Strongly Secure Universal Thresholdizer

Ehsan Ebrahimi
University of Luxembourg

Anshu Yadav
IST Austria

Threshold Cryptography - Motivation

Distributes privileged operation amongst multiple parties

↓

no single point of security failure

ThPKE, ThSignature, ThIBE, ThTraitorTracing, etc.

Threshold Cryptography - Motivation

Distributes privileged operation amongst multiple parties

↓

no single point of security failure

ThPKE, ThSignature, ThIBE, ThTraitorTracing, etc.

Interactive /non-interactive

classical / post-quantum assumptions

Threshold Cryptography - Motivation

Distributes privileged operation amongst multiple parties

no single point of security failure

ThPKE, ThSignature, ThIBE, ThTraitorTracing, etc.

Interactive /**non-interactive**

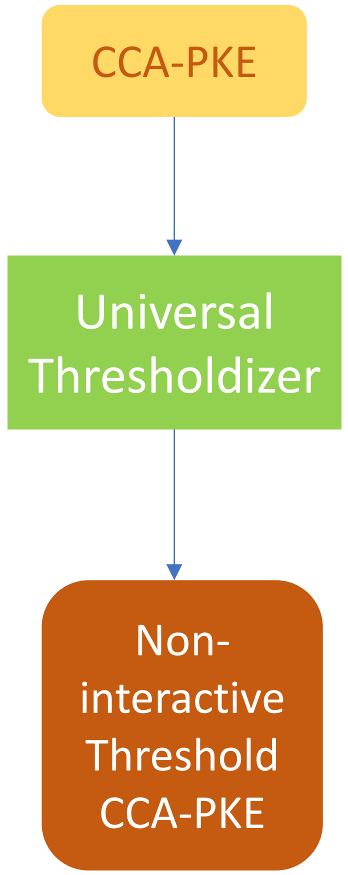
classical / **post-quantum** assumptions

Universal Thresholdizer [BGG+18]

Adds thresholdizing functionality to several cryptographic primitives

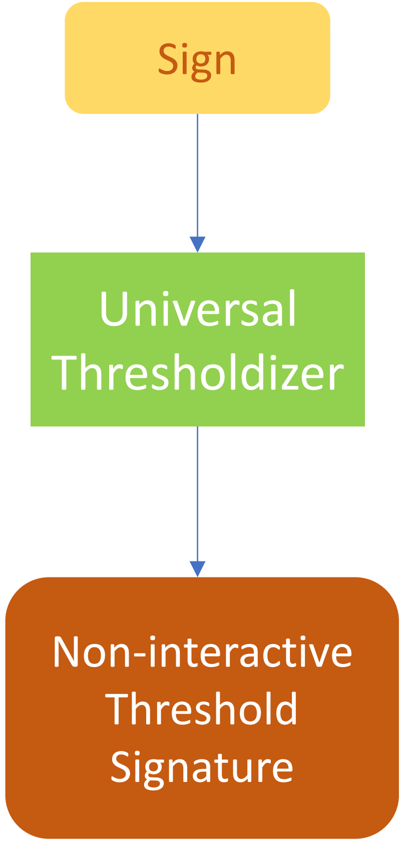
Universal Thresholdizer [BGG+18]

Adds thresholdizing functionality to several cryptographic primitives



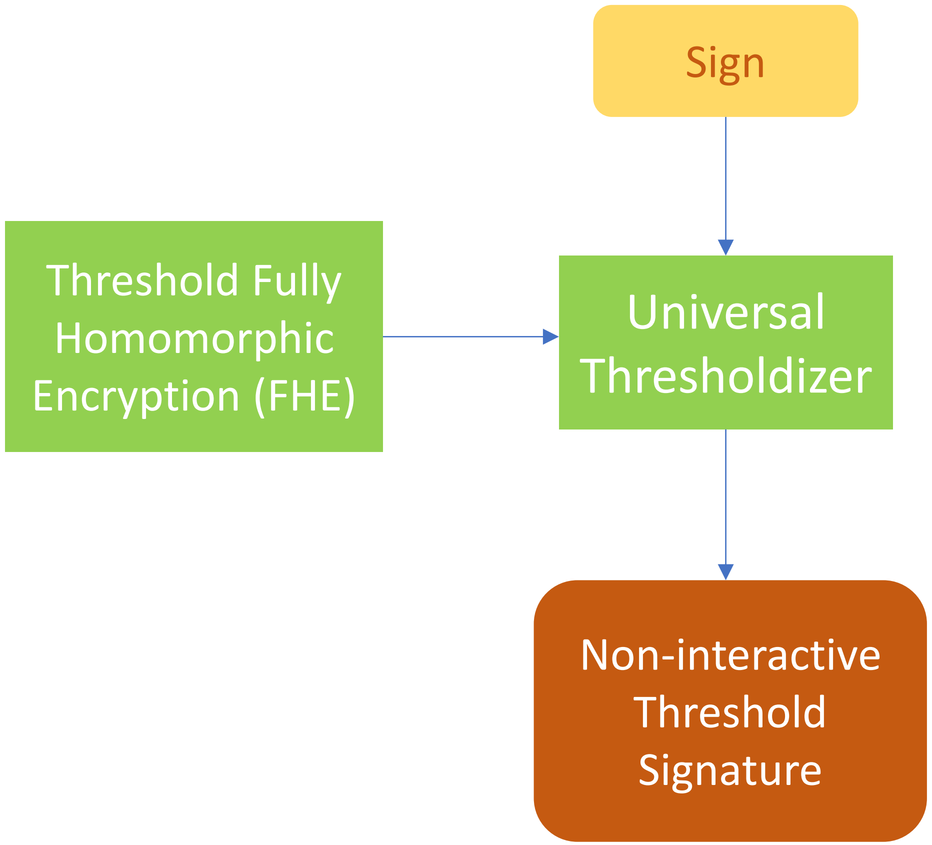
Universal Thresholdizer [BGG+18]

Adds thresholdizing functionality to several cryptographic primitives



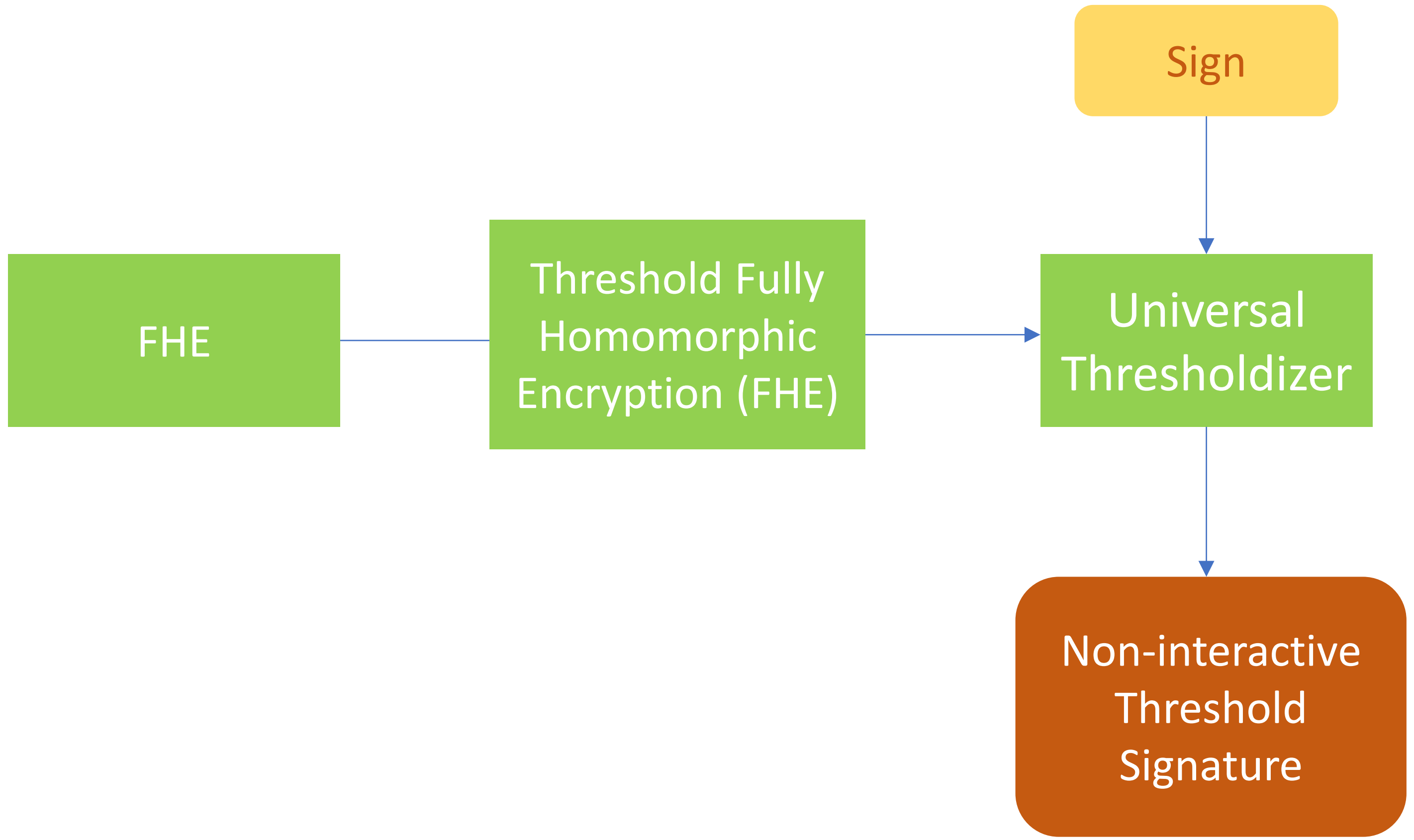
Universal Thresholdizer [BGG+18]

Adds thresholdizing functionality to several cryptographic primitives



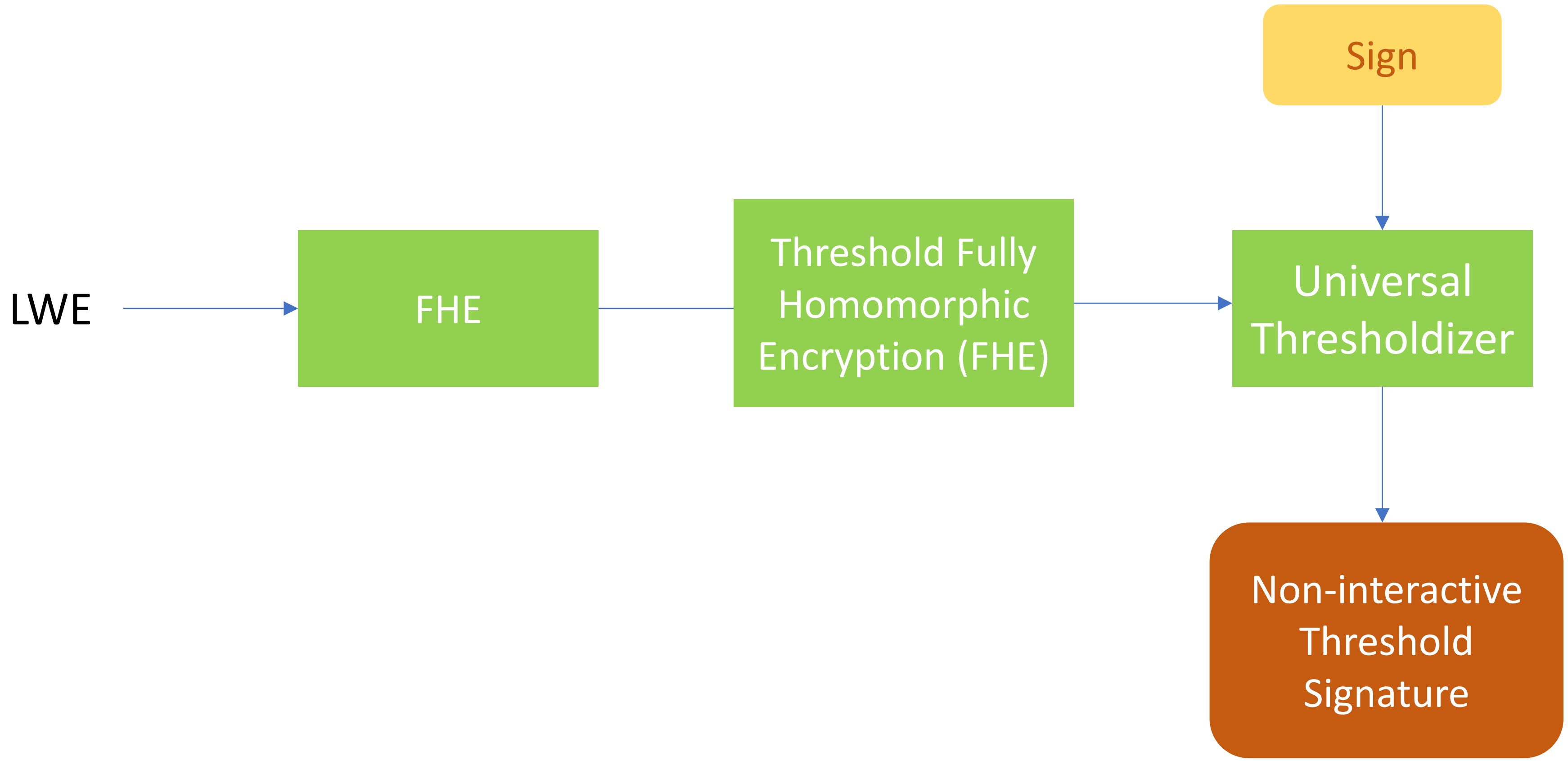
Universal Thresholdizer [BGG+18]

Adds thresholdizing functionality to several cryptographic primitives



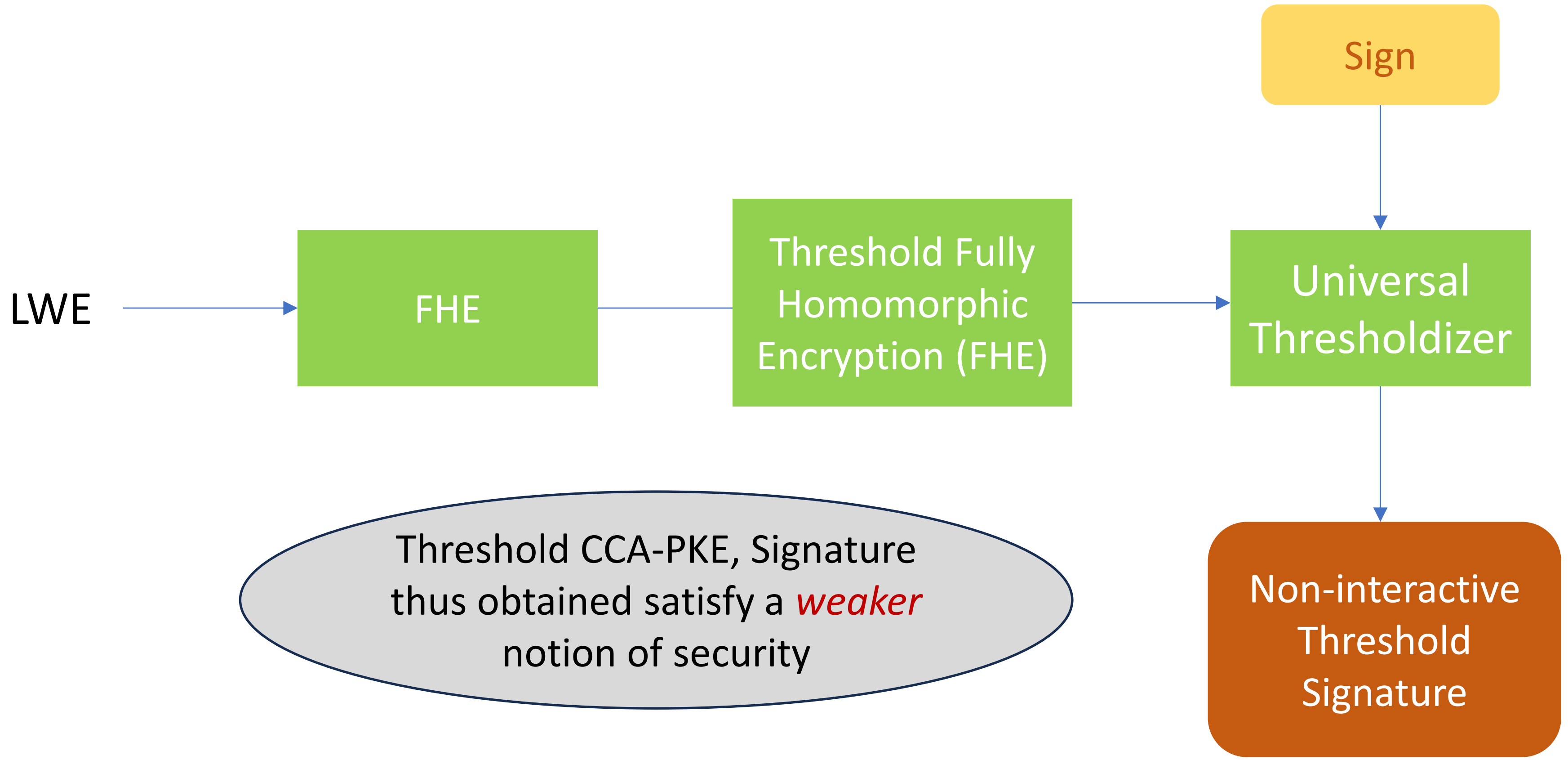
Universal Thresholdizer [BGG+18]

Adds thresholdizing functionality to several cryptographic primitives



Universal Thresholdizer [BGG+18]

Adds thresholdizing functionality to several cryptographic primitives



Our Contributions

- Define and build universal thresholdizer (UT) and threshold FHE (TFHE) with *stronger* security notions
  - Needed to achieve *stronger* security for primitives thresholdized using UT

- Using our universal thresholdizer we get the first non-interactive lattice based threshold signature scheme with the stronger security

- Also define various security notions for Threshold Signature and relations between them

Our Contributions

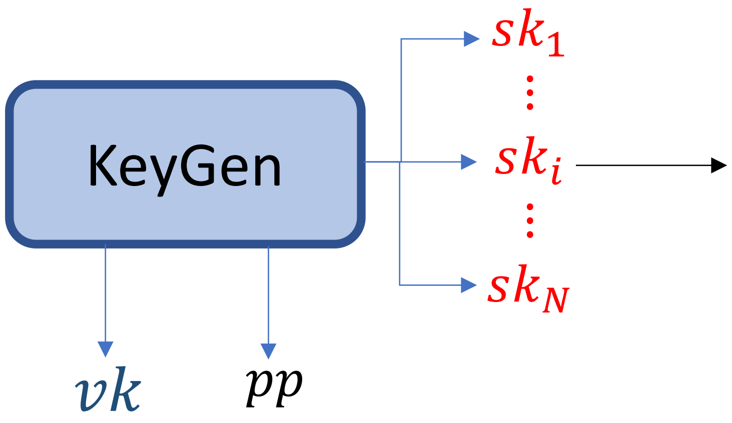
Related to (partial) adaptivity

- Define and build universal thresholdizer (UT) and threshold FHE (TFHE) with *stronger* security notions
  - Needed to achieve *stronger* security for primitives thresholdized using UT

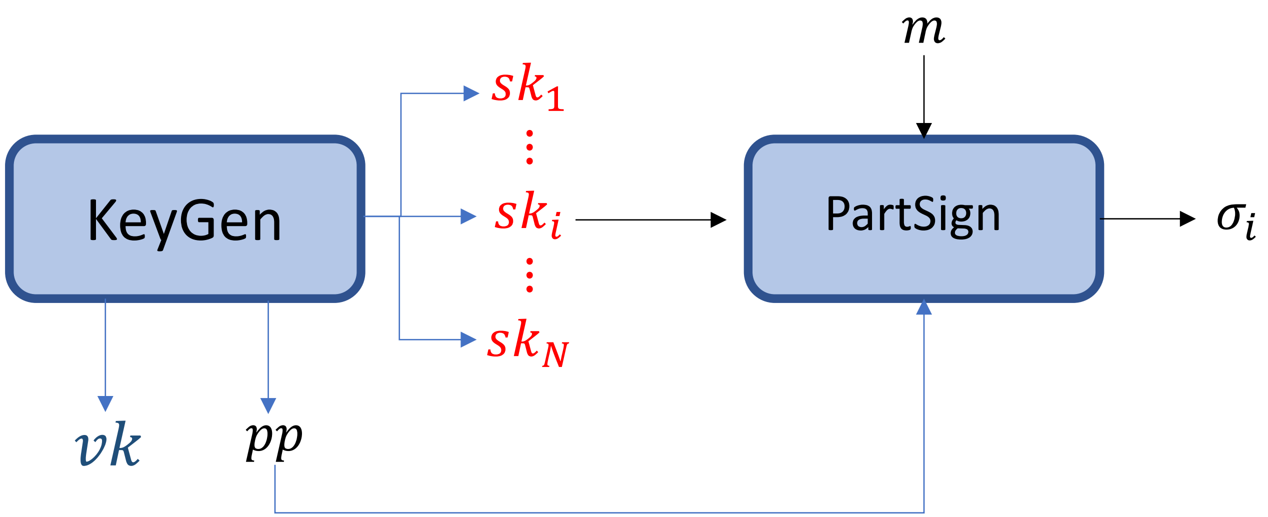
- Using our universal thresholdizer we get the first non-interactive lattice based threshold signature scheme with the stronger security

- Also define various security notions for Threshold Signature and relations between them

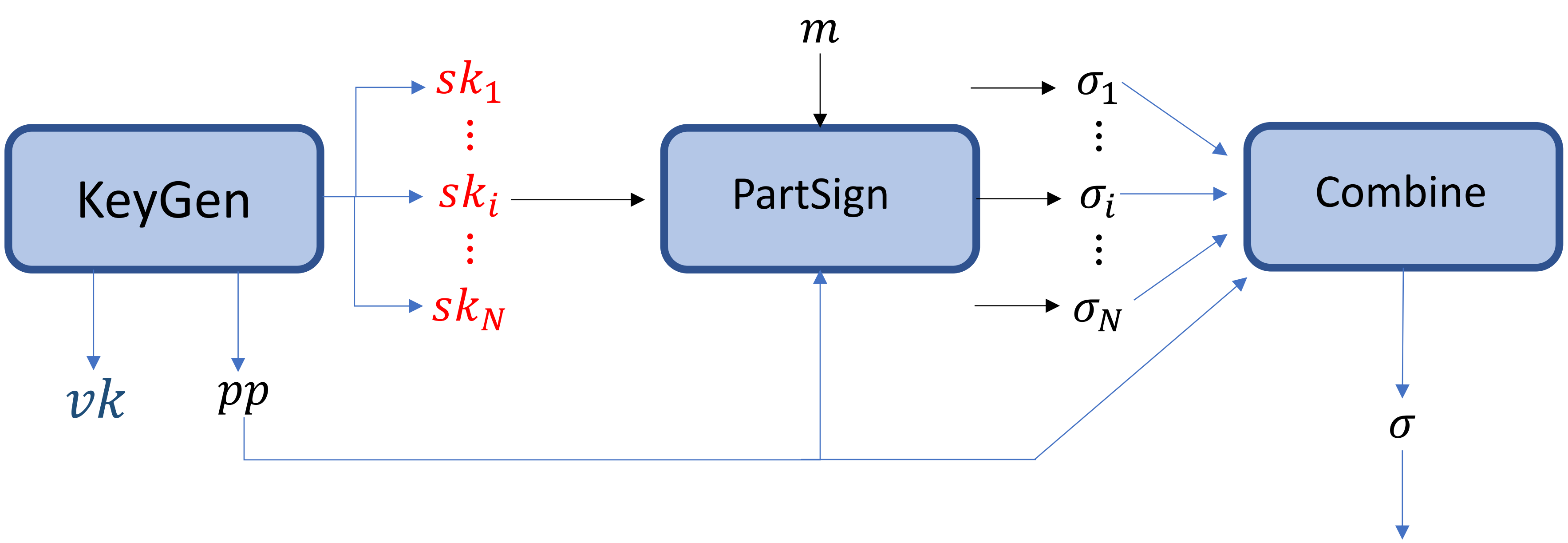
Threshold Signature Definition [BGG+18]



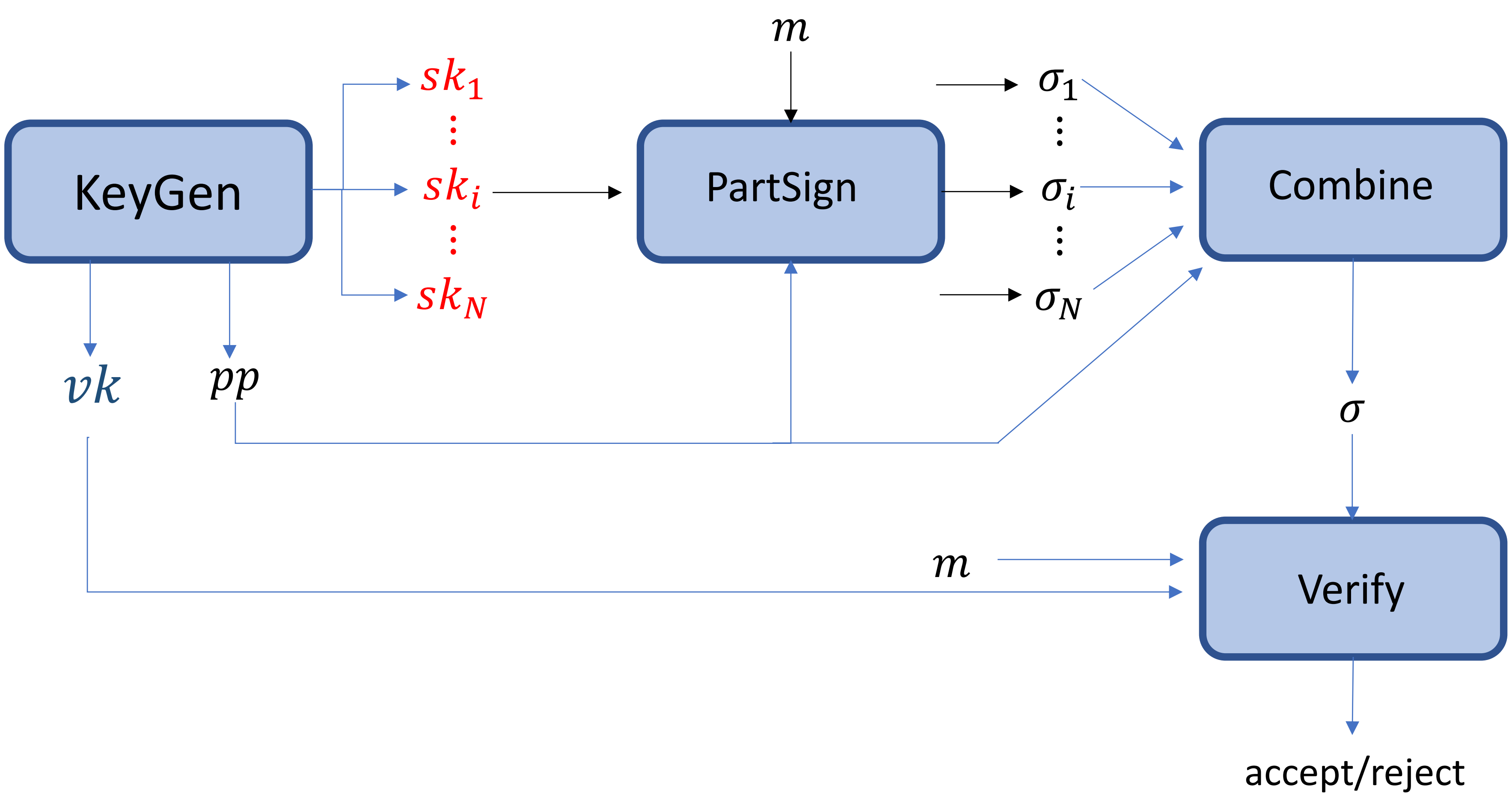
Threshold Signature Definition [BGG+18]



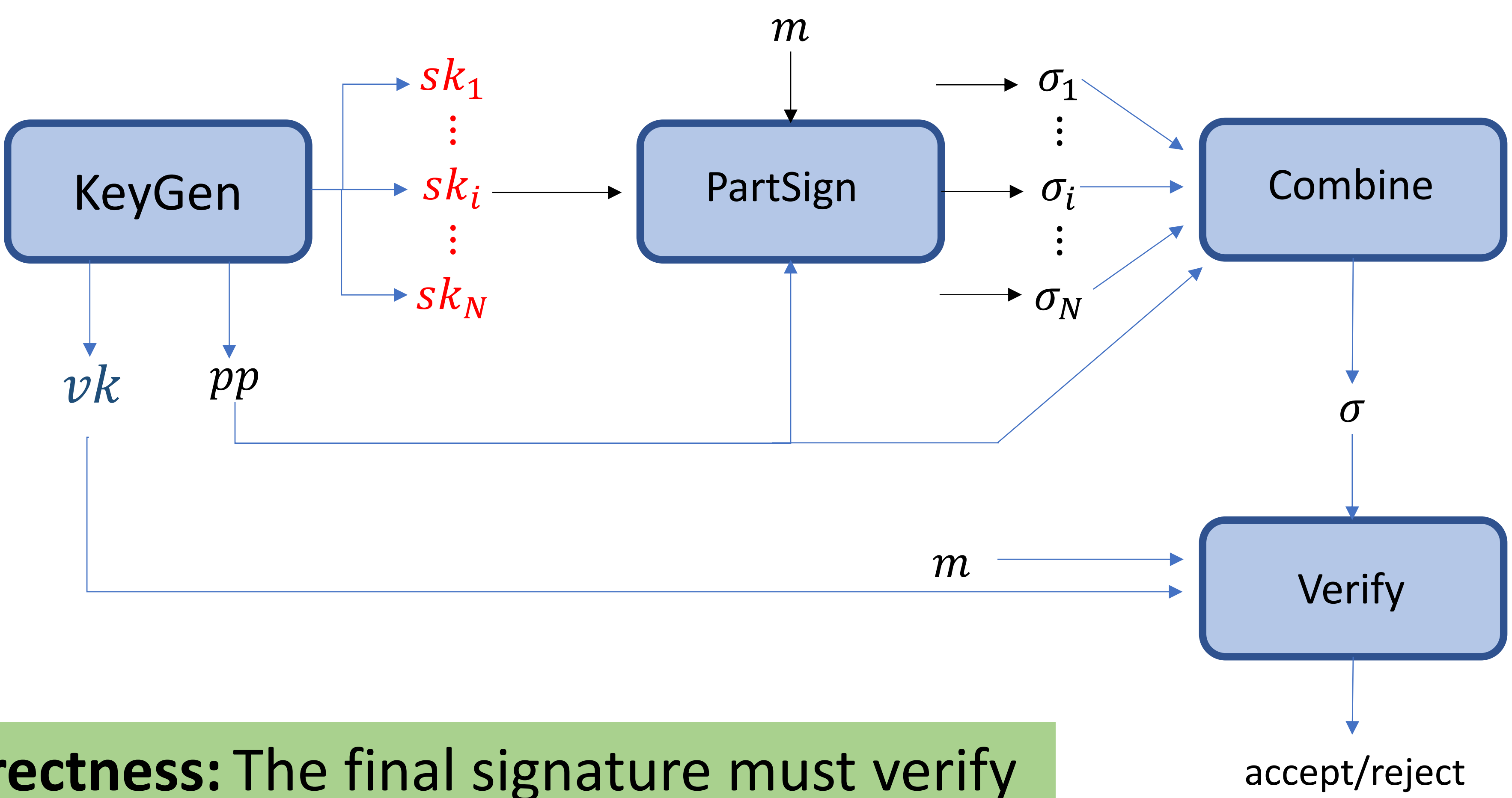
Threshold Signature Definition [BGG+18]



Threshold Signature Definition [BGG+18]



Threshold Signature Definition [BGG+18]



Correctness: The final signature must verify

Threshold Signature Security

No polynomial time adversary must be able to generate a signature on any message m^* even given

Threshold Signature Security

No polynomial time adversary must be able to generate a signature on any message m^* even given

✓ Partial signing keys from upto $t-1$ parties of adversary's choice

Threshold Signature Security

No polynomial time adversary must be able to generate a signature on any message m^* even given

- ✓ Partial signing keys from upto $t - 1$ parties of adversary's choice
- ✓ Partial/complete signatures on any number of other messages of adversary's choice

Threshold Signature Security

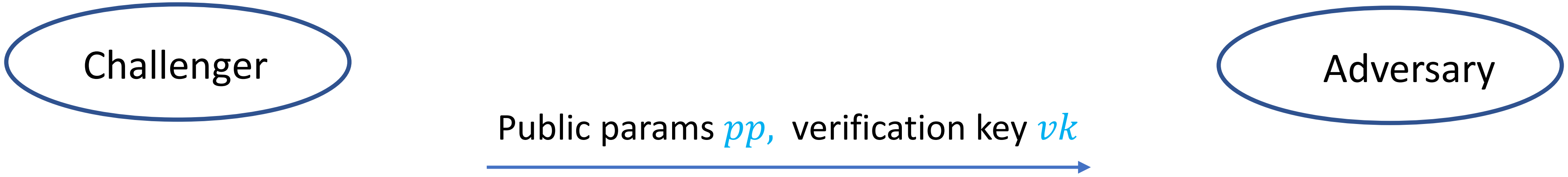
t-out-of-N access structure

Challenger

Adversary

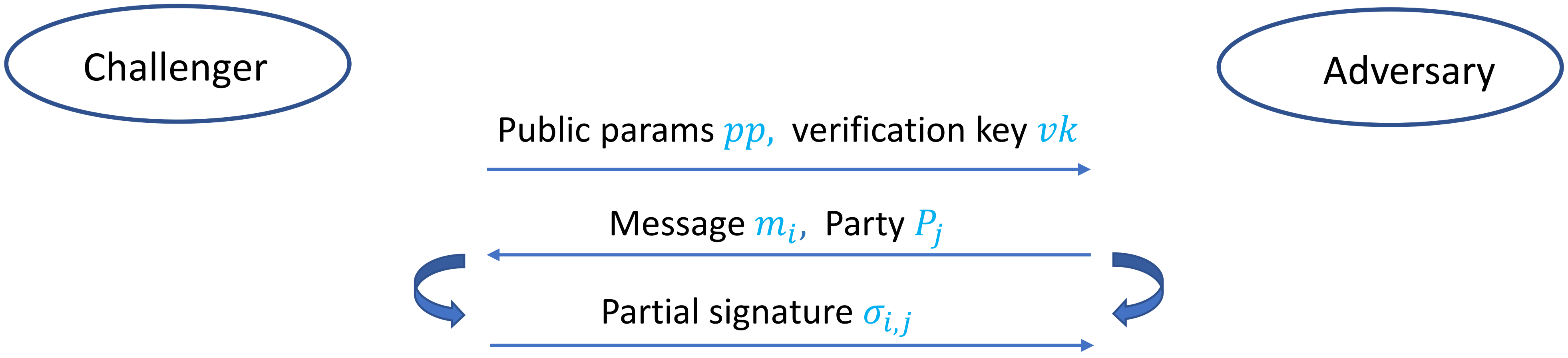
Threshold Signature Security

t-out-of-N access structure



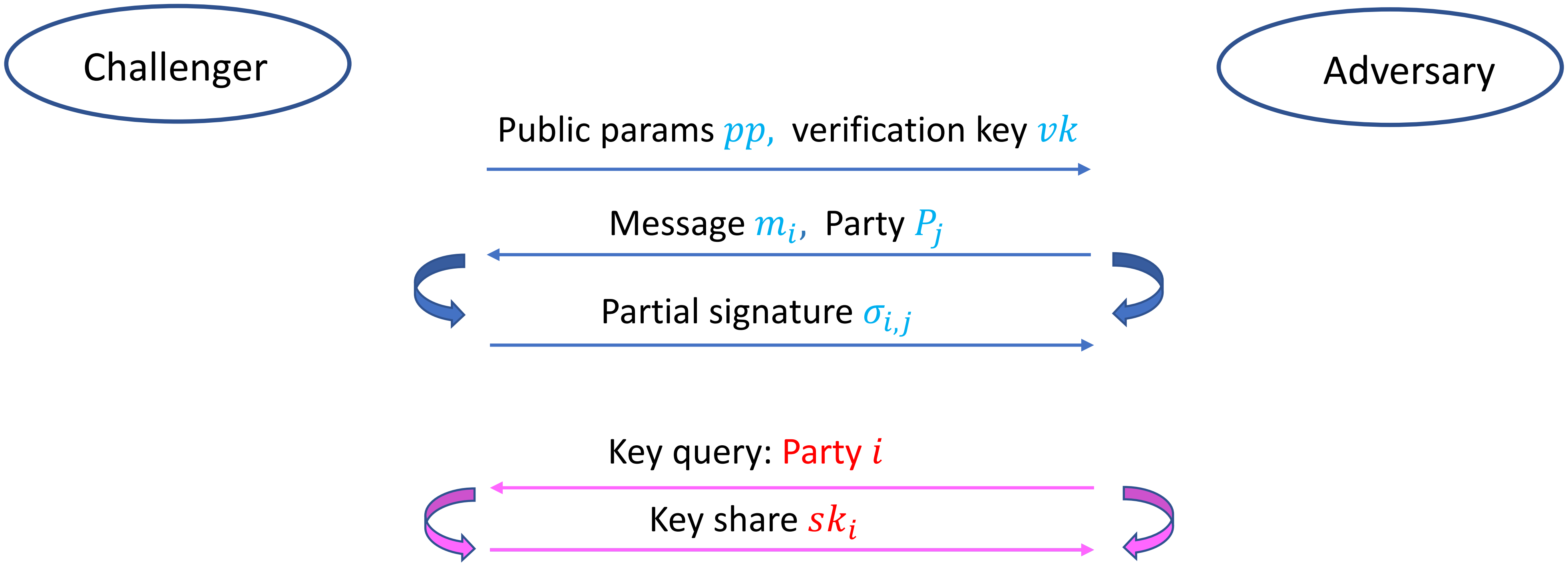
Threshold Signature Security

t-out-of-N access structure



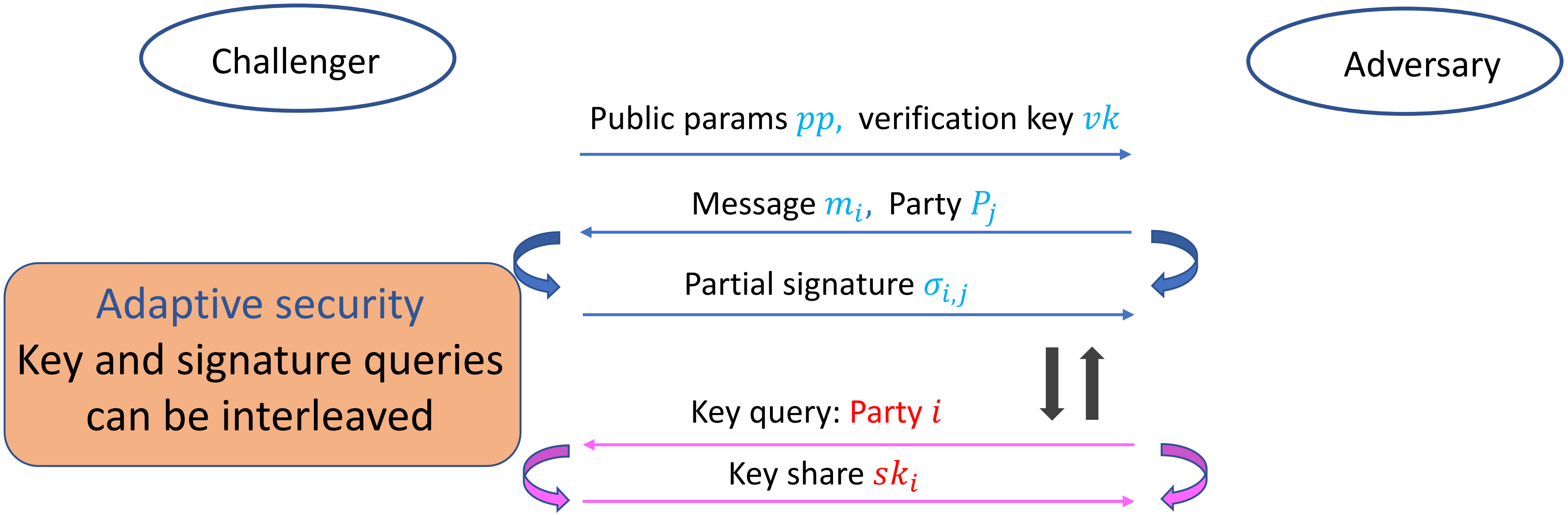
Threshold Signature Security

t-out-of-N access structure



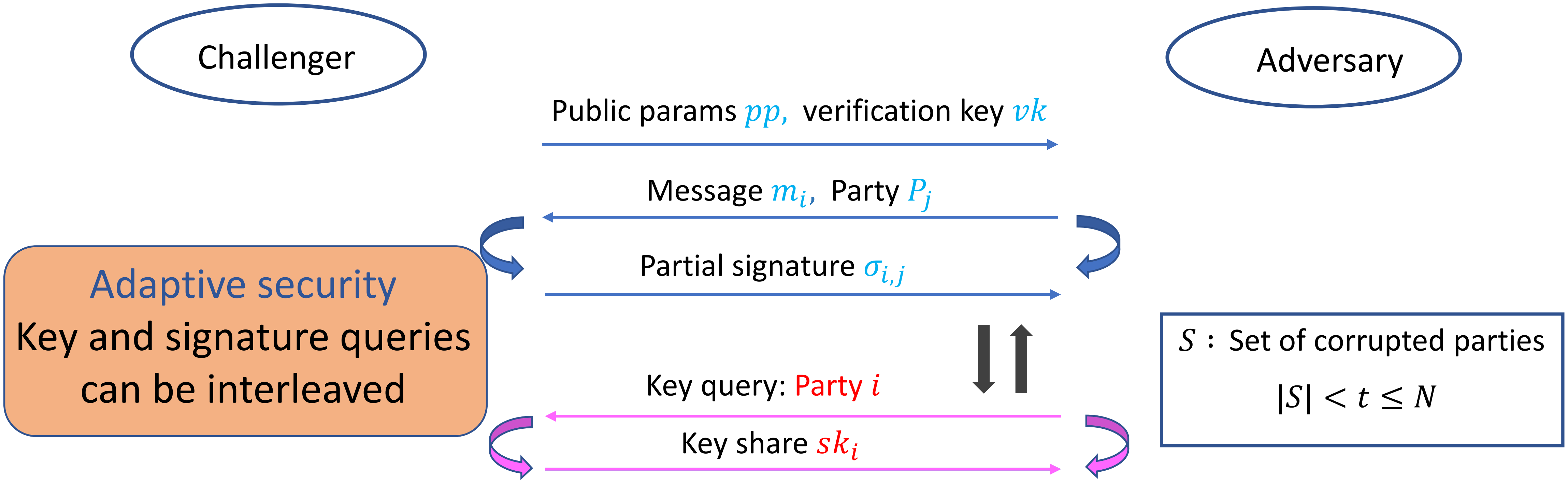
Threshold Signature Security

t-out-of-N access structure



Threshold Signature Security

t-out-of-N access structure



Threshold Signature Security

t-out-of-N access structure

Challenger

Adversary

Public params pp, verification key vk →

← Message m_i, Party P_j

Partial signature $\sigma_{i,j}$ →

Adaptive security
Key and signature queries can be interleaved

← Key query: Party i

Key share sk_i →

S : Set of corrupted parties

$|S| < t \leq N$

← Message-signature (m^*, σ^*)

Threshold Signature Security

t-out-of-N access structure

Challenger

Adversary

Public params pp, verification key vk

Message m_i, Party P_j

Partial signature $\sigma_{i,j}$

Key query: Party i

Key share sk_i

Message-signature (m^*, σ^*)

Adaptive security
Key and signature queries can be interleaved

S : Set of corrupted parties

$|S| < t \leq N$

The adversary wins if σ^* is a valid signature on m^* and a signature on m^* cannot be trivially computed

Threshold Signature Security

t-out-of-N access structure

Challenger

Adversary

Generally hard to achieve

Message m_i, Party P_j

Partial signature $\sigma_{i,j}$

Adaptive security

Key and signature queries can be interleaved

S : Set of corrupted parties

$|S| < t \leq N$

Key query: Party i

Key share sk_i

Message-signature (m^*, σ^*)

The adversary wins if σ^* is a valid signature on m^* and a signature on m^* cannot be trivially computed

Threshold Signature Security

t-out-of-N access structure

Challenger

Adversary

Generally hard to achieve

Message m_i, Party P_j

Partial signature $\sigma_{i,j}$

Adaptive security
Key and signature queries can be interleaved

Key query: Party i

Key share sk_i

S : Set of corrupted parties

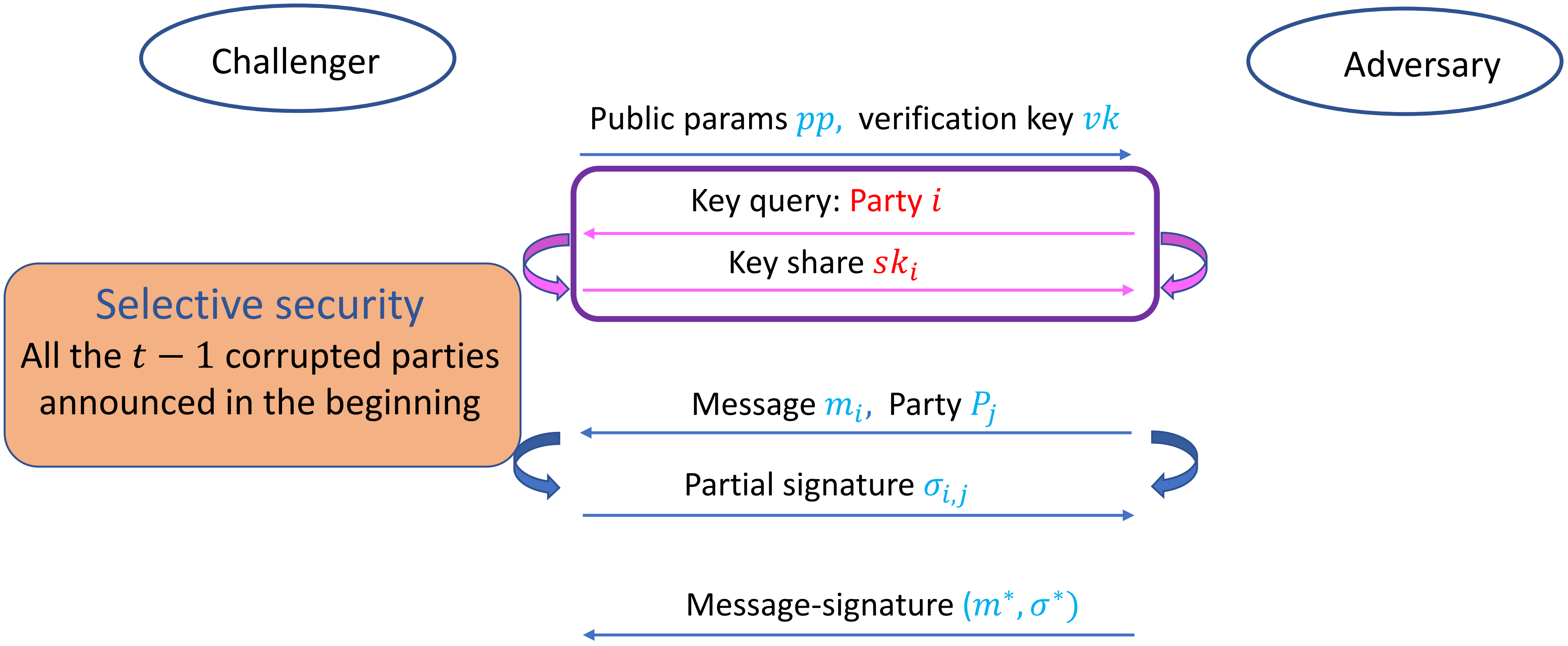
$|S| < t \leq N$

Message-signature (m^*, σ^*)

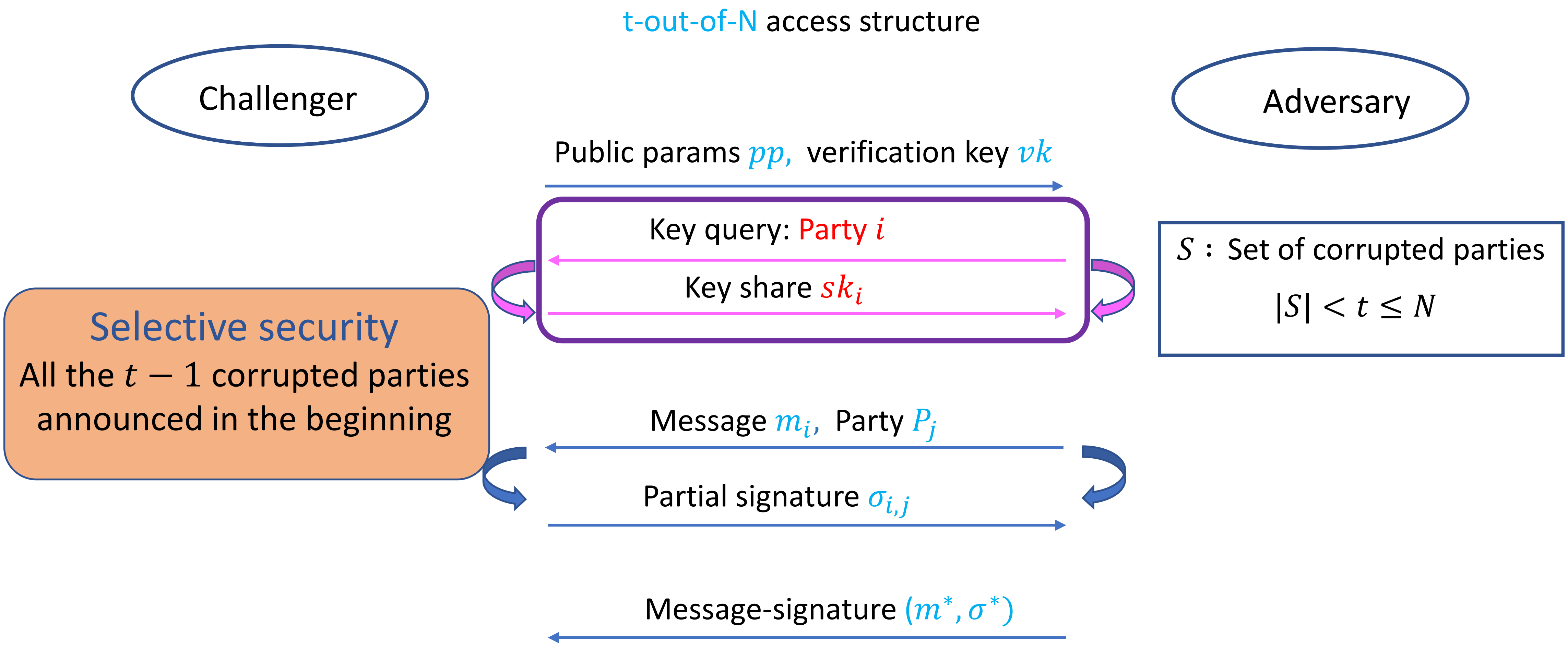
The adversary wins if σ^* is a valid signature on m^* and a signature on m^* cannot be trivially computed

Threshold Signature Security

t-out-of-N access structure



Threshold Signature Security



Threshold Signature Security

t-out-of-N access structure

Challenger

Adversary

Public params pp, verification key vk →

Key query: Party i ←

Key share sk_i →

S : Set of corrupted parties

$|S| < t \leq N$

Selective security
All the $t - 1$ corrupted parties announced in the beginning

Message m_i, Party P_j ←

Partial signature $\sigma_{i,j}$ →

Message-signature (m^*, σ^*) ←

The adversary wins if

- σ^* is a valid signature on m^* and
- no partial signature on m^* was queried

Threshold Signature Security

t-out-of-N access structure

Challenger

Adversary

Public params pp, verification key vk

Key query: Party i

Key share sk_i

S : Set of corrupted parties

$|S| < t \leq N$

Selective security

All the $t-1$ corrupted parties announced in the beginning

Enforces strict selectivity in choosing the $t-1$ parties

Message m_i, Party P_j

Partial signature $\sigma_{i,j}$

Message-signature (m^*, σ^*)

The adversary wins if

- σ^* is a valid signature on m^* and
- no partial signature on m^* was queried

Threshold Signature Security Definitions

Adaptive key queries

Part Sign on m^* ✓

At least as strong (adaptivity) as

Selective key queries [BGG+18]
– all $t-1$ key queries in the beginning of the game

Part Sign on m^* ✗

At least as strong (adaptive) as

Threshold Signature Security Definitions

Adaptive key queries

Part Sign on m^* ✓

Partially adaptive key queries [AS**Y**22]
– all $t - 1$ key queries in the middle of the game (but all at once)

Part Sign on m^* ✗

Selective key queries [BGG+18]
– all $t - 1$ key queries in the beginning of the game

Part Sign on m^* ✗

At least as strong (adaptivity) as

At least as strong (adaptive) as

Threshold Signature Security Definitions

Adaptive key queries

Part Sign on m^* ✓

Partially adaptive key queries [ASY22]
– all $t-1$ key queries in the middle of the game (but all at once)

Part Sign on m^* ✗

Selective key queries [BGG+18]
– all $t-1$ key queries in the beginning of the game

Part Sign on m^* ✗

Selective key queries – any key query(ies) in the beginning (defined in [BCK+22])

Part Sign on m^* ✓

At least as strong (adaptivity) as

At least as strong (adaptive) as

Threshold Signature Security Definitions

Adaptive key queries

Part Sign on m^* ✓

Partially adaptive key queries [ASY22]
– all $t - 1$ key queries in the middle of the game (but all at once)

Part Sign on m^* ✗

Partially adaptive key queries
– any key query(ies) in the middle of the game (but all at once)

Part Sign on m^* ✓

Selective key queries [BGG+18]
– all $t - 1$ key queries in the beginning of the game

Part Sign on m^* ✗

Selective key queries – any key query(ies) in the beginning (defined in [BCK+22])

Part Sign on m^* ✓

At least as strong (adaptivity) as

At least as strong (adaptive) as

Threshold Signature Security Definitions

Adaptive key queries

Part Sign on m^* ✗

Partially adaptive key queries [ASY22]
– all $t-1$ key queries in the middle of the game (but all at once)

Part Sign on m^* ✗

Selective key queries [BGG+18]
– all $t-1$ key queries in the beginning of the game

Part Sign on m^* ✗

Adaptive key queries

Part Sign on m^* ✓

Partially adaptive key queries
– any key query(ies) in the middle of the game (but all at once)

Part Sign on m^* ✓

Selective key queries – any key query(ies) in the beginning (defined in [BCK+22])

Part Sign on m^* ✓

At least as strong (adaptivity) as

At least as strong (adaptive) as

Threshold Signature Security Definitions

Adaptive key queries

Part Sign on m^* ✗

Loss of factor Q (# of signing queries)

Adaptive key queries

Part Sign on m^* ✓

Partially adaptive key queries [ASY22]
– all $t-1$ key queries in the middle of the game (but all at once)

Part Sign on m^* ✗

Partially adaptive key queries
– any key query(ies) in the middle of the game (but all at once)

Part Sign on m^* ✓

Selective key queries [BGG+18]
– all $t-1$ key queries in the beginning of the game

Part Sign on m^* ✗

Selective key queries – any key query(ies) in the beginning (defined in [BCK+22])

Part Sign on m^* ✓

At least as strong (adaptivity) as

At least as strong (adaptive) as

Threshold Signature Security Definitions

At least as strong (adaptivity) as

Adaptive key queries

Part Sign on m^* ✗

Loss of factor Q (# of signing queries)

Adaptive key queries

Part Sign on m^* ✓

Partially adaptive key queries [AS**Y**22]
– all $t-1$ key queries in the middle of the game (but all at once)

Part Sign on m^* ✗

Partially adaptive key queries
– any key query(ies) in the middle of the game (but all at once)

Part Sign on m^* ✓

Selective key queries [BGG+18]
– all $t-1$ key queries in the beginning of the game

Part Sign on m^* ✗

Selective key queries – any key query(ies) in the beginning (defined in [BCK+22])

Part Sign on m^* ✓

At least as strong (adaptive) as

Threshold Signature Security Definitions

At least as strong (adaptivity) as

Adaptive key queries

Part Sign on m^* ✗

Loss of factor Q (# of signing queries)

Adaptive key queries

Part Sign on m^* ✓

Partially adaptive key queries [ASY22]
– all $t-1$ key queries in the middle of the game (but all at once)

Part Sign on m^* ✗

Partially adaptive key queries
– any key query(ies) in the middle of the game (but all at once)

Part Sign on m^* ✓

Selective key queries [BGG+18]
– all $t-1$ key queries in the beginning of the game

Part Sign on m^* ✗

Selective key queries – any key query(ies) in the beginning (defined in [BCK+22])

Part Sign on m^* ✓

At least as strong (adaptive) as

Our Construction

Our Construction

Key Homomorphic
PRF (KHPRF)

+

BGG+18

Our Construction

$F(K_1, x) + F(K_2, x)$
$= F(K_1 + K_2, x)$

Obs: $F(0, x) = 0$

Key Homomorphic
PRF (KHPRF)

+

BGG+18

[BGG+18] Construction of Threshold Signatures

Building Blocks

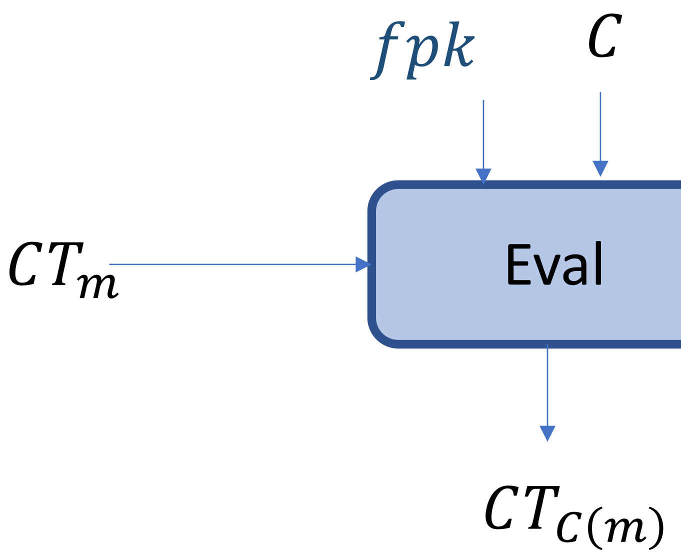
Standard signature scheme $(keys\text{: } sigvk, sigsk)$

FHE scheme $(keys\text{: } fpk, fsk)$

A Linear secret sharing scheme

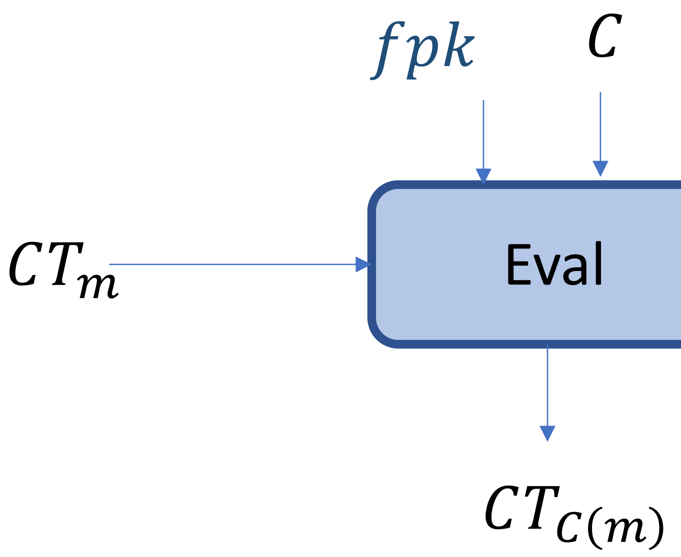
Fully Homomorphic Encryption

Same as public key encryption scheme with added functionality



Fully Homomorphic Encryption

Same as public key encryption scheme with added functionality

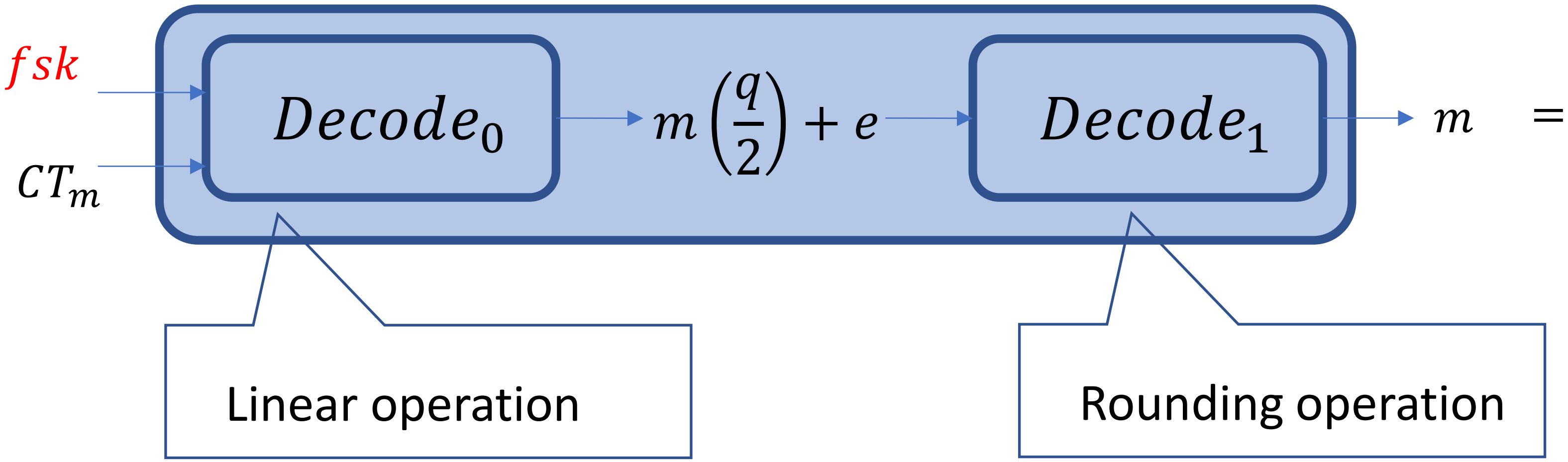


Public operation
Deterministic

Special Fully Homomorphic Encryption

Secret key fsk is a vector

Decrypt



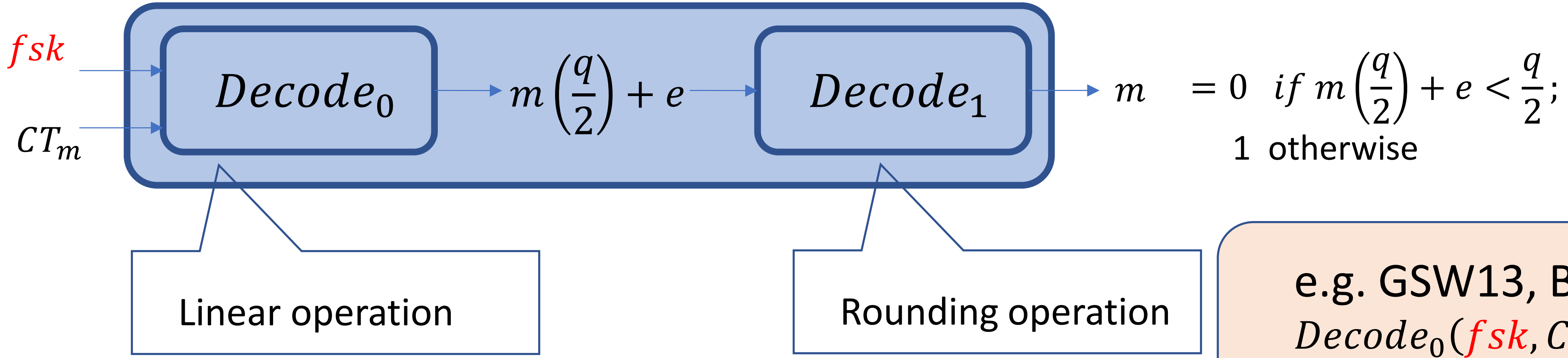
$= 0 \;\; if\ m\left(\frac{q}{2}\right) + e < \frac{q}{2};$

1 otherwise

Special Fully Homomorphic Encryption

Secret key fsk is a vector

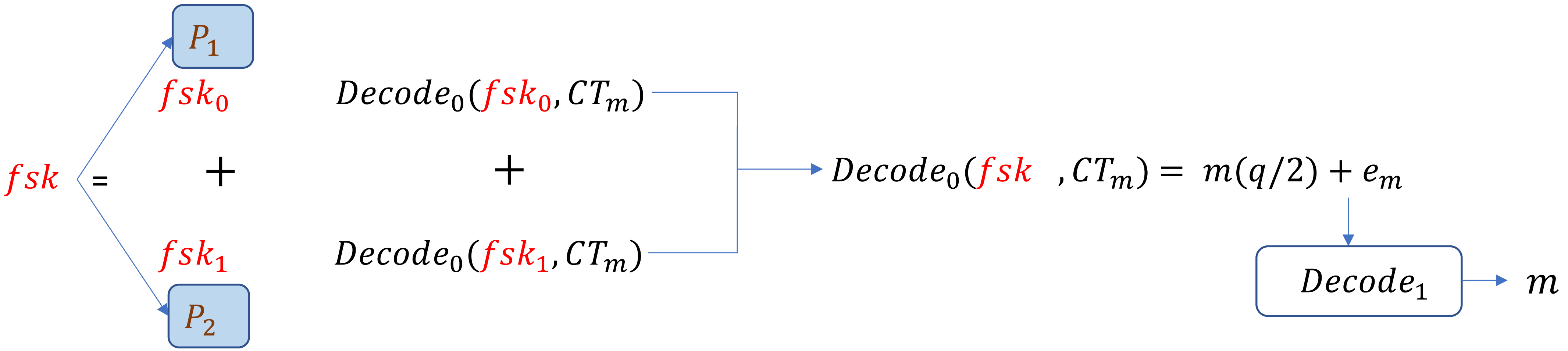
Decrypt



e.g. GSW13, BV11
$Decode_0(fsk, CT_m)$
$= \langle fsk, CT_m \rangle$

Usefulness of Linearity of $Decode_0$

Usefulness of Linearity of $Decode_0$



Construction Overview [BGG+18]

for 2-out-of-2 scheme

Key shares

PartSign(m)

Combine

Public params (pp)

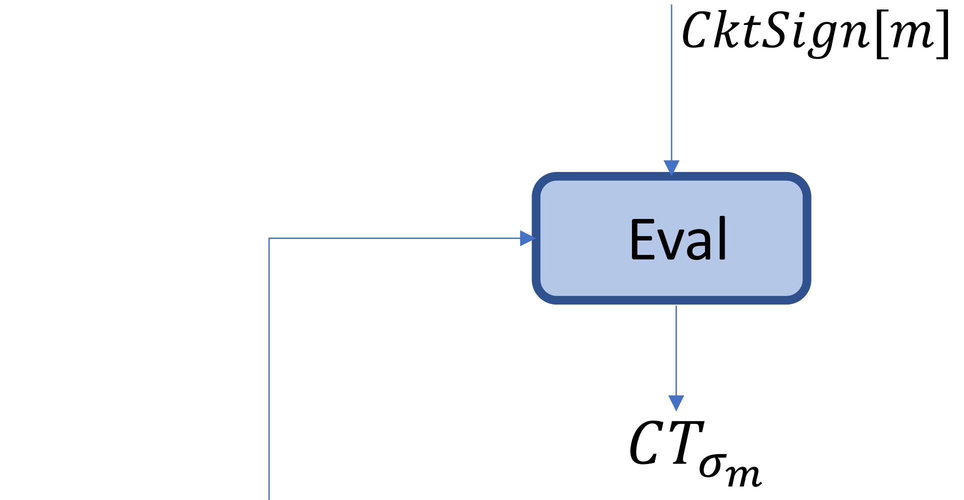
Construction Overview [BGG+18]

for 2-out-of-2 scheme

Key shares

PartSign(m)

Combine

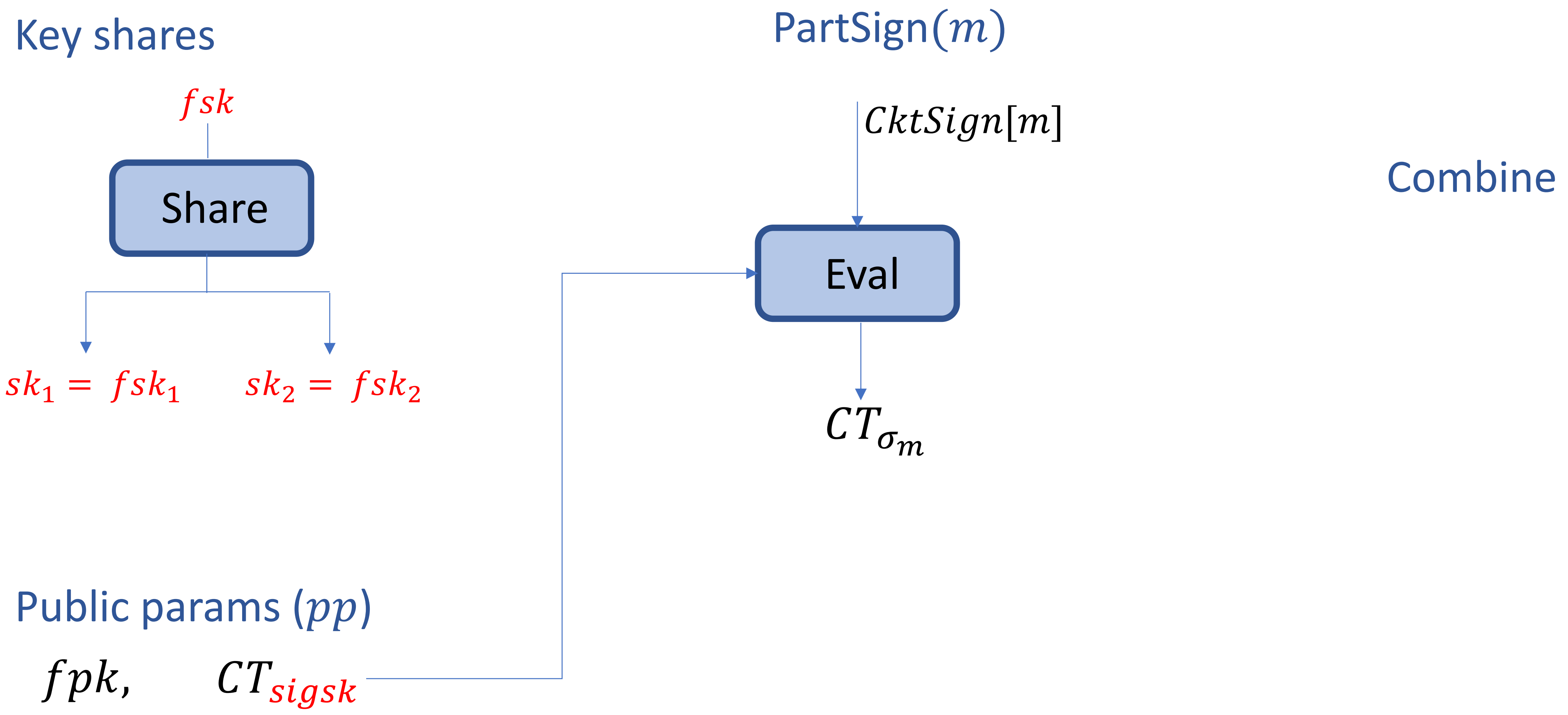


Public params (pp)

$fpk,\quad CT_{sigsk}$

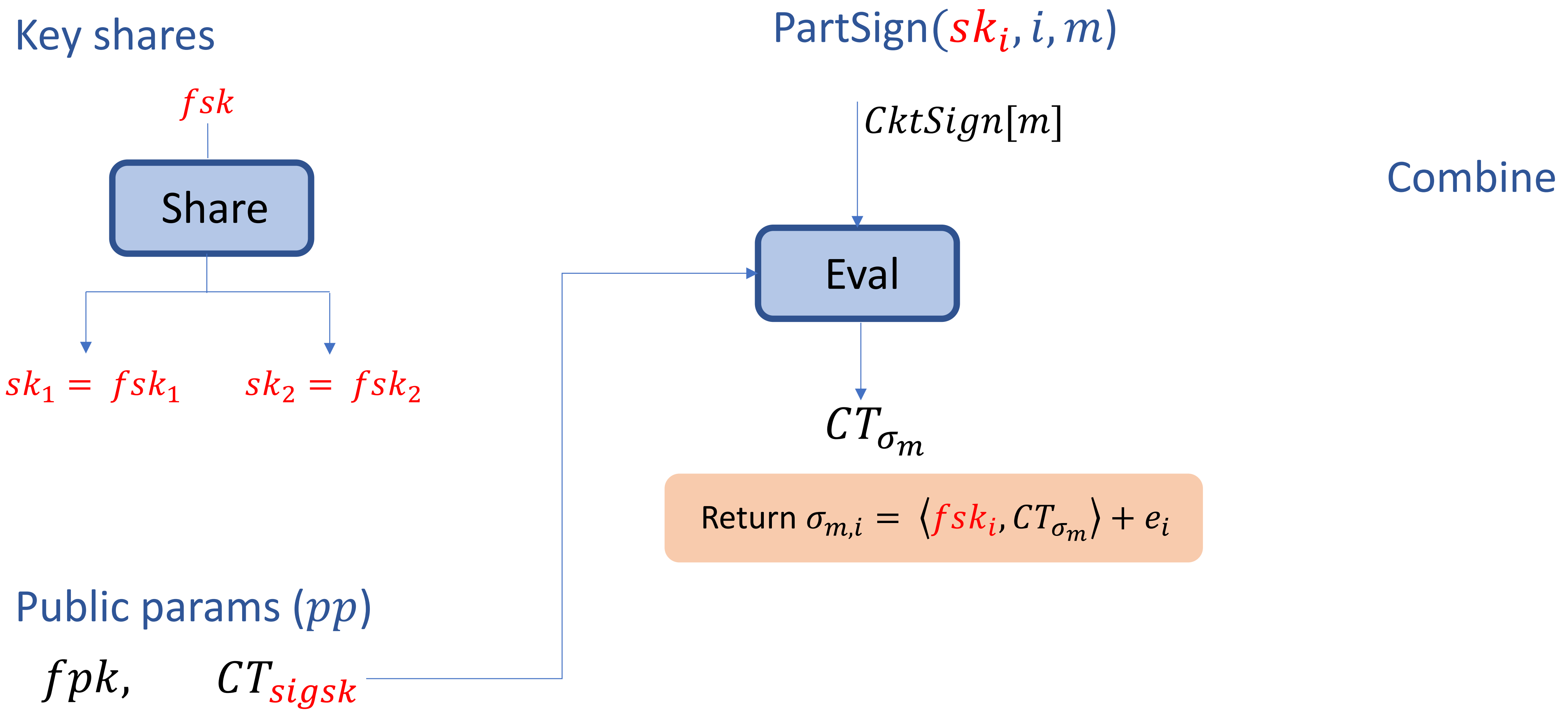
Construction Overview [BGG+18]

for 2-out-of-2 scheme



Construction Overview [BGG+18]

for 2-out-of-2 scheme



Construction Overview [BGG+18]

for 2-out-of-2 scheme

Key shares

fsk

Share

$sk_1 = fsk_1$ $sk_2 = fsk_2$

Public params (pp)

$fpk,$ CT_{sigsk}

PartSign(sk_i, i, m)

$CktSign[m]$

Eval

CT_{σ_m}

Return $\sigma_{m,i} = \langle fsk_i, CT_{\sigma_m} \rangle + e_i$

Combine

$$\sigma_m = \text{round}(\sigma_{m,1} + \sigma_{m,2})$$

Security Sketch

(Selective Security, No PartSign on m^*)

Let the adversary gets partial signing key from P1

Security Sketch

(Selective Security, No PartSign on m^*)

Let the adversary gets partial signing key from P1

H0:
The honest game

$mpk = CT_{sigsk}$; $\sigma_{m,2} = \langle fsk_2, CT_{\sigma_m} \rangle$

Security Sketch

(Selective Security, No PartSign on m^*)

Let the adversary gets partial signing key from P1

H0: $mpk = CT_{sigsk};$ $\sigma_{m,2} = \langle fsk_2, CT_{\sigma_m} \rangle$

The honest game

H1: $mpk = CT_{sigsk};$ $\sigma_{m,2} = \lfloor q/2 \rceil \sigma_m - \langle fsk_1, CT_{\sigma_m} \rangle$

Security Sketch

(Selective Security, No PartSign on m^*)

Let the adversary gets partial signing key from P1

H0: $mpk = CT_{sigsk}$; $\sigma_{m,2} = \langle fsk_2, CT_{\sigma_m} \rangle$

The honest game

H1: $mpk = CT_{sigsk}$; $\sigma_{m,2} = \lfloor q/2 \rceil \sigma_m - \langle fsk_1, CT_{\sigma_m} \rangle$

H2: $mpk = CT_0$; $\sigma_{m,2} = \lfloor q/2 \rceil \sigma_m - \langle fsk_1, CT_{\sigma_m} \rangle$

Security Sketch

(Selective Security, No PartSign on m^*)

Let the adversary gets partial signing key from P1

H0: The honest game $mpk = CT_{sigsk}$; $\sigma_{m,2} = \langle fsk_2, CT_{\sigma_m} \rangle$

H1: $mpk = CT_{sigsk}$; $\sigma_{m,2} = \lfloor q/2 \rceil \sigma_m - \langle fsk_1, CT_{\sigma_m} \rangle$

H2: $mpk = CT_0$; $\sigma_{m,2} = \lfloor q/2 \rceil \sigma_m - \langle fsk_1, CT_{\sigma_m} \rangle$

Reduction to Sign Security in H2

Security Sketch – Reduction to Sign Security in H2

(Selective Security, No PartSign on m^*)

Let the adversary gets partial signing key from P1

Security Sketch – Reduction to Sign Security in H2

(Selective Security, No PartSign on m^*)

Let the adversary gets partial signing key from P1

Sign-Challenger | Reduction | TSig-Adversary

Sign-Challenger → Reduction: svk

Reduction:

Sample (fpk, fsk); discard fsk

Sample fsk_1 randomly

Compute CT_0

Reduction → TSig-Adversary: svk, CT_0, fsk_1

TSig-Adversary → Reduction: PartSign Query: $(m, 2)$

Reduction → Sign-Challenger: Sign query: m

Sign-Challenger → Reduction: σ_m

Reduction:

Compute CT_{σ_m} using HE.Eval with CT_0

Compute $\sigma_{m,2} = \lfloor q/2 \rceil \sigma_m - \langle fsk_1, CT_{\sigma_m} \rangle$

Reduction → TSig-Adversary: $\sigma_{m,2}$

TSig-Adversary → Reduction: m^*, σ^*

Reduction → Sign-Challenger: m^*, σ^*

Security Sketch – Reduction to Sign Security in H2

(Selective Security, No PartSign on m^*)

Let the adversary gets partial signing key from P1

Sign-Challenger | Reduction | TSig-Adversary

Sign-Challenger → Reduction: svk

Reduction:

Sample (fpk, fsk); discard fsk

Sample fsk_1 randomly

Compute CT_0

Reduction → TSig-Adversary: svk, CT_0, fsk_1

TSig-Adversary → Reduction: PartSign Query: $(m, 2)$

Reduction → Sign-Challenger: Sign query: m

Sign-Challenger → Reduction: σ_m

Reduction:

Compute CT_{σ_m} using HE.Eval with CT_0

Compute $\sigma_{m,2} = \lfloor q/2 \rceil \sigma_m - \langle fsk_1, CT_{\sigma_m} \rangle$

Reduction → TSig-Adversary: $\sigma_{m,2}$

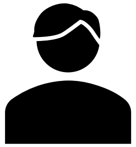
TSig-Adversary → Reduction: m^*, σ^*

Reduction → Sign-Challenger: m^*, σ^*

Challenges in Proving Stronger Security

Selective Key Query, PartSign on m^* allowed

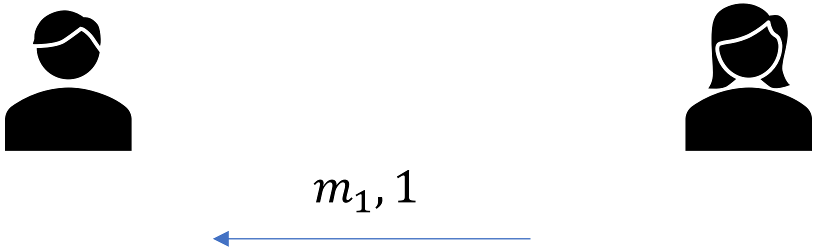
Let us consider 2-out-of-2 scheme. The adversary does not issue any key query



Challenges in Proving Stronger Security

Selective Key Query, PartSign on m^* allowed

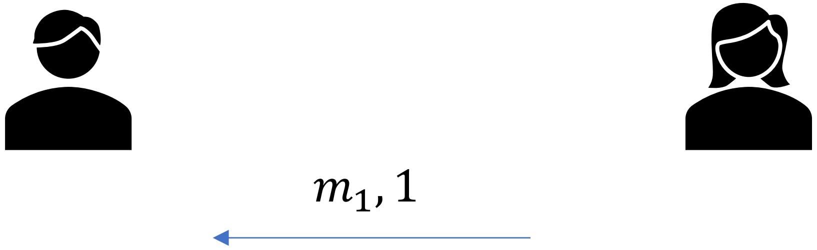
Let us consider 2-out-of-2 scheme. The adversary does not issue any key query



Challenges in Proving Stronger Security

Selective Key Query, PartSign on m^* allowed

Let us consider 2-out-of-2 scheme. The adversary does not issue any key query



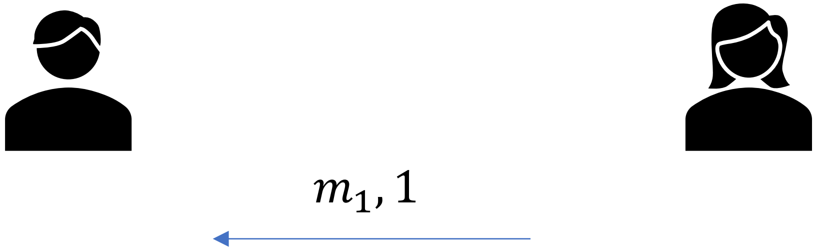
How to compute $\sigma_{m_1,1}$?

- Using fsk_1 as $\langle fsk_1, CT_{\sigma_{m_1}} \rangle$ or
- Simulate using fsk_2 and σ_{m_1}

Challenges in Proving Stronger Security

Selective Key Query, PartSign on m^* allowed

Let us consider 2-out-of-2 scheme. The adversary does not issue any key query



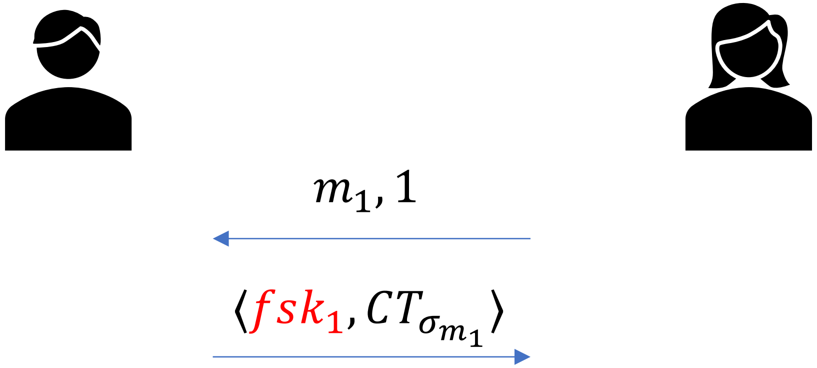
How to compute $\sigma_{m_1,1}$?

- **Using fsk_1 as $\langle fsk_1, CT_{\sigma_{m_1}} \rangle$** or
- Simulate using fsk_2 and σ_{m_1}

Challenges in Proving Stronger Security

Selective Key Query, PartSign on m^* allowed

Let us consider 2-out-of-2 scheme. The adversary does not issue any key query



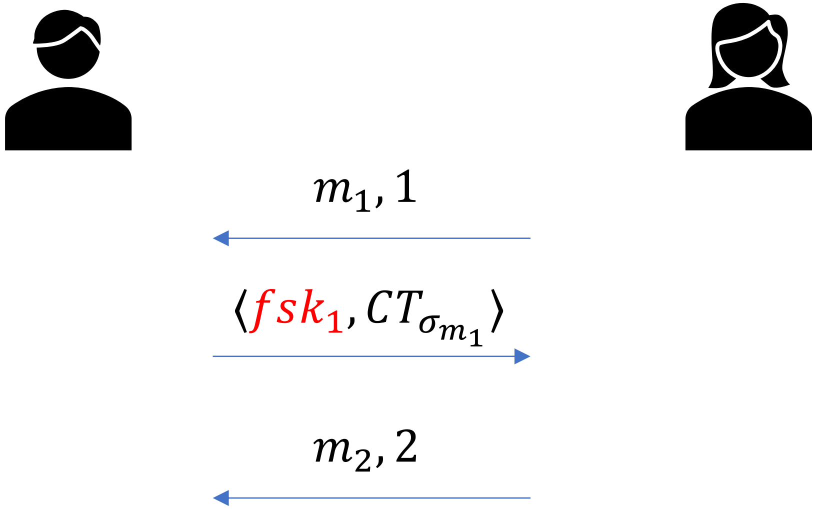
How to compute $\sigma_{m_1,1}$?

- **Using fsk_1 as $\langle fsk_1, CT_{\sigma_{m_1}} \rangle$** or
- Simulate using fsk_2 and σ_{m_1}

Challenges in Proving Stronger Security

Selective Key Query, PartSign on $\boldsymbol{m}^*$ allowed

Let us consider 2-out-of-2 scheme. The adversary does not issue any key query



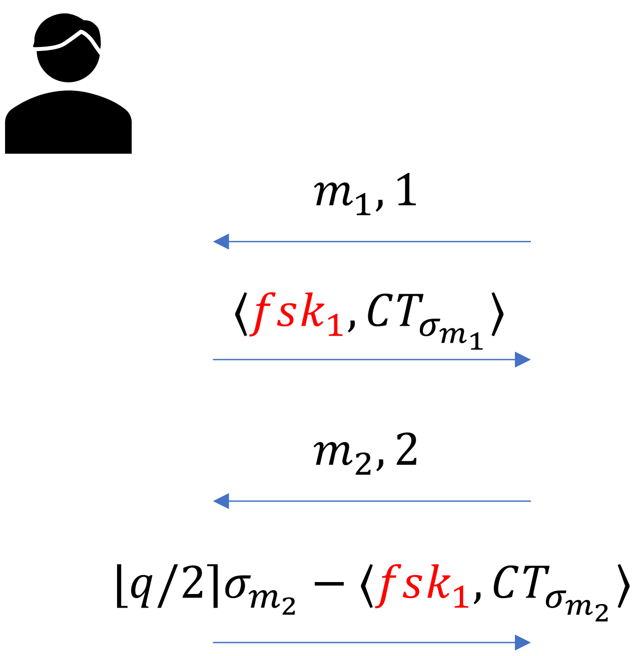
How to compute $\sigma_{m_1,1}$?

- **Using $\boldsymbol{fsk_1}$ as $\langle \boldsymbol{fsk_1}, \boldsymbol{CT_{\sigma_{m_1}}} \rangle$** or
- Simulate using fsk_2 and σ_{m_1}

Challenges in Proving Stronger Security

Selective Key Query, PartSign on m^* allowed

Let us consider 2-out-of-2 scheme. The adversary does not issue any key query



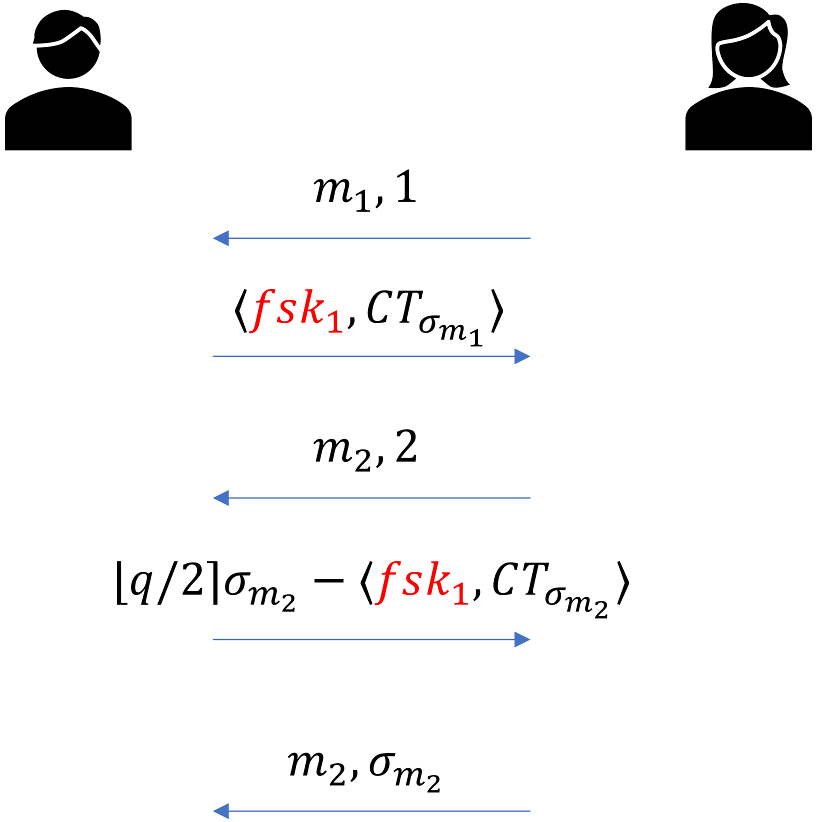
How to compute $\sigma_{m_1,1}$?

- **Using fsk_1 as $\langle fsk_1, CT_{\sigma_{m_1}} \rangle$** or
- Simulate using fsk_2 and σ_{m_1}

Challenges in Proving Stronger Security

Selective Key Query, PartSign on m^* allowed

Let us consider 2-out-of-2 scheme. The adversary does not issue any key query



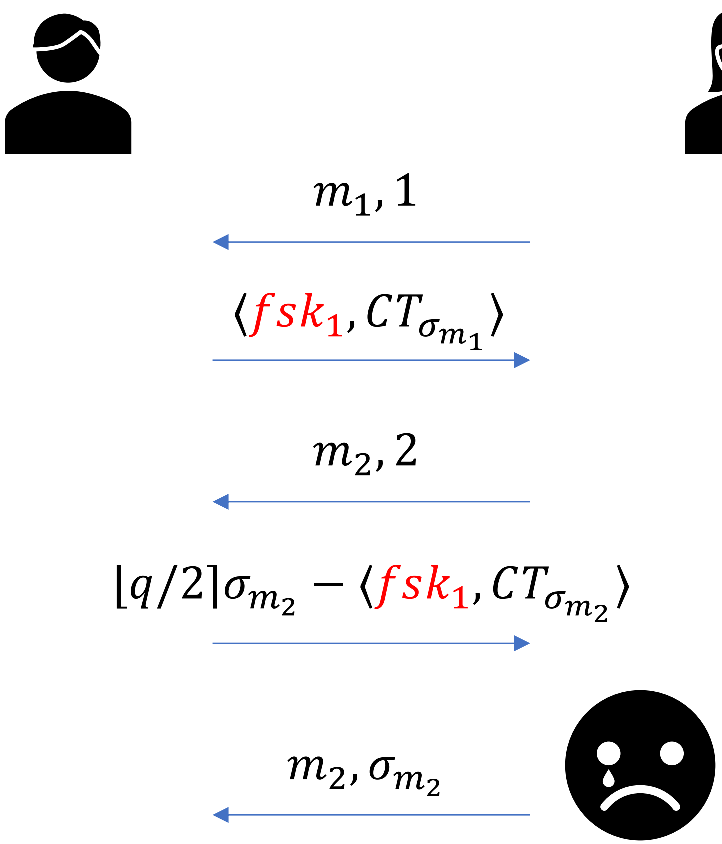
How to compute $\sigma_{m_1,1}$?

- **Using fsk_1 as $\langle fsk_1, CT_{\sigma_{m_1}} \rangle$** or
- Simulate using fsk_2 and σ_{m_1}

Challenges in Proving Stronger Security

Selective Key Query, PartSign on $\boldsymbol{m}^*$ allowed

Let us consider 2-out-of-2 scheme. The adversary does not issue any key query



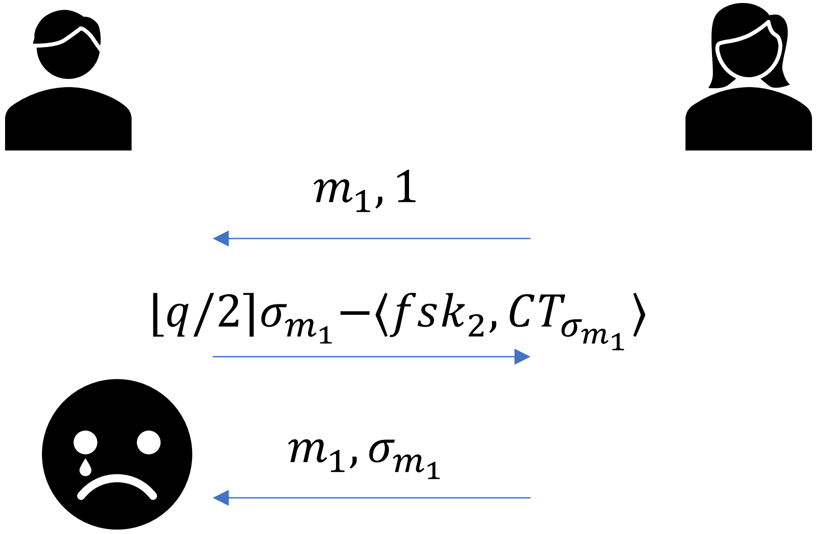
How to compute $\sigma_{m_1,1}$?

- **Using $\boldsymbol{fsk_1}$ as $\langle \boldsymbol{fsk_1}, \boldsymbol{CT_{\sigma_{m_1}}} \rangle$** or
- Simulate using fsk_2 and σ_{m_1}

Challenges in Proving Stronger Security

Selective Key Query, PartSign on $\boldsymbol{m}^*$ allowed

Let us consider 2-out-of-2 scheme. The adversary does not issue any key query



How to compute $\sigma_{m_1,1}$?

- Using fsk_1 as $\langle fsk_1, CT_{\sigma_{m_1}} \rangle$ or
- **Simulate using $\boldsymbol{fsk_2}$ and $\boldsymbol{\sigma_{m_1}}$**

Attempting Different Hybrid Policy

Selective Key Query, PartSign on m^* allowed

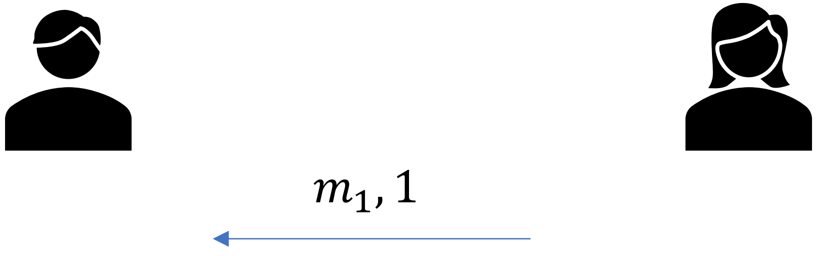
Let us consider 2-out-of-2 scheme. The adversary does not issue any key query

For each message, answer the first PartSign(m, i) query with a randomly sampled $\sigma_{m,i}$

Attempting Different Hybrid Policy

Selective Key Query, PartSign on m^* allowed

Let us consider 2-out-of-2 scheme. The adversary does not issue any key query

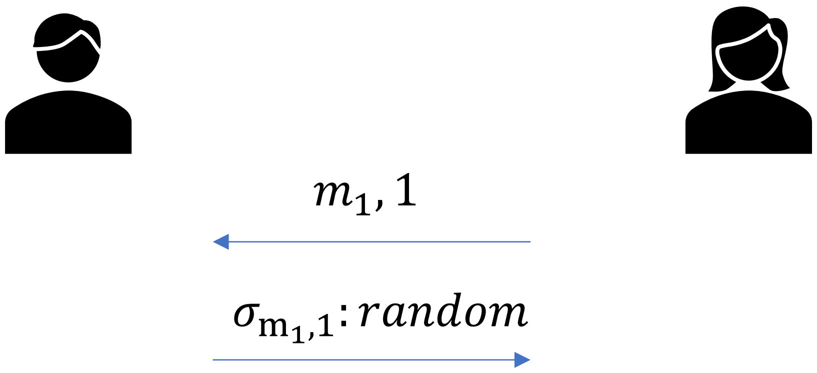


For each message, answer the first PartSign(m, i) query with a randomly sampled $\sigma_{m,i}$

Attempting Different Hybrid Policy

Selective Key Query, PartSign on m^* allowed

Let us consider 2-out-of-2 scheme. The adversary does not issue any key query

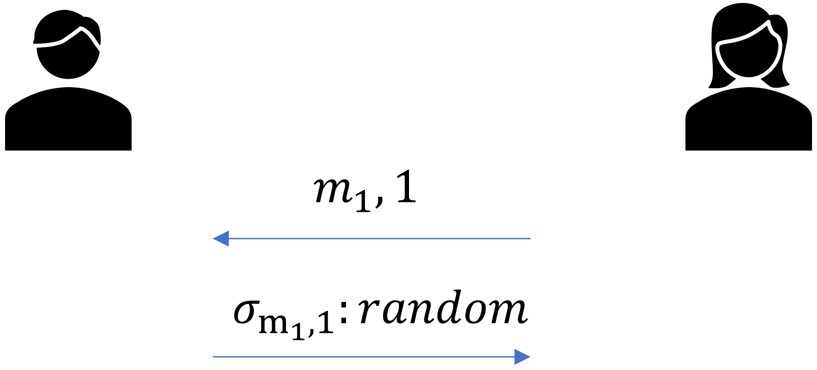


For each message, answer the first PartSign(m, i) query with a randomly sampled $\sigma_{m,i}$

Attempting Different Hybrid Policy

Selective Key Query, PartSign on m^* allowed

Let us consider 2-out-of-2 scheme. The adversary does not issue any key query



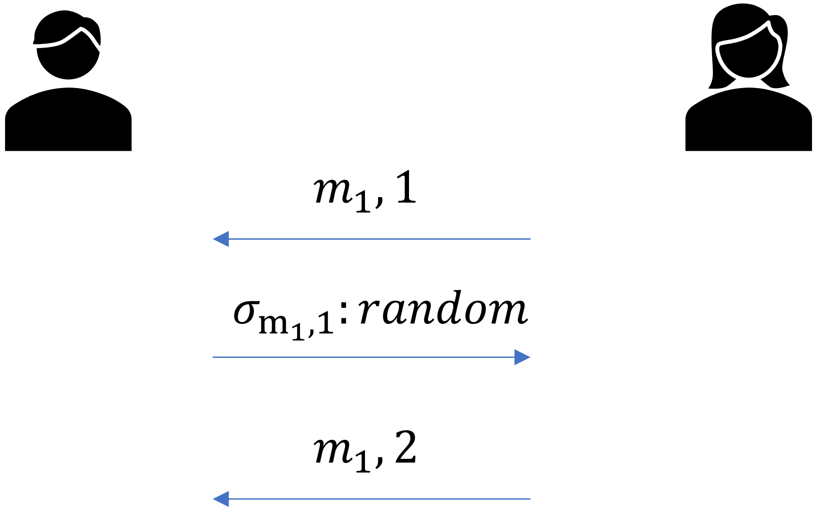
For each message, answer the first PartSign(m, i) query with a randomly sampled $\sigma_{m,i}$

If PartSign$(m, 3 - i)$: compute $\sigma_{m,3-i} = \left\lfloor \frac{q}{2} \right\rceil \sigma_m - \sigma_{m,i}$

Attempting Different Hybrid Policy

Selective Key Query, PartSign on m^* allowed

Let us consider 2-out-of-2 scheme. The adversary does not issue any key query



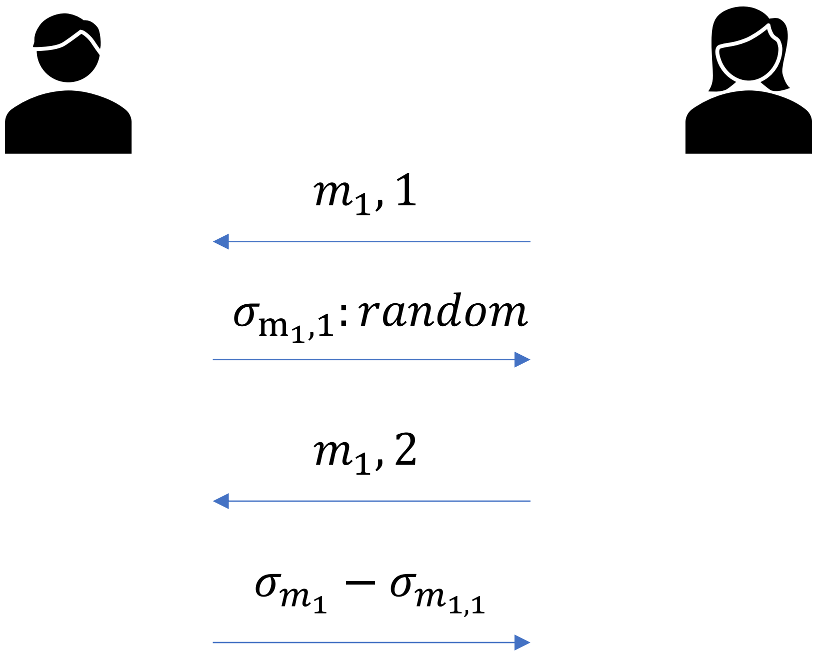
For each message, answer the first PartSign(m, i) query with a randomly sampled $\sigma_{m,i}$

If PartSign$(m, 3-i)$: compute $\sigma_{m,3-i} = \left\lfloor \frac{q}{2} \right\rfloor \sigma_m - \sigma_{m,i}$

Attempting Different Hybrid Policy

Selective Key Query, PartSign on m^* allowed

Let us consider 2-out-of-2 scheme. The adversary does not issue any key query



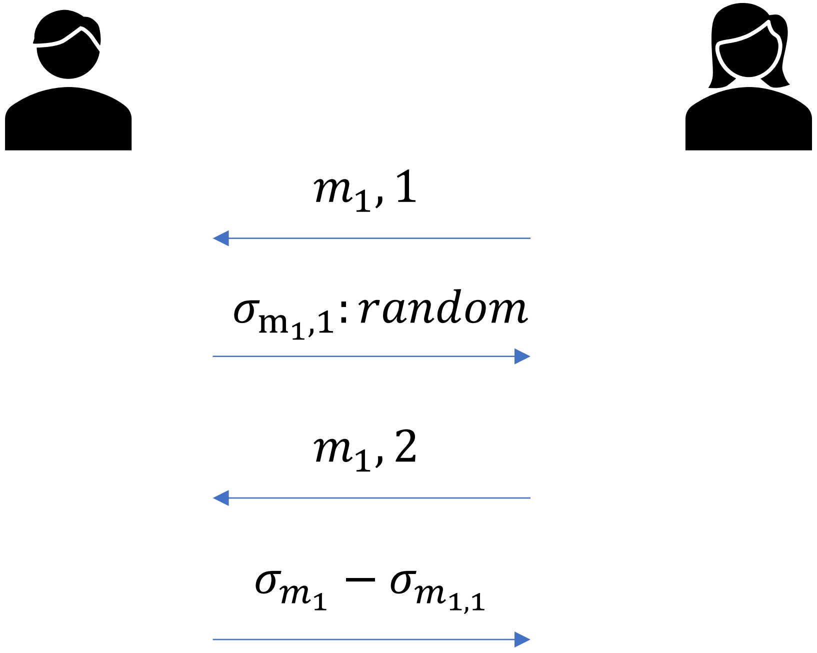
For each message, answer the first PartSign(m, i) query with a randomly sampled $\sigma_{m,i}$

If PartSign$(m, 3-i)$: compute $\sigma_{m,3-i} = \left\lfloor \frac{q}{2} \right\rceil \sigma_m - \sigma_{m,i}$

Attempting Different Hybrid Policy

Selective Key Query, PartSign on m^* allowed

Let us consider 2-out-of-2 scheme. The adversary does not issue any key query



For each message, answer the first PartSign(m, i) query with a randomly sampled $\sigma_{m,i}$

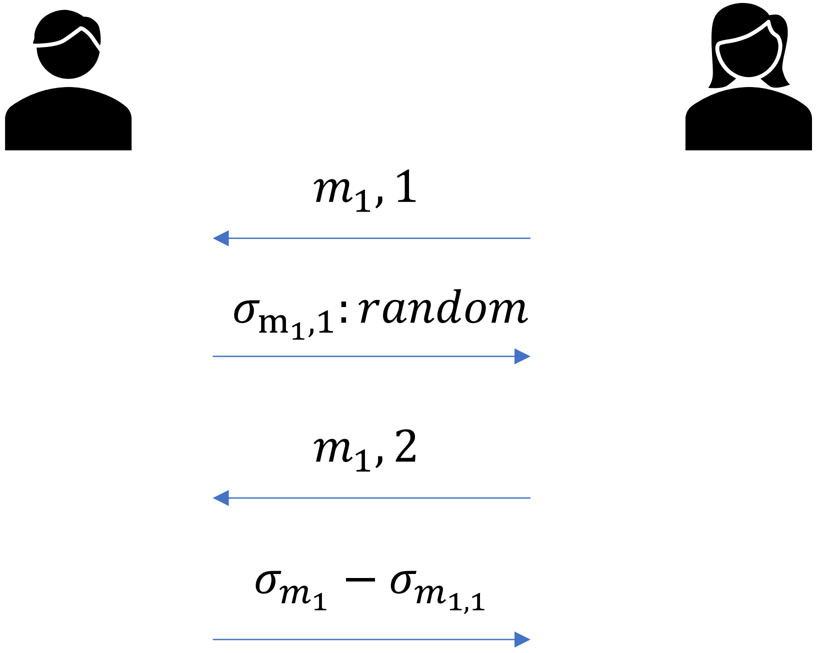
If PartSign$(m, 3-i)$: compute $\sigma_{m,3-i} = \left\lfloor \frac{q}{2} \right\rceil \sigma_m - \sigma_{m,i}$

It is safe to use σ_m now, because the second query on m ensures that m can not be m^*

Attempting Different Hybrid Policy

Selective Key Query, PartSign on m^* allowed

Let us consider 2-out-of-2 scheme. The adversary does not issue any key query



For each message, answer the first PartSign(m, i) query with a randomly sampled $\sigma_{m,i}$

If PartSign$(m, 3-i)$: compute $\sigma_{m,3-i} = \left\lfloor \frac{q}{2} \right\rceil \sigma_m - \sigma_{m,i}$

It is safe to use σ_m now, because the second query on m ensures that m can not be m^*

Problem: Cannot argue the following indistinguishability

$$\langle fsk_i, CT_{\sigma_m} \rangle \approx random$$

Our Solution

PartSign(pp, fsk_1, m)

$$\sigma_{m,1} = Decode_0(fsk_1, CT_{\sigma_m}) + e_1' + \mathrm{r}_{\mathrm{m,1}}$$

Our Solution

PartSign(pp, fsk_1, m)

$$\sigma_{m,1} = Decode_0\left(fsk_1, CT_{\sigma_m}\right) + e_1' + \mathrm{r}_{m,1}$$

PartSign(pp, fsk_2, m)

$$\sigma_{m,2} = Decode_0\left(fsk_2, CT_{\sigma_m}\right) + e_2' + \mathrm{r}_{m,2}$$

Our Solution

PartSign(pp, fsk_1, m)

$$\sigma_{m,1} = Decode_0(fsk_1, CT_{\sigma_m}) + e_1' + \mathrm{r}_{m,1}$$

PartSign(pp, fsk_2, m)

$$\sigma_{m,2} = Decode_0(fsk_2, CT_{\sigma_m}) + e_2' + \mathrm{r}_{m,2}$$

$r_{m,1}, r_{m,2}$ *are random under the constaint that* $r_{m,1} + r_{m,2} = 0$

Our Solution

PartSign(pp, fsk_1, m)

$\sigma_{m,1} = Decode_0(fsk_1, CT_{\sigma_m}) + e'_1 + \mathrm{r}_{m,1}$

PartSign(pp, fsk_2, m)

$\sigma_{m,2} = Decode_0(fsk_2, CT_{\sigma_m}) + e'_2 + \mathrm{r}_{m,2}$

$r_{m,1}, r_{m,2}$ *are random under the constaint that* $r_{m,1} + r_{m,2} = 0$

- Does not affect Correctness since the added randomness add to zero

Our Solution

PartSign(pp, fsk_1, m)

$\sigma_{m,1} = Decode_0(fsk_1, CT_{\sigma_m}) + e_1' + \mathrm{r_{m,1}}$

PartSign(pp, fsk_2, m)

$\sigma_{m,2} = Decode_0(fsk_2, CT_{\sigma_m}) + e_2' + \mathrm{r_{m,2}}$

$r_{m,1}, r_{m,2}$ *are random under the constaint that* $r_{m,1} + r_{m,2} = 0$

- Does not affect Correctness since the added randomness add to zero
- Can now argue

  $(\langle fsk_1, CT_{\sigma_m}\rangle + r_{m,1}, \langle fsk_2, CT_{\sigma_m}\rangle + r_{m,2}) \approx (r'_{m,1}, r'_{m,2}) : r'_{m,1}$ and $r'_{m,2}$ are random shares of $\left\lfloor \frac{q}{2} \right\rceil \sigma_m$

Our Solution

PartSign(pp, fsk_1, m)

$$\sigma_{m,1} = Decode_0(fsk_1, CT_{\sigma_m}) + e'_1 + \mathrm{r}_{m,1}$$

PartSign(pp, fsk_2, m)

$$\sigma_{m,2} = Decode_0(fsk_2, CT_{\sigma_m}) + e'_2 + \mathrm{r}_{m,2}$$

$$r_{m,1}, r_{m,2} \ are\ random\ under\ the\ constaint\ that\ r_{m,1} + r_{m,2} = 0$$

- Does not affect Correctness since the added randomness add to zero
- Can now argue

$(\langle fsk_1, CT_{\sigma_m}\rangle + r_{m,1}, \langle fsk_2, CT_{\sigma_m}\rangle + r_{m,2}) \approx (r'_{m,1}, r'_{m,2}) : r'_{m,1}$ and $r'_{m,2}$ are random shares of $\left\lfloor \frac{q}{2} \right\rceil \sigma_m$

This releases both fsk_1 and fsk_2 as desired

Our Solution

PartSign(pp, fsk_1, m)

$$\sigma_{m,1} = Decode_0(fsk_1, CT_{\sigma_m}) + e'_1 + \mathrm{r}_{m,1}$$

PartSign(pp, fsk_2, m)

$$\sigma_{m,2} = Decode_0(fsk_2, CT_{\sigma_m}) + e'_2 + \mathrm{r}_{m,2}$$

$r_{m,1}, r_{m,2}$ *are random under the constaint that* $r_{m,1} + r_{m,2} = 0$

- Does not affect Correctness since the added randomness add to zero
- Can now argue

$(\langle fsk_1, CT_{\sigma_m}\rangle + r_{m,1}, \langle fsk_2, CT_{\sigma_m}\rangle + r_{m,2}) \approx (r'_{m,1}, r'_{m,2}) : r'_{m,1}$ and $r'_{m,2}$ are random shares of $\left\lfloor \frac{q}{2} \right\rceil \sigma_m$

This releases both fsk_1 and fsk_2 as desired

Problem: How to ensure that $r_{m,1}$ and $r_{m,2}$ which are chosen independently by P1 and P2 add to zero

Our Solution

PartSign(pp, fsk_1, m)

$$\sigma_{m,1} = Decode_0(fsk_1, CT_{\sigma_m}) + e'_1 + \mathrm{r}_{m,1}$$

PartSign(pp, fsk_2, m)

$$\sigma_{m,2} = Decode_0(fsk_2, CT_{\sigma_m}) + e'_2 + \mathrm{r}_{m,2}$$

$r_{m,1}, r_{m,2}$ *are random under the constaint that* $r_{m,1} + r_{m,2} = 0$

- Does not affect Correctness since the added randomness add to zero
- Can now argue

$(\langle fsk_1, CT_{\sigma_m}\rangle + r_{m,1}, \langle fsk_2, CT_{\sigma_m}\rangle + r_{m,2}) \approx (r'_{m,1}, r'_{m,2})$: $r'_{m,1}$ and $r'_{m,2}$ are random shares of $\left\lfloor \frac{q}{2} \right\rceil \sigma_m$

This releases both fsk_1 and fsk_2 as desired

Problem: How to ensure that $r_{m,1}$ and $r_{m,2}$ which are chosen independently by P1 and P2 add to zero

We use Key Homomorphic PRF to generate $r_{m,1}$ and $r_{m,2}$

Our Solution

PartSign(pp, fsk_1, m)

$\sigma_{m,1} = Decode_0(fsk_1, CT_{\sigma_m}) + e'_1 + \mathrm{r}_{m,1}$

PartSign(pp, fsk_2, m

$\sigma_{m,2} = Decode_0(fsk$

Key Homomorphic PRF:
$F(K_1, x) + F(K_2, x) = F(K_1 + K_2, x)$

Obs: $F(0, x) = 0$

$r_{m,1}, r_{m,2}$ *are random under the constaint that* $r_{m,1} + r_{m,2} = 0$

- Does not affect Correctness since the added randomness add to zero
- Can now argue

$(\langle fsk_1, CT_{\sigma_m}\rangle + r_{m,1}, \langle fsk_2, CT_{\sigma_m}\rangle + r_{m,2}) \approx (r'_{m,1}, r'_{m,2}) : r'_{m,1}$ and $r'_{m,2}$ are random shares of $\left\lfloor \frac{q}{2} \right\rfloor \sigma_m$

This releases both fsk_1 and fsk_2 as desired

Problem: How to ensure that $r_{m,1}$ and $r_{m,2}$ which are chosen independently by P1 and P2 add to zero

We use Key Homomorphic PRF to generate $r_{m,1}$ and $r_{m,2}$

Final Working Solution

PartSign(pp, fsk_1, m)

$\sigma_{m,1} \;=\; Decode_0\left(fsk_1, CT_{\sigma_m}\right) + e_1' + F(K_1, m)$

PartSign(pp, fsk_2, m)

$\sigma_{m,2} \;=\; Decode_0\left(fsk_2, CT_{\sigma_m}\right) + e_2' + F(K_2, m)$

Final Working Solution

$K_1 + K_2 = 0$

K_i is include in the partial signing key of P_i

PartSign(pp, fsk_1, m)

$$\sigma_{m,1} = Decode_0\left(fsk_1, CT_{\sigma_m}\right) + e_1' + F(K_1, m)$$

PartSign(pp, fsk_2, m)

$$\sigma_{m,2} = Decode_0\left(fsk_2, CT_{\sigma_m}\right) + e_2' + F(K_2, m)$$

Final Working Solution
(Security)

$K_1 + K_2 = 0$

K_i is include in the partial signing key of P_i

PartSign(pp, fsk_1, m)

PartSign(pp, fsk_2, m)

H0: $\sigma_{m,1} = Decode_0\left(fsk_1, CT_{\sigma_m}\right) + e_1' + F(K_1, m)$ $\quad$ $\sigma_{m,2} = Decode_0\left(fsk_2, CT_{\sigma_m}\right) + e_2' + F(K_2, m)$

Final Working Solution
(Security)

$K_1 + K_2 = 0$

K_i is include in the partial signing key of P_i

PartSign(pp, fsk_1, m)

PartSign(pp, fsk_2, m)

H0: $\sigma_{m,1} = Decode_0(fsk_1, CT_{\sigma_m}) + e'_1 + F(K_1, m)$ $\quad \sigma_{m,2} = Decode_0(fsk_2, CT_{\sigma_m}) + e'_2 + F(K_2, m)$

H1: $\sigma_{m,1} = Decode_0(fsk_1, CT_{\sigma_m}) + e'_1 + r_{m,1}$ $\quad \sigma_{m,2} = Decode_0(fsk_2, CT_{\sigma_m}) + e'_2 + r_{m,2}$

Final Working Solution
(Security)

$K_1 + K_2 = 0$

K_i is include in the partial signing key of P_i

PartSign(pp, fsk_1, m)

PartSign(pp, fsk_2, m)

H0: $\sigma_{m,1} = Decode_0(fsk_1, CT_{\sigma_m}) + e_1' + F(K_1, m)$ $\quad$ $\sigma_{m,2} = Decode_0(fsk_2, CT_{\sigma_m}) + e_2' + F(K_2, m)$

H1: $\sigma_{m,1} = Decode_0(fsk_1, CT_{\sigma_m}) + e_1' + r_{m,1}$ $\quad$ $\sigma_{m,2} = Decode_0(fsk_2, CT_{\sigma_m}) + e_2' + r_{m,2}$

$(x, F(K_1, x), F(K_2, x)) \approx (x, r_1, r_2)$, where both K_1 and K_2 as well as r_1 and r_2 are secret shares of 0

Final Working Solution
(Security)

$K_1 + K_2 = 0$

K_i is include in the partial signing key of P_i

PartSign(pp, fsk_1, m)

PartSign(pp, fsk_2, m)

H0: $\sigma_{m,1} = Decode_0\left(fsk_1, CT_{\sigma_m}\right) + e_1' + F(K_1, m)$ $\quad \sigma_{m,2} = Decode_0\left(fsk_2, CT_{\sigma_m}\right) + e_2' + F(K_2, m)$

H1: $\sigma_{m,1} = Decode_0\left(fsk_1, CT_{\sigma_m}\right) + e_1' + r_{m,1}$ $\quad \sigma_{m,2} = Decode_0\left(fsk_2, CT_{\sigma_m}\right) + e_2' + r_{m,2}$

$\left(x, F(K_1, x), F(K_2, x)\right) \approx (x, r_1, r_2)$, where both K_1 and K_2 as well as r_1 and r_2 are secret shares of 0

H2: $\sigma_{m,1} = r'_{m,1}\ (random) + e_1'$ $\quad \sigma_{m,2} = r'_{m,2} = \left\lfloor \frac{q}{2} \right\rfloor \sigma_m - r'_{m,1} + e_2'$

H3: $mpk = CT_0$ instead of CT_{sigsk}

Reduction to Sign security in H3

Remarks and Conclusions

- Lattice based KHPRF do not satisfy exact homomorphism

$$F(K_1, x) + F(K_2, x) = F(K_1 + K_2, x) + \delta$$

We use flooding to hide δ

Remarks and Conclusions

- Lattice based KHPRF do not satisfy exact homomorphism

$$F(K_1, x) + F(K_2, x) = F(K_1 + K_2, x) + \delta$$

We use flooding to hide δ

- We implement these ideas at the level of threshold FHE

Remarks and Conclusions

- Lattice based KHPRF do not satisfy exact homomorphism

$$F(K_1, x) + F(K_2, x) = F(K_1 + K_2, x) + \delta$$

We use flooding to hide δ

- We implement these ideas at the level of threshold FHE ⟹ stronger threshold FHE

Remarks and Conclusions

- Lattice based KHPRF do not satisfy exact homomorphism

$$F(K_1, x) + F(K_2, x) = F(K_1 + K_2, x) + \delta$$

We use flooding to hide δ

- We implement these ideas at the level of threshold FHE ⟹ stronger threshold FHE

⇓

Stronger UT

Remarks and Conclusions

- Lattice based KHPRF do not satisfy exact homomorphism

$$F(K_1, x) + F(K_2, x) = F(K_1 + K_2, x) + \delta$$

We use flooding to hide δ

- We implement these ideas at the level of threshold FHE ⟹ stronger threshold FHE

Stronger UT

Stronger threshold signature/ CCA-PKE, etc

Thank You