# Stronger Security for Threshold Blind Signatures

Anja Lehmann[1], Phillip Nazarian[2], Cavit Özbay[1]

[1]Hasso Plattner Institute, University of Potsdam, Germany
[2]University of California Irvine, USA

Eurocrypt 2025

# Signatures

A **signature scheme** consists of algorithms $(\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$:

- $\mathsf{KeyGen} \rightarrow (pk, sk)$
- $\mathsf{Sign}(sk, m) \rightarrow \sigma$
- $\mathsf{Verify}(pk, m, \sigma) \rightarrow 0/1$

# Signatures

A **signature scheme** consists of algorithms (KeyGen, Sign, Verify):

- KeyGen $\rightarrow (pk, sk)$
- Sign$(sk, m) \rightarrow \sigma$
- Verify$(pk, m, \sigma) \rightarrow 0/1$

**Unforgeability** is the security property.

It is defined by the Chosen Message Attack (CMA) game, which allows the attacker to request signatures on arbitrary messages.

The attacker wins if they output a **non-trivial** $(m, \sigma)$ pair.

A message/signature pair $(m, \sigma)$ is **trivial** if the attacker queried a signature on $m$.

# Threshold Signatures

A **threshold signature scheme** is a protocol for 1 user and $n$ issuers, with signing threshold $t$. It consists of algorithms (KeyGen, ISign, USign, Verify):

- $\mathsf{KeyGen}(n, t) \rightarrow (pk, \{sk_1, sk_2, \ldots, sk_n\})$
- $\mathsf{ISign}(i, sk_i, m) \rightarrow pm_i$
- $\mathsf{USign}(pk, m, \mathcal{S}, \{pm_i\}_{i \in \mathcal{S}}) \rightarrow \sigma$ $\quad // \ |\mathcal{S}| = t$
- $\mathsf{Verify}(pk, m, \sigma) \rightarrow 0/1$

# Threshold Signatures

A **threshold signature scheme** is a protocol for 1 user and $n$ issuers, with signing threshold $t$. It consists of algorithms (KeyGen, ISign, USign, Verify):

- $\mathsf{KeyGen}(n, t) \to (pk, \{sk_1, sk_2, \ldots, sk_n\})$
- $\mathsf{ISign}(i, sk_i, m) \to pm_i$
- $\mathsf{USign}(pk, m, \mathcal{S}, \{pm_i\}_{i \in \mathcal{S}}) \to \sigma$ $\qquad // \ |\mathcal{S}| = t$
- $\mathsf{Verify}(pk, m, \sigma) \to 0/1$

We now need some notion of **threshold unforgeability**.

Similar to before, the attacker arbitrarily queries "partial signatures" from issuers and wins by outputting a non-trivial message/signature pair. But how should triviality be defined?

A further complication: up to $t - 1$ issuers may be corrupted.

# Threshold Unforgeability Hierarchy

[BTZ22] address this question by proposing a **hierarchy** of unforgeability notions for threshold signature schemes.

These notions grant the issuers increasingly strong abilities to control how their partial signatures are aggregated (e.g. the ability to restrict to a certain set of co-signers).

They show that the definition used by most prior works is weaker than expected in the case of $< t - 1$ corruptions.

# Blind Signatures

A **blind signature scheme** is a protocol for 1 user and 1 issuer. It consists of algorithms $(\mathsf{KeyGen}, \mathsf{USign}_0, \mathsf{ISign}, \mathsf{USign}_1, \mathsf{Verify})$:

- $\mathsf{KeyGen} \rightarrow (pk, sk)$
- $\mathsf{USign}_0(pk, m) \rightarrow (st_U, pm_U)$
- $\mathsf{ISign}(sk, pm_U) \rightarrow pm_I$
- $\mathsf{USign}_1(st_U, pm_I) \rightarrow \sigma$
- $\mathsf{Verify}(pk, m, \sigma) \rightarrow 0/1$

# Blind Signatures

A **blind signature scheme** is a protocol for 1 user and 1 issuer. It consists of algorithms $(\mathsf{KeyGen}, \mathsf{USign}_0, \mathsf{ISign}, \mathsf{USign}_1, \mathsf{Verify})$:

- $\mathsf{KeyGen} \to (pk, sk)$
- $\mathsf{USign}_0(pk, m) \to (st_U, pm_U)$
- $\mathsf{ISign}(sk, pm_U) \to pm_I$
- $\mathsf{USign}_1(st_U, pm_I) \to \sigma$
- $\mathsf{Verify}(pk, m, \sigma) \to 0/1$

**One-more unforgeability** is the security property.

Again the attacker acts as a dishonest user who queries the issuer arbitrarily. The issuer doesn't see $m$, though, so no $(m, \sigma)$ pairs can be discounted as trivial. Instead the security game keeps a query counter $allow$. The attacker wins if they output $allow + 1$ pairs $(m, \sigma)$.

# Blind Signatures

A **blind signature scheme** is a protocol for 1 user and 1 issuer. It consists of algorithms $(\mathsf{KeyGen}, \mathsf{USign}_0, \mathsf{ISign}, \mathsf{USign}_1, \mathsf{Verify})$:

- $\mathsf{KeyGen} \rightarrow (pk, sk)$
- $\mathsf{USign}_0(pk, m) \rightarrow (st_U, pm_U)$
- $\mathsf{ISign}(sk, pm_U) \rightarrow pm_I$
- $\mathsf{USign}_1(st_U, pm_I) \rightarrow \sigma$
- $\mathsf{Verify}(pk, m, \sigma) \rightarrow 0/1$

We also have a **blindness** property that ensures a dishonest issuer cannot link $(m, \sigma)$ to its signing session.

The definition of blindness is standard and not impacted by thresholdizing the issuer, so we do not focus on it.

# Threshold Blind Signatures

A **threshold blind signature scheme** is a protocol for 1 user and $n$ issuers, with signing threshold $t$. It consists of algorithms $(\mathsf{KeyGen}, \mathsf{USign}_0, \mathsf{ISign}, \mathsf{USign}_1, \mathsf{Verify})$:

- $\mathsf{KeyGen} \rightarrow (pk, \{sk_1, sk_2, \ldots, sk_n\})$
- $\mathsf{USign}_0(pk, m) \rightarrow (st_U, pm_U)$
- $\mathsf{ISign}(i, sk_i, pm_U) \rightarrow pm_i$
- $\mathsf{USign}_1(st_U, \mathcal{S}, \{pm_i\}_{i \in \mathcal{S}}) \rightarrow \sigma \qquad // \ |\mathcal{S}| = t$
- $\mathsf{Verify}(pk, m, \sigma) \rightarrow 0/1$

# Threshold Blind Signatures

A **threshold blind signature scheme** is a protocol for 1 user and $n$ issuers, with signing threshold $t$. It consists of algorithms $(\mathsf{KeyGen}, \mathsf{USign}_0, \mathsf{ISign}, \mathsf{USign}_1, \mathsf{Verify})$:

- $\mathsf{KeyGen} \rightarrow (pk, \{sk_1, sk_2, \ldots, sk_n\})$
- $\mathsf{USign}_0(pk, m) \rightarrow (st_U, pm_U)$
- $\mathsf{ISign}(i, sk_i, pm_U) \rightarrow pm_i$
- $\mathsf{USign}_1(st_U, \mathcal{S}, \{pm_i\}_{i \in \mathcal{S}}) \rightarrow \sigma$ $\quad // \; |\mathcal{S}| = t$
- $\mathsf{Verify}(pk, m, \sigma) \rightarrow 0/1$

Until now, there has been no consensus on how to define **threshold one-more unforgeability.**

We aim to clarify the matter by establishing a hierarchy similar to [BTZ22].

# One-More Unforgeability

$\mathsf{Exp}^{\mathsf{OMUF\text{-}x}}_{\mathsf{TB},\mathcal{A},n,t}(\lambda)$

$S := \emptyset;$

$\mathcal{C} \leftarrow \mathcal{A}(n,t) \quad \textbf{if } |\mathcal{C}| \geq t \;:\; \textbf{return } \mathsf{false}$

$(pk, \{sk_i\}_{i\in[n]}) \leftarrow \mathsf{KeyGen}(n,t)$

$(\ell, \{(m_k^*, \sigma_k^*)\}_{k\in[\ell]}) \leftarrow \mathcal{A}^{\mathcal{O}^{\mathsf{ISign}}}(pk, \{sk_i\}_{i\in\mathcal{C}})$

$$\textbf{return } \left(\ell > \mathsf{allow_x}\right) \wedge \forall k \in [\ell] \;:\; \begin{pmatrix} \mathsf{Verify}(pk, \sigma_k^*, m_k^*) \\ \wedge\ \forall j \in [\ell]\setminus\{k\} \;:\; (m_k^*, \sigma_k^*) \neq (m_j^*, \sigma_j^*) \end{pmatrix}$$

$\mathcal{O}^{\mathsf{ISign}}(sid, i, pm_U)$ // *issuer* $i \in [n]$

$\textbf{if } (i, sid) \in S \;:\; \textbf{return } \bot \quad \textbf{else } S := S \cup \{(i, sid)\}$

$pm_i \leftarrow \mathsf{ISign}(i, sk_i, pm_U)$

update $\mathsf{allow_x}$

$\textbf{return } pm_i$

# OMUF-0 and OMUF-1 Security

Issuers $\mathcal{C} \subseteq [n]$ are corrupted. Issuers $\mathcal{H} = [n] \setminus \mathcal{C}$ are uncorrupted.

## OMUF-0

$\mathit{allow}_0$ is the total number of interactions with issuers in $\mathcal{H}$.

# OMUF-0 and OMUF-1 Security

Issuers $\mathcal{C} \subseteq [n]$ are corrupted. Issuers $\mathcal{H} = [n] \setminus \mathcal{C}$ are uncorrupted.

## OMUF-0

$allow_0$ is the total number of interactions with issuers in $\mathcal{H}$.

## OMUF-1

$allow_1$ is the greatest integer such that the interactions with issuers in $\mathcal{H}$ can be divided into $allow_1$ groups where each group contains interactions with $\geq t - |\mathcal{C}|$ different issuers.

If $|\mathcal{C}| = t - 1$, OMUF-0 and OMUF-1 are equivalent.

# OMUF-0 vs. OMUF-1

Suppose $\mathcal{C} = \emptyset$ and $(q_1, q_2, \ldots, q_5)$ are the number of signing sessions of each issuer for a *3-out-of-5* threshold blind signature scheme.

| $(q_1, q_2, \ldots, q_5)$ | $\mathit{allow}_0$ | $\mathit{allow}_1$ |
|---|---|---|
| $(2, 2, 2, 2, 1)$ | 9 | 3 |
| $(4, 2, 1, 1, 1)$ | 9 | 2 |

# OMUF-0 vs. OMUF-1

Suppose $\mathcal{C} = \emptyset$ and $(q_1, q_2, \ldots, q_5)$ are the number of signing sessions of each issuer for a *3-out-of-5* threshold blind signature scheme.

| $(q_1, q_2, \ldots, q_5)$ | $allow_0$ | $allow_1$ |
|---|---|---|
| $(2, 2, 2, 2, 1)$ | 9 | 3 |
| $(4, 2, 1, 1, 1)$ | 9 | 2 |

Some prior works have only analyzed threshold blind signature security in the case $|\mathcal{C}| = t - 1$, but we show that OMUF-0 does not imply OMUF-1 in general.

# OMUF-1 Secure tBlindBLS-1

Public key: $pk := g^{\kappa}$ BLS signature: $\sigma = H(msg)^{\kappa}$

| *User* | | *Issuer* $i$ |
|---|---|---|
| | | Share $k_i$ of $\kappa$ |
| $r \leftarrow_{\$} \mathbb{Z}_q,\ a := H(msg)^r$ | $\xrightarrow{\quad a \quad}$ | |
| | | $b_i := a^{k_i}$ |
| $\sigma := \left(\prod_{i \in \mathcal{S}} b_i^{\Lambda_i^{\mathcal{S}}}\right)^{1/r}$ | $\xleftarrow{\quad b_i \quad}$ | |

- Was shown to be OMUF-0 secure under OMDH assumption [VZK03].
- We show OMUF-1 security under T-BOMDH assumption

# OMUF-2 Security

- Notion of signing session with ssid:

$$\mathsf{ISign}(i, sk_i, pm_U) \rightarrow (pm_i, \mathsf{ssid})$$

# OMUF-2 Security

- Notion of signing session with ssid:

$$\mathsf{ISign}(i, sk_i, pm_U) \rightarrow (pm_i, \text{ssid})$$

**OMUF-2**

$allow_2$ is the number of ssid "postfixes" that have been outputted by a signing set of $\geq t - |\mathcal{C}|$ different issuers.

# OMUF-2 Security

- Notion of signing session with ssid:

$$\mathsf{ISign}(i, sk_i, pm_U) \rightarrow (pm_i, \mathsf{ssid})$$

**OMUF-2**

*allow*$_2$ is the number of ssid "postfixes" that have been outputted by a signing set of $\geq t - |\mathcal{C}|$ different issuers.

- Each partial signature is bound to a signing session.

# OMUF-2 Insecurity of tBlindBLS-1

| *User* | | *Issuer $i$* |
|---|---|---|
| | | Share $k_i$ of $\kappa$ |
| $r \leftarrow_\$ \mathbb{Z}_q,\ a := H(msg)^r$ | $\xrightarrow{a}$ | |
| | | $b_i := a^{k_i}$ |
| $\sigma := \left(\prod_{i \in \mathcal{S}} b_i^{\Lambda_i^{\mathcal{S}}}\right)^{1/r}$ | $\xleftarrow{b_i}$ | $\boxed{\textbf{return}\ \text{ssid} = a}$ |

- Attack against OMUF-2 security:

# OMUF-2 Insecurity of tBlindBLS-1

| *User* | | *Issuer $i$* |
|---|---|---|
| | | Share $k_i$ of $\kappa$ |
| $r \leftarrow_\$ \mathbb{Z}_q,\ a := H(msg)^r$ | $\xrightarrow{a}$ | |
| | | $b_i := a^{k_i}$ |
| | $\xleftarrow{b_i}$ | |
| $\sigma := \left(\prod_{i \in \mathcal{S}} b_i^{\Lambda_i^{\mathcal{S}}}\right)^{1/r}$ | | **return** ssid $= a$ |

- Attack against OMUF-2 security:
  Choose $a_1 := H(msg)^{r_1}$ and $a_2 := H(msg)^{r_2}$.

# OMUF-2 Insecurity of tBlindBLS-1

| *User* | | *Issuer $i$* |
|---|---|---|
| | | Share $k_i$ of $\kappa$ |
| $r \leftarrow_\$ \mathbb{Z}_q, \; a := H(msg)^r$ | $\xrightarrow{\quad a \quad}$ | |
| | | $b_i := a^{k_i}$ |
| | $\xleftarrow{\quad b_i \quad}$ | |
| $\sigma := \left( \prod_{i \in \mathcal{S}} b_i^{\Lambda_i^{\mathcal{S}}} \right)^{1/r}$ | | $\boxed{\textbf{return } \mathsf{ssid} = a}$ |

- Attack against OMUF-2 security:
  Choose $a_1 := H(msg)^{r_1}$ and $a_2 := H(msg)^{r_2}$.
  Get partial signatures $b_1 := a_1^{k_1}$ and $b_2 := a_2^{k_2}$.

# OMUF-2 Insecurity of tBlindBLS-1

| *User* | | *Issuer* $i$ |
|---|---|---|
| | | Share $k_i$ of $\kappa$ |
| $r \leftarrow_\$ \mathbb{Z}_q, \ a := H(msg)^r$ | $\xrightarrow{a}$ | |
| | | $b_i := a^{k_i}$ |
| | $\xleftarrow{b_i}$ | |
| $\sigma := \left(\prod_{i \in \mathcal{S}} b_i^{\Lambda_i^{\mathcal{S}}}\right)^{1/r}$ | | $\boxed{\textbf{return } \text{ssid} = a}$ |

- Attack against OMUF-2 security:
  Choose $a_1 := H(msg)^{r_1}$ and $a_2 := H(msg)^{r_2}$.
  Get partial signatures $b_1 := a_1^{k_1}$ and $b_2 := a_2^{k_2}$.
  Combine them: $\sigma := (b_1^{\Lambda_1^{\mathcal{S}}})^{1/r_1} \cdot (b_2^{\Lambda_2^{\mathcal{S}}})^{1/r_2}$

# OMUF-2 Insecurity of tBlindBLS-1

| *User* | | *Issuer* $i$ |
|---|---|---|
| | | Share $k_i$ of $\kappa$ |
| $r \leftarrow_\$ \mathbb{Z}_q, \ a := H(msg)^r$ | $\xrightarrow{\quad a \quad}$ | |
| | | $b_i := a^{k_i}$ |
| $\sigma := \left(\prod_{i \in \mathcal{S}} b_i^{\Lambda_i^{\mathcal{S}}}\right)^{1/r}$ | $\xleftarrow{\quad b_i \quad}$ | **return** ssid $= a$ |

- Attack against OMUF-2 security:
  Choose $a_1 := H(msg)^{r_1}$ and $a_2 := H(msg)^{r_2}$.
  Get partial signatures $b_1 := a_1^{k_1}$ and $b_2 := a_2^{k_2}$.
  Combine them: $\sigma := (b_1^{\Lambda_1^{\mathcal{S}}})^{1/r_1} \cdot (b_2^{\Lambda_2^{\mathcal{S}}})^{1/r_2}$
- Problem: the signature share $b_i$ is not bound to the signing session – determined by $a$.

# OMUF-2 Secure tBlindBLS-2

| *User* | | *Issuer* $i$ |
|---|---|---|
| | | Share $k_i$ of $\kappa$ |
| $r \leftarrow_\$ \mathbb{Z}_q, \ a := H(msg)^r$ | $\xrightarrow{\quad a \quad}$ | |
| | | $b_i := a^{k_i}$ |
| $\sigma := \left(\prod_{i \in \mathcal{S}} b_i^{\Lambda_i^{\mathcal{S}}}\right)^{1/r}$ | $\xleftarrow{\quad b_i \quad}$ | **return** ssid $= a$ |

- Problem: The signature share $b_i$ is not bound to the signing session – determined by $a$.

# OMUF-2 Secure tBlindBLS-2

| *User* | | *Issuer* $i$ |
|---|---|---|
| | | Share $k_i$ of $\kappa$, share $z_i$ of 0 |
| $r \leftarrow_\$ \mathbb{Z}_q,\ a := H(msg)^r$ | $\xrightarrow{a}$ | |
| | | $b_i := a^{k_i} \cdot \tilde{H}(a)^{z_i}$ |
| $\sigma := \left(\prod_{i \in \mathcal{S}} b_i^{\Lambda_i^{\mathcal{S}}}\right)^{1/r}$ | $\xleftarrow{b_i}$ | **return** ssid $= a$ |

- Problem: The signature share $b_i$ is not bound to the signing session – determined by $a$.
- Solution: Bind signature shares together with a session-specific blinding.

# OMUF-2 Secure tBlindBLS-2

| *User* | | *Issuer i* |
|---|---|---|
| | | Share $k_i$ of $\kappa$, share $z_i$ of 0 |
| $r \leftarrow_{\$} \mathbb{Z}_q,\ a := H(msg)^r$ | $\xrightarrow{a}$ | |
| | | $b_i := a^{k_i} \cdot \tilde{H}(a)^{z_i}$ |
| | $\xleftarrow{b_i}$ | |
| $\sigma := \left(\prod_{i \in \mathcal{S}} b_i^{\Lambda_i^{\mathcal{S}}}\right)^{1/r}$ | | **return** ssid $= a$ |

- Problem: The signature share $b_i$ is not bound to the signing session – determined by $a$.
- Solution: Bind signature shares together with a session-specific blinding.
- Blindings only cancel each other when $t$ shares for the same ssid $= a$ are combined. They *look like random* otherwise (under DDH).

# OMUF-2 Secure tBlindBLS-2

| *User* | | *Issuer* $i$ |
|---|---|---|
| | | Share $k_i$ of $\kappa$, share $z_i$ of 0 |
| $r \leftarrow_\$ \mathbb{Z}_q,\; a := H(msg)^r$ | $\xrightarrow{a}$ | |
| | | $b_i := a^{k_i} \cdot \tilde{H}(a)^{z_i}$ |
| | $\xleftarrow{b_i}$ | |
| $\sigma := \left(\prod_{i \in \mathcal{S}} b_i^{\Lambda_i^{\mathcal{S}}}\right)^{1/r}$ | | **return** ssid $= a$ |

- Problem: The signature share $b_i$ is not bound to the signing session – determined by $a$.
- Solution: Bind signature shares together with a session-specific blinding.
- Blindings only cancel each other when $t$ shares for the same ssid $= a$ are combined. They *look like random* otherwise (under DDH).
- OMUF-2 secure under BOMDH and DDH assumptions.

# OMUF-3 Security

- An issuer chooses the intended set of co-issuers with $\mathcal{S}$ for the session ssid:

$$\mathsf{ISign}(i, sk_i, pm_U) \rightarrow (pm_i, \mathsf{ssid}, \mathcal{S})$$

# OMUF-3 Security

- An issuer chooses the intended set of co-issuers with $\mathcal{S}$ for the session ssid:

$$\mathsf{ISign}(i, sk_i, pm_U) \rightarrow (pm_i, \boxed{\mathsf{ssid}}, \boxed{\mathcal{S}})$$

**OMUF-3**

$allow_3$ is the number of ssid "postfixes" that have been outputted by a signing set of $\geq t - |\mathcal{C}|$ different issuers, each of whom outputted a supposed signing set $\mathcal{S}$ s.t. $\mathcal{S} \cap \mathcal{H}$ is a subset of the actual signing set.

# OMUF-3 Security

- An issuer chooses the intended set of co-issuers with $\mathcal{S}$ for the session ssid:

$$\mathsf{ISign}(i, sk_i, pm_U) \rightarrow (pm_i, \boxed{\mathsf{ssid}}, \boxed{\mathcal{S}})$$

**OMUF-3**

*allow*$_3$ is the number of ssid "postfixes" that have been outputted by a signing set of $\geq t - |\mathcal{C}|$ different issuers, each of whom outputted a supposed signing set $\mathcal{S}$ s.t. $\mathcal{S} \cap \mathcal{H}$ is a subset of the actual signing set.

- Each partial signature is bound to a signing session *and* the issuer is aware of its co-issuers.

# OMUF-3 Secure tBlindBLS-3

| *User* | | *Issuer $i$* |
|---|---|---|
| | | Share $k_i$ of $\kappa$ |
| $r \leftarrow_\$ \mathbb{Z}_q, a := H(msg)^r$ | | Pairwise PRF key $w_{i,j}$ for $j \in [n] \setminus \{i\}$ |
| | $\xrightarrow{a, \mathcal{S}}$ | |
| | | $b_i := (a^{k_i})^{\Lambda_i^{\mathcal{S}}} \cdot \prod_{j \in \mathcal{S}, j<i} F_{w_{i,j}}(a, \mathcal{S}) \cdot \prod_{j \in \mathcal{S}, j>i} F_{w_{i,j}}(a, \mathcal{S})^{-1}$ |
| | $\xleftarrow{b_i}$ | |
| $\sigma := \left(\prod_{i \in \mathcal{S}} b_i\right)^{1/r}$ | | **return** (ssid $= a, \mathcal{S}$) |

- Partial signatures are bound to a session *and issuer set*-specific blinding.

# OMUF-3 Secure tBlindBLS-3

| *User* | | *Issuer $i$* |
|---|---|---|
| | | Share $k_i$ of $\kappa$ |
| $r \leftarrow_\$ \mathbb{Z}_q, a := H(msg)^r$ | | Pairwise PRF key $w_{i,j}$ for $j \in [n] \setminus \{i\}$ |
| | $\xrightarrow{a, \mathcal{S}}$ | |
| | | $b_i := (a^{k_i})^{\Lambda_i^{\mathcal{S}}} \cdot \prod_{j \in \mathcal{S}, j<i} F_{w_{i,j}}(a, \mathcal{S}) \cdot \prod_{j \in \mathcal{S}, j>i} F_{w_{i,j}}(a, \mathcal{S})^{-1}$ |
| | $\xleftarrow{b_i}$ | |
| $\sigma := \left(\prod_{i \in \mathcal{S}} b_i\right)^{1/r}$ | | **return** (ssid $= a, \mathcal{S}$) |

- Partial signatures are bound to a session *and issuer set*-specific blinding.
- Blindings only cancel each other when $t$ shares for the same ssid $= a$ are combined. They *look like random* otherwise (if $F$ is a secure PRF).

# OMUF-3 Secure tBlindBLS-3

| *User* | | *Issuer $i$* |
|---|---|---|
| | | Share $k_i$ of $\kappa$ |
| $r \leftarrow_\$ \mathbb{Z}_q, a := H(msg)^r$ | $\xrightarrow{a,\ \mathcal{S}}$ | Pairwise PRF key $w_{i,j}$ for $j \in [n] \setminus \{i\}$ |
| | | $b_i := (a^{k_i})^{\Lambda_i^{\mathcal{S}}} \cdot \prod_{j \in \mathcal{S}, j<i} F_{w_{i,j}}(a, \mathcal{S}) \cdot \prod_{j \in \mathcal{S}, j>i} F_{w_{i,j}}(a, \mathcal{S})^{-1}$ |
| $\sigma := \left(\prod_{i \in \mathcal{S}} b_i\right)^{1/r}$ | $\xleftarrow{b_i}$ | **return** (ssid $= a, \mathcal{S}$) |

- Partial signatures are bound to a session *and issuer set*-specific blinding.
- Blindings only cancel each other when $t$ shares for the same ssid $= a$ are combined. They *look like random* otherwise (if $F$ is a secure PRF).
- OMUF-3 secure under BOMDH assumption and pseudorandomness of $F$.

# Our Hierarchy and OMUF-SB

OMUF-0: A single partial signature on a message can give a valid signature.

---

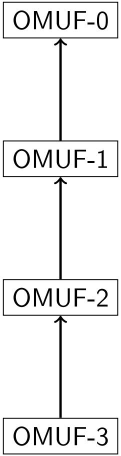
OMUF-1: $t$ partial signatures can give a valid signature.

---

OMUF-2: $t$ partial signatures from the same signing session gives a valid signature.

---

OMUF-3: $t$ partial signatures from the same signing session gives a valid signature and each issuer knows their co-issuers.



# Our Hierarchy and OMUF-SB

OMUF-0: A single partial signature on a message can give a valid signature.

---

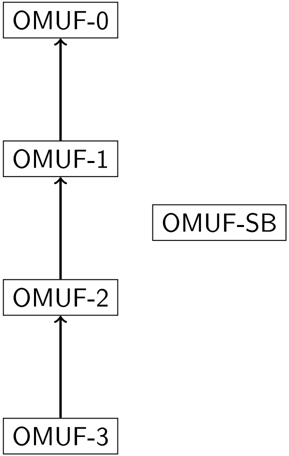
OMUF-1: $t$ partial signatures can give a valid signature.

---

OMUF-2: $t$ partial signatures from the same signing session gives a valid signature.

---

OMUF-3: $t$ partial signatures from the same signing session gives a valid signature and each issuer knows their co-issuers.




**OMUF-SB**

[CKM+23] proposed OMUF-SB while showing the security of the Snowblind scheme.

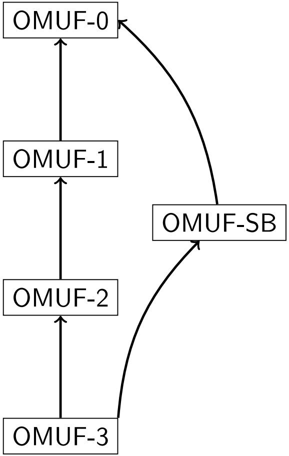
# Our Hierarchy and OMUF-SB

OMUF-0: A single partial signature on a message can give a valid signature.

OMUF-1: $t$ partial signatures can give a valid signature.

OMUF-2: $t$ partial signatures from the same signing session gives a valid signature.

OMUF-3: $t$ partial signatures from the same signing session gives a valid signature and each issuer knows their co-issuers.




**OMUF-SB**

[CKM+23] proposed OMUF-SB while showing the security of the Snowblind scheme.

- Incomparable to OMUF-1,2.

# Our Hierarchy and OMUF-SB

OMUF-0: A single partial signature on a message can give a valid signature.

---

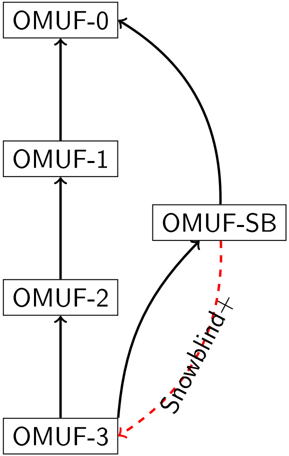
OMUF-1: $t$ partial signatures can give a valid signature.

---

OMUF-2: $t$ partial signatures from the same signing session gives a valid signature.

---

OMUF-3: $t$ partial signatures from the same signing session gives a valid signature and each issuer knows their co-issuers.




**OMUF-SB**

[CKM+23] proposed OMUF-SB while showing the security of the Snowblind scheme.

- Incomparable to OMUF-1,2.
- OMUF-3 secure Snowblind+ scheme by relying on PRFs and Snowblind.

# Open Problems

- Our transformation from OMUF-SB gives a 4-round OMUF-3 secure signature scheme. What about more efficient pairing-free constructions targeting OMUF-1/2/3?

# Open Problems

- Our transformation from OMUF-SB gives a 4-round OMUF-3 secure signature scheme. What about more efficient pairing-free constructions targeting OMUF-1/2/3?
- What about the security of the original Snowblind scheme in our hierarchy?

# Open Problems

- Our transformation from OMUF-SB gives a 4-round OMUF-3 secure signature scheme. What about more efficient pairing-free constructions targeting OMUF-1/2/3?
- What about the security of the original Snowblind scheme in our hierarchy?
- What about the security against adaptive corruptions?

# Open Problems

- Our transformation from OMUF-SB gives a 4-round OMUF-3 secure signature scheme. What about more efficient pairing-free constructions targeting OMUF-1/2/3?
- What about the security of the original Snowblind scheme in our hierarchy?
- What about the security against adaptive corruptions?

Thanks for listening...

# References I


- Mihir Bellare, Stefano Tessaro, and Chenzhi Zhu, *Stronger security for non-interactive threshold signatures: BLS and FROST*, Cryptology ePrint Archive, Paper 2022/833, 2022.
- Elizabeth Crites, Chelsea Komlo, Mary Maller, Stefano Tessaro, and Chenzhi Zhu, *Snowblind: A threshold blind signature in pairing-free groups*, Advances in Cryptology – CRYPTO 2023, 2023.
- Duc Liem Vo, Fangguo Zhang, and Kwangjo Kim, *A new threshold blind signature scheme from pairings*, SCIS2003, SCIS, 2003, pp. 233–238.