Galindo-Garcia Identity-Based Signature Revisited.

Sanjit Chatterjee, Chethan Kamath and Vikas Kumar

Indian Institute of Science, Bangalore

November 2, 2013

Table of contents

FORMAL DEFINITIONS

Definition–Public-Key Signature

An PKS scheme consists of three PPT algorithms $\{\mathcal{K}, \mathcal{S}, \mathcal{V}\}$

- Key Generation, $\mathcal{K}$
  - Used by the user to generate the public-private key pair $(\mathtt{pk}, \mathtt{sk})$
  - $\mathtt{pk}$ is published and the $\mathtt{sk}$ kept secret
  - Run on a *security parameter* κ

$$(\mathtt{pk}, \mathtt{sk}) \xleftarrow{\$} \mathcal{K}(\kappa)$$

- Signing, $\mathcal{S}$
  - Used by the user to generate signature on some message m
  - The secret key $\mathtt{sk}$ used for signing

$$\sigma \xleftarrow{\$} \mathcal{S}(\mathtt{sk}, m)$$

- Verification, $\mathcal{V}$
  - Outputs 1 if σ is a valid signature on m; else, outputs 0

$$\mathtt{b} \leftarrow \mathcal{V}(\sigma, m, \mathtt{pk})$$

Definition–Identity-Based Signature

An IBS scheme consists of four PPT algorithms $\{\mathcal{G}, \mathcal{E}, \mathcal{S}, \mathcal{V}\}$

- Set-up, $\mathcal{G}$
  - Used by the PKG to generate the public parameters ($\mathtt{mpk}$) and master secret ($\mathtt{msk}$)
  - $\mathtt{mpk}$ is published and the $\mathtt{msk}$ kept secret
  - Run on a *security parameter* κ

$$(\mathtt{mpk}, \mathtt{msk}) \xleftarrow{\$} \mathcal{G}(\kappa)$$

- Key Extraction, $\mathcal{E}$
  - Used by the PKG to generate the user secret key ($\mathtt{usk}$)
  - $\mathtt{usk}$ is then distributed through a secure channel

$$\mathtt{usk} \xleftarrow{\$} \mathcal{E}(\mathtt{id}, \mathtt{msk})$$

Definition–Identity-Based Signature...

An IBS scheme consists of four PPT algorithms $\{\mathcal{G}, \mathcal{E}, \mathcal{S}, \mathcal{V}\}$

- Signing, $\mathcal{S}$
  - Used by a user with identity $\mathtt{id}$ to generate signature on some message m
  - The user secret key $\mathtt{usk}$ used for signing

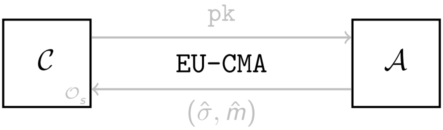
$$\sigma \xleftarrow{\$} \mathcal{S}(\mathtt{usk}, \mathtt{id}, m, \mathtt{mpk})$$

- Verification, $\mathcal{V}$
  - Outputs 1 if σ is a valid signature on m by the user with identity $\mathtt{id}$
  - Otherwise, outputs 0

$$\mathtt{b} \leftarrow \mathcal{V}(\sigma, \mathtt{id}, m, \mathtt{mpk})$$

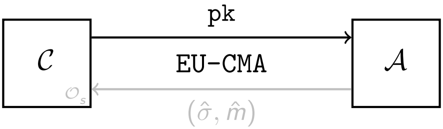
SECURITY MODELS FOR PKS AND IBS

Security Model for PKS–EU-CMA



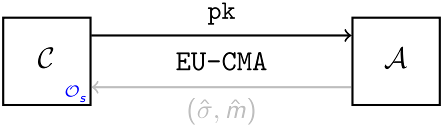
- Existential unforgeability under chosen-message attack

Security Model for PKS–EU-CMA



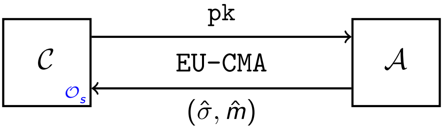
- Existential unforgeability under chosen-message attack
- $\mathcal{C}$ generates key-pair $(\mathtt{pk}, \mathtt{sk})$ and passes $\mathtt{pk}$ to $\mathcal{A}$.

Security Model for PKS–EU-CMA



- Existential unforgeability under chosen-message attack
- $\mathcal{C}$ generates key-pair $(\mathtt{pk}, \mathtt{sk})$ and passes $\mathtt{pk}$ to $\mathcal{A}$.
- Signature Queries: Access to a signing oracle $\mathcal{O}_s$

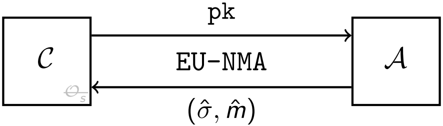
Security Model for PKS–EU-CMA



- Existential unforgeability under chosen-message attack
- $\mathcal{C}$ generates key-pair $(\mathtt{pk}, \mathtt{sk})$ and passes $\mathtt{pk}$ to $\mathcal{A}$.
- Signature Queries: Access to a signing oracle $\mathcal{O}_s$
- Forgery: $\mathcal{A}$ wins if
  - $\hat{\sigma}$ is a *valid* signature on $\hat{m}$.
  - $\mathcal{A}$ has *not* made a signature query on $\hat{m}$.
- Adversary's advantage in the game:

$$\Pr\left[1 \leftarrow \mathcal{V}(\hat{\sigma}, \hat{m}, \mathtt{pk}) \mid (\mathtt{sk}, \mathtt{pk}) \xleftarrow{\$} \mathcal{K}(\kappa); (\hat{\sigma}, \hat{m}) \xleftarrow{\$} \mathcal{A}^{\mathcal{O}_s}(\mathtt{pk})\right]$$

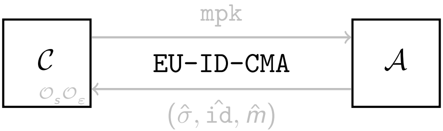
Security Model for PKS–EU-NMA



- Existential unforgeability under no-message attack
- $\mathcal{C}$ generates key-pair $(\mathtt{pk}, \mathtt{sk})$ and passes $\mathtt{pk}$ to $\mathcal{A}$.
- ~~Signature Queries: Access to a signing oracle $\mathcal{O}_s$~~
- Forgery: $\mathcal{A}$ wins if
  - $\hat{\sigma}$ is a *valid* signature on $\hat{m}$.
  - ~~$\mathcal{A}$ has *not* made a signature query on $\hat{m}$.~~
- Adversary's advantage in the game:

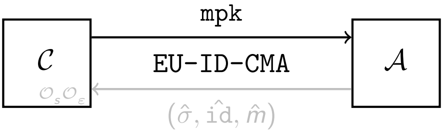
$$\Pr\left[1 \leftarrow \mathcal{V}(\hat{\sigma}, \hat{m}, \mathtt{pk}) \mid (\mathtt{sk}, \mathtt{pk}) \xleftarrow{\$} \mathcal{K}(\kappa); (\hat{\sigma}, \hat{m}) \xleftarrow{\$} \mathcal{A}(\mathtt{pk})\right]$$

Security Model for IBS: EU-ID-CMA



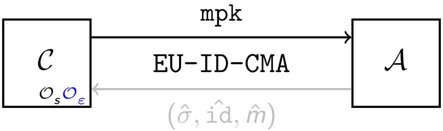
- Existential unforgeability with adaptive identity under no-message attack

Security Model for IBS: EU-ID-CMA



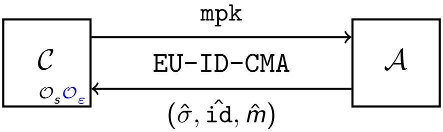
- Existential unforgeability with adaptive identity under no-message attack
- $\mathcal{C}$ generates key-pair $(\mathtt{mpk}, \mathtt{msk})$ and passes $\mathtt{mpk}$ to $\mathcal{A}$.

Security Model for IBS: EU-ID-CMA



- Existential unforgeability with adaptive identity under no-message attack
- $\mathcal{C}$ generates key-pair $(\mathtt{mpk}, \mathtt{msk})$ and passes $\mathtt{mpk}$ to $\mathcal{A}$.
- Extract Queries, Signature Queries

Security Model for IBS: EU-ID-CMA

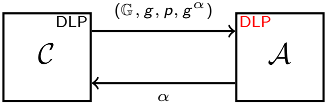


- Existential unforgeability with adaptive identity under no-message attack
- $\mathcal{C}$ generates key-pair $(\mathtt{mpk}, \mathtt{msk})$ and passes $\mathtt{mpk}$ to $\mathcal{A}$.
- Extract Queries, Signature Queries
- Forgery: $\mathcal{A}$ wins if
  - $\hat{\sigma}$ is a *valid* signature on $\hat{m}$ by $\hat{\mathtt{id}}$.
  - $\mathcal{A}$ has *not* made an extract query on $\hat{\mathtt{id}}$.
  - $\mathcal{A}$ has *not* made a signature query on $(\hat{\mathtt{id}}, \hat{m})$.
- Adversary's advantage in the game:

$$\Pr\left[1 \leftarrow \mathcal{V}(\hat{\sigma}, \hat{\mathtt{id}}, \hat{m}, \mathtt{mpk}) \mid (\mathtt{msk}, \mathtt{mpk}) \xleftarrow{\$} \mathcal{G}(\kappa); (\hat{\sigma}, \hat{\mathtt{id}}, \hat{m}) \xleftarrow{\$} \mathcal{A}^{\mathcal{O}_{\{s,\varepsilon\}}}(\mathtt{mpk})\right]$$

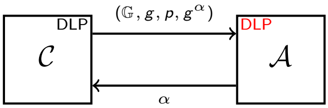
Hardness Assumption: Discrete-log Assumption

Discrete-log problem for a group $\mathbb{G} = \langle g \rangle$ and $|\mathbb{G}| = p$



Hardness Assumption: Discrete-log Assumption

Discrete-log problem for a group $\mathbb{G} = \langle g \rangle$ and $|\mathbb{G}| = p$



Definition. The DLP in $\mathbb{G}$ is to find α given g^{α}, where $\alpha \in_R \mathbb{Z}_p$. An adversary $\mathcal{A}$ has advantage ϵ in solving the DLP if

$$\Pr\left[\alpha' = \alpha \mid \alpha \in_R \mathbb{Z}_p; \alpha' \leftarrow \mathcal{A}(\mathbb{G}, p, g, g^{\alpha})\right] \geq \epsilon.$$

The (ϵ, t)-discrete-log assumption *holds* in $\mathbb{G}$ if no adversary has advantage at least ϵ in solving the DLP in time at most t.

GALINDO-GARCIA IBS

Galindo-Garcia IBS - Salient Features

- Derived from Schnorr signature scheme
- Based on the *discrete-log* assumption
- Efficient, simple and does not use *pairing*
- Security argued using *oracle replay* attacks
- Uses the *random oracle* heuristic

SCHNORR SIGNATURE AND THE ORACLE REPLAY ATTACK

Schnorr Signature

The Setting.

1. We work in group $\mathbb{G} = \langle g \rangle$ of prime order p.
2. A hash function $\mathsf{H} : \{0,1\}^* \rightarrow \mathbb{Z}_p$ is used.

Key Generation. $\mathcal{K}(\kappa)$:

1. Select $z \in_R \mathbb{Z}_p$ as the secret key $\mathtt{sk}$
2. Set $Z := g^z$ as the public key $\mathtt{pk}$

Signing. $\mathcal{S}(m, \mathtt{sk})$:

1. Let $\mathtt{sk} = z$. Select $r \in_R \mathbb{Z}_p$, set $R := g^r$ and $c := \mathsf{H}(m, R)$.
2. The signature on m is $\sigma := (y, R)$ where

$$y := r + zc$$

Verification. $\mathcal{V}(\sigma, m)$:

1. Let $\sigma = (y, R)$ and $c = \mathsf{H}(m, R)$.
2. σ is valid if

$$g^y = RZ^c$$

Security of Schnorr Signature–An Intuition

- Consider an adversary $\mathcal{A}$ with ability to launch chosen-message attack on the Schnorr signature scheme.
- Let $\{\sigma_0, \ldots, \sigma_{n-1}\}$ with $\sigma_i = (y_i = r_i + zc_i, R_i)$ on m_i be the signatures that $\mathcal{A}$ receives.

$$\begin{pmatrix} 1 & 0 & \cdots & 0 & c_0 \\ 0 & 1 & \cdots & 0 & c_1 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & c_{n-1} \end{pmatrix} \times \begin{pmatrix} r_0 \\ r_1 \\ \vdots \\ r_{n-1} \\ z \end{pmatrix} = \begin{pmatrix} y_0 \\ y_1 \\ \vdots \\ r_{n-1} \end{pmatrix}$$

Security of Schnorr Signature–An Intuition...

- *However*, $\mathcal{A}$ can solve for x if it gets two equations containing the same r but different c, i.e.

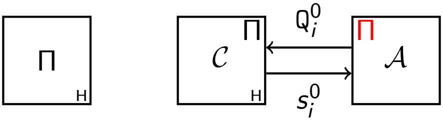
$$y = r + zc \quad \text{and} \quad \bar{y} = r + z\bar{c}$$

implies

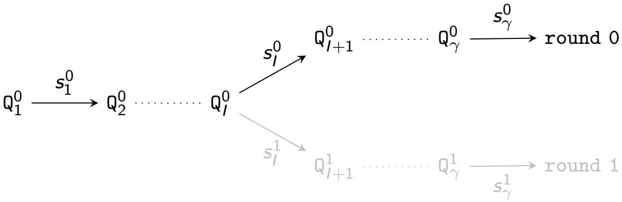
$$z = \frac{y - \bar{y}}{c - \bar{c}} \sqcap$$

The Oracle Replay Attack

- Random oracle H–i^{th} random oracle query $\mathtt{Q}_i^0$ replied with s_i^0.

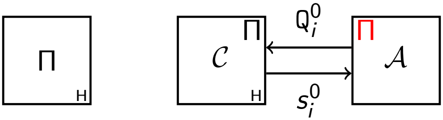


Tape re-wound to $\mathtt{Q}_I^0$
Simulation in `round 1` from $\mathtt{Q}_I^0$ using a *different* random function

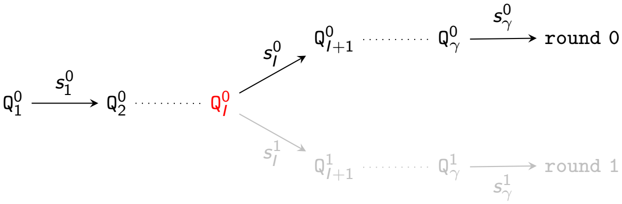


The Oracle Replay Attack

- Random oracle H–i^{th} random oracle query $\mathtt{Q}_i^0$ replied with s_i^0.

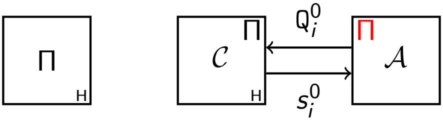


1. Tape re-wound to $\mathtt{Q}_I^0$
   Simulation in `round 1` from $\mathtt{Q}_I^0$ using a *different* random function

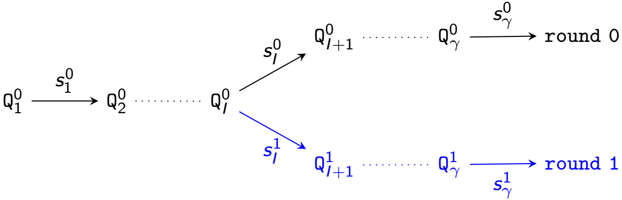


The Oracle Replay Attack

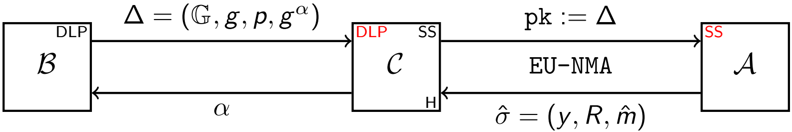
- Random oracle H–i^{th} random oracle query $\mathtt{Q}_i^0$ replied with s_i^0.

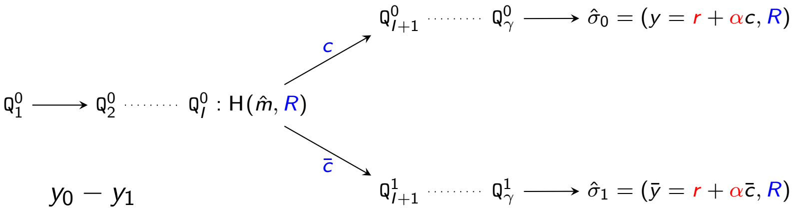


1. Tape re-wound to $\mathtt{Q}_I^0$
2. Simulation in `round 1` from $\mathtt{Q}_I^0$ using a *different* random function



Proving Security of Schnorr Signature using ORA





$$\alpha = \frac{y_0 - y_1}{c - \bar{c}}$$

Forking Lemma

- The oracle replay attack formalised through the forking algorithm
- The *forking lemma* gives a lower bound on the success probability of the oracle replay attack (frk) in terms of the success probability of the adversary during a particular run (acc)

Forking Lemma

- The oracle replay attack formalised through the forking algorithm
- The *forking lemma* gives a lower bound on the success probability of the oracle replay attack (frk) in terms of the success probability of the adversary during a particular run (acc)
- Types of forking algorithms

Forking Algorithm	#Oracles	#Replay Attacks	Success Prob. ($\approx$)
GF–*General Forking* - $\mathcal{F}_{\mathcal{W}}$	1	1 (*i.e.* 2 runs)	$\frac{acc^2}{\gamma}$
MF–*Multiple-Forking(n)* - $\mathcal{M}_{\mathcal{W},n}$	2	2n-1 (*i.e.* 2n runs)	$\frac{acc^n}{\gamma^{2n}}$

γ–Upper bound on the number of oracle queries

Forking Lemma...

E.g. Multiple-forking algorithm for $n = 3$.

$$
\begin{tikzcd}
 & & & & & \mathtt{Q}^0_{I_0+1} \arrow[r, dotted, no head] & \mathtt{Q}^0_{\gamma} \arrow[r] & \sigma\ \texttt{[round 0]} \\
 & & & \mathtt{Q}^0_{J_0+1} \arrow[r, dotted, no head] & {\color{blue}\mathtt{Q}^0_{I_0}} \arrow[ur, "s^0_{I_0}"] \arrow[dr, "s^1_{I_0}"'] & & & \\
 & & & & & \mathtt{Q}^1_{I_0+1} \arrow[r, dotted, no head] & \mathtt{Q}^1_{\gamma} \arrow[r] & \sigma_1\ \texttt{[round 1]} \\
\mathtt{Q}^0_1 \arrow[r] & \mathtt{Q}^0_2 \arrow[r, dotted, no head] & {\color{red}\mathtt{Q}^0_{J_0}} \arrow[uur, "s^0_{J_0}"] \arrow[ddr, "s^2_{J_0}"'] & & & & & \\
 & & & & & \mathtt{Q}^2_{I_0+1} \arrow[r, dotted, no head] & \mathtt{Q}^2_{\gamma} \arrow[r] & \sigma_2\ \texttt{[round 2]} \\
 & & & \mathtt{Q}^2_{J_0+1} \arrow[r, dotted, no head] & {\color{blue}\mathtt{Q}^2_{I_0}} \arrow[ur, "s^2_{I_0}"] \arrow[dr, "s^3_{I_0}"'] & & & \\
 & & & & & \mathtt{Q}^3_{I_0+1} \arrow[r, dotted, no head] & \mathtt{Q}^3_{\gamma} \arrow[r] & \sigma_3\ \texttt{[round 3]}
\end{tikzcd}
$$

GALINDO-GARCIA IBS–CONSTRUCTION

The Construction

Set-up. $\mathcal{G}(\kappa)$:

1. Let $\mathbb{G} = \langle g \rangle$ be a group of prime order p.
2. Return $z \in_R \mathbb{Z}_p$ as `msk` and $(\mathbb{G}, p, g, g^z, \mathsf{H}, \mathsf{G})$ as `mpk`, where H and G are hash functions

$$\mathsf{H} : \{0,1\}^* \to \mathbb{Z}_p \;\text{ and }\; \mathsf{G} : \{0,1\}^* \to \mathbb{Z}_p.$$

The Construction

Set-up. $\mathcal{G}(\kappa)$:

1. Let $\mathbb{G} = \langle g \rangle$ be a group of prime order p.
2. Return $z \in_R \mathbb{Z}_p$ as $\mathtt{msk}$ and $(\mathbb{G}, p, g, g^z, \mathsf{H}, \mathsf{G})$ as $\mathtt{mpk}$, where H and G are hash functions

$$\mathsf{H} : \{0,1\}^* \to \mathbb{Z}_p \quad \text{and} \quad \mathsf{G} : \{0,1\}^* \to \mathbb{Z}_p.$$

Key Extraction. $\mathcal{E}(\mathtt{id}, \mathtt{msk}, \mathtt{mpk})$:

1. Select $r \in_R \mathbb{Z}_p$ and set $R := g^r$.
2. Return $\mathtt{usk} := (y, R)$ as $\mathtt{usk}$, where

$$y := r + zc \quad \text{and} \quad c := \mathsf{H}(R, \mathtt{id}).$$

The Construction

Set-up. $\mathcal{G}(\kappa)$:

1. Let $\mathbb{G} = \langle g \rangle$ be a group of prime order p.
2. Return $z \in_R \mathbb{Z}_p$ as $\mathtt{msk}$ and $(\mathbb{G}, p, g, g^z, \mathsf{H}, \mathsf{G})$ as $\mathtt{mpk}$, where H and G are hash functions

$$\mathsf{H} : \{0,1\}^* \to \mathbb{Z}_p \quad \text{and} \quad \mathsf{G} : \{0,1\}^* \to \mathbb{Z}_p.$$

Key Extraction. $\mathcal{E}(\mathtt{id}, \mathtt{msk}, \mathtt{mpk})$:

1. Select $r \in_R \mathbb{Z}_p$ and set $R := g^r$.
2. Return $\mathtt{usk} := (y, R)$ as $\mathtt{usk}$, where

$$y := r + zc \quad \text{and} \quad c := \mathsf{H}(R, \mathtt{id}).$$

Signing. $\mathcal{S}(\mathtt{id}, m, \mathtt{usk}, \mathtt{mpk})$:

1. Let $\mathtt{usk} = (y, R)$. Select $a \in_R \mathbb{Z}_p$ and set $A := g^a$.
2. Return $\sigma := (A, b, R)$ as the signature, where

$$b := a + yd \quad \text{and} \quad d := \mathsf{G}(\mathtt{id}, A, m).$$

The Construction

Verification. $\mathcal{V}(\sigma, \mathtt{id}, m, \mathtt{mpk})$:

1. Let $\sigma = (A, b, R)$, $c := \mathsf{H}(R, \mathtt{id})$ and $d := \mathsf{G}(\mathtt{id}, A, m)$.
2. The signature is valid if

$$g^b = A(R \cdot (g^z)^c)^d.$$

ORIGINAL SECURITY ARGUMENT

Original Security Argument

- Let $\hat{\sigma} = (b, A, R)$ be the forgery produced by $\mathcal{A}$ on $(\hat{\mathtt{id}}, \hat{m})$.

$$\begin{tikzcd}
 & \mathsf{U} \arrow[dl] \arrow[dr] & \\
\mathsf{E} \arrow[d] & & \bar{\mathsf{E}} \arrow[d] \\
\mathcal{B}_1 & & \mathcal{B}_2
\end{tikzcd}$$

E: Event that $\mathcal{A}$ forges using the same randomiser R as given by $\mathcal{C}$ as part of signature query on $\hat{\mathtt{id}}$.

Original Security Argument

- Let $\hat{\sigma} = (b, A, R)$ be the forgery produced by $\mathcal{A}$ on $(\hat{\mathtt{id}}, \hat{m})$.

$$\begin{tikzcd}
 & \mathsf{U} \arrow[dl] \arrow[dr] & \\
\mathsf{E} \arrow[d] & & \bar{\mathsf{E}} \arrow[d] \\
\mathcal{B}_1 & & \mathcal{B}_2
\end{tikzcd}$$

E: Event that $\mathcal{A}$ forges using the same randomiser R as given by $\mathcal{C}$ as part of signature query on $\hat{\mathtt{id}}$.

- In both $\mathcal{B}_1$ and $\mathcal{B}_2$, solving DLP is *reduced* to breaking the IBS.

In a Nutshell

Reduction	Success Prob. ($\approx$)	Forking Used
$\mathcal{B}_1$	$\frac{\epsilon^2}{q_{\mathsf{G}}^3}$	General Forking–$\mathcal{F}_{\mathcal{W}}$
$\mathcal{B}_2$	$\frac{\epsilon^4}{(q_{\mathsf{H}} q_{\mathsf{G}})^6}$	Multiple-Forking–$\mathcal{M}_{\mathcal{W},3}$

Our Contribution

- We found several problems with $\mathcal{B}_1$ and $\mathcal{B}_2$
    1. $\mathcal{B}_1$: Fails in the standard security model for IBS
    2. $\mathcal{B}_2$: All the adversarial strategies were not covered

Our Contribution

- We found several problems with $\mathcal{B}_1$ and $\mathcal{B}_2$
  1. $\mathcal{B}_1$: Fails in the standard security model for IBS
  2. $\mathcal{B}_2$: All the adversarial strategies were not covered
- The adversary *is able to* distinguish a simulation from the real execution of the protocol.

Our Contribution

- We found several problems with $\mathcal{B}_1$ and $\mathcal{B}_2$
  1. $\mathcal{B}_1$: Fails in the standard security model for IBS
  2. $\mathcal{B}_2$: All the adversarial strategies were not covered
- The adversary *is able to* distinguish a simulation from the real execution of the protocol.
- Positive contribution:
  1. We give a *detailed* new security argument
  2. *Tighter* than the original security argument

NEW SECURITY ARGUMENT

New Security Argument

- Let $\hat{\sigma} = (b, A, R)$ be the forgery produced by $\mathcal{A}$ on $(\hat{\mathtt{id}}, \hat{m})$.

New Security Argument

- Let $\hat{\sigma} = (b, A, R)$ be the forgery produced by $\mathcal{A}$ on $(\hat{\mathtt{id}}, \hat{m})$.

$$\begin{tikzcd}
 & & \mathsf{U} \arrow[dl] \arrow[dr] & & \\
 & \mathsf{E} \arrow[d] & & \bar{\mathsf{E}} \arrow[dl] \arrow[dr] & \\
 & \mathcal{R}_1 & \mathsf{F} \arrow[d] & & \bar{\mathsf{F}} \arrow[d] \\
 & & \mathcal{R}_2 & & \mathcal{R}_3
\end{tikzcd}$$

F: Event that $\mathcal{A}$ calls $\mathsf{G}(\hat{\mathtt{id}}, A, \hat{m})$ before $\mathsf{H}(R, \hat{\mathtt{id}})$.

New Security Argument

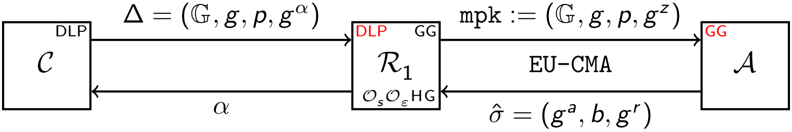
- Let $\hat{\sigma} = (b, A, R)$ be the forgery produced by $\mathcal{A}$ on $(\hat{\mathtt{id}}, \hat{m})$.

$$\begin{tikzcd}
 & \mathsf{U} \arrow[dl] \arrow[dr] & & \\
\mathsf{E} \arrow[d] & & \bar{\mathsf{E}} \arrow[dl] \arrow[dr] & \\
\mathcal{R}_1 & \mathsf{F} \arrow[d] & & \bar{\mathsf{F}} \arrow[d] \\
 & \mathcal{R}_2 & & \mathcal{R}_3
\end{tikzcd}$$

F: Event that $\mathcal{A}$ calls $\mathsf{G}(\hat{\mathtt{id}}, A, \hat{m})$ before $\mathsf{H}(R, \hat{\mathtt{id}})$.

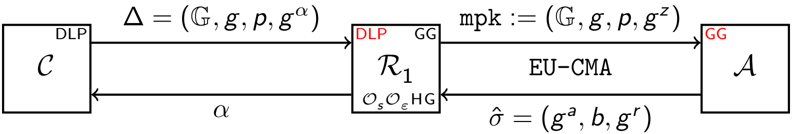
1. Problems with $\mathcal{B}_1$ addressed in $\mathcal{R}_1$
2. $\mathcal{R}_2$ covers the unaddressed adversarial strategy in $\mathcal{B}_2$
3. $\mathcal{R}_3$ same as the original reduction $\mathcal{B}_2$

Reduction $\mathcal{R}_1$



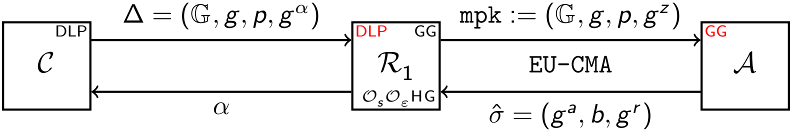
- Problem instance plugged in the randomiser R (as in $\mathcal{B}_1$)

Reduction $\mathcal{R}_1$



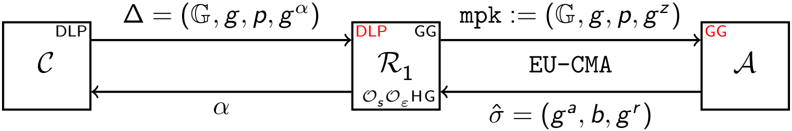
- Problem instance plugged in the randomiser R (as in $\mathcal{B}_1$)
- *Coron's technique* used to assign target identities (instead of guessing) – security degradation *reduced* to $\mathrm{O}(q_\varepsilon)$
- Signature Query. $\mathcal{O}_s(\mathtt{id}, m)$ –
  - Toss a biased coin β

Reduction $\mathcal{R}_1$



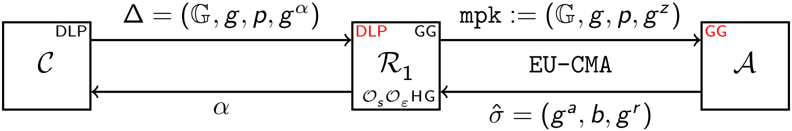
- Problem instance plugged in the randomiser R (as in $\mathcal{B}_1$)
- *Coron's technique* used to assign target identities (instead of guessing) – security degradation *reduced* to $\mathrm{O}(q_\varepsilon)$
- Signature Query. $\mathcal{O}_s(\mathtt{id}, m)$ –
  - Toss a biased coin β
    1. If $\beta = 0$, signature given with randomiser R containing g^{α}
    2. Else, $\mathcal{R}_1$ uses knowledge of $\mathtt{msk}$ to generate user private key for $\mathtt{id}$ and then computes signature using $\mathcal{S}$

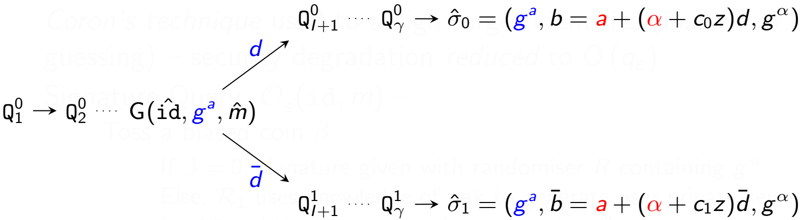
Reduction $\mathcal{R}_1$



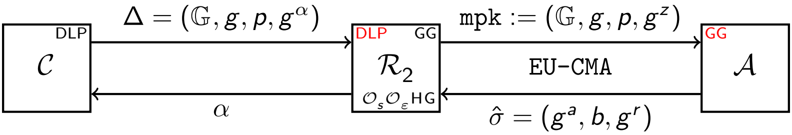
- Problem instance plugged in the randomiser R (as in $\mathcal{B}_1$)
- *Coron's technique* used to assign target identities (instead of guessing) – security degradation *reduced* to $\mathrm{O}(q_\varepsilon)$
- Signature Query. $\mathcal{O}_s(\mathtt{id}, m)$ –
  - Toss a biased coin β
    1. If $\beta = 0$, signature given with randomiser R containing g^{α}
    2. Else, $\mathcal{R}_1$ uses knowledge of msk to generate user private key for id and then computes signature using $\mathcal{S}$
- *General forking algorithm* $(\mathcal{F}_{\mathcal{W}})$ used to solve DLP (as in $\mathcal{B}_1$)

Reduction $\mathcal{R}_1$



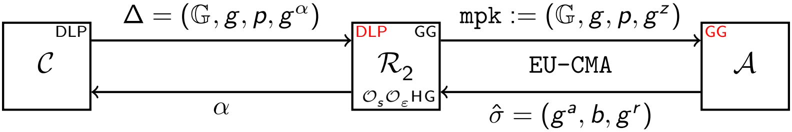


Reduction $\mathcal{R}_2$



- Problem instance plugged in the public key pk (as in $\mathcal{B}_2$)
- Signature queries are handled as in $\mathcal{B}_2$
- *However*, Multiple-forking with $n = 1$ ($\mathcal{M}_{\mathcal{W},1}$) used to solve the DLP
- Hence, tighter than $\mathcal{B}_2$

Reduction $\mathcal{R}_2$



Problem instance plugged in the public key pk (as in $\mathcal{B}_2$)

Signature queries are handled as in $\mathcal{B}_2$

Improved Multiple-forking with $n = 1$ ($\mathcal{M}_{\mathcal{W},1}$) used to solve the DLP

Hence, *tighter* than $\mathcal{B}_2$

$$\mathtt{Q}^0_{I_0+1} \cdots \mathtt{Q}^0_{\gamma} \rightarrow \hat{\sigma}_0 = (g^{a}, b = a + (r + \alpha c)d, g^{r})$$

$$\mathtt{Q}^0_1 \rightarrow \mathtt{Q}^0_2 \cdots \mathsf{G}(\hat{\mathtt{id}}, g^{a}, \hat{m}) \xrightarrow{d} \mathtt{Q}^0_{J_0+1} \cdots \mathsf{H}(\hat{\mathtt{id}}, g^{r}) \begin{cases} \xrightarrow{c} \\ \xrightarrow{\bar{c}} \end{cases}$$

$$\mathtt{Q}^1_{I_0+1} \cdots \mathtt{Q}^1_{\gamma} \rightarrow \hat{\sigma}_1 = (g^{a}, \bar{b} = a + (r + \alpha \bar{c})d, g^{r})$$

In a Nutshell

Reduction	Success Prob. ($\approx$)	Forking Used
$\mathcal{R}_1$	$\frac{\epsilon^2}{q_G q_\varepsilon}$	$\mathcal{F}_\mathcal{W}$
$\mathcal{R}_2$	$\frac{\epsilon^2}{(q_H+q_G)^2}$	$\mathcal{M}_{\mathcal{W},1}$
$\mathcal{R}_3$	$\frac{\epsilon^4}{(q_H+q_G)^6}$	$\mathcal{M}_{\mathcal{W},3}$

Conclusion and Future Work

We revisited the Galindo-Garcia IBS security argument

- Analysed the original security proof; fixed ambiguities
- Provided an improved security proof

Future Work

- Replacing the ‘costly’ multiple-forking for even tighter reductions–*dependent* random oracles.

THANK YOU!