Analysis of "Count Me In! Extendability for Threshold Ring Signatures"

Dimitrios Vassiliou Stylianos Zarifis

National Technical University of Athens

el19830@mail.ntua.com

el20435@mail.ntua.com

Computational Cryptography - Semester Project

March 22, 2024

Presentation Overview

Ring Signatures

Definition

Ring signatures are a type of digital signature that can be performed by any member of a group of users that each have keys.

- A message is signed by a member on behalf of the group.
- Computationally infeasible to determine which member signed the message.

Ring Signature Scheme Algorithms

- **Setup(1^λ) $\rightarrow$ pp:** Generates public parameters.
- **KeyGen() $\rightarrow$ (pk, sk):** Creates a key pair (pk, sk).
- **Sign(μ, $\{\mathbf{pk}_j\}_{j\in\mathcal{R}}$, $\mathbf{sk}_i$) $\rightarrow \sigma$:** Produces a signature.
- **Verify(μ, $\{\mathbf{pk}_i\}_{i\in\mathcal{R}}$, σ) $\rightarrow$ `accept/reject`:** Validates the signature.

Threshold Ring Signatures

Definition

Threshold ring signatures extend ring signatures by allowing any t signers to anonymize themselves among a ring of signers R where $t \leq |R|$. A verifier can then check that at least t signers in the ring R signed the same message.

Threshold Ring Signature Scheme Algorithms

- **Combisign**$(\{\sigma_i\}_{i \in \mathcal{S}}) \rightarrow \sigma$**:** Combines partial signatures.
- **Join**$(\mu,\ \{\mathbf{pk}_j\}_{j \in \mathcal{R}},\ \mathbf{sk},\ \sigma) \rightarrow \sigma'$**:** Introduces a new signer.

Signatures of Knowledge

Definition

Signatures of Knowledge (SoKs) are a form of digital signatures. Instead of using a public key, they use a statement from a specific language related to computational complexity. One may not create a new signature without knowing a certain piece of information, the witness, related to the statement.

- Statement ϕ: Problem to prove knowledge about.
- Witness w: Secret piece of information that proves ϕ is true.

Example: Discrete Logarithm Problem

Statement ϕ: "There exists an x such that $a = g^x$ in a given group G". Witness w: Is x, the discrete logarithm of a to base g. Knowing x, one can prove they know the discrete logarithm of a to base g without revealing x (e.g. $\Sigma -$ Protocols).

Extendable Ring Signatures

Definition

Extendable Ring Signatures (ERS) expand traditional ring signatures by adding more members to the group after signing, while preserving anonymity.

- **Ladders of rings**: Tuples $\mathsf{lad} = \left(i, \mathcal{R}^{(1)}, \mathcal{R}^{(2)}, \ldots, \mathcal{R}^{(l)}\right)$, where i is the signer identity, and $\mathcal{R}^{(i)}$ are sets of signer identifiers.

Extendable Ring Signature Scheme Algorithms

- **Extend** $(\mu, \{\mathsf{pk}_i\}_{i\in\mathcal{R}}, \sigma, \{\mathsf{pk}_j\}_{j\in\mathcal{R}'}) \to \sigma'$**:** Outputs a modified signature which verifies under $\mathcal{R} \cup \mathcal{R}'$.
- **Process**$(\mu, \mathsf{L}_{\mathsf{keys}}, \mathsf{lad}) \to \sigma$: Generates extendable ring signature by applying the `Extend` algorithm for each identifier's set $\mathcal{R}^{(i)}$.

Unforgeability Game

Unforgeability: Negligible success for adversaries in generating valid signatures.

$\mathsf{Exp}^{\mathrm{cmEUF}}_{\mathcal{A},\mathbf{ERS}}(\lambda)$

$\mathsf{L}_{\mathsf{keys}}, \mathsf{L}_{\mathsf{corr}}, \mathsf{L}_{\mathsf{sign}} \leftarrow \varnothing$

$\mathsf{pp} \leftarrow \mathbf{ERS}.\mathsf{Setup}(1^\lambda)$

$O \leftarrow \{O\mathsf{Sign}, O\mathsf{KeyGen}, O\mathsf{Corrupt}\}$

$(\mu^*, \mathcal{R}^*, \sigma^*) \leftarrow \mathcal{A}^O(\mathsf{pp})$

// rule out trivial wins due to ring expansion

if $\exists\ (\mu^*, \mathcal{R}, \cdot) \in \mathsf{L}_{\mathsf{sign}}\ s.t.$
$\{\mathsf{pk}_j\}_{j\in\mathcal{R}} \subseteq \{\mathsf{pk}_j\}_{j\in\mathcal{R}^*}$

  return `lose`

// rule out trivial wins due to key duplication

if $\{\mathsf{pk}_j\}_{j\in\mathcal{R}^*} \cap \{\mathsf{pk}_j\}_{j\in\mathsf{L}_{\mathsf{corr}}} \neq \varnothing$

  return `lose`

if $\mathsf{Verify}(\mu^*, \{\mathsf{pk}_j\}_{j\in\mathcal{R}^*}, \sigma^*) = \mathtt{reject}$

  return `lose`

return `win`

O*Sign* Oracle

Signs messages with a signer's secret key if the signer is valid, uncorrupted, and part of the ring.

O*KeyGen* Oracle

Generates key pairs for signers. If given a public key, corrupts the signer, associating their key with the provided public key.

O*Corrupt* Oracle

Corrupts a signer, revealing their keys.

Anonymity & Anonymous Extendability Games

Anonymity: Probability of adversary correctly identifying the ladder used in generating a challenge signature close to random guessing.

Anonymous Extendability: Expands the Anonymity property to apply on extended ring signatures.

$\mathsf{Exp}^{\mathrm{ANEXT}}_{\mathcal{A},\mathbf{ERS}}(\lambda)$

1 : $b \leftarrow_R \{0,1\}$
2 : $\mathsf{L}_{\mathsf{keys}}, \mathsf{L}_{\mathsf{corr}}, \mathsf{L}_{\mathsf{sign}} \leftarrow \varnothing$
3 : $\mathsf{pp} \leftarrow \mathbf{ERS}.\mathsf{Setup}(1^\lambda)$
// handle of oracles, for compact notation
4 : $O \leftarrow \{O\mathsf{Sign}, O\mathsf{KeyGen}, O\mathsf{Corrupt}\}$
5 : $(\mu^*, \mathtt{lad}_0^*, \mathtt{lad}_1^*) \leftarrow \mathcal{A}^O(\mathsf{pp})$
6 : $\bar{\sigma} \leftarrow \mathsf{Chal}_b(\mu^*, \mathtt{lad}_0^*, \mathtt{lad}_1^*)$
7 : $b^* \leftarrow \mathcal{A}^O(\bar{\sigma})$
// make sure $\mathcal{A}$ did not corrupt the challenge
// keys during the second query phase
8 : **if** $i_0 \in \mathsf{L}_{\mathsf{corr}} \vee i_1 \in \mathsf{L}_{\mathsf{corr}}$
9 : **return** lose
10 : **if** $b^* \neq b$
11 : **return** lose
12 : **return** win

$\mathsf{Chal}_b(\mu^*, \mathtt{lad}_0^*, \mathtt{lad}_1^*)$

1 : **parse** $\mathtt{lad}_0^* = (i_0, \mathcal{R}_0^{(1)}, \ldots, \mathcal{R}_0^{(l_0)})$
2 : **parse** $\mathtt{lad}_1^* = (i_1, \mathcal{R}_1^{(1)}, \ldots, \mathcal{R}_1^{(l_1)})$
// challenge signing keys should not be corrupted
3 : **if** $i_0, i_1 \in \mathsf{L}_{\mathsf{corr}}$ **return** $\perp$
// sign and extend following the instructions
// in both ladders
4 : $\sigma_0 \leftarrow \mathsf{Process}(\mu, \mathsf{L}_{\mathsf{keys}}, \mathtt{lad}_0^*)$
5 : $\sigma_1 \leftarrow \mathsf{Process}(\mu, \mathsf{L}_{\mathsf{keys}}, \mathtt{lad}_1^*)$
6 : **if** $\sigma_0 = \perp$ **or** $\sigma_1 = \perp$ **return** $\perp$
// check that ladders end with the same ring
7 : **if** $\mathcal{R}_0^{(1)} \cup \cdots \cup \mathcal{R}_0^{(l_0)} \neq \mathcal{R}_1^{(1)} \cup \cdots \cup \mathcal{R}_1^{(l_1)}$
8 : **return** $\perp$
// set the challenge signature according to b
9 : $\bar{\sigma} \leftarrow \sigma_b$
10 : **return** $\bar{\sigma}$

Same-Message Linkable Extendable Ring Signatures

Definition

A Same-Message Linkable Ring Signature Scheme (SMLRS) is a type of ring signature scheme that allows anyone to determine if two signatures on the same message were created by the same signer.

Definition

Same-Message Linkable Extendable Ring Signatures (SMLERS) expand SMLRS by allowing the addition of new members to a group signature.

SMLERS Algorithms

- **Link**$(\mu, (\sigma_0, \{pk_j\}_{j\in\mathcal{R}_0}), (\sigma_1, \{pk_j\}_{j\in\mathcal{R}_1})) \rightarrow$ **{linked, unlinked}:** Outputs linked if σ_0 and σ_1 were produced by the same signer, otherwise unlinked.

Same Message One-More Linkability Game

Same-Message One-more Linkability: The property of ensuring that an adversary can't create multiple unlinked signatures for the same message, preserving the linkability of the scheme.

$\mathsf{Exp}^{\mathrm{omlink}}_{\mathcal{A},\mathbf{SMLERS}}(\lambda)$

$\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$

$\mathsf{L}_{\mathsf{keys}}, \mathsf{L}_{\mathsf{corr}}, \mathsf{L}_{\mathsf{sign}} \leftarrow \varnothing$

$O \leftarrow \{O\mathsf{Sign}, O\mathsf{KeyGen}, O\mathsf{Corrupt}\}$

$(\mu^*, \{(\sigma^*_k, \mathcal{R}^*_k)\}_{k\in[1\ldots,t]}) \leftarrow \mathcal{A}^O(\mathsf{pp})$

// $\mathcal{A}$ has never seen a signature for the message and a subring of the forgery rings

if $\exists\ (\mu^*, \mathcal{R}, \cdot) \in \mathsf{L}_{\mathsf{sign}}\ s.t.\ \mathcal{R} \subseteq \mathcal{R}^*_k$ for some $k \in [1, \ldots, t]$ **return** `lose`

// $\mathcal{A}$ holds at most $t-1$ secret keys, among the keys identified by the forgery rings

if $|(\mathcal{R}^*_1 \cup \cdots \cup \mathcal{R}^*_t) \cap \mathsf{L}_{\mathsf{corr}}| \geq t$ **return** `lose`

// all the signatures in the forgery verify (for the same message)

if $\exists\ k \in [1 \ldots, t]$ s.t. $\mathsf{Verify}(\mu^*, \{\mathtt{pk}_j\}_{j\in\mathcal{R}^*_k}, \sigma^*_k) = \mathtt{reject}$ **return** `lose`

// all signatures in the forgery are unlinked (here $k, l \in [1, \ldots, t]$)

if $\exists\ k \neq l$ s.t. $\mathsf{Link}(\mu^*, (\sigma^*_k, \{\mathtt{pk}_j\}_{j\in\mathcal{R}^*_k}), (\sigma^*_l, \{\mathtt{pk}_j\}_{j\in\mathcal{R}^*_l})) = \mathtt{linked}$

return `lose`

return `win`

Cross Message Unlinkability Game

Cross Message Unlinkability: No adversary can determine whether two signatures for different messages were produced by the same signer.

$\mathsf{Exp}^{\mathrm{cmunlink}}_{\mathcal{A},\mathbf{LRS}}(\lambda)$

$b \leftarrow_R \{0,1\}, \mathsf{L}_{\mathsf{keys}}, \mathsf{L}_{\mathsf{corr}}, \mathsf{L}_{\mathsf{sign}} \leftarrow \varnothing$
$\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$
$O \leftarrow \{O\mathsf{Sign}, O\mathsf{KeyGen}, O\mathsf{Corrupt}\}$
$(\{\mu_0, \mathcal{R}_0, i_0\}, \{\mu_1, \mathcal{R}_1, i_1\}) \leftarrow \mathcal{A}^O(\mathsf{pp})$
$(\bar{\sigma}_0, \bar{\sigma}_1) \leftarrow \mathsf{Chal}_b(\{\mu_0, \mathcal{R}_0, i_0\}, \{\mu_1, \mathcal{R}_1, i_1\})$
$b^* \leftarrow \mathcal{A}^O(\bar{\sigma}_0, \bar{\sigma}_1)$
// Rule out corruption of challenge identities
if $i_0 \in \mathsf{L}_{\mathsf{corr}} \vee i_1 \in \mathsf{L}_{\mathsf{corr}}$ **return** `lose`
// Rule out trivial attacks using Link
if $\mu_0 = \mu_1$ **return** `lose`
// Rule out trivial attacks using Link
if $(\mu_0, \cdot, i_0) \in \mathsf{L}_{\mathsf{sign}} \vee (\mu_0, \cdot, i_1) \in \mathsf{L}_{\mathsf{sign}}$
$\vee (\mu_1, \cdot, i_0) \in \mathsf{L}_{\mathsf{sign}} \vee (\mu_1, \cdot, i_1) \in \mathsf{L}_{\mathsf{sign}}$
return `lose`
if $b^* \neq b$ **return** `lose`
return `win`

$\mathsf{Chal}_b(\{\mu_0, \mathcal{R}_0, i_0\}, \{\mu_1, \mathcal{R}_1, i_1\})$

// the challenge identities must be uncorrupted
if $i_0 \in \mathsf{L}_{\mathsf{corr}} \vee i_1 \in \mathsf{L}_{\mathsf{corr}}$
return $\perp$
// one identity needs to be in both rings
if $i_0 \notin \mathcal{R}_0 \cap \mathcal{R}_1 \vee i_1 \notin \mathcal{R}_1$
return $\perp$
// signing keys must exist
if $\nexists\ (i_0, \mathtt{pk}_{i_0}, \mathtt{sk}_{i_0}) \in \mathsf{L}_{\mathsf{keys}}$ **return** $\perp$
if $\nexists\ (i_1, \mathtt{pk}_{i_1}, \mathtt{sk}_{i_1}) \in \mathsf{L}_{\mathsf{keys}}$ **return** $\perp$
// generate a signature
$\bar{\sigma}_0 \leftarrow \mathbf{LRS}.\mathsf{Sign}(\mu_0, \{\mathtt{pk}_i\}_{i \in \mathcal{R}_0}, \mathtt{sk}_{i_0})$
// generate the second signature according to
// the experiment's bit b
$\bar{\sigma}_1 \leftarrow \mathbf{LRS}.\mathsf{Sign}(\mu_1, \{\mathtt{pk}_i\}_{i \in \mathcal{R}_1}, \mathtt{sk}_{i_b})$
return $(\bar{\sigma}_0, \bar{\sigma}_1)$

Extendable Threshold Ring Signatures

Definition

Extendable threshold ring signature scheme allows parties to collectively produce a signature on a message μ for a ring $\mathcal{R}$, revealing that at least t out of $\mathcal{R}$ potential signers participated, while maintaining their anonymity.

- **Generalized ladders**: Support sequences of actions (Sign, Combine, Extend) to create valid threshold ring signatures.

Extendable Threshold Ring Signature Algorithms

- **Combine**$(\mu, \sigma_0, \sigma_1, \{pk_i\}_{i\in R}) \to \sigma'$: Combines two signatures for the same ring into one with the number of signers as threshold.
- **Extend**$(\mu, \sigma, \{pk_i\}_{i\in R}, \{pk_i\}_{i\in R'}) \to \sigma'$: Extends σ with threshold t for ring R into a new σ' with same threshold for a larger ring $R \cup R'$.
- **Process**$(\mu, \mathsf{L}_{\mathsf{keys}}, \mathsf{lad}) \to (\sigma^{(l)}, t^{(l)}, \mathcal{R}^{(l)})$: Produces the final signature, threshold and ring by applying the operations of the generalized Ladders.

Unforgeability Game

$\mathsf{Exp}^{\mathrm{cmEUF}}_{\mathcal{A},\mathbf{ETRS}}(\lambda)$

1 : $\mathsf{L}_{\mathsf{keys}}, \mathsf{L}_{\mathsf{corr}}, \mathsf{L}_{\mathsf{sign}} \leftarrow \varnothing$

2 : $\mathsf{pp} \leftarrow \mathsf{Setup}(1^{\lambda})$

3 : $O \leftarrow \{O\mathsf{Sign}, O\mathsf{KeyGen}, O\mathsf{Corrupt}\}$

4 : $(t^*, \mu^*, \mathcal{R}^*, \sigma^*) \leftarrow \mathcal{A}^{O}(\mathsf{pp})$

5 : $q \leftarrow |\{(\mu^*, \mathcal{R}, \cdot) \in \mathsf{L}_{\mathsf{sign}} \; s.t. \; \mathcal{R} \subseteq \mathcal{R}^*)\}|$

// rule out attacks if $\mathcal{A}$ knows too many sk:s or honestly

// generated signatures for μ^*

6 : **if** $|\mathcal{R}^* \cap \mathsf{L}_{\mathsf{corr}}| + q \geq t$ **return** `lose`

// rule out outputs that do not verify

7 : **if** $\mathsf{Verify}(t, \mu^*, \{\mathtt{pk}_j\}_{j \in \mathcal{R}^*}, \sigma^*) = \mathtt{reject}$

return `lose`

8 : **return** `win`

- **Number** q: The count of honestly generated signatures known to the adversary.
- **Set** $\mathbf{L_{corr}}$: The set of secret keys obtained by the adversary through corruption queries.
- **Algorithm**: Checks if the sum of known secret keys and q is equal to or exceeds the threshold, t.
- **Condition**: Prevents targeted attacks by ensuring $\mathcal{A}$ may not create t partial signatures, that could lead to a valid ETRS, via Combine.

Anonymity & Anonymous Extendability Games

Anonymity: Probability of adversary correctly identifying the ladder used in generating a challenge signature close to random guessing.

Anonymous Extendability: Expands the Anonymity property to apply on extended ring signatures.

$\mathsf{Exp}^{\mathrm{ANEXT}}_{\mathcal{A},\mathbf{ETRS}}(\lambda)$

1: $b \leftarrow_R \{0,1\}$
2: $\mathsf{L}_{\mathsf{keys}}, \mathsf{L}_{\mathsf{corr}}, \mathsf{L}_{\mathsf{sign}} \leftarrow \varnothing$
3: $\mathsf{pp} \leftarrow \mathbf{ETRS}.\mathsf{Setup}(1^\lambda)$
4: $O \leftarrow \{O\mathsf{Sign}, O\mathsf{KeyGen}, O\mathsf{Corrupt}\}$
5: $(\mu^*, \mathtt{lad}_0^*, \mathtt{lad}_1^*) \leftarrow \mathcal{A}^O(\mathsf{pp})$
6: $\bar{\sigma} \leftarrow \mathsf{Chal}_b(\mu^*, \mathtt{lad}_0^*, \mathtt{lad}_1^*)$
7: $b^* \leftarrow \mathcal{A}^O(\bar{\sigma})$
8: **if** $\exists i \in \mathtt{lad}_0^*.\mathtt{signers}$ *s.t.* $i \in \mathsf{L}_{\mathsf{corr}}$
9: **return** lose
10: **if** $\exists i \in \mathtt{lad}_1^*.\mathtt{signers}$ *s.t.* $i \in \mathsf{L}_{\mathsf{corr}}$
11: **return** lose
12: **if** $\exists(\mu^*, \cdot, i) \in \mathsf{L}_{\mathsf{sign}}$ for $i \in \mathtt{lad}_0^*.\mathtt{signers}$
13: **return** lose
14: **if** $\exists(\mu^*, \cdot, i) \in \mathsf{L}_{\mathsf{sign}}$ for $i \in \mathtt{lad}_1^*.\mathtt{signers}$
15: **return** lose
16: **if** $b^* \neq b$
17: **return** lose
18: **return** win

$\mathsf{Chal}_b(\mu^*, \mathtt{lad}_0^*, \mathtt{lad}_1^*)$

1: **if** $\mathtt{lad}_0^*$ or $\mathtt{lad}_1^*$ is not well-formed
2: **return** $\perp$
3: **if** $\exists i \in \mathtt{lad}_0^*.\mathtt{signers}$ *s.t.* $i \in \mathsf{L}_{\mathsf{corr}}$
4: **return** $\perp$
5: **if** $\exists i \in \mathtt{lad}_1^*.\mathtt{signers}$ *s.t.* $i \in \mathsf{L}_{\mathsf{corr}}$
6: **return** $\perp$
// make sure the public keys are known / initialized
7: **if** $\exists i \in \mathtt{lad}_0^*.\mathtt{sr}$ *s.t.* $(\mathtt{pk}_i, \cdot) \notin \mathsf{L}_{\mathsf{keys}}$
8: **return** $\perp$
9: **if** $\exists i \in \mathtt{lad}_1^*.\mathtt{sr}$ *s.t.* $(\mathtt{pk}_i, \cdot) \notin \mathsf{L}_{\mathsf{keys}}$
10: **return** $\perp$
11: $(\sigma_0, t_0, \mathcal{R}_0) \leftarrow \mathsf{Process}(\mu^*, \mathsf{L}_{\mathsf{keys}}, \mathtt{lad}_0)$
12: $(\sigma_1, t_1, \mathcal{R}_1) \leftarrow \mathsf{Process}(\mu^*, \mathsf{L}_{\mathsf{keys}}, \mathtt{lad}_1)$
// rule out trivial attacks
13: **if** $\mathcal{R}_0 \neq \mathcal{R}_1$ or $t_0 \neq t_1$
14: **return** $\perp$
15: $\bar{\sigma} \leftarrow \sigma_b$
16: **return** $\bar{\sigma}$

Compiler for ETRS

Building ETRS from SMLERS

An Extendable Threshold Ring Signature (ETRS) can be built from any Same-Message Linkable Extendable Ring Signature Scheme (SMLERS) using a compiler.

Theorem: Security

Assuming that SMLERS is secure, the ETRS scheme satisfies the properties of Correctness, Unforgeability, and Anonymity.

Signature Generation Time

This figure illustrates the time taken for the Sign operation in the ERS, SMLERS and ETRS schemes, for various thresholds.

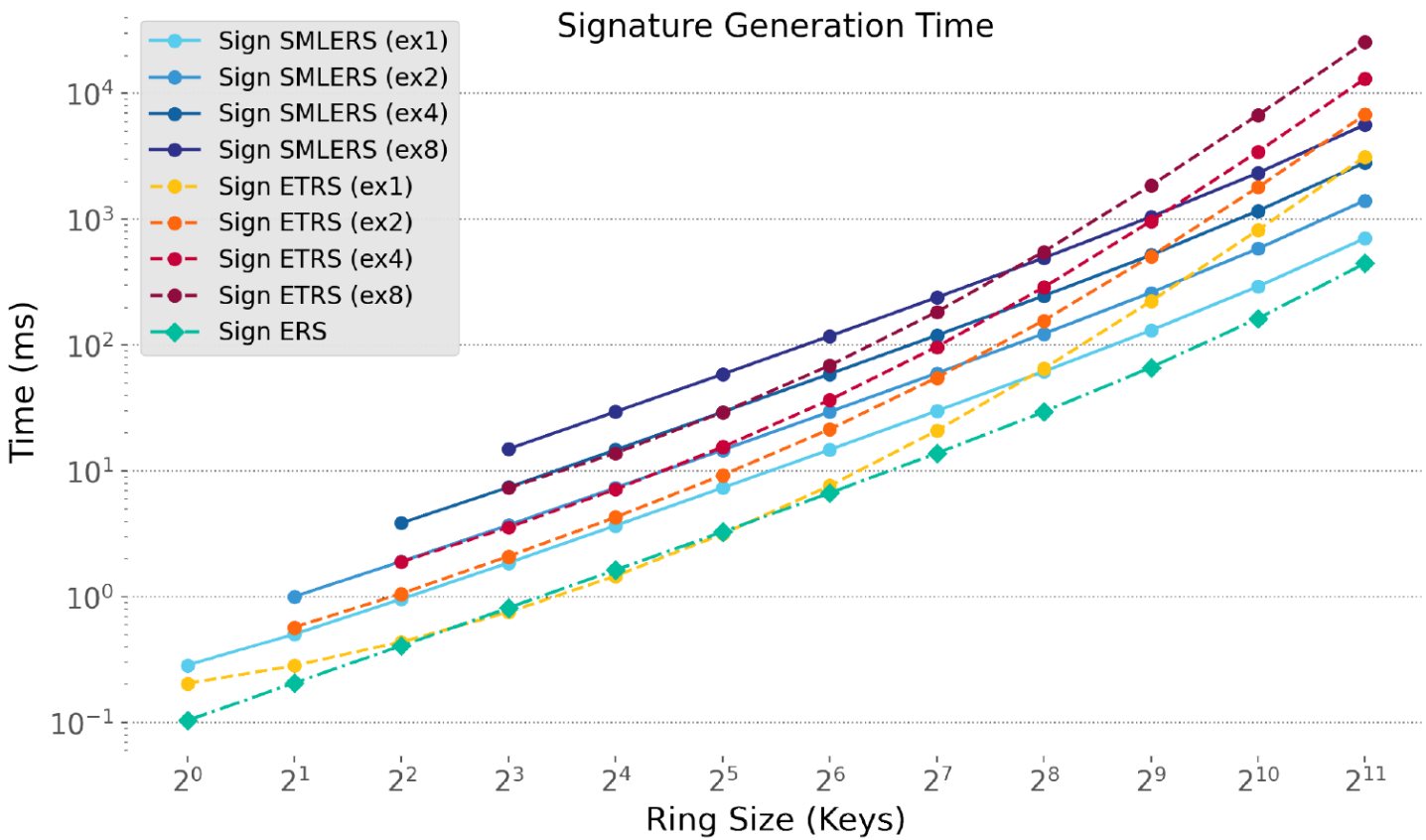

Figure: Clock time for `Sign` in the implemented schemes for different thresholds.

Signature Verification Time

This figure presents the time required for the Verify operation in the ERS, SMLERS, and ETRS schemes across different thresholds.

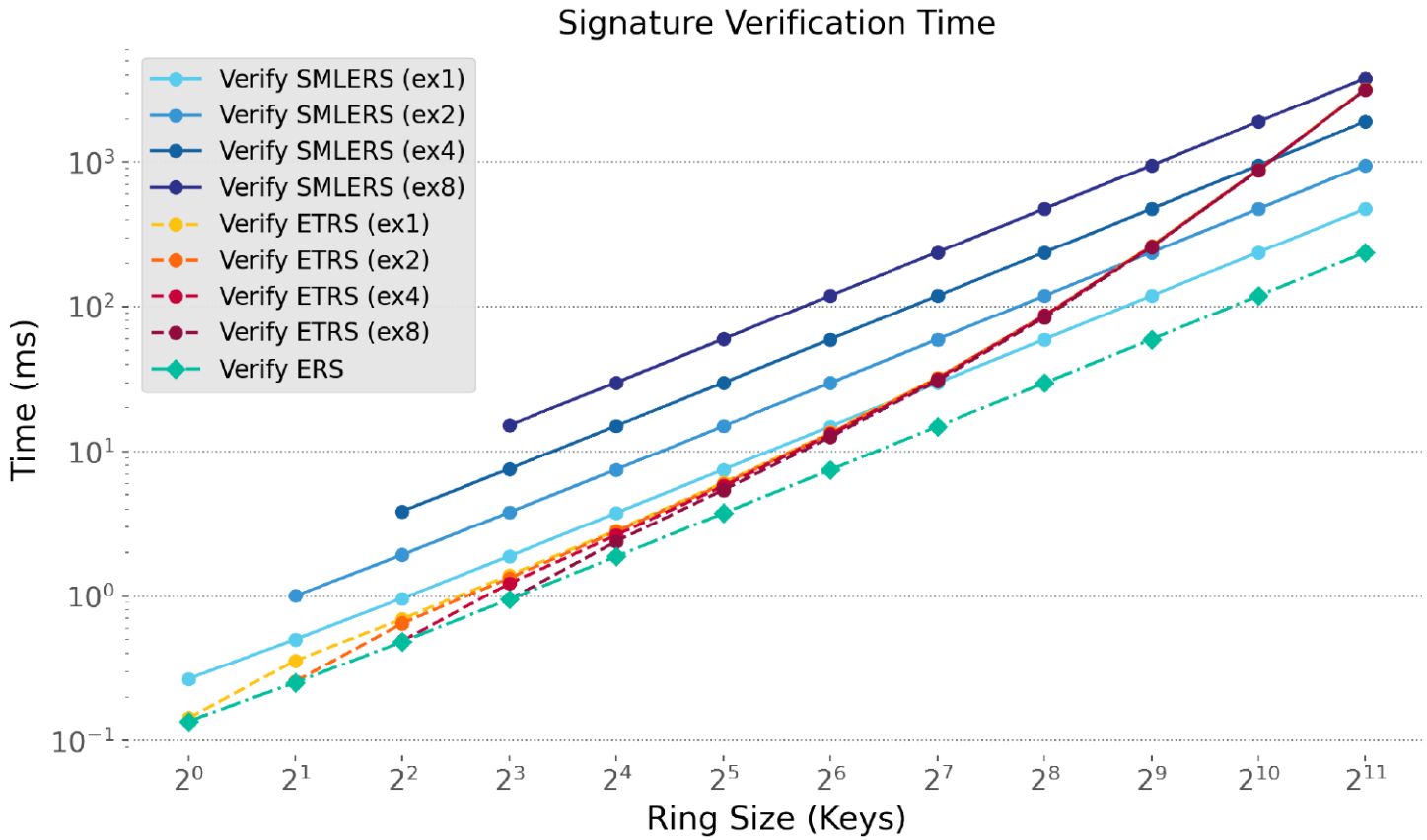

Figure: Clock time for `Verify` in the implemented schemes for different thresholds.

Signature Sizes

This figure showcases the signature sizes for the ERS, SMLERS, and ETRS schemes with varying thresholds.

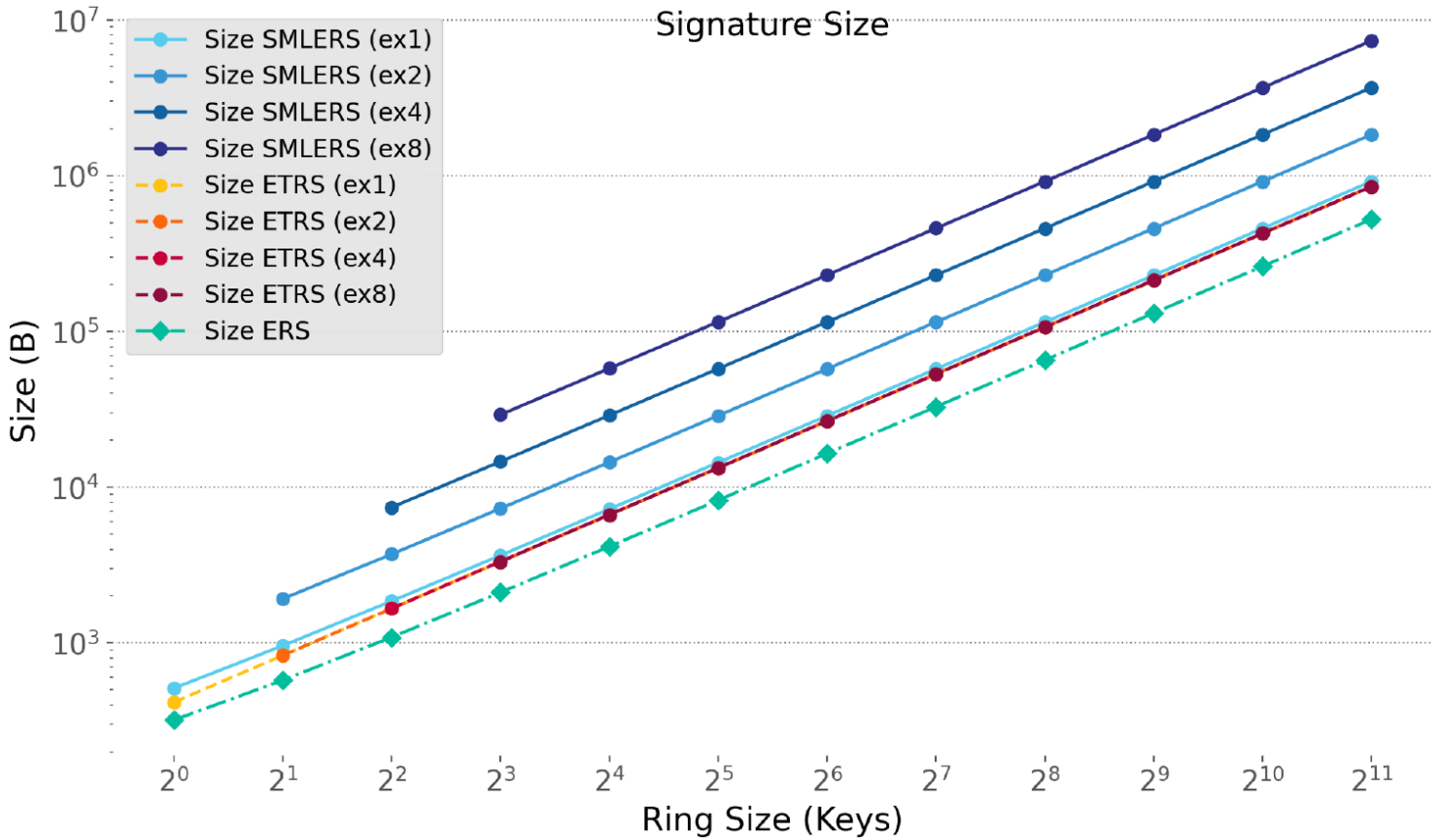

Figure: Signature sizes for all implemented schemes, with varying thresholds.

The End

Questions? Comments?