1-out-of-n Oblivious Signatures: Security Revisited and a Generic Construction with an Efficient Communication Cost

〇Masayuki Tezuka Keisuke Tanaka

Tokyo Institute of Technology

Version 2023/12/06

ICISC 2023 Full Presentation Slide

(1,n)-Oblivious Signatures Scheme

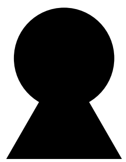
(1,n)-Oblivious Signatures [Chen94]

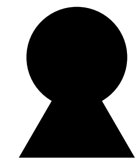
(1,n)-Oblivious Signatures [Chen94]

Make a list of n candidate messages  .

Signer

sk





User

vk

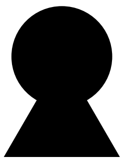
(1,n)-Oblivious Signatures [Chen94]

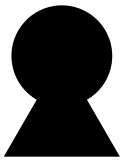
Selected

Make a list of n candidate messages  .

Signer

sk

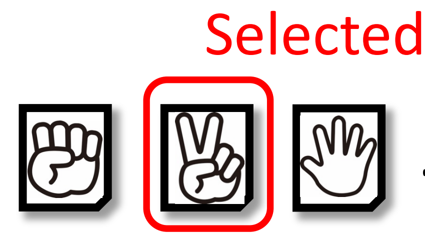


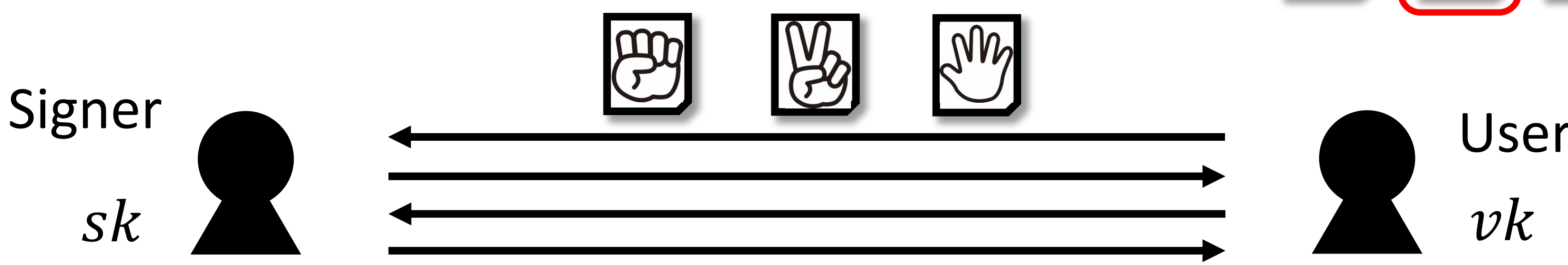


User

vk

(1,n)-Oblivious Signatures [Chen94]

Make a list of n candidate messages  .



During the interaction,
singer knows a message list
but has no idea which one
of message is selected.

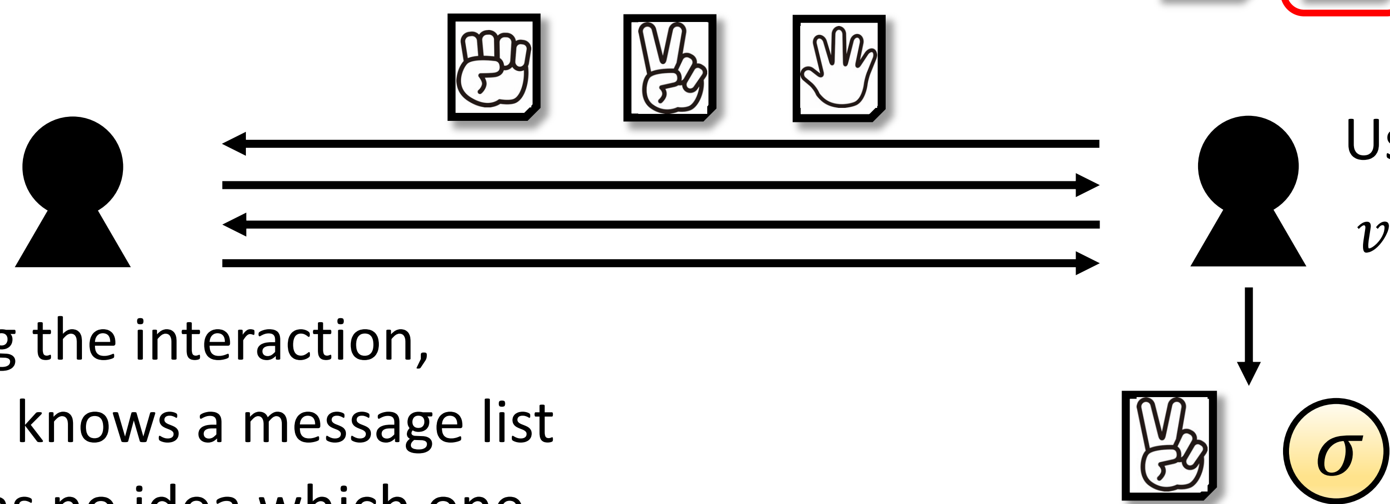
(1,n)-Oblivious Signatures [Chen94]

Selected

Make a list of n candidate messages .

Signer

sk



User

vk

During the interaction, singer knows a message list but has no idea which one of message is selected.

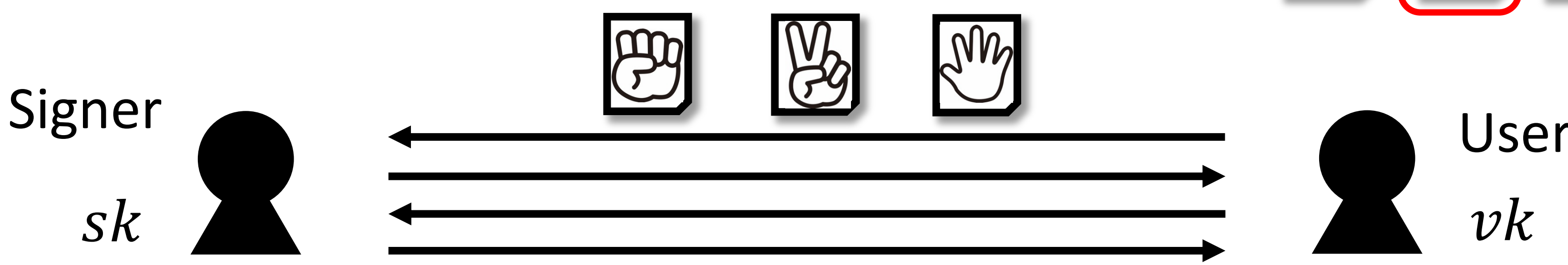
User obtain a signature σ on the selected message .

Anyone can verify a signature.

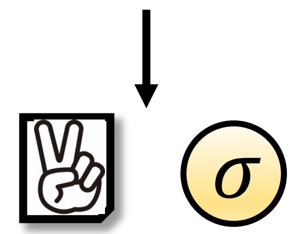
(1,n)-Oblivious Signatures [Chen94]

Selected

Make a list of n candidate messages .



During the interaction, singer knows a message list but has no idea which one of message is selected.



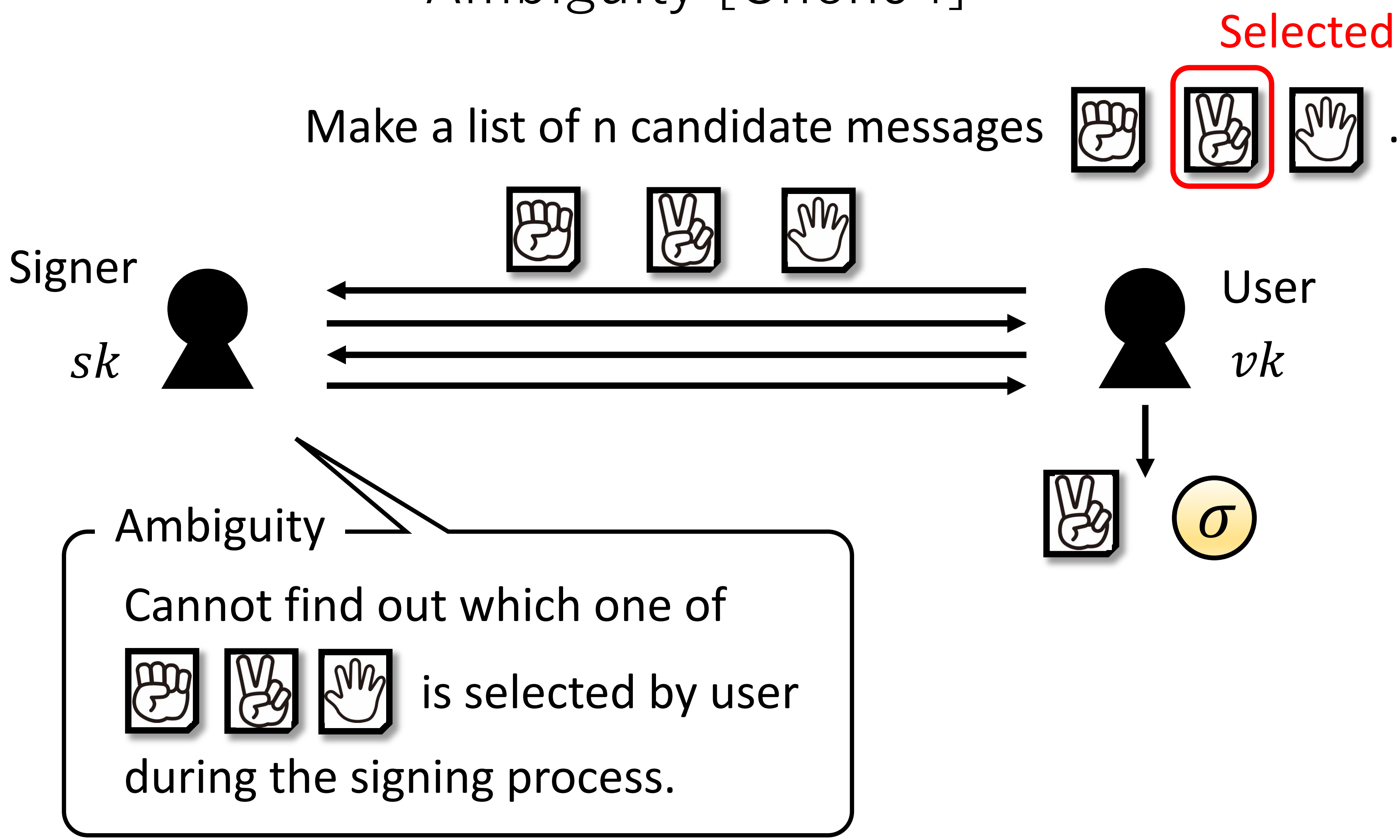
User obtain a signature σ on the selected message .

Anyone can verify a signature.

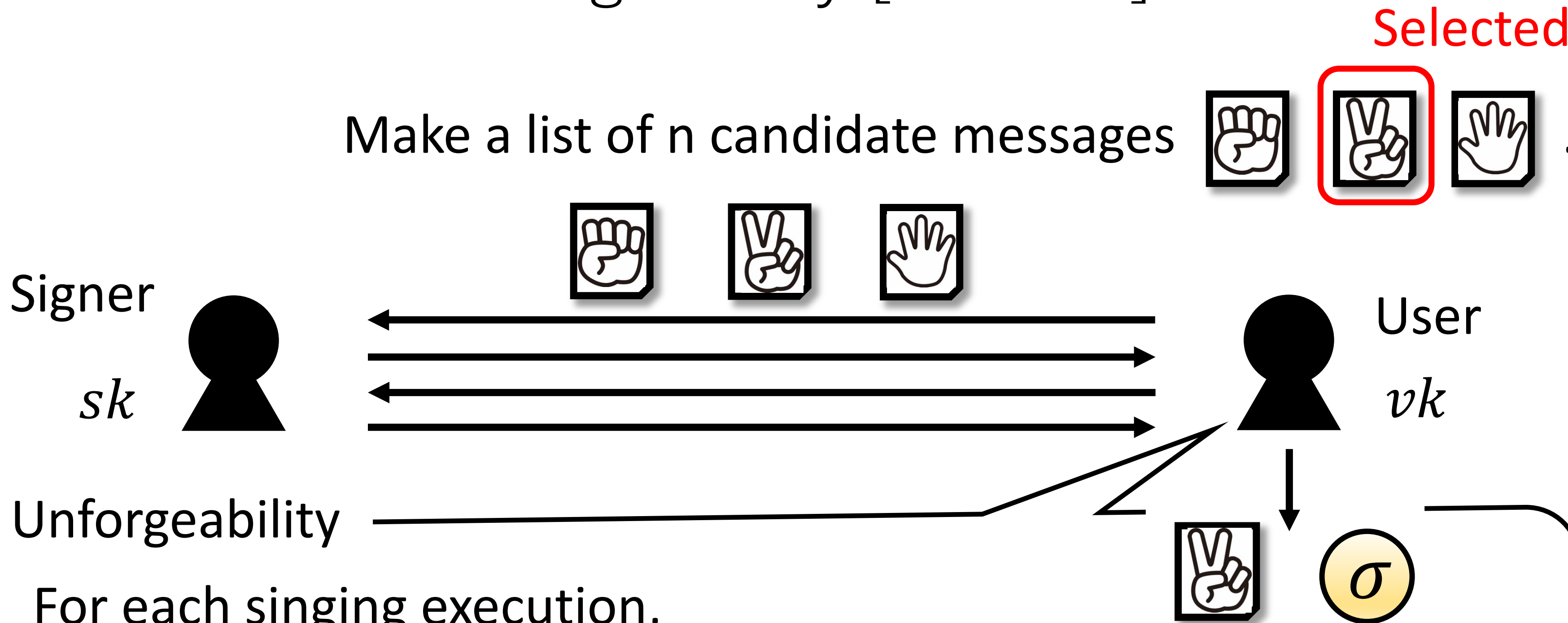
Security Requirements:

- Ambiguity
- Unforgeability

Ambiguity [Chen94]



Unforgeability [Chen94]

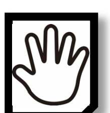


Unforgeability

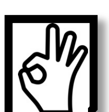
For each singing execution,

・Can derive a signature for only 1 of n messages in the list.

・Cannot obtain signatures on two or more messages in the list.

(e.g.  )

・Cannot obtain a signature on a message which is not in the list.

(e.g. )

Previous works for (1,n)-Oblivious Signatures

Chen [Chen94]

- Notion of (1, n)-oblivious signatures
- The first oblivious scheme

Tso et al. [TOO08]

- Formal definition and security model
- 2-move signing scheme
  based on DL assumption in ROM

Zhou et al. [ZLH22]

- Generic construction of 2-move signing scheme
  from commitment and a digital signature without ROM

Our Contributions

Chen [Chen94]

- Notion of (1, n)-oblivious signatures
- The first oblivious scheme

1. Revisit the unforgeability security model

Tso et al. [TOO08]

- Formal definition and security model
- 2-move signing scheme based on DL assumption in ROM

2. Second communication size improvement.

Zhou et al. [ZLH22]

- Generic construction of 2-move signing scheme from commitment and a digital signature without ROM

Syntax and Unforgeability Security model in the Previous Work

Syntax of (1,n)-Oblivious Signature Scheme

2-move (1,n)-OS (KGen, U_1, S_2, Derive, Verify)

Syntax of (1,n)-Oblivious Signature Scheme

2-move (1,n)-OS (KGen, U_1, S_2, Derive, Verify)

KGen$(1^{\lambda}) \rightarrow (vk, sk)$

Syntax of (1,n)-Oblivious Signature Scheme

2-move (1,n)-OS (KGen, U_1, S_2, Derive, Verify)

KGen$(1^\lambda) \to (vk, sk)$

Signing protocol (U$_1$, S$_2$, Derive)

Signer(sk)

User (vk, $(m_1, \cdots, m_n)$, j)

Syntax of (1,n)-Oblivious Signature Scheme

2-move (1,n)-OS (KGen, U_1, S_2, Derive, Verify)

$\mathrm{KGen}(1^{\lambda}) \rightarrow (vk, sk)$

Signing protocol (${\mathrm{U}_1}$, S_2, Derive)

Signer(sk)

User (vk, $(m_1, \cdots, m_n)$, j)

$\mathrm{U}_1(vk, (m_1, \cdots, m_n), j)$

↓

(μ, st)

Syntax of (1,n)-Oblivious Signature Scheme

2-move (1,n)-OS (KGen, U_1, S_2, Derive, Verify)

$\mathrm{KGen}(1^\lambda) \rightarrow (vk, sk)$

Signing protocol (U$_1$, S$_2$, Derive)

Signer(sk)

$\mathrm{S}_2(sk, (m_1, \cdots, m_n), \mu)$

$\downarrow$

ρ

$(m_1, \cdots, m_n), \mu$ ←

User (vk, $(m_1, \cdots, m_n), j$)

$\mathrm{U}_1(vk, (m_1, \cdots, m_n), j)$

$\downarrow$

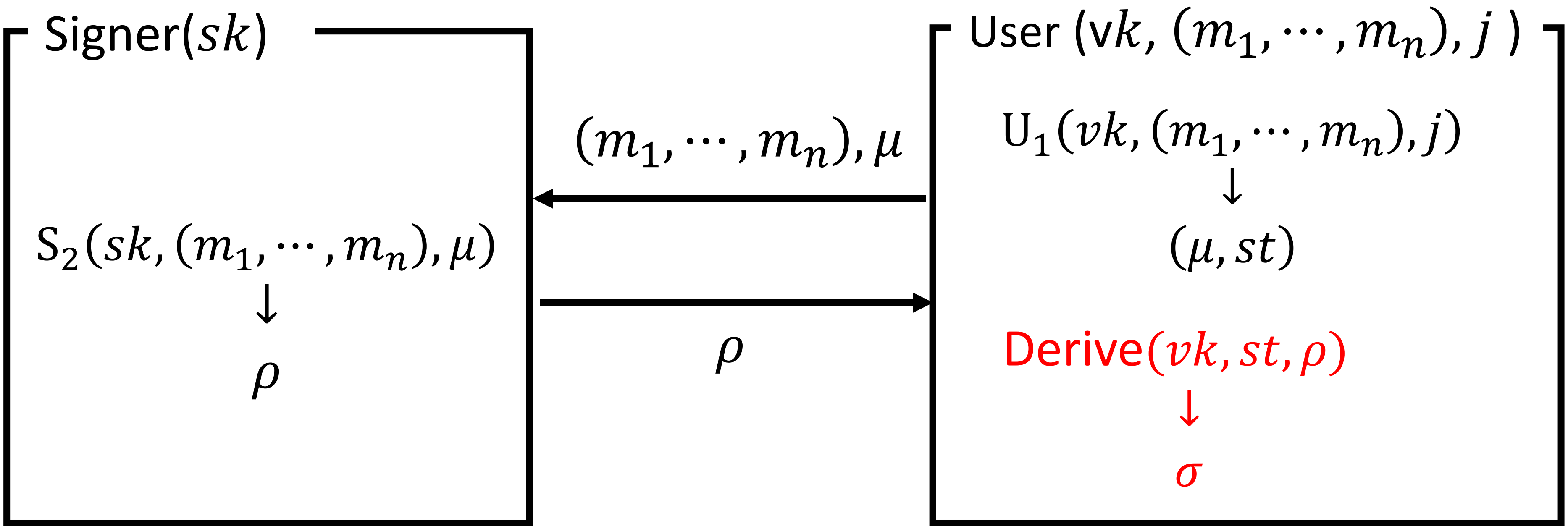
(μ, st)

Syntax of (1,n)-Oblivious Signature Scheme

2-move (1,n)-OS (KGen, U_1, S_2, Derive, Verify)

KGen$(1^\lambda) \rightarrow (vk, sk)$

Signing protocol (U$_1$, S$_2$, Derive)

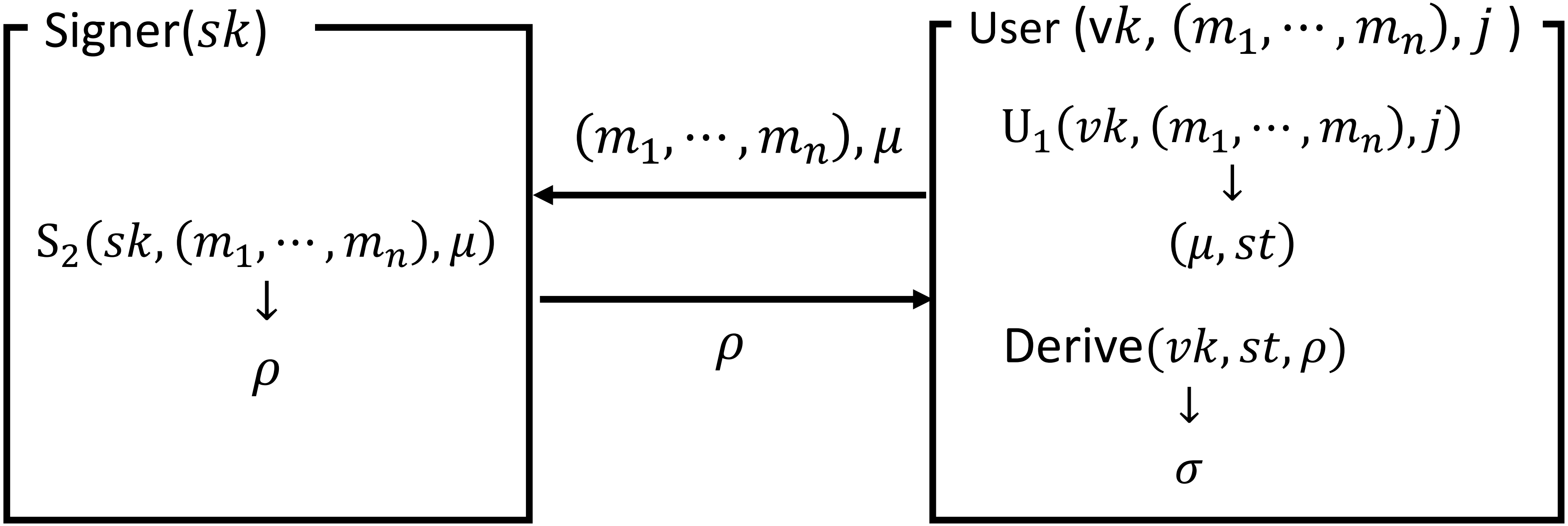


Syntax of (1,n)-Oblivious Signature Scheme

2-move (1,n)-OS (KGen, U_1, S_2, Derive, Verify)

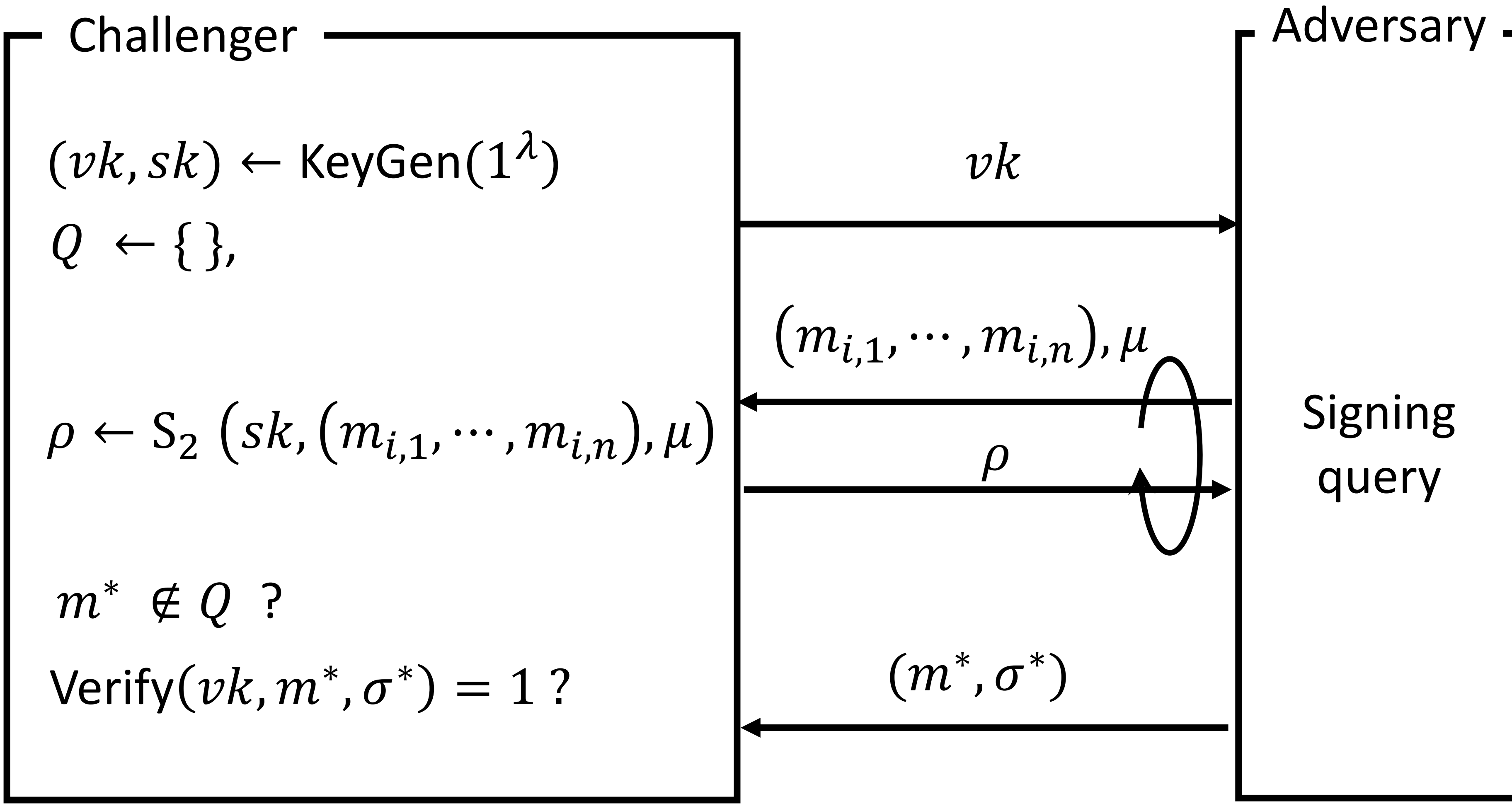
$\mathrm{KGen}(1^\lambda) \rightarrow (vk, sk)$

Signing protocol (U_1, S_2, Derive)



$\mathrm{Verify}(vk, m, \sigma) \rightarrow 0 \text{ or } 1$

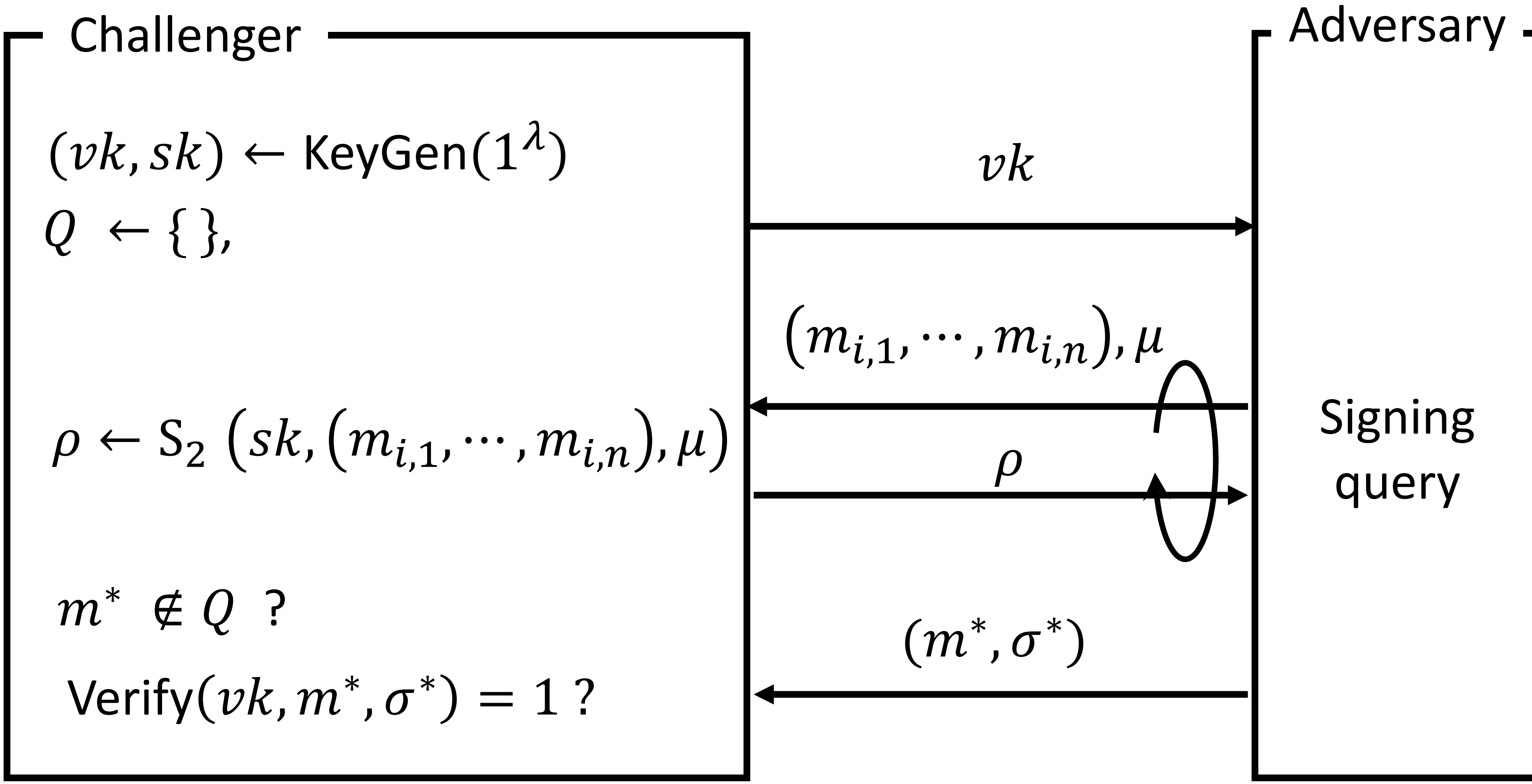
Unforgeability Security Game in [TOO08]



Q records signed messages that the adversary has obtained.

Problems in Unforgeability Security Model and Countermeasures

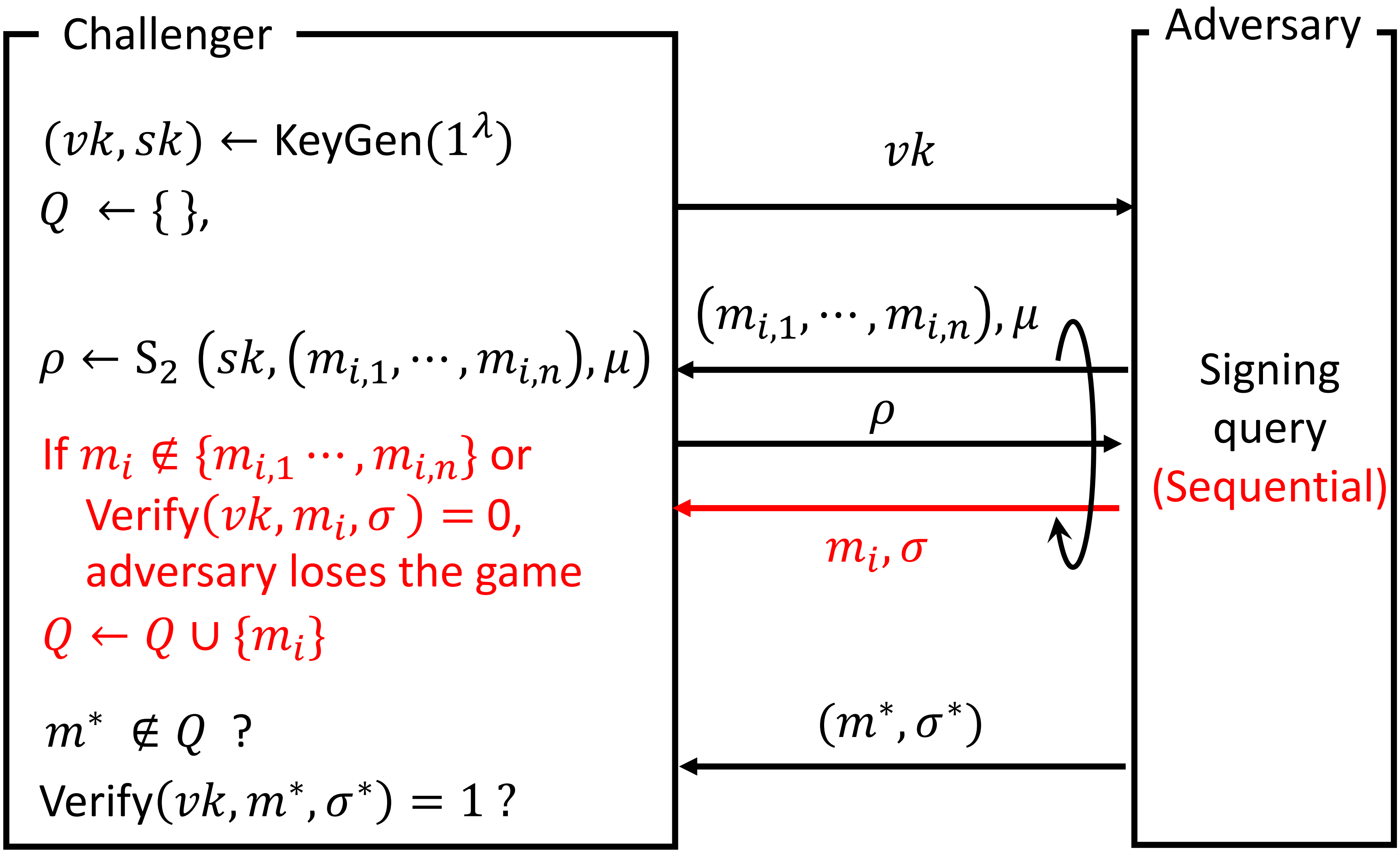
Problem 1 (How to Manage Set Q)



Q is a set of signed messages that the adversary has obtained.

By ambiguity, the challenger cannot know which one of message the adversary gets a signature in each signing query.

Countermeasure 1



Problem 2 (Trivial Attack)

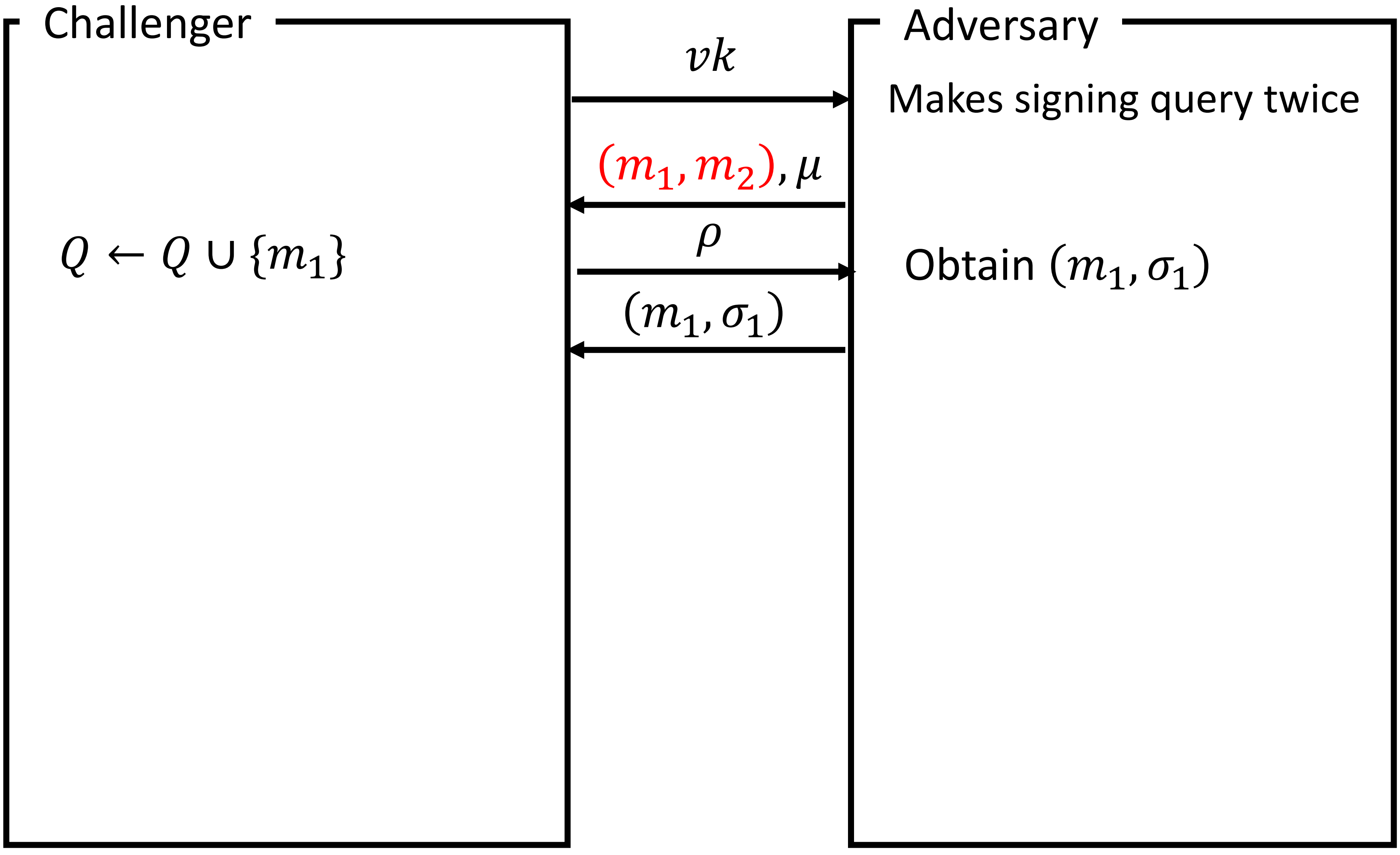
To simplify the discussion, we assume that n is 2.

Challenger

Adversary

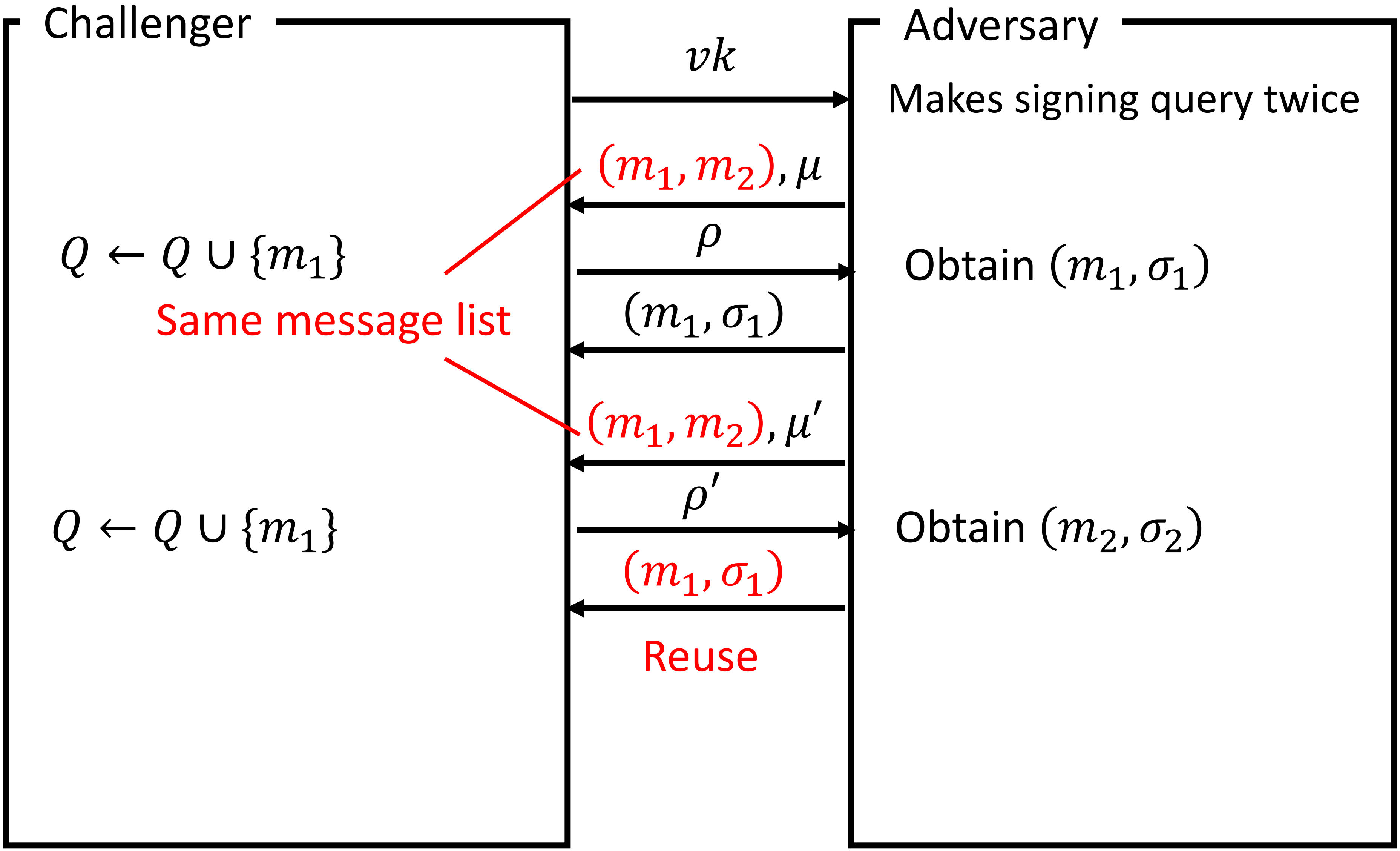
Problem 2 (Trivial Attack)

To simplify the discussion, we assume that n is 2.



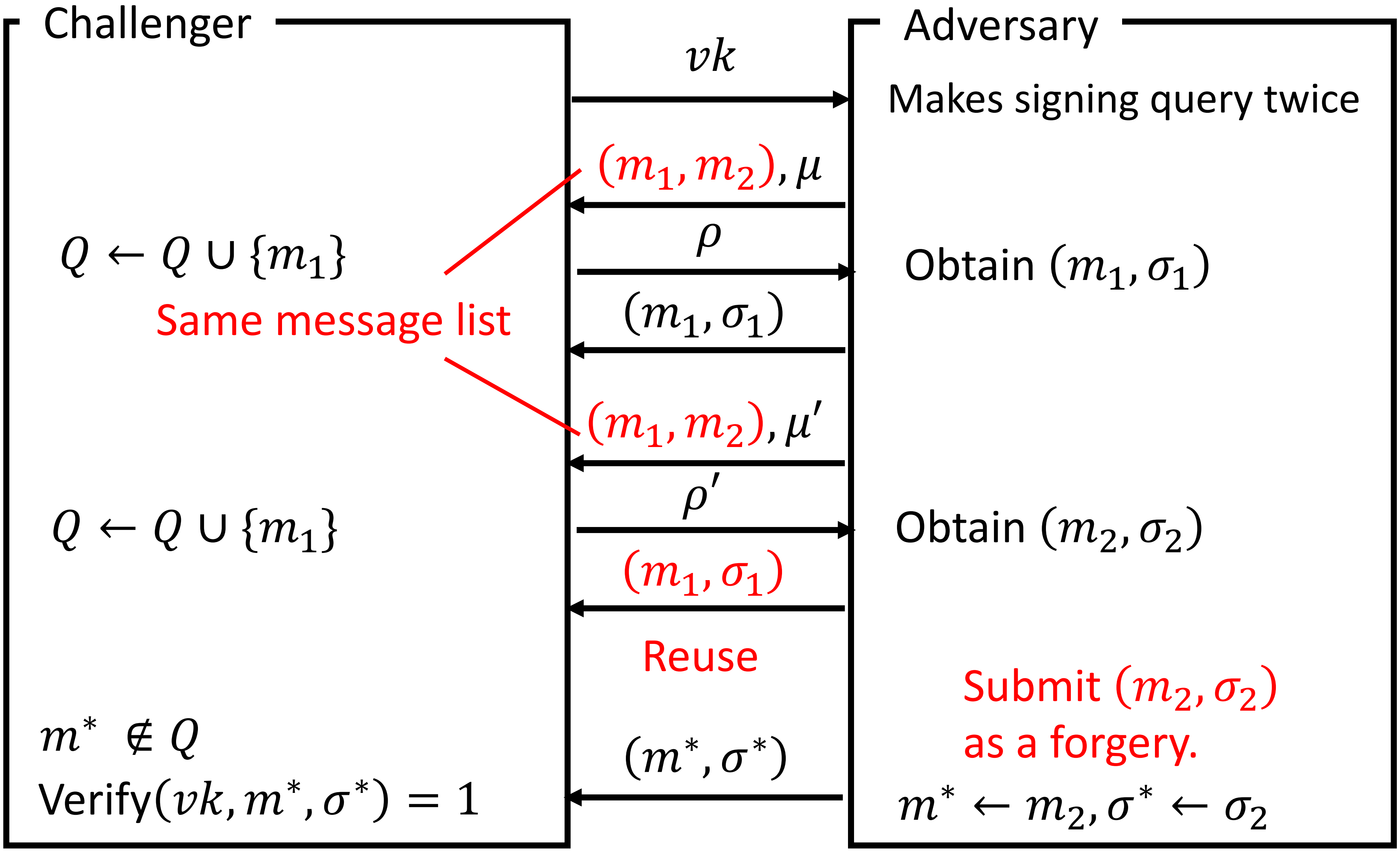
Problem 2 (Trivial Attack)

To simplify the discussion, we assume that n is 2.



Problem 2 (Trivial Attack)

To simplify the discussion, we assume that n is 2.



Countermeasure 2

Challenger

$(vk, sk) \leftarrow \mathsf{KeyGen}(1^{\lambda})$

$Q \leftarrow \{\,\}$,

$\rho \leftarrow S_2\left(sk, (m_{i,1}, \cdots, m_{i,n}), \mu\right)$

If $(m_i, \sigma) \in Q$ or
$m_i \notin \{m_{i,1} \cdots, m_{i,n}\}$ or
$\mathsf{Verify}(vk, m_i, \sigma) = 0$,
adversary loses the game

$Q \leftarrow Q \cup \{(m_i, \sigma)\}$

$(m^*, \sigma^*) \notin Q$?

$\mathsf{Verify}(vk, m^*, \sigma^*) = 1$?

Adversary

Signing query (Sequential)

Challenger → Adversary: vk

Adversary → Challenger: $(m_{i,1}, \cdots, m_{i,n}), \mu$

Challenger → Adversary: ρ

Adversary → Challenger: m_i, σ

Adversary → Challenger: (m^*, σ^*)

2. Prevent refreshing (reusing) signatures.
We makes sUF security as a defalt!

1. Signature resubmission check!

Problem 3 (Missing Adversary Strategy)

To simplify the discussion, we assume that n is 2.

Challenger

Adversary

Problem 3 (Missing Adversary Strategy)

To simplify the discussion, we assume that n is 2.

Challenger

$(vk, sk) \leftarrow \text{KeyGen}(1^\lambda)$

$Q \leftarrow \{\}$,

$\rho \leftarrow \mathrm{S}_2\ (sk, (m_1, m_2), \mu)$

vk

$(m_1, m_2), \mu$

ρ

Adversary

Problem 3 (Missing Adversary Strategy)

To simplify the discussion, we assume that n is 2.

Challenger

$(vk, sk) \leftarrow \text{KeyGen}(1^\lambda)$

$Q \leftarrow \{\}$,

$\rho \leftarrow \text{S}_2\ (sk, (m_1, m_2), \mu)$

If $(m_i, \sigma) \in Q$ or
$m_i \notin \{m_1, m_2\}$ or
$\text{Verify}(vk, m_i, \sigma) = 0$,
adversary loses the game.

Challenger → Adversary: vk

Adversary → Challenger: $(m_1, m_2), \mu$

Challenger → Adversary: ρ

Adversary → Challenger: Give up !

Adversary

Instead obtaining
(m^*, σ) s.t.
$m^* \notin (m_1, m_2)$

Problem 3 (Missing Adversary Strategy)

To simplify the discussion, we assume that n is 2.

Challenger

$(vk, sk) \leftarrow \mathsf{KeyGen}(1^\lambda)$

$Q \leftarrow \{\}$,

$\rho \leftarrow S_2\ (sk, (m_1, m_2), \mu)$

If $(m_i, \sigma) \in Q$ or
$m_i \notin \{m_1, m_2\}$ or
$\mathsf{Verify}(vk, m_i, \sigma) = 0$,
adversary loses the game.

Challenger → Adversary: vk

Adversary → Challenger: $(m_1, m_2), \mu$

Challenger → Adversary: ρ

Adversary → Challenger: Give up !

Adversary

Instead obtaining
(m^*, σ) s.t.
$m^* \notin (m_1, m_2)$
Always loses the game!

Unforgeability security must guarantee that the user cannot obtain a signature on a message which is not in the list!

→ This security model does not capture this requirement!

Countermeasure 3

Challenger

$(vk, sk) \leftarrow \text{KeyGen}(1^\lambda)$

$Q \leftarrow \{\}$,

$\rho \leftarrow S_2\left(sk, (m_{i,1}, \cdots, m_{i,n}), \mu\right)$

If $(m_i, \sigma) \in Q$ or
Verify$(vk, m_i, \sigma) = 0$,
adversary loses the game.

If $m_i \notin \{m_{i,1} \cdots, m_{i,n}\}$,
adversary wins the game.

If $m_i \in \{m_{i,1} \cdots, m_{i,n}\}$,
$Q \leftarrow Q \cup \{(m_i, \sigma)\}$

$(m^*, \sigma^*) \notin Q$?

Verify$(vk, m^*, \sigma^*) = 1$?

Adversary

vk

$(m_{i,1}, \cdots, m_{i,n}), \mu$

ρ

m_i, σ

Signing query (Sequential)

(m^*, σ^*)

Another winning path!
Capture the adversary with the missing strategy.

Communication Size Improvement Result in Our Scheme

Communication Message Size

Scheme	Building Block	First Message μ	Second Message ρ
[ZLH 22]	DS COM	1 com for m_j	n sigs on (m_i, μ)

We reduce the second message size !

Communication Size Improvement Result

Scheme	Building Block	First Message μ	Second Message ρ
[ZLH 22]	DS COM	1 com for m_j	n sigs on (m_i, μ)
Ours	DS COM Merkle Tree	1 com for m_j	1 sig on (root, μ)

root: Assigned root node value of the Merkle Tree on $(m_1, \cdots, m_n)$

Communication Size Improvement Result

Scheme	Building Block	First Message μ	Second Message ρ
[ZLH 22]	DS COM	1 com for m_j	n sigs on (m_i, μ)
Ours	DS COM Merkle Tree	1 com for m_j	1 sig on (root, μ)

root: Assigned root node value of the Merkle Tree on $(m_1, \cdots, m_n)$

Security of Our Scheme

Ambiguity Security: Hiding COM

Unforgeability Security: sEUF-CMA DS + sBinding COM + Coll resist H

Summary

- We revisited the unforgeability security model by Tso et al.
  We identify problems and redefine the security model.

- We improve the generic construction by Zhou et al.
  Our scheme offers the smaller second message size.

Thank you!

References

[Chen94] L. Chen. Oblivious signatures. (ESORICS 1994)

[TOO08] R. Tso, T. Okamoto, and E. Okamoto. 1-out-of-n oblivious signatures. (ISPEC 2008)

[YLTTM22] J. You, Z. Liu, R. Tso, Y. Tseng, and M. Mambo. Quantum-resistant 1-out-of-n oblivious signatures from lattices. (IWSEC 2022)

[ZLH22] Y. Zhou, S. Liu, and S. Han. Generic construction of 1-out-of-n oblivious signatures. (IEICE Trans. Inf. Syst. 2022)

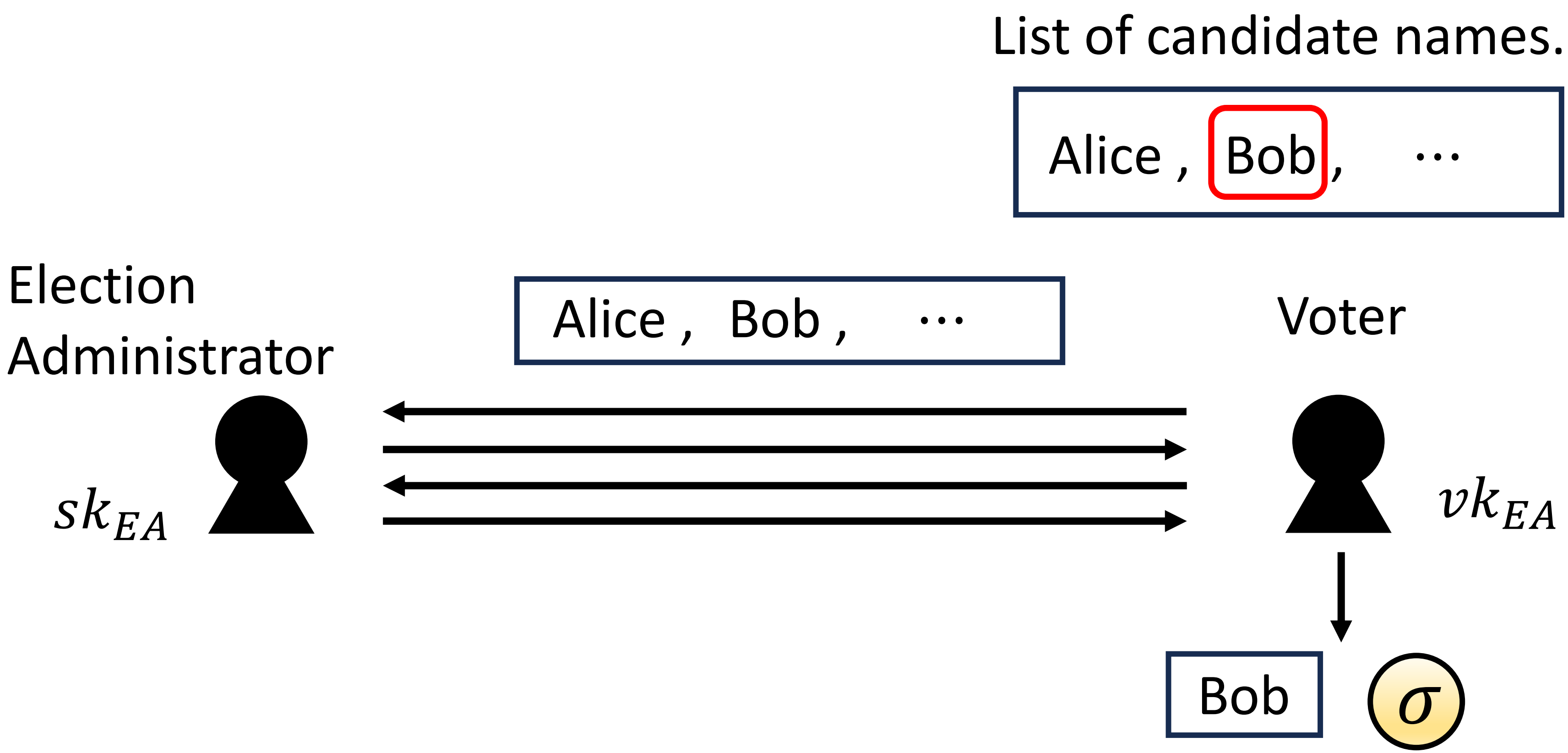
[SYL08] C. Song, X. Yin, and Y. Liu. A practical electronic voting protocol based upon oblivious signature scheme. (CIS 2008)

[CC18] S. Chiou and J. Chen. Design and implementation of a multiple-choice e-voting scheme on mobile system using novel t -out-of- n oblivious signature. (J. Inf. Sci. Eng. 2018).

Appendix

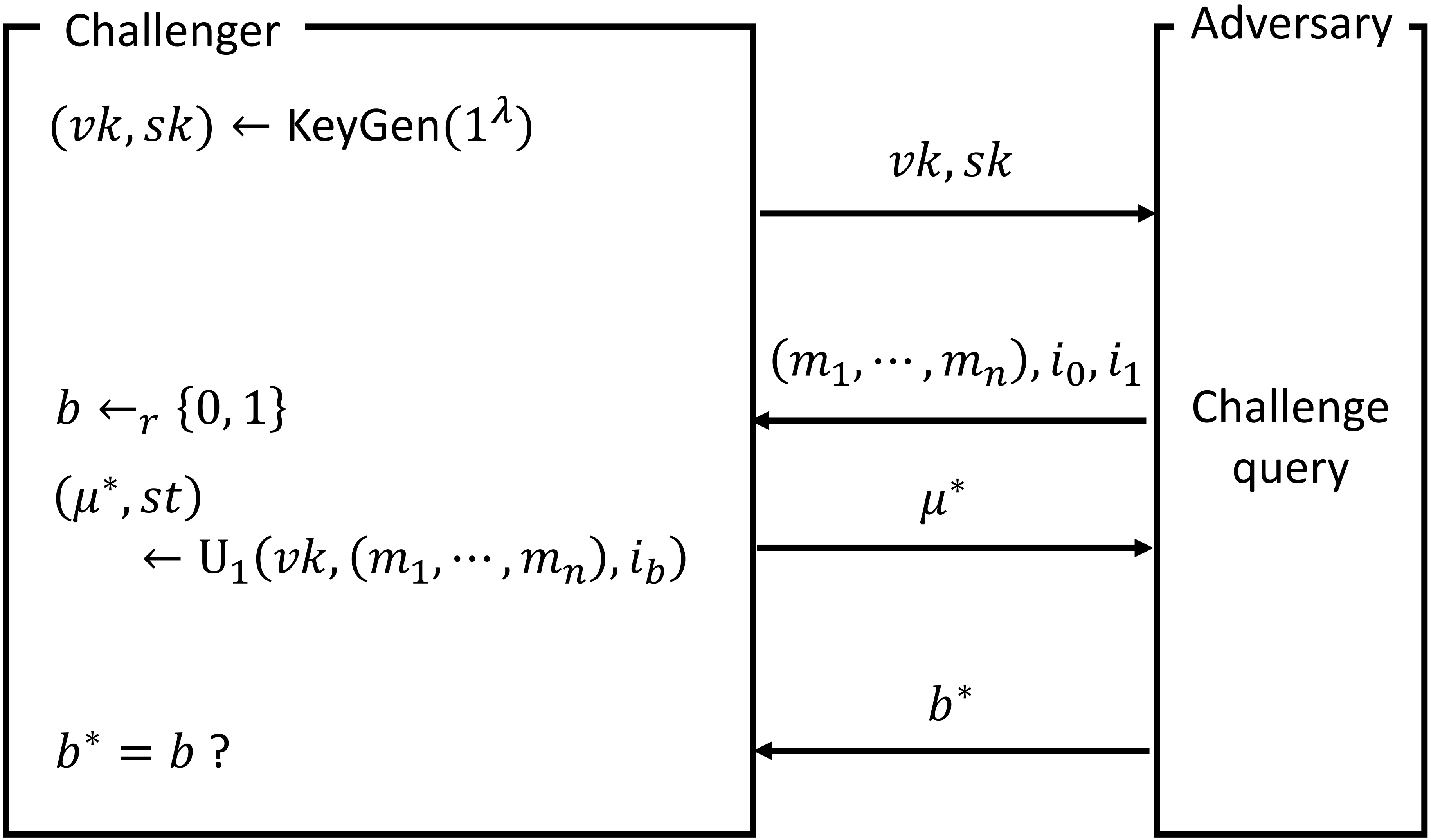
Application of (1,n)-Oblivious Signatures

E-voting system based on oblivious signatures [SYL08, CC18]



Voter cast the vote for "Bob" with the signature σ.

Ambiguity Security Game



Commitment Scheme

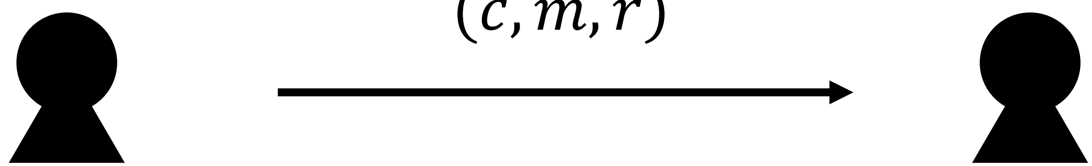
Commitment Scheme

$$ck \leftarrow \mathsf{KGen}(1^{\lambda})$$

Sender (ck, m)

Receiver(ck)

(c, m, r)



$c \leftarrow$Commit$(ck, m; r)$

Open$(ck, c, m; r) = 1$?

Security

Hiding: A commitment c hides the committed message m.

Binding: A commitment c can only be opened with the committed message m.

Digital Signature Scheme

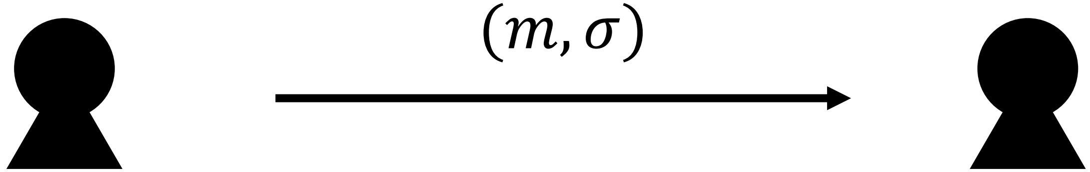
Digital Signature Scheme

$(vk, sk) \leftarrow \text{KGen}(1^\lambda)$

Signer (sk, m)

Verifier (vk)

(m, σ)



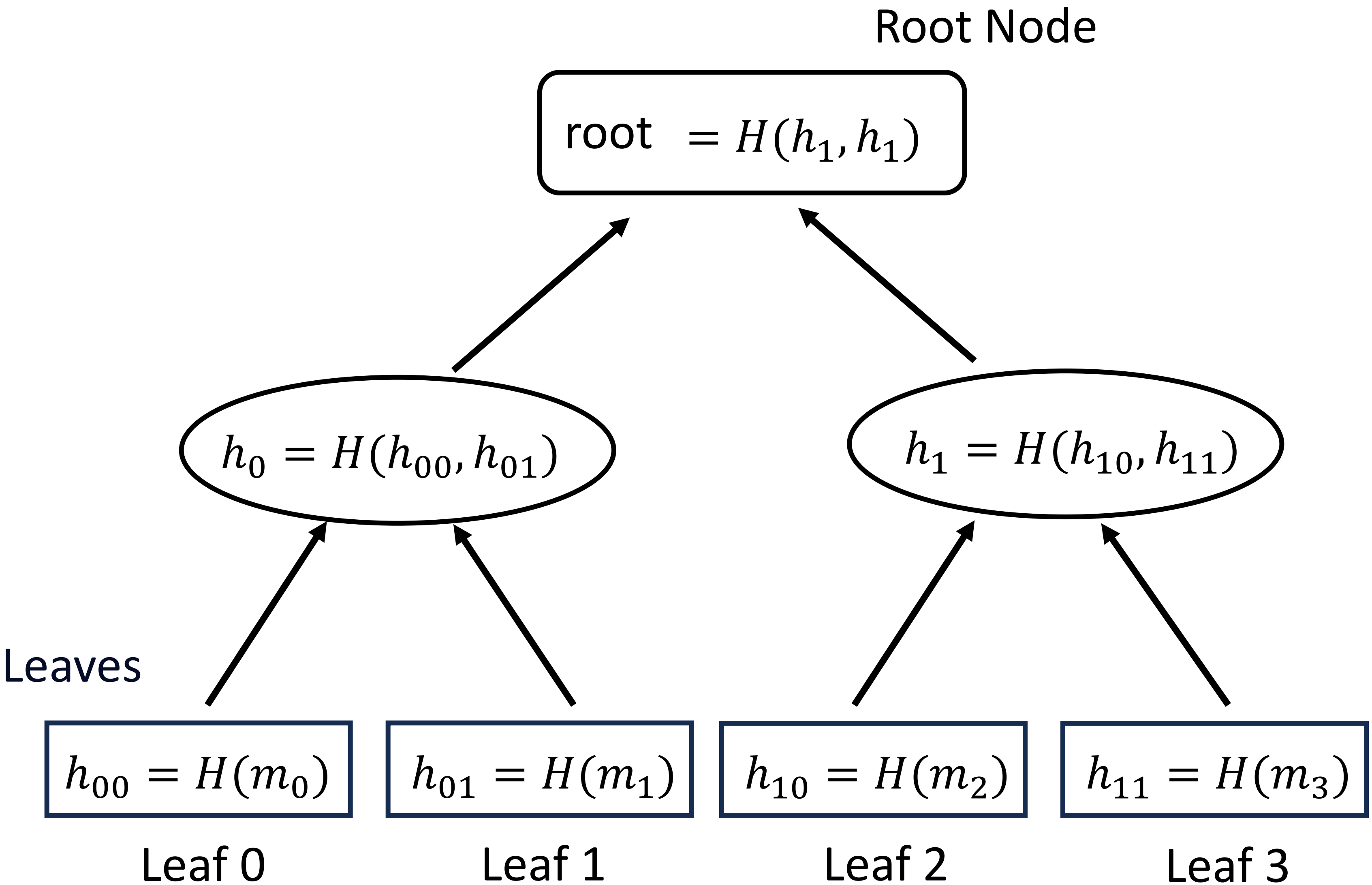
$\sigma \leftarrow \text{Sign}(sk, m)$

$\text{Verify}(vk, m, \sigma) = 1$?

Security (Strong EUF-CMA)

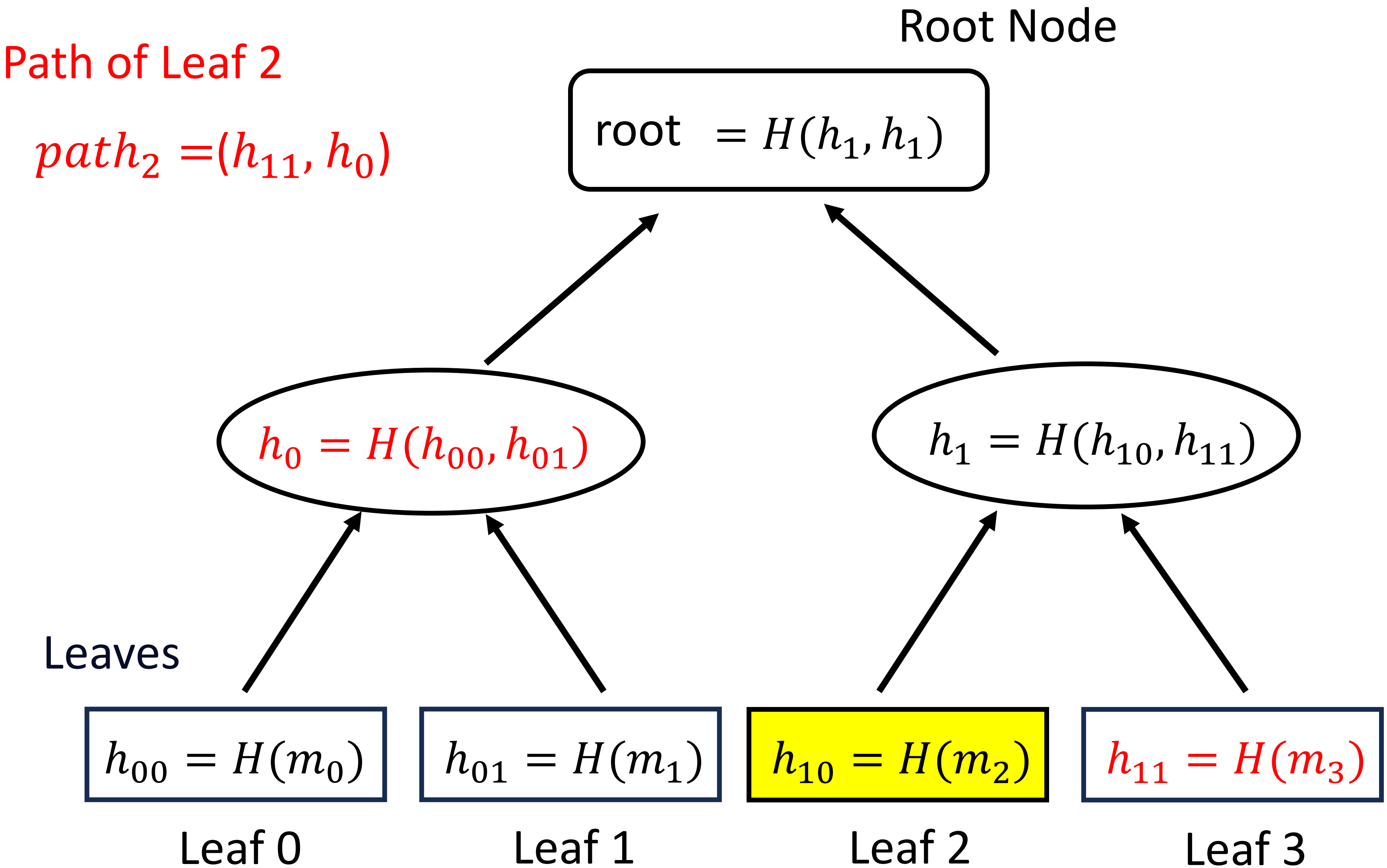
If an adversary obtains message-signature pairs $(m_i, \sigma_i)_i$ on their message choice via signing queries, it is difficult to generate a forgery (m^*, σ^*) which has not been outputted by singing queries.

Merkle Tree



Root Node
root $= H(h_1, h_1)$
$h_0 = H(h_{00}, h_{01})$
$h_1 = H(h_{10}, h_{11})$
Leaves
$h_{00} = H(m_0)$
$h_{01} = H(m_1)$
$h_{10} = H(m_2)$
$h_{11} = H(m_3)$
Leaf 0
Leaf 1
Leaf 2
Leaf 3

Merkle Tree


Path of Leaf 2
$path_2 = (h_{11}, h_0)$
Root Node
root $= H(h_1, h_1)$
$h_0 = H(h_{00}, h_{01})$
$h_1 = H(h_{10}, h_{11})$
Leaves
$h_{00} = H(m_0)$
$h_{01} = H(m_1)$
$h_{10} = H(m_2)$
$h_{11} = H(m_3)$
Leaf 0
Leaf 1
Leaf 2
Leaf 3

Generic Construction by Zhou et al. [ZLH22]

DS: Digital signature scheme
Com: Commitment scheme

User ($vk^{\mathrm{OS}} = (ck, vk^{\mathrm{DS}})$, $(m_1, \cdots, m_n)$, j)

Signer ($sk^{\mathrm{OS}} = sk^{\mathrm{DS}}$)

$\mathrm{U}_1(vk^{\mathrm{OS}}, (m_1, \cdots, m_n), j)$
$c \leftarrow$ Com.Commit$(ck, m_j; r)$

$(m_1, \cdots, m_n)$
$\mu = c$

$\mathrm{S}_2(sk^{\mathrm{OS}}, (m_1, \cdots, m_n), \mu)$
For $i \in [n]$,
$\sigma_i^{\mathrm{DS}} \leftarrow$ DS.Sign$(sk^{\mathrm{DS}}, (m_i, \mu))$

$\rho = (\sigma_i^{\mathrm{DS}})_{i \in [n]}$

Derive$(vk^{\mathrm{OS}}, st = (r, j), \rho)$
$\sigma^{\mathrm{OS}} \leftarrow (c, r, \sigma_j^{\mathrm{DS}})$

A second communication message ρ needs n signatures !
Signing on (m_i, μ) is seems redundant.

Our Improved Scheme

DS: Digital signature scheme
Com: Commitment scheme
H: Hash function

User $(vk^{\mathrm{OS}} = (ck, vk^{\mathrm{DS}}, H),$ $(m_1, \cdots, m_n),\ j)$

Signer $(sk^{\mathrm{OS}} = (sk^{\mathrm{DS}}, H))$

$\mathrm{U}_1(vk, (m_1, \cdots, m_n), j)$

c ←Com.Commit$(ck, m_j; r)$

$(m_1, \cdots, m_n)$
$\mu = c$

$\mathrm{S}_2(sk, (m_1, \cdots, m_n), \mu)$

Check messages in $(m_1, \cdots, m_n)$ are all distinct.

Compute the root of Merkle Tree from $(m_1, \cdots, m_n)$

σ^{DS} ←DS.Sign$(sk^{\mathrm{DS}}, (\mathrm{root}, \mu))$

$\rho = \sigma^{\mathrm{DS}}$

1 signature

Derive$(vk, st = (r,\ j), \rho)$

Compute root of Merkle Tree from $(m_1, \cdots, m_n)$.

Compute $path_j$.

$\sigma^{\mathrm{OS}} \leftarrow (c, r, \sigma_j^{\mathrm{DS}},$ $\mathrm{root}, path_j)$

Why Our Model Cannot Be Straightforwardly Extended to Concurrent Signing Model ?

If we extend our model to concurrent signing setting, there is a problem.

Challenger

1st signing query

2nd signing query

Fin 1st signing query

$m^* \notin (m_{1,1}, m_{1,2})$

Really forgery ?

Messages

vk (Challenger → Adversary)

$(m_{1,1}, m_{1,2}), \mu_1$ (Adversary → Challenger)

ρ_1 (Challenger → Adversary)

$(m_{2,1}, m_{2,2}), \mu_2$ (Adversary → Challenger)

ρ_2 (Challenger → Adversary)

(m^*, σ^*) (Adversary → Challenger)

Adversary

Makes signing query twice concurrently

Obtain $(m_{1,1}, \sigma_1)$

Obtain $(m_{2,1}, \sigma_2)$

$m^* \leftarrow m_{2,1}, \sigma^* \leftarrow \sigma_2$