to \mathbb{G}$.

Public parameters:

$g, (h_1, \cdots, h_N) = (g^{x_1}, \cdots, g^{x_N})$

$\tau(g), \tau(h_1), \cdots, \tau(h_N)$

$\tau(g), \tau(h_1), \cdots, \tau(h_N)$ ←

Pick $\mathfrak{y} = \tau(y)$

Query $\mathtt{DLog}_g(\mathfrak{y})$ →

Restricted Discrete-Log Oracle in the GGM

Consider the generic group model for a cyclic group $(G = \langle g \rangle, \cdot)$ of prime order q with random injective encoding map $\tau : G \to \mathbb{G}$.

Public parameters:
$g, (h_1, \cdots, h_N) = (g^{x_1}, \cdots, g^{x_N})$

$\tau(g), \tau(h_1), \cdots, \tau(h_N)$

$\xleftarrow{\tau(g), \tau(h_1), \cdots, \tau(h_N)}$

Pick $\mathfrak{y} = \tau(y)$

$\xrightarrow{\text{Query } \mathtt{DLog}_g(\mathfrak{y})}$

If $\mathfrak{y}$ is "fresh",
$y = g^z$

Restricted Discrete-Log Oracle in the GGM

Consider the generic group model for a cyclic group $(G = \langle g \rangle, \cdot)$ of prime order q with random injective encoding map $\tau : G \to \mathbb{G}$.

Public parameters:

$$g, (h_1, \cdots, h_N) = (g^{x_1}, \cdots, g^{x_N})$$

$\tau(g), \tau(h_1), \cdots, \tau(h_N)$

$\tau(g), \tau(h_1), \cdots, \tau(h_N)$

Pick $\mathfrak{y} = \tau(y)$

Query $\mathtt{DLog}_g(\mathfrak{y})$

$\tau(y) \neq \tau(g), \tau(h_1), \cdots, \tau(h_N)$, and
$\tau(y) \neq$ (output of prior `GO` query)

If $\mathfrak{y}$ is "fresh",
$y = g^z$

Restricted Discrete-Log Oracle in the GGM

Consider the generic group model for a cyclic group $(G = \langle g \rangle, \cdot)$ of prime order q with random injective encoding map $\tau : G \to \mathbb{G}$.

Public parameters:

$g, (h_1, \cdots, h_N) = (g^{x_1}, \cdots, g^{x_N})$

$\tau(g), \tau(h_1), \cdots, \tau(h_N)$

$\xleftarrow{\tau(g), \tau(h_1), \cdots, \tau(h_N)}$

Pick $\mathfrak{y} = \tau(y)$

$\xrightarrow{\text{Query } \mathtt{DLog}_g(\mathfrak{y})}$

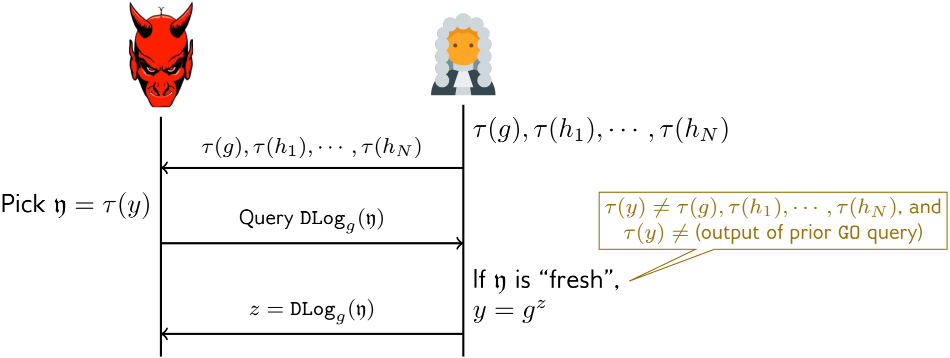
$\tau(y) \neq \tau(g), \tau(h_1), \cdots, \tau(h_N)$, and $\tau(y) \neq$ (output of prior GO query)

If $\mathfrak{y}$ is "fresh",
$y = g^z$

$\xleftarrow{z = \mathtt{DLog}_g(\mathfrak{y})}$

Restricted Discrete-Log Oracle in the GGM

Public parameters:
$g, (h_1, \cdots, h_N) = (g^{x_1}, \cdots, g^{x_N})$



Why restricting $\mathtt{DLog}_g(\cdot)$ to "fresh" queries?

- Trivial attack: Pick random $r \in \mathbb{Z}_q$, compute $\tau(h_i g^r)$ using $\mathtt{Mult}$ oracle and query $\mathtt{DLog}_g(\tau(h_i g^r))$

We are now at...

The 1-out-of-N Generic Signature Forgery Game

- Multi-user security in the "1-out-of-N" setting
- The probability that the attacker can forge ***any one*** of N signatures is negligible

The 1-out-of-N Generic Signature Forgery Game $\mathsf{SigForge}^{\mathtt{GO},N}_{\mathcal{A},\Pi}(k)$:

Consider $G = \langle g \rangle$ of prime order $q \approx 2^{2k}$ and $\tau : G \to \mathbb{G}$.

Run $\mathsf{Kg}(1^k)$ N times

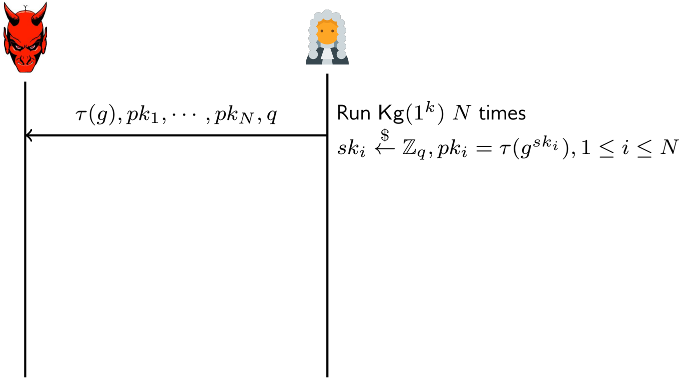
$sk_i \xleftarrow{\$} \mathbb{Z}_q, pk_i = \tau(g^{sk_i}), 1 \le i \le N$

The 1-out-of-N Generic Signature Forgery Game

- Multi-user security in the "1-out-of-N" setting
- The probability that the attacker can forge ***any one*** of N signatures is negligible

The 1-out-of-N Generic Signature Forgery Game $\mathsf{SigForge}^{\mathtt{GO},N}_{\mathcal{A},\Pi}(k)$:

Consider $G = \langle g \rangle$ of prime order $q \approx 2^{2k}$ and $\tau : G \to \mathbb{G}$.

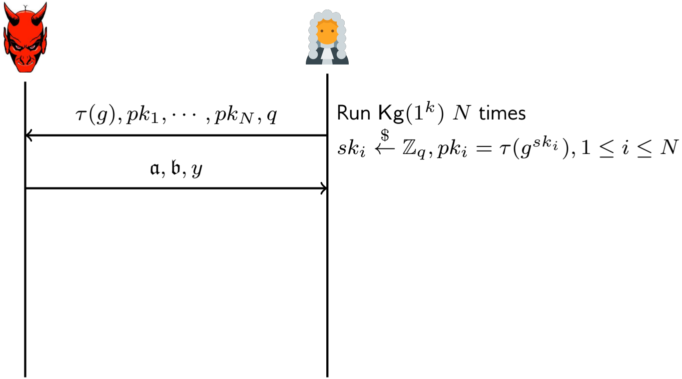


The 1-out-of-N Generic Signature Forgery Game

- Multi-user security in the "1-out-of-N" setting
- The probability that the attacker can forge ***any one*** of N signatures is negligible

The 1-out-of-N Generic Signature Forgery Game $\mathsf{SigForge}^{\mathtt{GO},N}_{\mathcal{A},\Pi}(k)$:

Consider $G = \langle g \rangle$ of prime order $q \approx 2^{2k}$ and $\tau : G \to \mathbb{G}$.

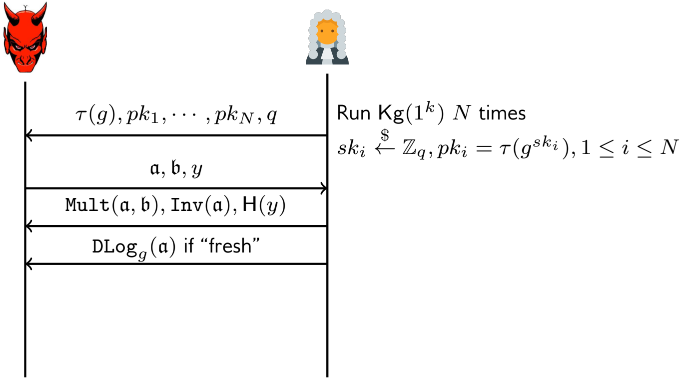


The 1-out-of-N Generic Signature Forgery Game

- Multi-user security in the "1-out-of-N" setting
- The probability that the attacker can forge ***any one*** of N signatures is negligible

The 1-out-of-N Generic Signature Forgery Game $\mathsf{SigForge}^{\mathtt{GO},N}_{\mathcal{A},\Pi}(k)$:

Consider $G = \langle g \rangle$ of prime order $q \approx 2^{2k}$ and $\tau : G \to \mathbb{G}$.

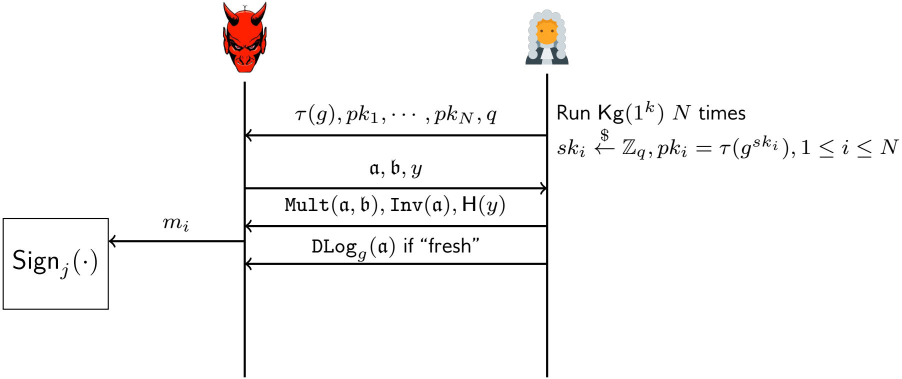


The 1-out-of-N Generic Signature Forgery Game

- Multi-user security in the "1-out-of-N" setting
- The probability that the attacker can forge ***any one*** of N signatures is negligible

The 1-out-of-N Generic Signature Forgery Game $\mathsf{SigForge}_{\mathcal{A},\Pi}^{\mathtt{GO},N}(k)$:

Consider $G = \langle g \rangle$ of prime order $q \approx 2^{2k}$ and $\tau : G \to \mathbb{G}$.

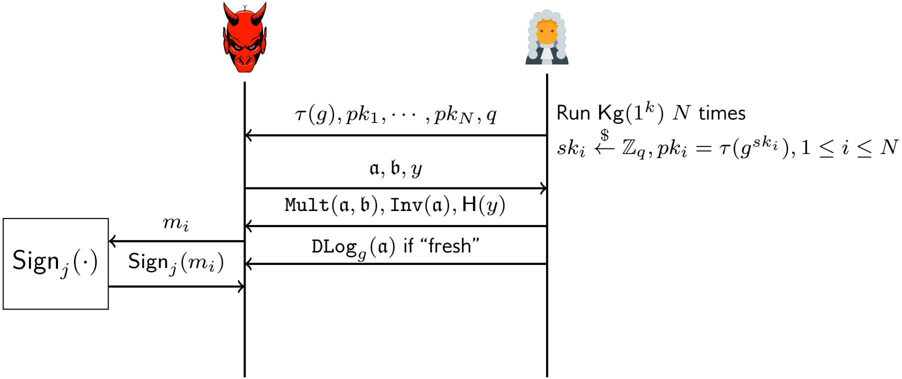


The 1-out-of-N Generic Signature Forgery Game

- Multi-user security in the "1-out-of-N" setting
- The probability that the attacker can forge ***any one*** of N signatures is negligible

The 1-out-of-N Generic Signature Forgery Game $\mathsf{SigForge}^{\mathtt{GO},N}_{\mathcal{A},\Pi}(k)$:

Consider $G = \langle g \rangle$ of prime order $q \approx 2^{2k}$ and $\tau : G \to \mathbb{G}$.

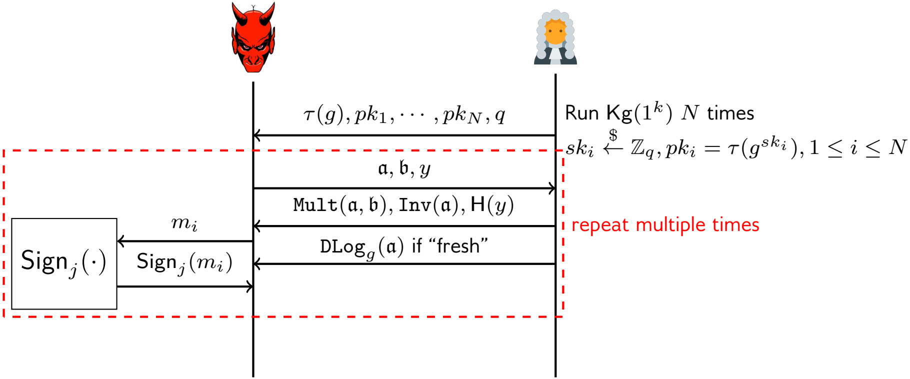


The 1-out-of-N Generic Signature Forgery Game

- Multi-user security in the "1-out-of-N" setting
- The probability that the attacker can forge ***any one*** of N signatures is negligible

The 1-out-of-N Generic Signature Forgery Game $\mathsf{SigForge}^{\mathtt{GO},N}_{\mathcal{A},\Pi}(k)$:

Consider $G = \langle g \rangle$ of prime order $q \approx 2^{2k}$ and $\tau : G \to \mathbb{G}$.

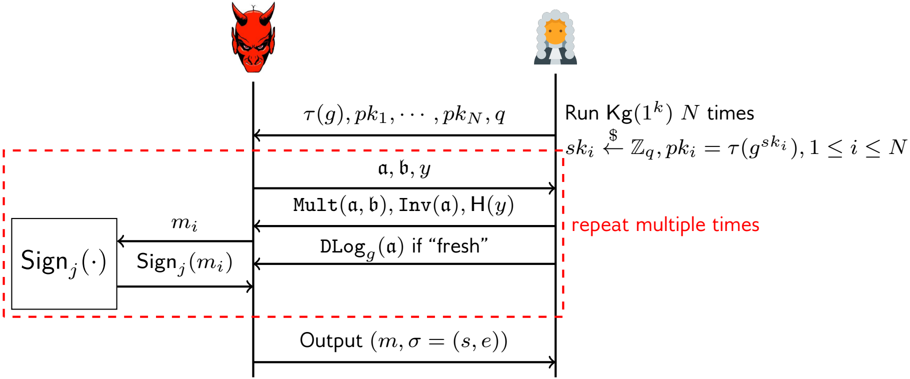


The 1-out-of-N Generic Signature Forgery Game

- Multi-user security in the "1-out-of-N" setting
- The probability that the attacker can forge ***any one*** of N signatures is negligible

The 1-out-of-N Generic Signature Forgery Game $\mathsf{SigForge}^{\mathtt{GO},N}_{\mathcal{A},\Pi}(k)$:

Consider $G = \langle g \rangle$ of prime order $q \approx 2^{2k}$ and $\tau : G \to \mathbb{G}$.

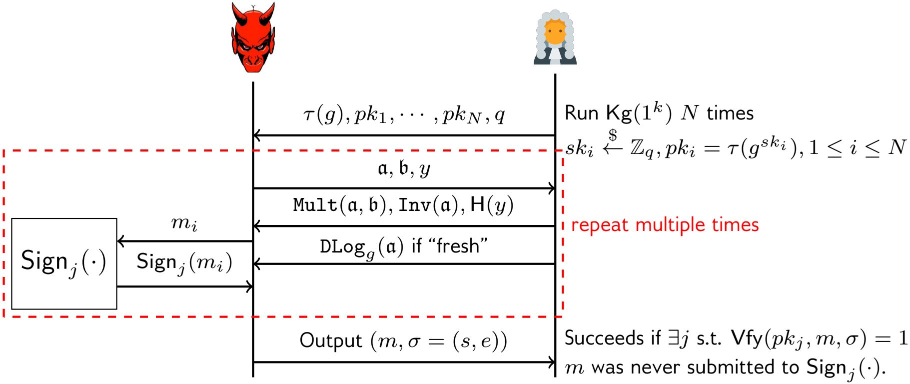


The 1-out-of-N Generic Signature Forgery Game

- Multi-user security in the "1-out-of-N" setting
- The probability that the attacker can forge ***any one*** of N signatures is negligible

The 1-out-of-N Generic Signature Forgery Game $\mathsf{SigForge}^{\mathtt{GO},N}_{\mathcal{A},\Pi}(k)$:

Consider $G = \langle g \rangle$ of prime order $q \approx 2^{2k}$ and $\tau : G \to \mathbb{G}$.

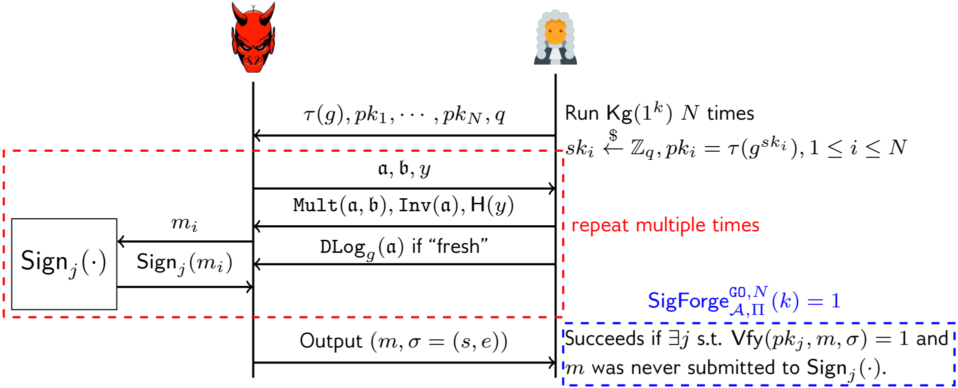


The 1-out-of-N Generic Signature Forgery Game

- Multi-user security in the "1-out-of-N" setting
- The probability that the attacker can forge ***any one*** of N signatures is negligible

The 1-out-of-N Generic Signature Forgery Game $\mathsf{SigForge}^{\mathtt{GO},N}_{\mathcal{A},\Pi}(k)$:

Consider $G = \langle g \rangle$ of prime order $q \approx 2^{2k}$ and $\tau : G \to \mathbb{G}$.



Multi-User Security Definition

Definition

We say that a signature scheme $\Pi = (\mathsf{Kg}, \mathsf{Sign}, \mathsf{Vfy})$ is $(t, N, q_{\mathtt{RO}}, q_{\mathtt{GO}}, q_{\mathsf{Sign}}, \epsilon)$-***MU-UF-CMA secure (multi-user unforgeable against chosen message attack)*** if for every adversary $\mathcal{A}$ running in time at most t and making at most $q_{\mathtt{RO}}$ (resp. $q_{\mathtt{GO}}$, q_{Sign}) queries to the random oracle (resp. generic group, signature oracles), the following bound holds:

$$\Pr\left[\mathsf{SigForge}_{\mathcal{A},\Pi}^{\mathtt{GO},N}(k) = 1\right] \leq \epsilon.$$

The Multi-User Bridge Game

Recall

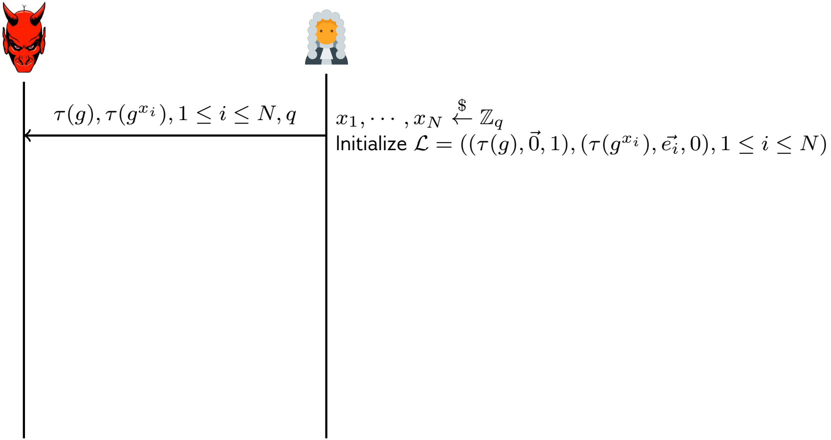
The event BRIDGE^N occurs if $\mathcal{L}$ ever contains two distinct tuples $(\mathfrak{y}_1, \vec{a}_1, b_1)$ and $(\mathfrak{y}_2, \vec{a}_2, b_2)$ such that $\mathfrak{y}_1 = \mathfrak{y}_2$ but $(\vec{a}_1, b_1) \neq (\vec{a}_2, b_2)$.

- As long as the event BRIDGE^N has not occurred we can (essentially) view $x_1, \ldots, x_N$ as uniformly random values that that yet to be selected.
- More precisely, the values $x_1, \ldots, x_N$ are selected subject to a few constraints, e.g., if we know $\mathfrak{f}_1 = \tau(g^{\vec{a}_1 \cdot \vec{x} + b_1}) \neq \mathfrak{f}_2 = \tau(g^{\vec{a}_2 \cdot \vec{x} + b_2})$ then we have the constraint that $\vec{a}_1 \cdot \vec{x} + b_1 \neq \vec{a}_2 \cdot \vec{x} + b_2$.

The Multi-User Bridge Game

The 1-out-of-N Generic BRIDGE^N**-Finding Game** $\mathsf{BridgeChal}_{\mathcal{A}}^{\mathtt{GO},N}(k)$:

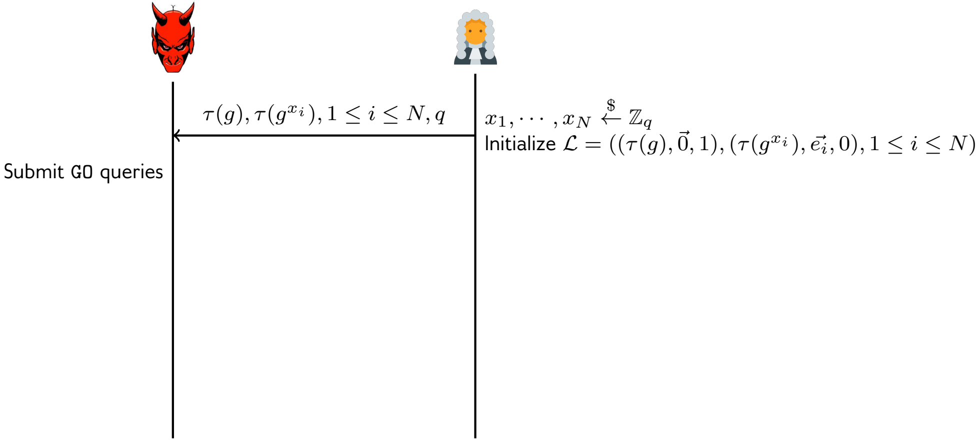
Consider $G = \langle g \rangle$ of prime order $q \approx 2^{2k}$ and $\tau : G \to \mathbb{G}$.



The Multi-User Bridge Game

The 1-out-of-N Generic BRIDGE^N**-Finding Game** BridgeChal$_{\mathcal{A}}^{\texttt{GO},N}(k)$:

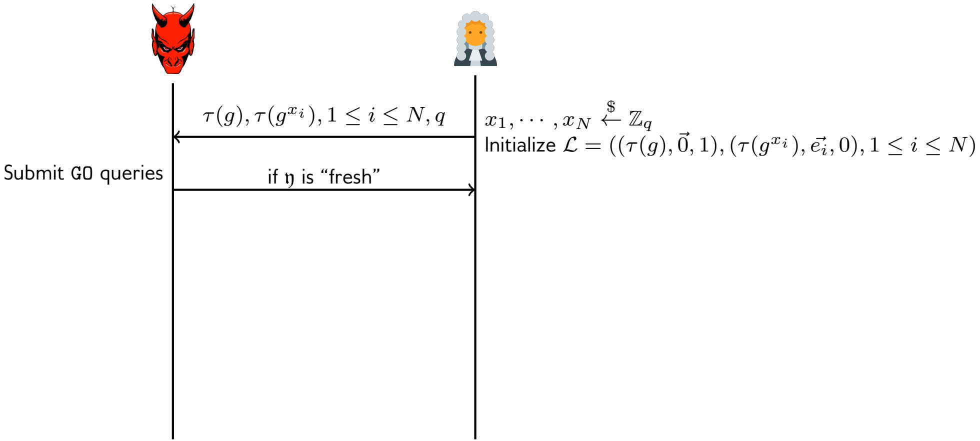
Consider $G = \langle g \rangle$ of prime order $q \approx 2^{2k}$ and $\tau : G \to \mathbb{G}$.



The Multi-User Bridge Game

The 1-out-of-N Generic BRIDGE^N**-Finding Game** $\text{BridgeChal}_{\mathcal{A}}^{\texttt{GO},N}(k)$:

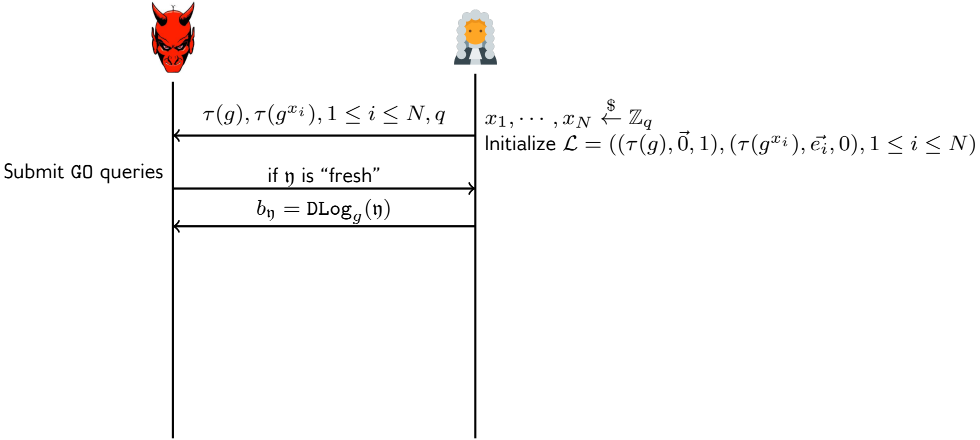
Consider $G = \langle g \rangle$ of prime order $q \approx 2^{2k}$ and $\tau : G \rightarrow \mathbb{G}$.



The Multi-User Bridge Game

The 1-out-of-N Generic BRIDGE^N**-Finding Game** $\mathsf{BridgeChal}_{\mathcal{A}}^{\mathtt{GO},N}(k)$:

Consider $G = \langle g \rangle$ of prime order $q \approx 2^{2k}$ and $\tau : G \to \mathbb{G}$.



The Multi-User Bridge Game

The 1-out-of-N Generic BRIDGE^N**-Finding Game** BridgeChal$^{\mathtt{GO},N}_{\mathcal{A}}(k)$:

Consider $G = \langle g \rangle$ of prime order $q \approx 2^{2k}$ and $\tau : G \to \mathbb{G}$.

$\tau(g), \tau(g^{x_i}), 1 \le i \le N, q$ ←

$x_1, \cdots, x_N \xleftarrow{\$} \mathbb{Z}_q$

Initialize $\mathcal{L} = ((\tau(g), \vec{0}, 1), (\tau(g^{x_i}), \vec{e_i}, 0), 1 \le i \le N)$

Submit $\mathtt{GO}$ queries

→ if $\mathfrak{y}$ is "fresh"

← $b_{\mathfrak{y}} = \mathtt{DLog}_g(\mathfrak{y})$

Add $(\mathfrak{y}, \vec{0}, b_{\mathfrak{y}}) \in \mathcal{L}$

The Multi-User Bridge Game

The 1-out-of-N Generic BRIDGE^N**-Finding Game** $\mathsf{BridgeChal}_{\mathcal{A}}^{\mathtt{GO},N}(k)$:

Consider $G = \langle g \rangle$ of prime order $q \approx 2^{2k}$ and $\tau : G \to \mathbb{G}$.

$\tau(g), \tau(g^{x_i}), 1 \le i \le N, q$ ←

$x_1, \cdots, x_N \xleftarrow{\$} \mathbb{Z}_q$

Initialize $\mathcal{L} = ((\tau(g), \vec{0}, 1), (\tau(g^{x_i}), \vec{e_i}, 0), 1 \le i \le N)$

Submit $\mathtt{GO}$ queries

→ if $\mathfrak{y}$ is "fresh"

← $b_{\mathfrak{y}} = \mathtt{DLog}_g(\mathfrak{y})$

Add $(\mathfrak{y}, \vec{0}, b_{\mathfrak{y}}) \in \mathcal{L}$

→ if $\mathfrak{y}_1, \mathfrak{y}_2, \mathfrak{y} \in \mathcal{L}$

The Multi-User Bridge Game

The 1-out-of-N Generic BRIDGE^N**-Finding Game** $\mathsf{BridgeChal}_{\mathcal{A}}^{\mathtt{GO},N}(k)$:

Consider $G = \langle g \rangle$ of prime order $q \approx 2^{2k}$ and $\tau : G \to \mathbb{G}$.

$x_1, \cdots, x_N \xleftarrow{\$} \mathbb{Z}_q$

Initialize $\mathcal{L} = ((\tau(g), \vec{0}, 1), (\tau(g^{x_i}), \vec{e_i}, 0), 1 \le i \le N)$

$\longleftarrow$ $\tau(g), \tau(g^{x_i}), 1 \le i \le N, q$

Submit $\mathtt{GO}$ queries

$\longrightarrow$ if $\mathfrak{y}$ is "fresh"

$\longleftarrow$ $b_{\mathfrak{y}} = \mathtt{DLog}_g(\mathfrak{y})$

Add $(\mathfrak{y}, \vec{0}, b_{\mathfrak{y}}) \in \mathcal{L}$

$\longrightarrow$ if $\mathfrak{y}_1, \mathfrak{y}_2, \mathfrak{y} \in \mathcal{L}$

$\longleftarrow$ $\mathtt{Mult}(\mathfrak{y}_1, \mathfrak{y}_2), \mathtt{Inv}(\mathfrak{y})$

The Multi-User Bridge Game

The 1-out-of-N Generic BRIDGE^N**-Finding Game** BridgeChal$_{\mathcal{A}}^{\mathtt{GO},N}(k)$:

Consider $G = \langle g \rangle$ of prime order $q \approx 2^{2k}$ and $\tau : G \to \mathbb{G}$.

$x_1, \cdots, x_N \xleftarrow{\$} \mathbb{Z}_q$

Initialize $\mathcal{L} = ((\tau(g), \vec{0}, 1), (\tau(g^{x_i}), \vec{e_i}, 0), 1 \le i \le N)$

$\xleftarrow{\tau(g), \tau(g^{x_i}), 1 \le i \le N, q}$

Submit GO queries

$\xrightarrow{\text{if } \mathfrak{y} \text{ is “fresh”}}$

$\xleftarrow{b_{\mathfrak{y}} = \mathtt{DLog}_g(\mathfrak{y})}$

Add $(\mathfrak{y}, \vec{0}, b_{\mathfrak{y}}) \in \mathcal{L}$

$\xrightarrow{\text{if } \mathfrak{y}_1, \mathfrak{y}_2, \mathfrak{y} \in \mathcal{L}}$

$\xleftarrow{\mathtt{Mult}(\mathfrak{y}_1, \mathfrak{y}_2), \mathtt{Inv}(\mathfrak{y})}$

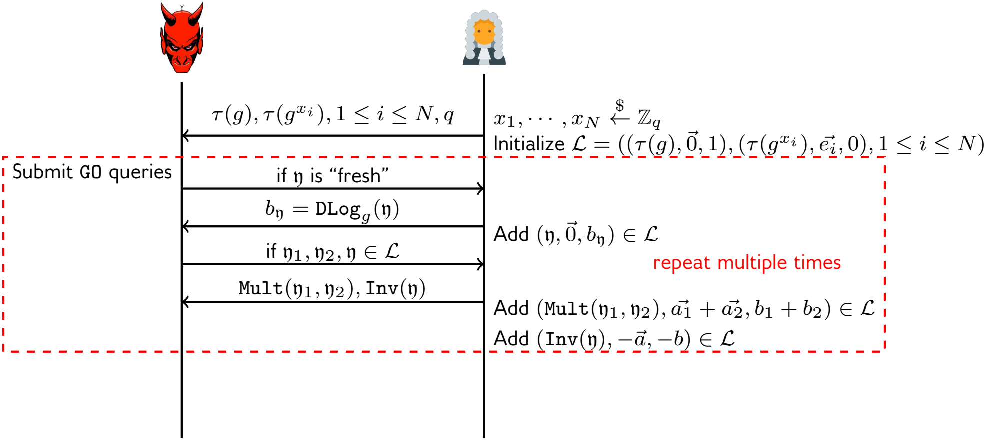
Add $(\mathtt{Mult}(\mathfrak{y}_1, \mathfrak{y}_2), \vec{a_1} + \vec{a_2}, b_1 + b_2) \in \mathcal{L}$

Add $(\mathtt{Inv}(\mathfrak{y}), -\vec{a}, -b) \in \mathcal{L}$

The Multi-User Bridge Game

The 1-out-of-N Generic BRIDGE^N**-Finding Game** $\text{BridgeChal}_{\mathcal{A}}^{\texttt{GO},N}(k)$:

Consider $G = \langle g \rangle$ of prime order $q \approx 2^{2k}$ and $\tau : G \to \mathbb{G}$.



The Multi-User Bridge Game

The 1-out-of-N Generic BRIDGE^N**-Finding Game** $\mathsf{BridgeChal}_{\mathcal{A}}^{\mathtt{GO},N}(k)$:

Consider $G = \langle g \rangle$ of prime order $q \approx 2^{2k}$ and $\tau : G \to \mathbb{G}$.

$\tau(g), \tau(g^{x_i}), 1 \leq i \leq N, q$ ←

$x_1, \cdots, x_N \xleftarrow{\$} \mathbb{Z}_q$

Initialize $\mathcal{L} = ((\tau(g), \vec{0}, 1), (\tau(g^{x_i}), \vec{e_i}, 0), 1 \leq i \leq N)$

Submit GO queries

→ if $\mathfrak{y}$ is "fresh"

← $b_{\mathfrak{y}} = \mathtt{DLog}_g(\mathfrak{y})$

Add $(\mathfrak{y}, \vec{0}, b_{\mathfrak{y}}) \in \mathcal{L}$

→ if $\mathfrak{y}_1, \mathfrak{y}_2, \mathfrak{y} \in \mathcal{L}$

repeat multiple times

← $\mathtt{Mult}(\mathfrak{y}_1, \mathfrak{y}_2), \mathtt{Inv}(\mathfrak{y})$

Add $(\mathtt{Mult}(\mathfrak{y}_1, \mathfrak{y}_2), \vec{a_1} + \vec{a_2}, b_1 + b_2) \in \mathcal{L}$

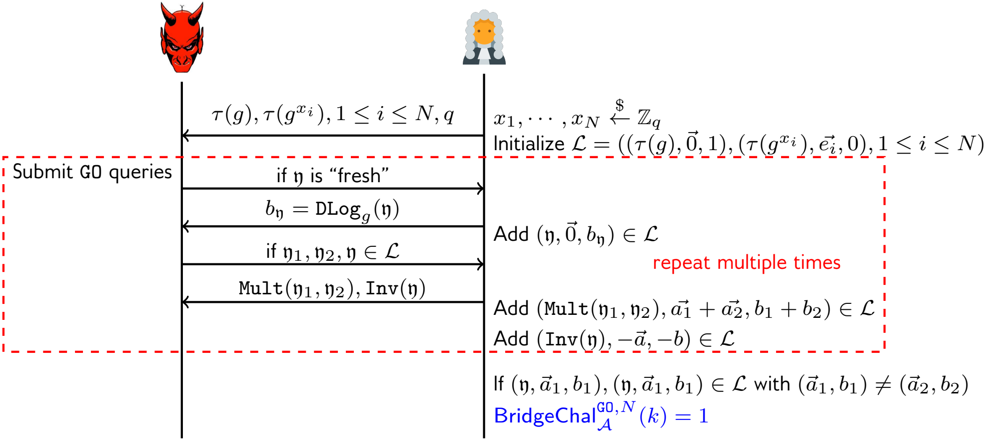
Add $(\mathtt{Inv}(\mathfrak{y}), -\vec{a}, -b) \in \mathcal{L}$

If $(\mathfrak{y}, \vec{a}_1, b_1), (\mathfrak{y}, \vec{a}_1, b_1) \in \mathcal{L}$ with $(\vec{a}_1, b_1) \neq (\vec{a}_2, b_2)$

The Multi-User Bridge Game

The 1-out-of-N Generic BRIDGE^N**-Finding Game** BridgeChal$^{\mathtt{GO},N}_{\mathcal{A}}(k)$:

Consider $G = \langle g \rangle$ of prime order $q \approx 2^{2k}$ and $\tau : G \to \mathbb{G}$.



The Multi-User Bridge Game

Theorem

The probability an attacker $\mathcal{A}$ running in time t wins the 1-out-of-N generic BRIDGE^N-finding game (even with access to the restricted $\mathtt{DLog}$ oracle) is at most

$$\Pr\left[\mathsf{BridgeChal}_{\mathcal{A}}^{\mathtt{GO},N}(k)=1\right] \le \frac{tN+3t(t+1)/2}{q-(N+3t+1)^2-N} = \mathcal{O}\left(\frac{(t+N)^2}{q}\right)$$

where q is the order of the group G.

Corollary

For any attacker $\mathcal{A}$ running in time $t' = t + 2\log q$ we have

$$\Pr\left[\mathsf{1ofNDLog}_{\mathcal{A}}^{\mathtt{GO},N}(k)=1\right] \le \frac{tN+3t(t+1)/2}{q-(N+3t+1)^2-N} = \mathcal{O}\left(\frac{(t+N)^2}{q}\right)$$

where q is the order of the group G.

We are now at...

Main Theorem

Theorem

In the generic group model of prime order $q \approx 2^{2k}$ and the programmable random oracle model the short Schnorr signature scheme is $(t, N, q_{\mathtt{RO}}, q_{\mathtt{GO}}, q_{\mathsf{Sign}}, \epsilon)$-MU-UF-CMA secure with $\epsilon = \frac{tN+3t(t+2)/2}{q-(N+3t+1)^2-N} + \frac{t^2}{q} + \frac{t+1}{2^k} = \mathcal{O}\left(\frac{t+N}{2^k}\right)$.

- Our result provides k-bits of multi-user security of "short" Schnorr signatures since usually $t \gg N$ ($t \approx 2^{80}, N \approx 2^{32}$).

Proof Sketch: Security Reduction

Reduction $\mathcal{A}_{\text{bridge}}$

Given: $\mathfrak{g} = \tau(g), \tau(h_i) = \tau(g^{x_i}), 1 \leq i \leq N, q$

Initialize $\mathcal{L} = \{(\tau(g), \vec{0}, 1), (\tau(g^{x_i}), \vec{e_i}, 0) \text{ for } 1 \leq i \leq N\}, H_{\text{resp}} = \{\}$

/* begin simulation */

$\texttt{Mult}(\cdot)$

$\texttt{Inv}(\cdot)$

$\texttt{DLog}_g(\cdot)$

$\mathsf{H}(\cdot)$

$\mathcal{A}_{\text{sig}}$

$\{m_i\}_{i=1}^N$

$\mathsf{Sign}_j(\cdot)$

$\{\sigma_i\}_{i=1}^N$

$\sigma = (s, e), m$

$\mathsf{Sign}_j(m_i)$ without secret key x_j $(1 \leq j \leq N)$

1 : Pick s_i, e_i randomly

2 : Compute $\tau(g^{s_i}), \tau(g^{x_j e_i}) = \tau(\tau^{-1}(h_j)^{e_i})$

3 : Compute $I_i = \tau(g^{s_i} \cdot g^{-x_j e_i})$

4 : **if** $\mathsf{H}(I_i || m_i)$ previously queried **then**

5 : **return** $\perp$

6 : **else**

7 : Program $\mathsf{H}(I_i || m_i) := e_i$

8 : **return** $\sigma_i = (s_i, e_i)$

/* end simulation */

Compute: $I_\sigma = \tau(g^s \cdot g^{-xe}), e_\sigma = \mathsf{H}(I_\sigma || m)$ and check if $(I_\sigma, \vec{a}, b) \in \mathcal{L}$

if no such triple exists then **return** $\perp$

If $\vec{a}$ has only one nonzero element a and if $a + e = 0$ then **return** $\perp$

Otherwise we find a BRIDGE^N instance

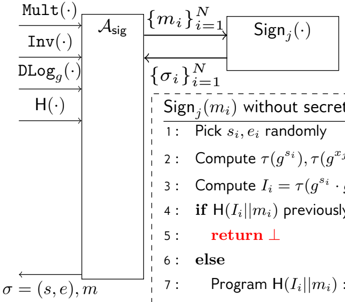
Proof Sketch: Security Reduction

Reduction $\mathcal{A}_{\text{bridge}}$

Given: $\mathfrak{g} = \tau(g), \tau(h_i) = \tau(g^{x_i}), 1 \le i \le N, q$

Initialize $\mathcal{L} = \{(\tau(g), \vec{0}, 1), (\tau(g^{x_i}), \vec{e_i}, 0) \text{ for } 1 \le i \le N\}, H_{\text{resp}} = \{\}$

/* begin simulation */



Sign$_j(m_i)$ without secret key x_j $(1 \le j \le N)$

1 : Pick s_i, e_i randomly

2 : Compute $\tau(g^{s_i}), \tau(g^{x_j e_i}) = \tau(\tau^{-1}(h_j)^{e_i})$

3 : Compute $I_i = \tau(g^{s_i} \cdot g^{-x_j e_i})$

4 : **if** $\mathsf{H}(I_i || m_i)$ previously queried **then**

5 : **return** $\perp$

6 : **else**

7 : Program $\mathsf{H}(I_i || m_i) := e_i$

8 : **return** $\sigma_i = (s_i, e_i)$

/* end simulation */

Compute: $I_\sigma = \tau(g^s \cdot g^{-xe}), e_\sigma = \mathsf{H}(I_\sigma || m)$ and check if $(I_\sigma, \vec{a}, b) \in \mathcal{L}$

if no such triple exists then **return** $\perp$

If $\vec{a}$ has only one nonzero element a and if $a + e = 0$ then **return** $\perp$

Otherwise we find a BRIDGE^N instance

Probability of outputting $\perp$

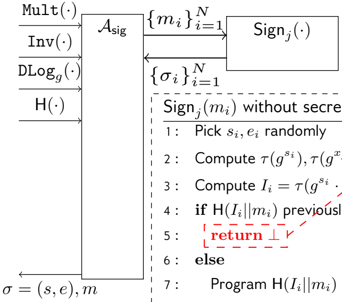
Proof Sketch: Security Reduction

Reduction $\mathcal{A}_{\text{bridge}}$

Given: $\mathfrak{g} = \tau(g), \tau(h_i) = \tau(g^{x_i}), 1 \le i \le N, q$

Initialize $\mathcal{L} = \{(\tau(g), \vec{0}, 1), (\tau(g^{x_i}), \vec{e_i}, 0) \text{ for } 1 \le i \le N\}, H_{\text{resp}} = \{\}$

/ begin simulation */*



$\text{Sign}_j(m_i)$ without secret key x_j $(1 \le j \le N)$

1 : Pick s_i, e_i randomly

2 : Compute $\tau(g^{s_i}), \tau(g^{x_j e_i}) = \tau(\tau^{-1}(h_j)^{e_i})$

3 : Compute $I_i = \tau(g^{s_i} \cdot g^{-x_j e_i})$

4 : **if** $\mathsf{H}(I_i || m_i)$ previously queried **then**

5 : $\quad$ **return** $\perp$

6 : **else**

7 : $\quad$ Program $\mathsf{H}(I_i || m_i) := e_i$

8 : $\quad$ **return** $\sigma_i = (s_i, e_i)$

/ end simulation */*

Compute: $I_\sigma = \tau(g^s \cdot g^{-xe}), e_\sigma = \mathsf{H}(I_\sigma || m)$ and check if $(I_\sigma, \vec{a}, b) \in \mathcal{L}$

if no such triple exists then **return** $\perp$

If $\vec{a}$ has only one nonzero element a and if $a + e = 0$ then **return** $\perp$

Otherwise we find a BRIDGE^N instance

Probability of outputting $\perp$

$$\le q_{\text{Sign}} \times \frac{q_{\text{RO}} + q_{\text{Sign}}}{q} = \mathcal{O}\left(\frac{t^2}{q}\right)$$

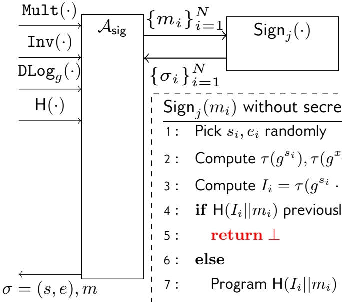
Proof Sketch: Security Reduction

Reduction $\mathcal{A}_{\text{bridge}}$

Given: $\mathfrak{g} = \tau(g), \tau(h_i) = \tau(g^{x_i}), 1 \le i \le N, q$

Initialize $\mathcal{L} = \{(\tau(g), \vec{0}, 1), (\tau(g^{x_i}), \vec{e_i}, 0) \text{ for } 1 \le i \le N\}, H_{\text{resp}} = \{\}$

/* begin simulation */



$\mathsf{Sign}_j(m_i)$ without secret key x_j $(1 \le j \le N)$

1 : Pick s_i, e_i randomly

2 : Compute $\tau(g^{s_i}), \tau(g^{x_j e_i}) = \tau(\tau^{-1}(h_j)^{e_i})$

3 : Compute $I_i = \tau(g^{s_i} \cdot g^{-x_j e_i})$

4 : **if** $\mathsf{H}(I_i || m_i)$ previously queried **then**

5 : **return** $\perp$

6 : **else**

7 : Program $\mathsf{H}(I_i || m_i) := e_i$

8 : **return** $\sigma_i = (s_i, e_i)$

/* end simulation */

Compute: $I_\sigma = \tau(g^s \cdot g^{-xe}), e_\sigma = \mathsf{H}(I_\sigma || m)$ and check if $(I_\sigma, \vec{a}, b) \in \mathcal{L}$

if no such triple exists then **return** $\perp$

If $\vec{a}$ has only one nonzero element a and if $a + e = 0$ then **return** $\perp$

Otherwise we find a BRIDGE^N instance

Probability of outputting $\perp$

$$\le q_{\text{Sign}} \times \frac{q_{\text{RO}} + q_{\text{Sign}}}{q} = \mathcal{O}\left(\frac{t^2}{q}\right)$$

$$\le \frac{q_{\text{RO}} + q_{\text{Sign}}}{q - |\mathcal{L}|} + \frac{1}{2^k} = \mathcal{O}\left(\frac{t}{2^k}\right)$$

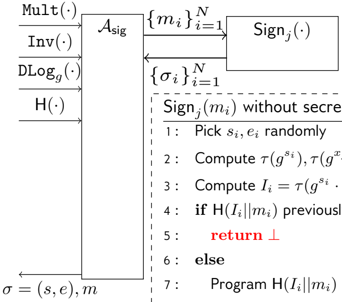
Proof Sketch: Security Reduction

Reduction $\mathcal{A}_{\text{bridge}}$

Given: $\mathfrak{g} = \tau(g), \tau(h_i) = \tau(g^{x_i}), 1 \le i \le N, q$

Initialize $\mathcal{L} = \{(\tau(g), \vec{0}, 1), (\tau(g^{x_i}), \vec{e_i}, 0) \text{ for } 1 \le i \le N\}, H_{\text{resp}} = \{\}$

/* begin simulation */



Sign$_j(m_i)$ without secret key x_j $(1 \le j \le N)$

Pick s_i, e_i randomly

Compute $\tau(g^{s_i}), \tau(g^{x_j e_i}) = \tau(\tau^{-1}(h_j)^{e_i})$

Compute $I_i = \tau(g^{s_i} \cdot g^{-x_j e_i})$

if H$(I_i || m_i)$ previously queried **then**

  return $\perp$

else

  Program H$(I_i || m_i) := e_i$

  return $\sigma_i = (s_i, e_i)$

/* end simulation */

Compute: $I_\sigma = \tau(g^s \cdot g^{-xe}), e_\sigma =$ H$(I_\sigma || m)$ and check if $(I_\sigma, \vec{a}, b) \in \mathcal{L}$

  if no such triple exists then **return** $\perp$

If $\vec{a}$ has only one nonzero element a and if $a + e = 0$ then **return** $\perp$

Otherwise we find a BRIDGE^N instance

Probability of outputting $\perp$

$$\le q_{\text{Sign}} \times \frac{q_{\text{RO}} + q_{\text{Sign}}}{q} = \mathcal{O}\left(\frac{t^2}{q}\right)$$

$$\le \frac{q_{\text{RO}} + q_{\text{Sign}}}{q - |\mathcal{L}|} + \frac{1}{2^k} = \mathcal{O}\left(\frac{t}{2^k}\right)$$

$$\le \frac{q_{\text{RO}}}{2^k} = \mathcal{O}\left(\frac{t}{2^k}\right)$$

Proof Sketch: Security Reduction

Reduction $\mathcal{A}_{\text{bridge}}$

Given: $\mathfrak{g} = \tau(g), \tau(h_i) = \tau(g^{x_i}), 1 \le i \le N, q$

Initialize $\mathcal{L} = \{(\tau(g), \vec{0}, 1), (\tau(g^{x_i}), \vec{e_i}, 0) \text{ for } 1 \le i \le N\}, H_{\text{resp}} = \{\}$

/* begin simulation */

$\texttt{Mult}(\cdot)$

$\texttt{Inv}(\cdot)$

$\texttt{DLog}_g(\cdot)$

$\mathsf{H}(\cdot)$

$\mathcal{A}_{\text{sig}}$

$\{m_i\}_{i=1}^N$

$\mathsf{Sign}_j(\cdot)$

$\{\sigma_i\}_{i=1}^N$

$\sigma = (s, e), m$

$\mathsf{Sign}_j(m_i)$ without secret key x_j $(1 \le j \le N)$

1 : Pick s_i, e_i randomly

2 : Compute $\tau(g^{s_i}), \tau(g^{x_j e_i}) = \tau(\tau^{-1}(h_j)^{e_i})$

3 : Compute $I_i = \tau(g^{s_i} \cdot g^{-x_j e_i})$

4 : **if** $\mathsf{H}(I_i || m_i)$ previously queried **then**

5 : **return** $\perp$

6 : **else**

7 : Program $\mathsf{H}(I_i || m_i) := e_i$

8 : **return** $\sigma_i = (s_i, e_i)$

/* end simulation */

Compute: $I_\sigma = \tau(g^s \cdot g^{-xe}), e_\sigma = \mathsf{H}(I_\sigma || m)$ and check if $(I_\sigma, \vec{a}, b) \in \mathcal{L}$

if no such triple exists then **return** $\perp$

If $\vec{a}$ has only one nonzero element a and if $a + e = 0$ then **return** $\perp$

Otherwise we find a BRIDGE^N instance

Probability of outputting $\perp$

$$\le q_{\text{Sign}} \times \frac{q_{\text{RO}} + q_{\text{Sign}}}{q} = \mathcal{O}\left(\frac{t^2}{q}\right)$$

$$\le \frac{q_{\text{RO}} + q_{\text{Sign}}}{q - |\mathcal{L}|} + \frac{1}{2^k} = \mathcal{O}\left(\frac{t}{2^k}\right)$$

$$\le \frac{q_{\text{RO}}}{2^k} = \mathcal{O}\left(\frac{t}{2^k}\right)$$

comes with "short" Schnorr signatures

Proof Sketch: Security Reduction

Reduction $\mathcal{A}_{\text{bridge}}$

Given: $\mathfrak{g} = \tau(g), \tau(h_i) = \tau(g^{x_i}), 1 \le i \le N, q$

Initialize $\mathcal{L} = \{(\tau(g), \vec{0}, 1), (\tau(g^{x_i}), \vec{e_i}, 0) \text{ for } 1 \le i \le N\}, H_{\text{resp}} = \{\}$

/* begin simulation */

$\mathtt{Mult}(\cdot)$, $\mathtt{Inv}(\cdot)$, $\mathtt{DLog}_g(\cdot)$, $\mathsf{H}(\cdot)$ $\rightarrow$ $\mathcal{A}_{\text{sig}}$

$\mathcal{A}_{\text{sig}} \xrightarrow{\{m_i\}_{i=1}^N} \mathsf{Sign}_j(\cdot)$

$\mathsf{Sign}_j(\cdot) \xrightarrow{\{\sigma_i\}_{i=1}^N} \mathcal{A}_{\text{sig}}$

$\mathcal{A}_{\text{sig}} \rightarrow \sigma = (s, e), m$

$\mathsf{Sign}_j(m_i)$ without secret key x_j $(1 \le j \le N)$

1 : Pick s_i, e_i randomly

2 : Compute $\tau(g^{s_i}), \tau(g^{x_j e_i}) = \tau(\tau^{-1}(h_j)^{e_i})$

3 : Compute $I_i = \tau(g^{s_i} \cdot g^{-x_j e_i})$

4 : **if** $\mathsf{H}(I_i || m_i)$ previously queried **then**

5 : **return** $\perp$

6 : **else**

7 : Program $\mathsf{H}(I_i || m_i) := e_i$

8 : **return** $\sigma_i = (s_i, e_i)$

/* end simulation */

Compute: $I_\sigma = \tau(g^s \cdot g^{-xe}), e_\sigma = \mathsf{H}(I_\sigma || m)$ and check if $(I_\sigma, \vec{a}, b) \in \mathcal{L}$

if no such triple exists then **return** $\perp$

If $\vec{a}$ has only one nonzero element a and if $a + e = 0$ then **return** $\perp$

Otherwise we find a BRIDGE^N instance

Probability of outputting $\perp$

$$\le q_{\mathtt{Sign}} \times \frac{q_{\mathtt{RO}} + q_{\mathtt{Sign}}}{q} = \mathcal{O}\left(\frac{t^2}{q}\right)$$

$$\le \frac{q_{\mathtt{RO}} + q_{\mathtt{Sign}}}{q - |\mathcal{L}|} + \frac{1}{2^k} = \mathcal{O}\left(\frac{t}{2^k}\right)$$

$$\le \frac{q_{\mathtt{RO}}}{2^k} = \mathcal{O}\left(\frac{t}{2^k}\right)$$

comes with "short" Schnorr signatures

$$\therefore \Pr\left[\mathsf{SigForge}^{\mathtt{GO},N}_{\mathcal{A}_{\text{sig}},\Pi}(k) = 1\right]$$

$$\le \Pr\left[\mathsf{BridgeChal}^{\mathtt{GO},N}_{\mathcal{A}_{\text{bridge}}}(k) = 1\right] + \mathcal{O}\left(\frac{t}{2^k}\right)$$

$$\le \mathcal{O}\left(\frac{t+N}{2^k}\right).$$

Conclusion and Future Work

Our Contributions

- We showed that the ***short*** Schnorr signatures provides k-bits of security in ***both*** single and multi-user settings under the programmable ROM and the GGM.
- Breaking multi-user security of short Schnorr signatures in "1-out-of-N" setting is not ***easier*** than breaking a single instance.
- The short Schnorr signature is still secure even if we allow a restricted discrete-log oracle in the GGM.
- We provide a new proof technique which keeps track of the known and the partially known set in a global list.

Future Work

- Security of (short) Schnorr signatures against preprocessing attacks [CK18].
  - Preprocessing attacks are used to criticize non-standard generic group models proposed earlier [SJ00, KMP16].
  - Preprocessing phase is not doable in both non-standard models, whereas it is clearly captured by the original model.

References I

Daniel J. Bernstein, ***Multi-user Schnorr security, revisited***, Cryptology ePrint Archive, Report 2015/996, 2015, `http://eprint.iacr.org/2015/996`.

Henry Corrigan-Gibbs and Dmitry Kogan, ***The discrete-logarithm problem with preprocessing***, EUROCRYPT 2018, Part II (Jesper Buus Nielsen and Vincent Rijmen, eds.), LNCS, vol. 10821, Springer, Heidelberg, April / May 2018, pp. 415–447.

Alexander W. Dent, ***Adapting the weaknesses of the random oracle model to the generic group model***, ASIACRYPT 2002 (Yuliang Zheng, ed.), LNCS, vol. 2501, Springer, Heidelberg, December 2002, pp. 100–109.

Nils Fleischhacker, Tibor Jager, and Dominique Schröder, ***On tight security proofs for Schnorr signatures***, ASIACRYPT 2014, Part I (Palash Sarkar and Tetsu Iwata, eds.), LNCS, vol. 8873, Springer, Heidelberg, December 2014, pp. 512–531.

David Freeman, Michael Scott, and Edlyn Teske, ***A taxonomy of pairing-friendly elliptic curves***, Journal of Cryptology **23** (2010), no. 2, 224–280.

S. Galbraith, J. Malone-Lee, and N. P. Smart, ***Public key signatures in the multi-user setting***, Inf. Process. Lett. **83** (2002), no. 5, 263–266.

Don Johnson, Alfred Menezes, and Scott Vanstone, ***The elliptic curve digital signature algorithm (ecdsa)***, International Journal of Information Security **1** (2001), no. 1, 36–63.

Eike Kiltz, Daniel Masny, and Jiaxin Pan, ***Optimal security proofs for signatures from identification schemes***, CRYPTO 2016, Part II (Matthew Robshaw and Jonathan Katz, eds.), LNCS, vol. 9815, Springer, Heidelberg, August 2016, pp. 33–61.

Gregory Neven, Nigel P. Smart, and Bogdan Warinschi, ***Hash function requirements for schnorr signatures***, Journal of Mathematical Cryptology **3** (2009).

References II

David Pointcheval and Jacques Stern, ***Security proofs for signature schemes***, EUROCRYPT'96 (Ueli M. Maurer, ed.), LNCS, vol. 1070, Springer, Heidelberg, May 1996, pp. 387–398.

Yannick Seurin, ***On the exact security of Schnorr-type signatures in the random oracle model***, EUROCRYPT 2012 (David Pointcheval and Thomas Johansson, eds.), LNCS, vol. 7237, Springer, Heidelberg, April 2012, pp. 554–571.

Victor Shoup, ***Lower bounds for discrete logarithms and related problems***, EUROCRYPT'97 (Walter Fumy, ed.), LNCS, vol. 1233, Springer, Heidelberg, May 1997, pp. 256–266.

Claus-Peter Schnorr and Markus Jakobsson, ***Security of signed ElGamal encryption***, ASIACRYPT 2000 (Tatsuaki Okamoto, ed.), LNCS, vol. 1976, Springer, Heidelberg, December 2000, pp. 73–89.

Questions?