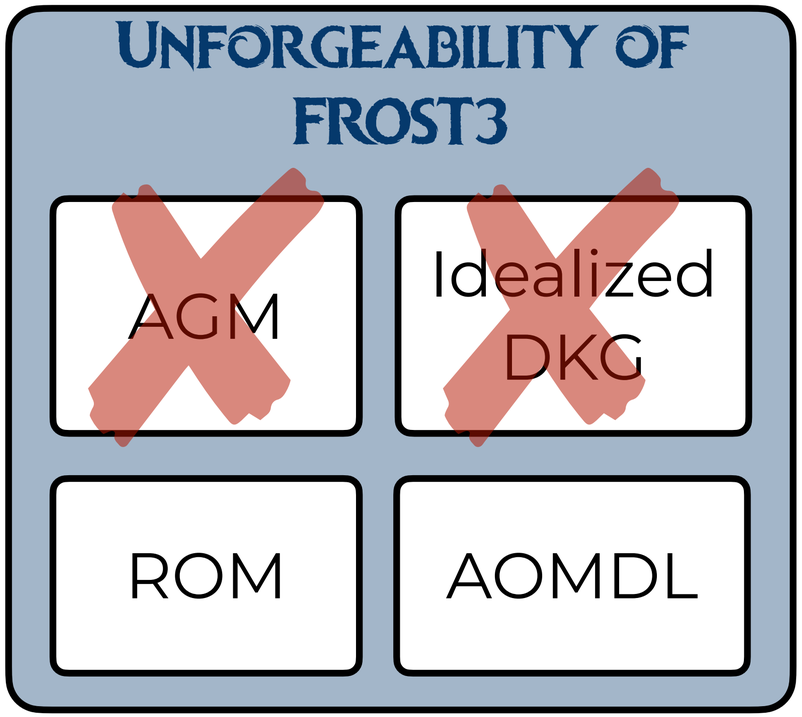
Practical Schnorr Threshold Signatures Without the Algebraic Group Model

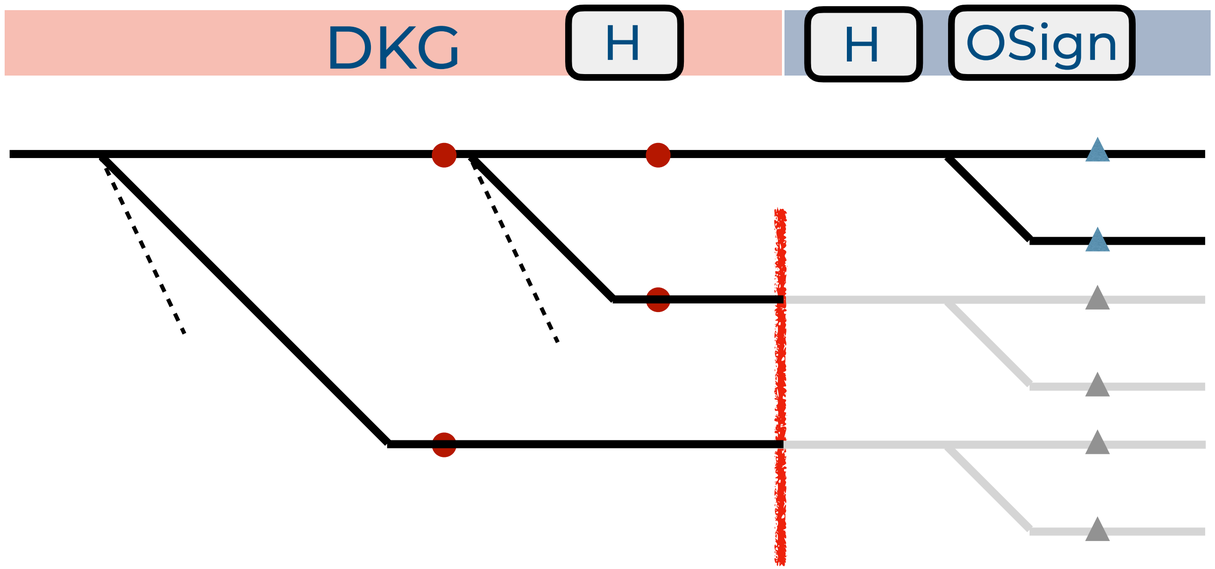
Hien Chu[1], Paul Gerhart[1], Tim Ruffing[2], and Dominique Schröder[1]

FAU[1]

Blockstream[2]

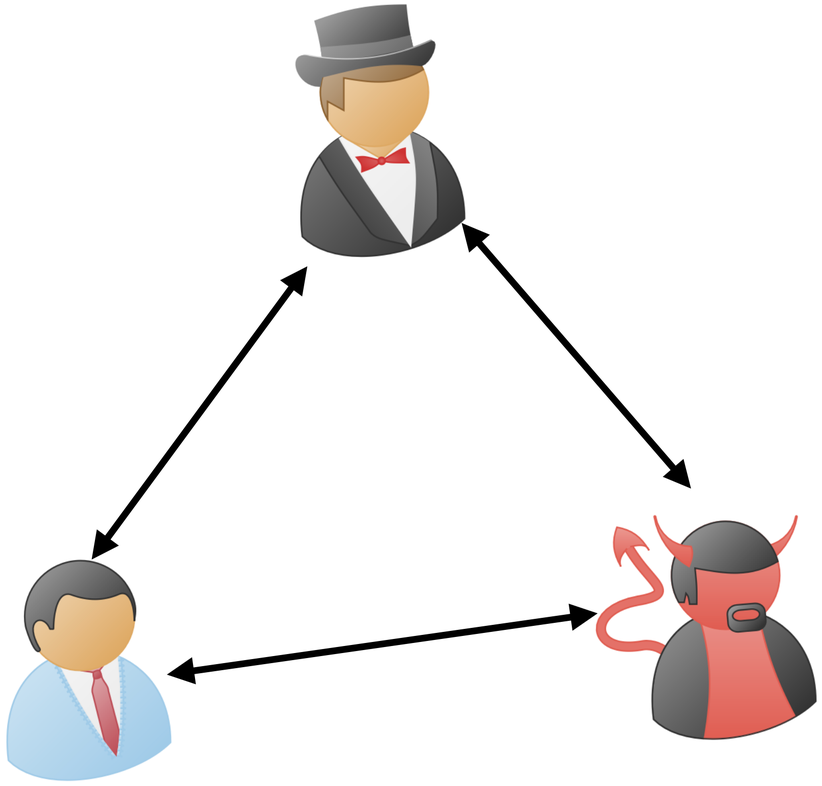
Our Contributions





Blockstream ChaAC FAU

Threshold Signatures [DES88, DF90]



$$m = \text{LET'S PROVE FROST3}$$

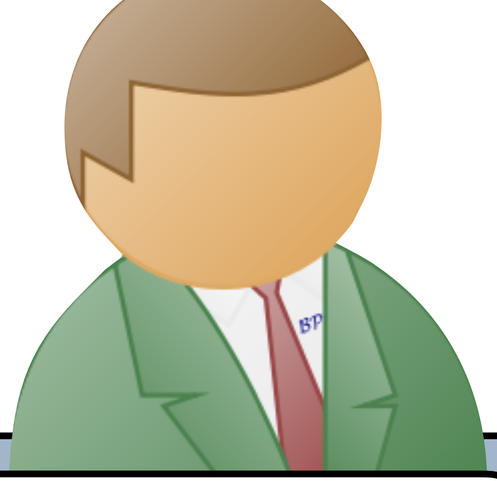
$$(\sigma_1, \sigma_2, \sigma_3) \xrightarrow{\text{Aggregate}} \sigma = (R, s)$$

$$c := \mathsf{H}(\mathsf{pk}, R, m)$$

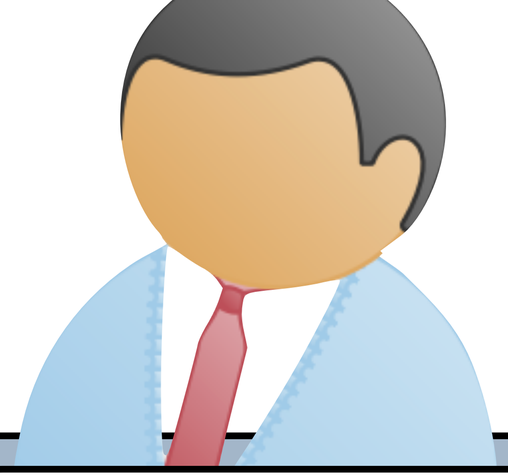
$$R \stackrel{?}{=} g^s \mathsf{pk}^{-c}$$

Blockstream ChaAC FAU

Komlo and Goldberg
SAC 2020



FROST3



PreRound(pk)

$\rho_i := (D_i, E_i)$

$\rho_j := (D_j, E_j)$

PreRound(pk)

PreAgg(pk, ρ_i, ρ_j)

Preprocessing

SignRound(sk_i, pk, ρ, m)

σ_i

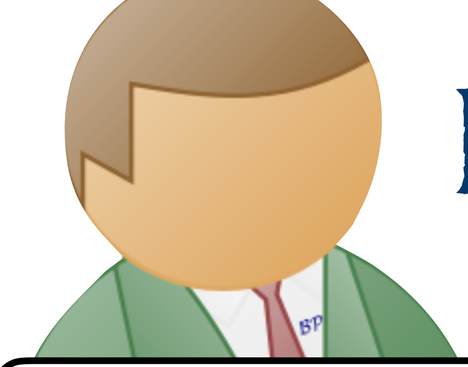
σ_j

SignRound(sk_j, pk, ρ, m)

SignAgg(pk, (σ_i), m)

Signing

Blockstream ChaAC FAU



FROST3 PREPROCESSING

(SIMPLIFIED)

PreRound(pk)

$d_i \leftarrow^{\$} \mathbb{Z}_p, \quad e_i \leftarrow^{\$} \mathbb{Z}_p$

$D_i \leftarrow g^{d_i}, \quad E_i \leftarrow g^{e_i}$

$\rho_i \leftarrow (D_i, E_i)$

(D_i, E_i) →

← (D_j, E_j)

PreRound(pk)

$d_j \leftarrow^{\$} \mathbb{Z}_p, \quad e_j \leftarrow^{\$} \mathbb{Z}_p$

$D_j \leftarrow g^{d_j}, \quad E_j \leftarrow g^{e_j}$

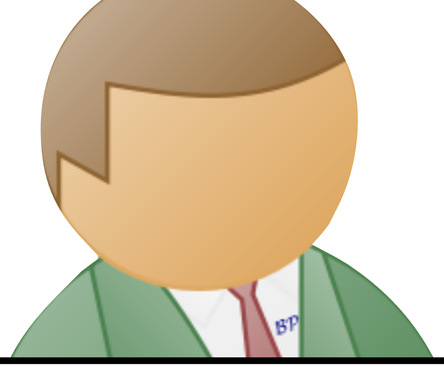
$\rho_j \leftarrow (D_j, E_j)$

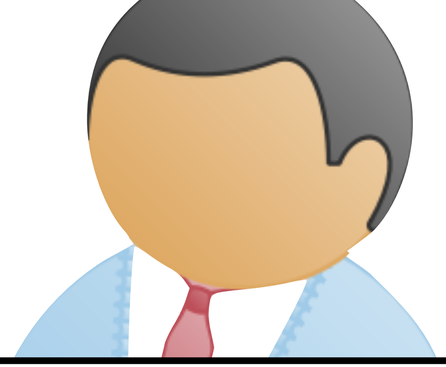
PreAgg(pk, ρ_i, ρ_j)

$D := D_i \cdot D_j \qquad E := E_i \cdot E_j$

Blockstream ChaAC FAU

FROST3 SIGNING





SignRound(sk_i, pk, S, $state_i$, ρ, m)

$b \leftarrow \mathsf{H}_{non}(\mathsf{pk}, S, \rho, m)$

$R \leftarrow DE^b$ $r = d + be$

$c \leftarrow \mathsf{H}_{sig}(X, R, m)$

$\Lambda_i \leftarrow \mathsf{Lagrange}(S, i)$

$\sigma_i \leftarrow d_i + be_i + c\Lambda_i x_i$

σ_i →

← σ_j

SignRound(sk_j, pk, S, $state_j$, ρ, m)

$b \leftarrow \mathsf{H}_{non}(X, S, \rho, m)$

$R \leftarrow DE^b$

$c \leftarrow \mathsf{H}_{sig}(X, R, m)$

$\Lambda_j \leftarrow \mathsf{Lagrange}(S, j)$

$\sigma_j \leftarrow d_j + be_j + c\Lambda_j x_j$

SignAgg(pk, ρ, σ_i, σ_j, S, m)

$b \leftarrow \mathsf{H}_{non}(X, S, \rho, m)$

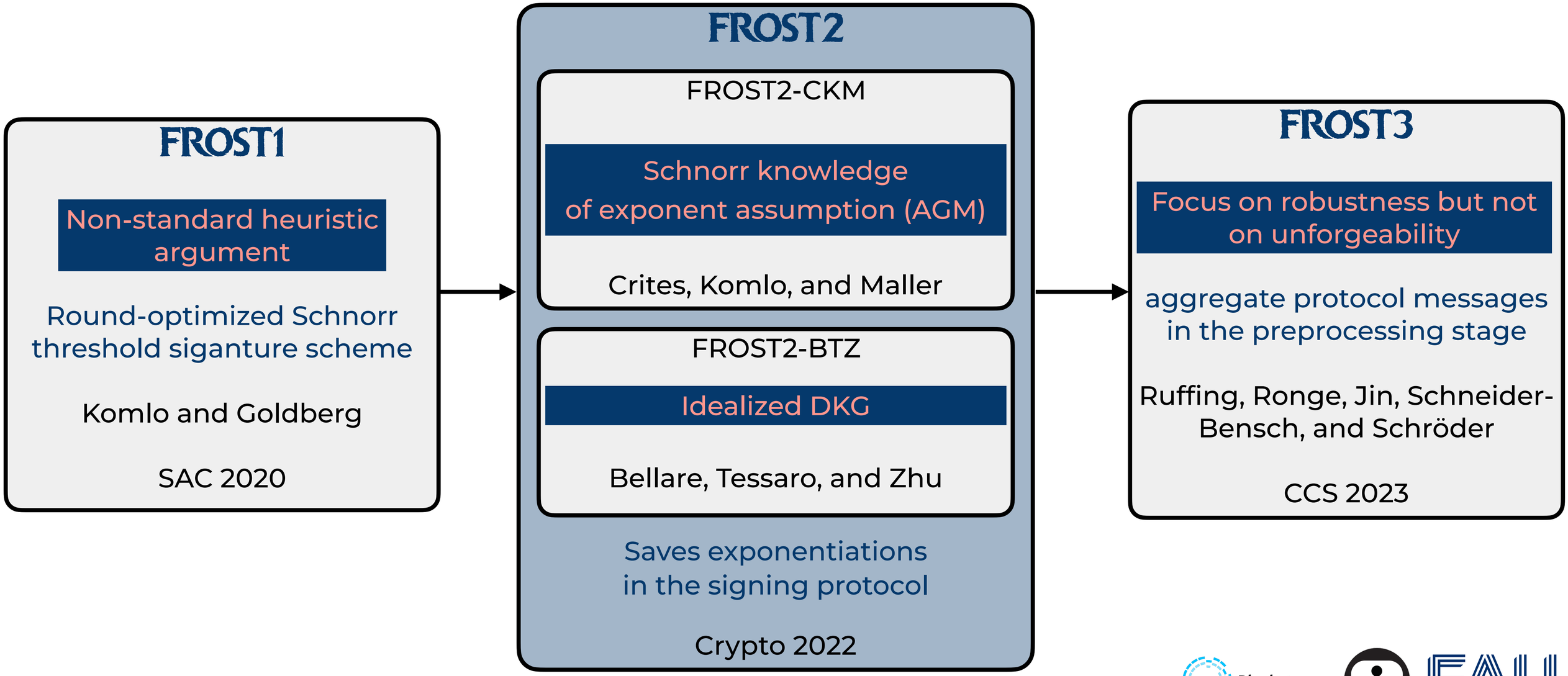
$R \leftarrow DE^b$

$s' \leftarrow \sigma_i + \sigma_j = (d + be) + cx = r + cx$

$\sigma \leftarrow (R, s) \quad = (g^r, r + cx)$

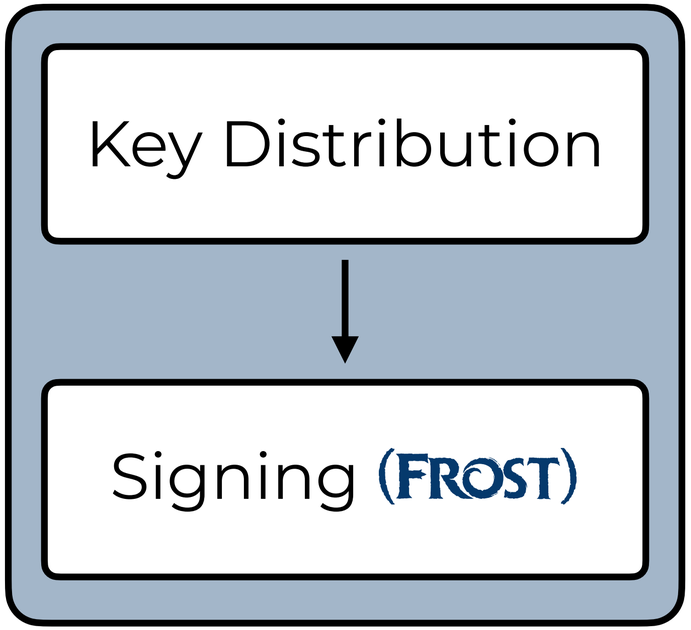
Blockstream ChaAC FAU

The History of Proving FROST Secure



Blockstream ChaAC FAU

Idealized DKGs [GJKR99,GJKR07,KGS23]



- No fully simulatable DKG in the dishonest majority setting $(n/2 \leq t-1)$
- Simulatable DKGs are not as efficient as the Pedersen DKG

Blockstream ChaAC FAU

Pedersen DKG with PoP [KG20, CKM21]

$$f_i(Z) = a_{i,0} + a_{i,1}Z + \cdots + a_{i,t-1}Z^{t-1}$$

$$f(Z) = \sum_{i=1}^{n} f_i(Z)$$

$$x_i = f(i), pk = g^{f(0)}$$

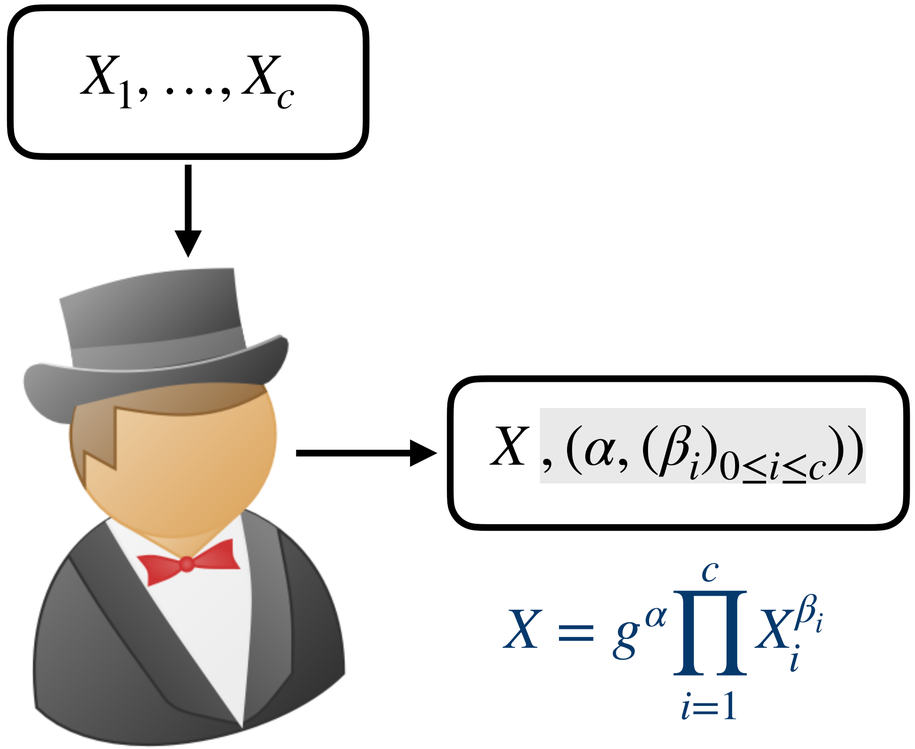
$$A_{i,k} = g^{a_{i,k}}$$

$$\text{PoP} = (R, s)$$

- ✓ Additive sharing of a secret key
  $x = x_1 + \ldots + x_n = a_{1,0} + \ldots + a_{n_0}$
- ✗ Not fully simulatable DKG due to a possible shift [GJKR06]

Blockstream ChaAC FAU

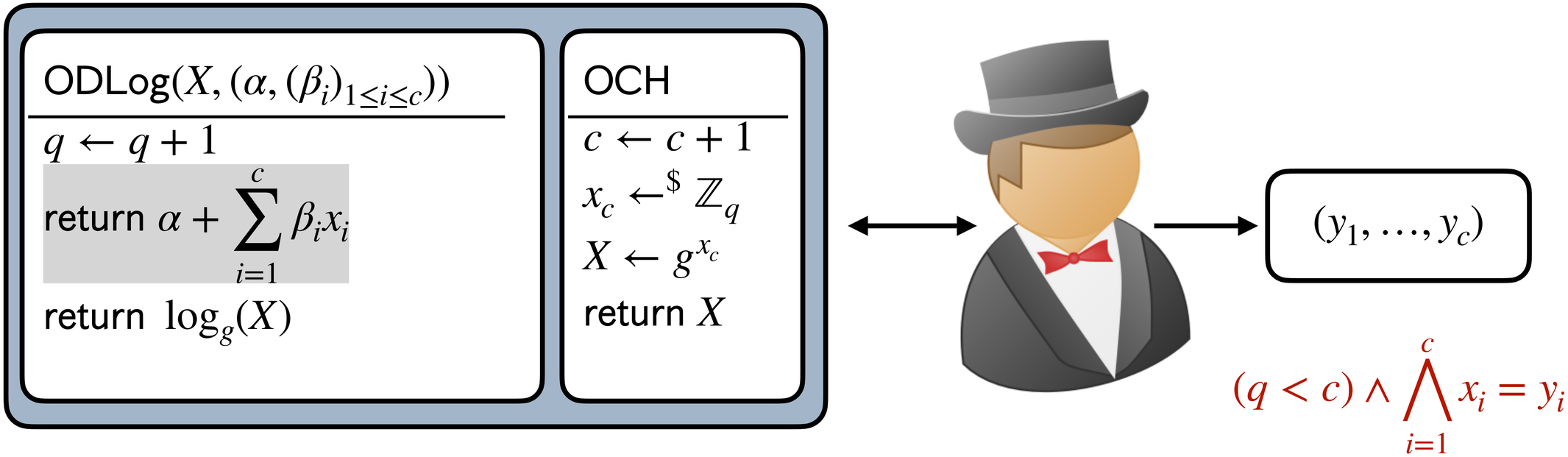
The AGM [PV05, FKL18]



- ✗ “Hardness in the AGM may not imply hardness in the GGM” [KZZ22]
- ✗ “[...] The AGM is restricted to the Type Safe model games.“ [Zha22]

Blockstream ChaAC FAU

AOMDL Assumption [NRS21]



Blockstream ChaAC FAU

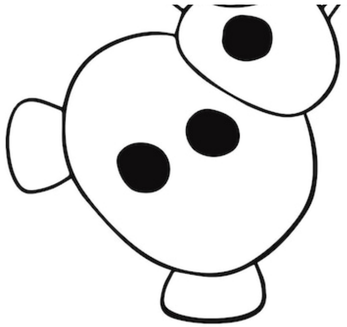
From FROST to OLAF

(HOPEFULLY) ONE LAST FROST

From FROST to OLAF

FROST3

+



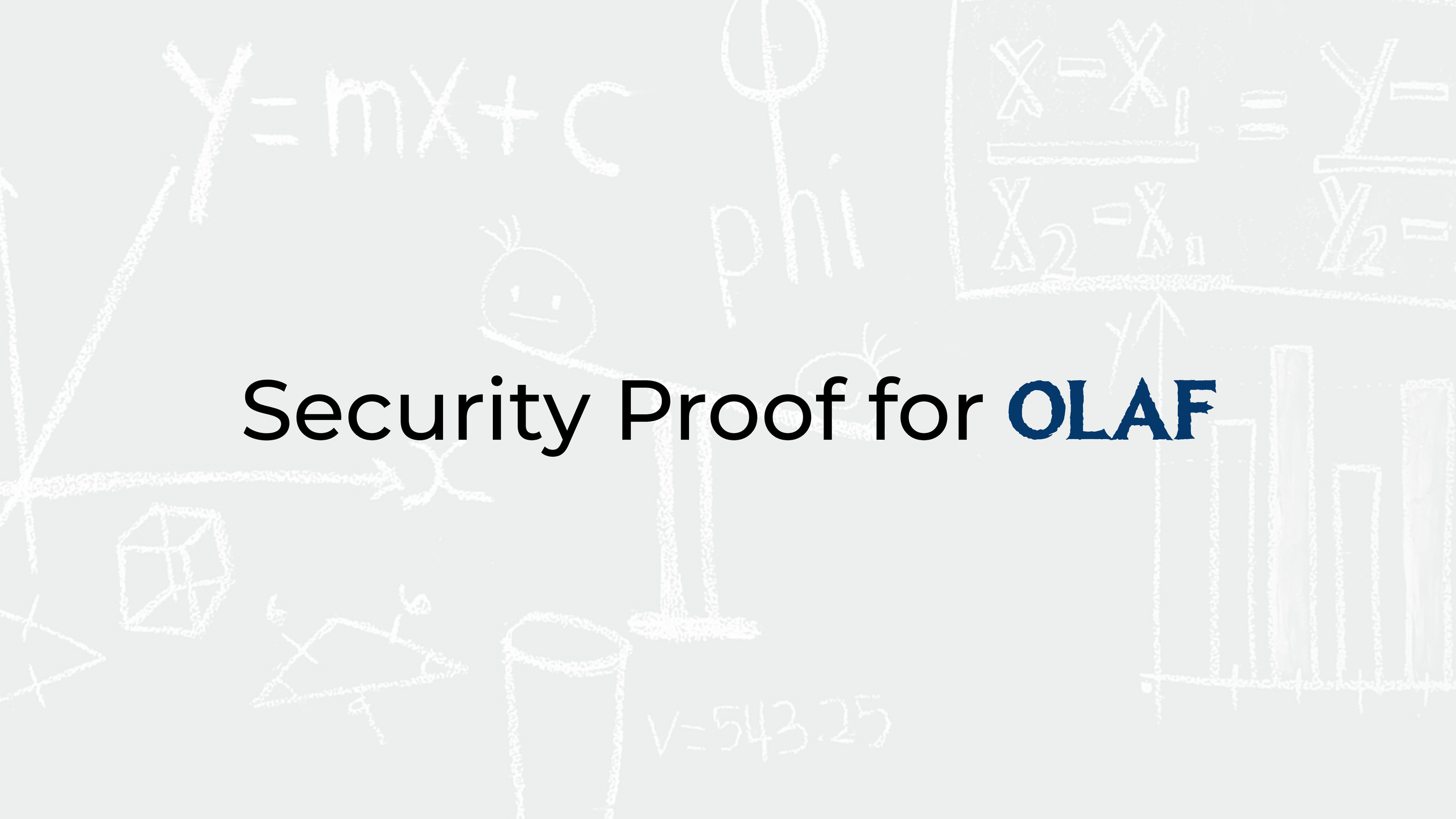
(Simplified)
Pedersen DKG

+

Avoid the AGM
and idealized DKGs

=

Blockstream ChaAC FAU

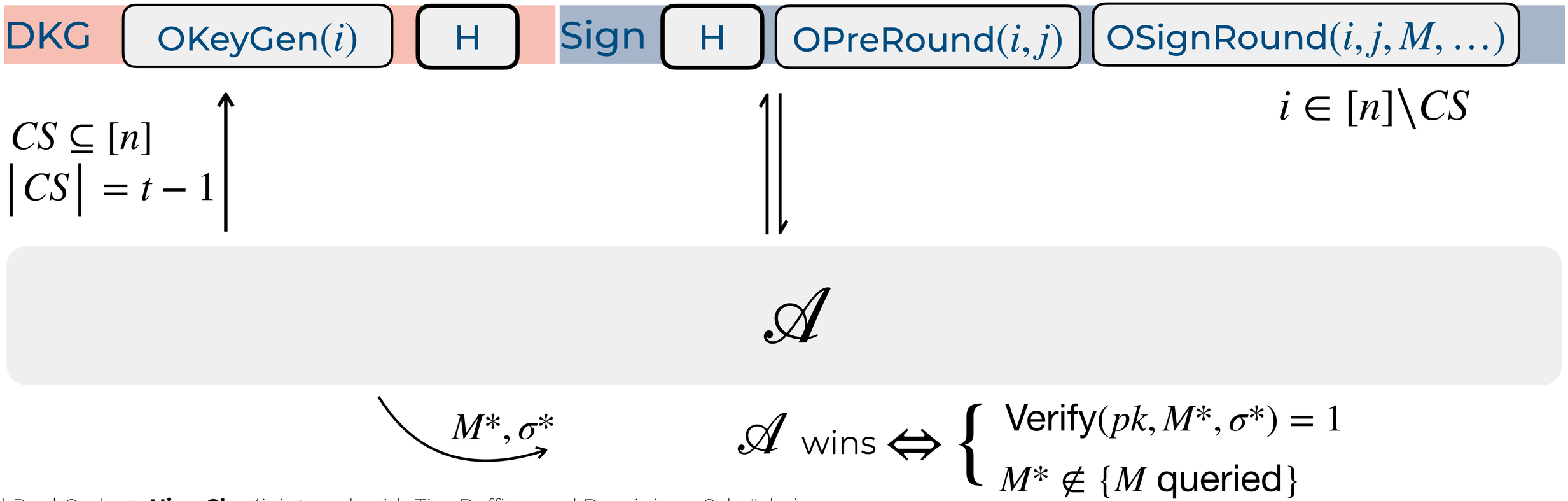


Security Proof for OLAF

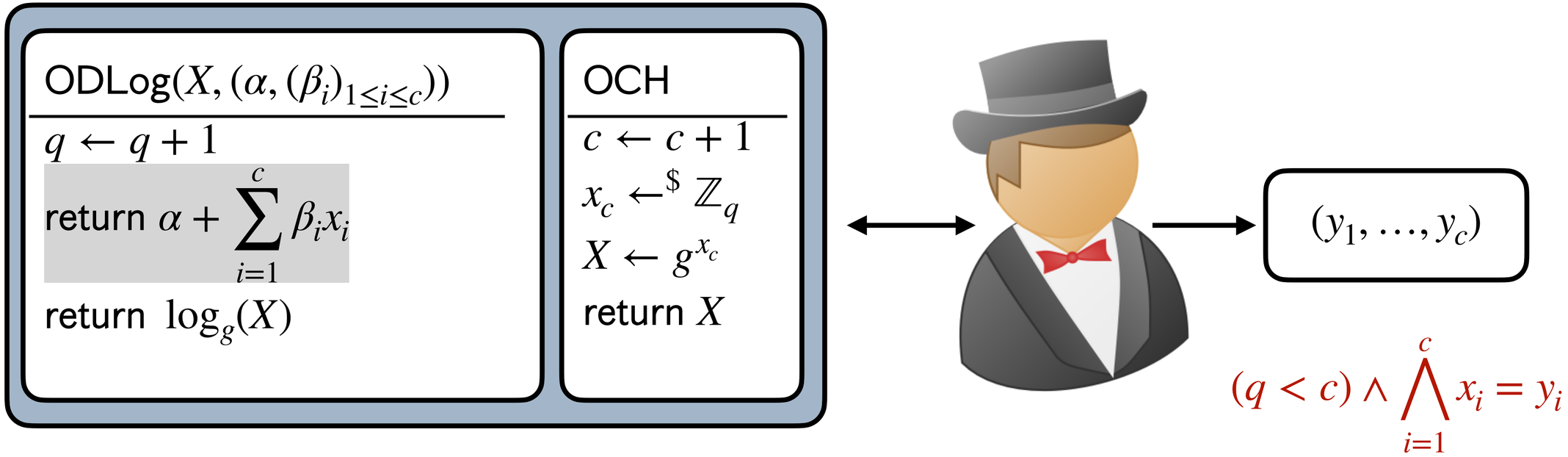
Our Unforgeability Model

Similar to **TS-UF-0** of [BTZ22]

An honest party accepts NO signing queries **before** key setup (DKG) has finished for it



AOMDL Assumption [NRS21]



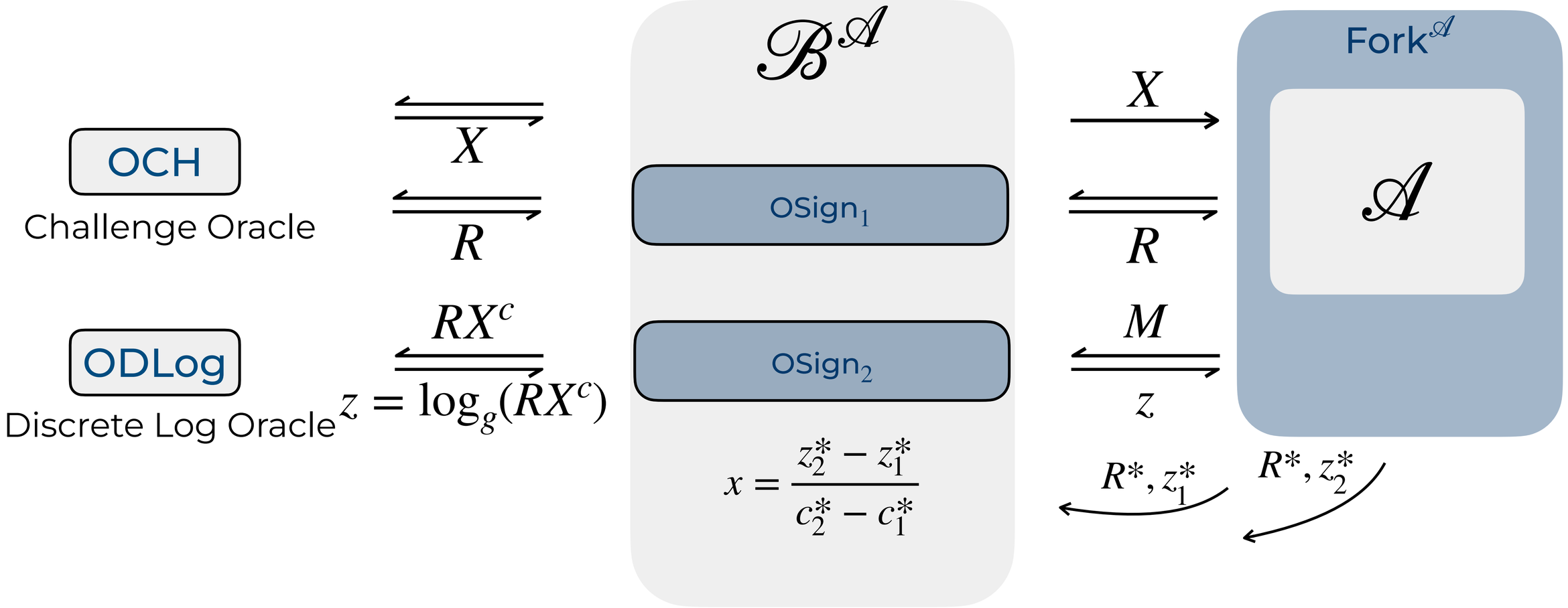
Blockstream ChaAC FAU

Proper (A)OMDL proof has less DLog queries than challenges.

Counting DLog queries could be the most tricky part.

Blockstream ChaAC FAU

AOMDL Proof for Single-Signer Schnorr



$\#\mathsf{ODLog} = \#\mathsf{OSign}_2 \leq \mathsf{OSign}_1 = \#\mathsf{OCH} - 1$

Blockstream ChaAC FAU

AOMDL Proof for Threshold Schnorr

Proof of single-signer Schnorr is well understood

1. Simple simulation of signing queries
2. Rely on forking [PS00], [BN06], optimal tightness [Seurin11, FJS19]

Blockstream ChaAC FAU

AOMDL Proof for Threshold Schnorr

Techniques do NOT carry over to the threshold setting

1. Using **two** nonces D, E instead of one nonce (see FROST [KG20])
   To answer two signing queries on **two executions** for same D, E
2. Consider **Pedersen DKG**
   Reduction needs to know each x_i in $\sum_i x_i = x$ instead of just x to solve AOMDL

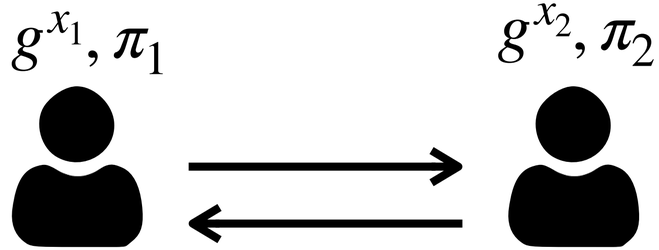
x_i's: signing key shares

Blockstream ChaAC FAU

We only have enough DLog queries for two executions of signing.

Blockstream ChaAC FAU

Our Proof Strategy for OLAF



1. Use Pedersen DKG with **proofs of possession (PoPs)** as in [CKM21]

   PoP is Schnorr-like proof of knowledge

2. Use a novel **forking proof technique** to avoid AGM

x_i's: signing key shares

π_i's: proof of possession

g^{x_i}'s: public key shares

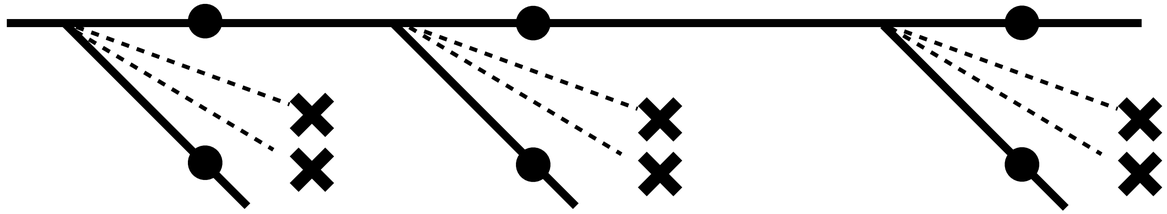
Blockstream ChaAC FAU

Existing Forking Techniques: Overview

[Bellare-Neven-05] bi-forking lemma $\epsilon' = \epsilon^2/q$

Blockstream ChaAC FAU

Existing Forking Techniques: Overview

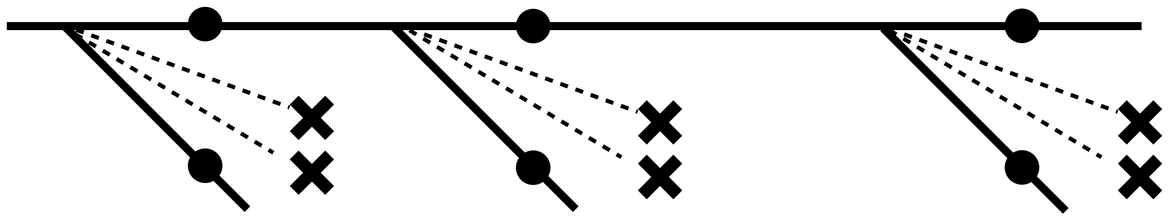


[BN06] bi-forking lemma $\epsilon' = \epsilon^2/q$

[BCJ08] multi-forking lemma $\epsilon' = \epsilon/8$

Blockstream ChaAC FAU

Existing Forking Techniques: Overview



[BN06] bi-forking lemma $\epsilon' = \epsilon^2/q$

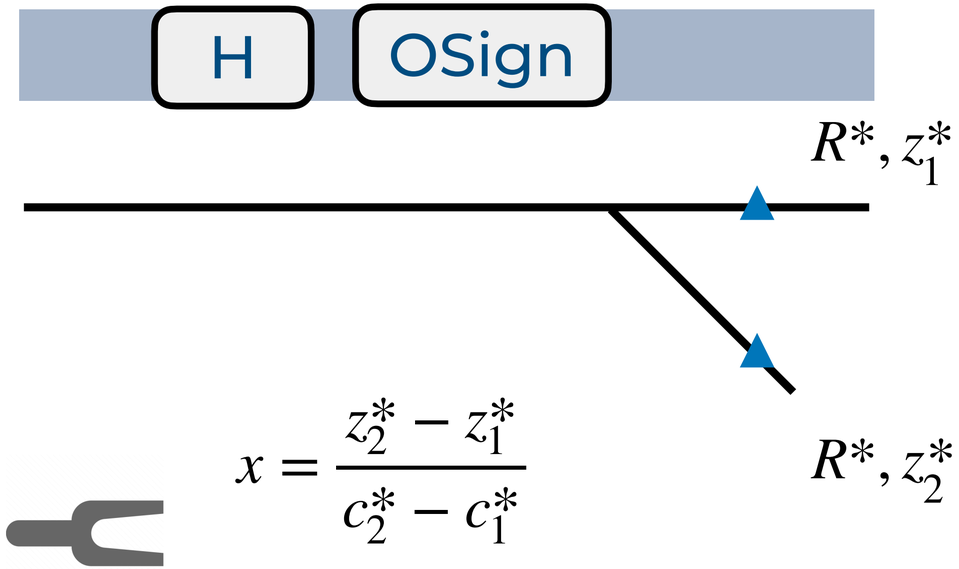
- ✓ Only two executions
- ✗ Multiple extraction (in composition) loses 2^t success probability

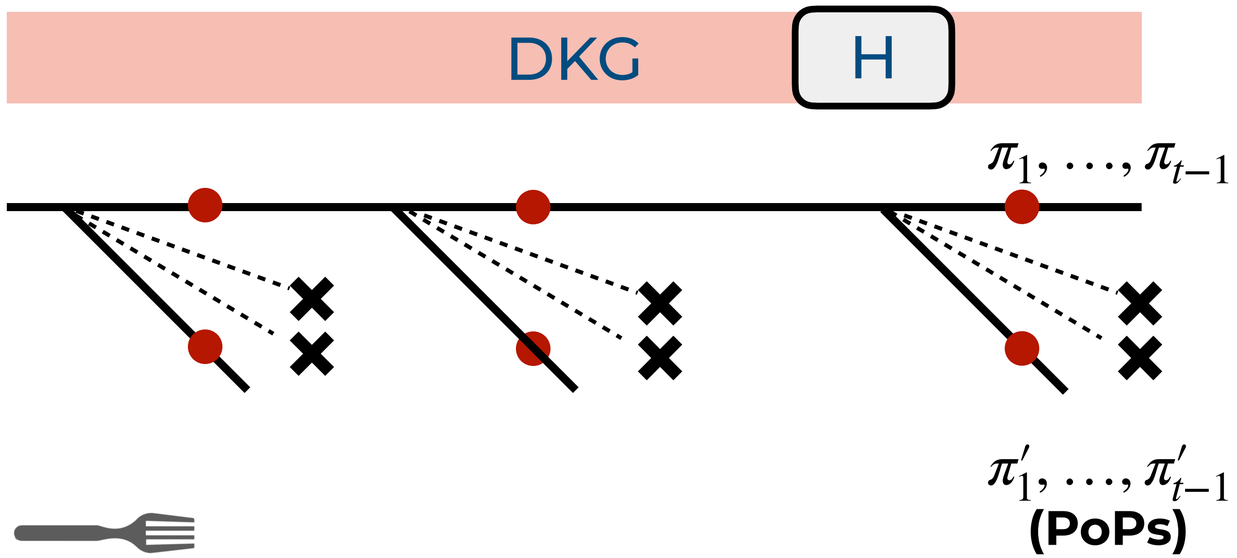
[BCJ08] multi-forking lemma $\epsilon' = \epsilon/8$

- ✓ Multiple extraction points
- ✗ Polynomial number of executions

Blockstream ChaAC FAU

Existing Forking Techniques: In Our Proof



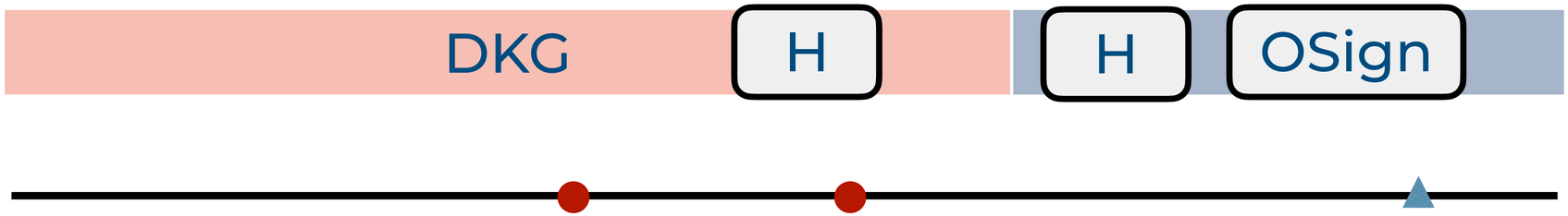


[BN06] bi-forking lemma $\epsilon' = \epsilon^2/q$ [BCJ08] multi-forking lemma $\epsilon' = \epsilon/8$

➡ Can extract x from forgeries ➡ Can extract $x_1, \ldots, x_{t-1}$ from PoPs

Blockstream ChaAC FAU

The Mixed Forking

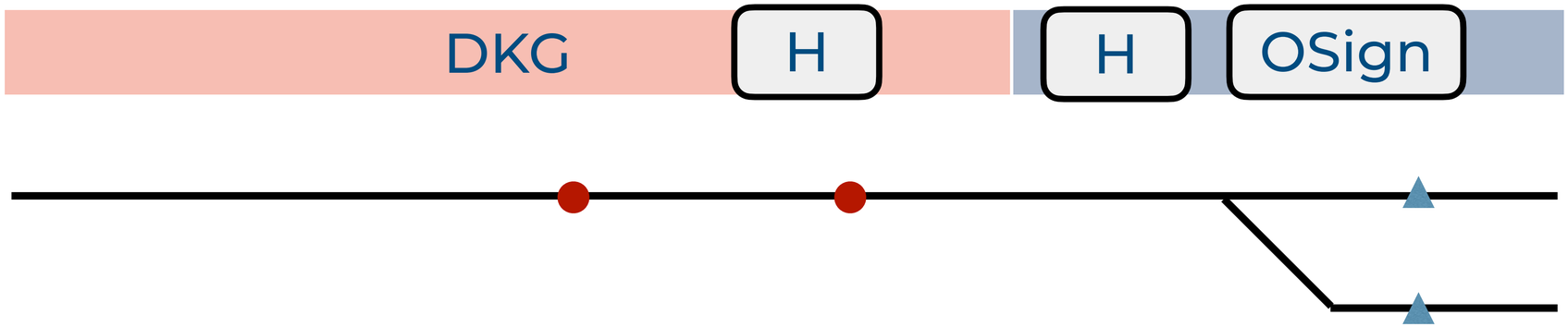


Two-step approach

$\mathscr{A}$

Blockstream ChaAC FAU

The Mixed Forking



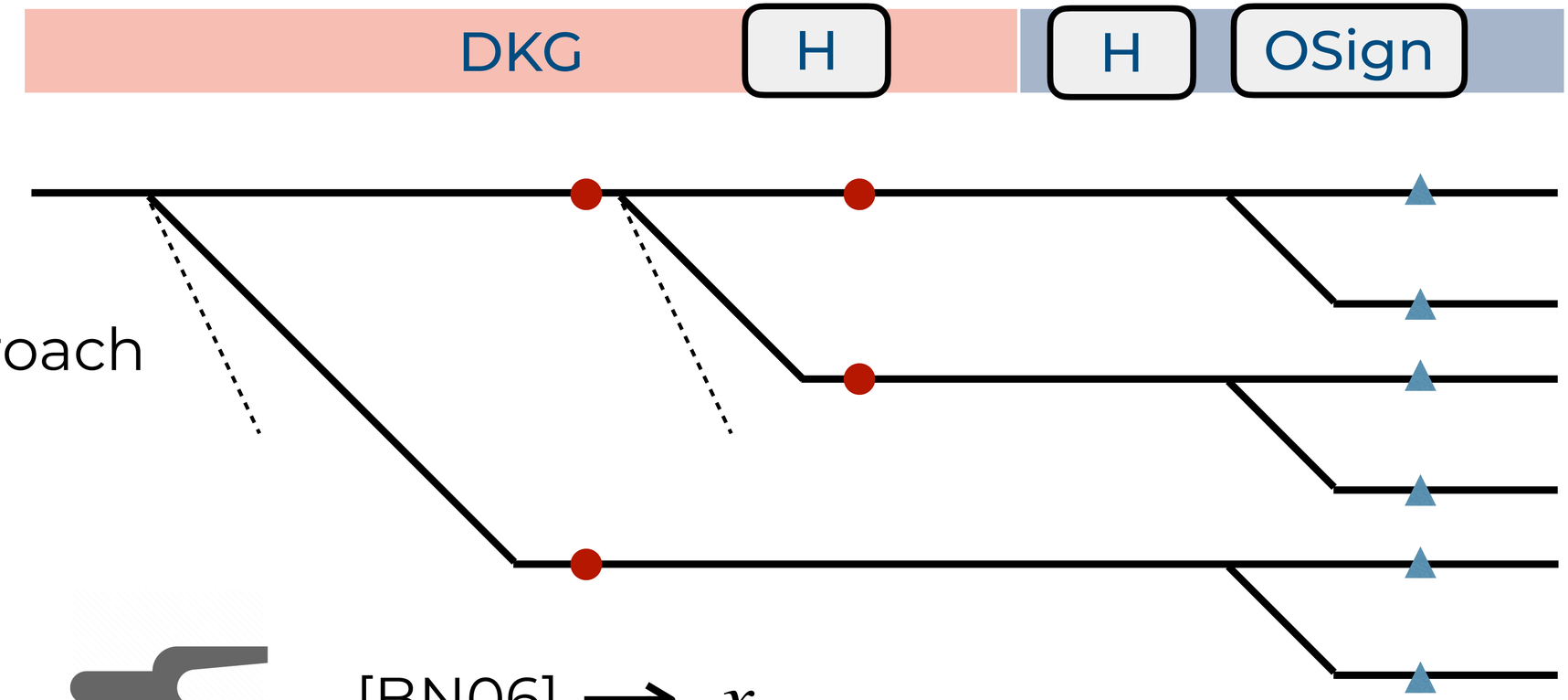
Two-step approach

$$\mathscr{A} \downarrow \mathscr{C}\left(\mathsf{Fork}^{\mathscr{A}}\right)$$

[BN06] $\longrightarrow x$

Blockstream ChaAC FAU

The Mixed Forking



Two-step approach

$\mathscr{A}$

$\downarrow$

$\mathscr{C}\left(\text{Fork}^{\mathscr{A}}\right)$ [BN06] $\longrightarrow$ x

$\downarrow$

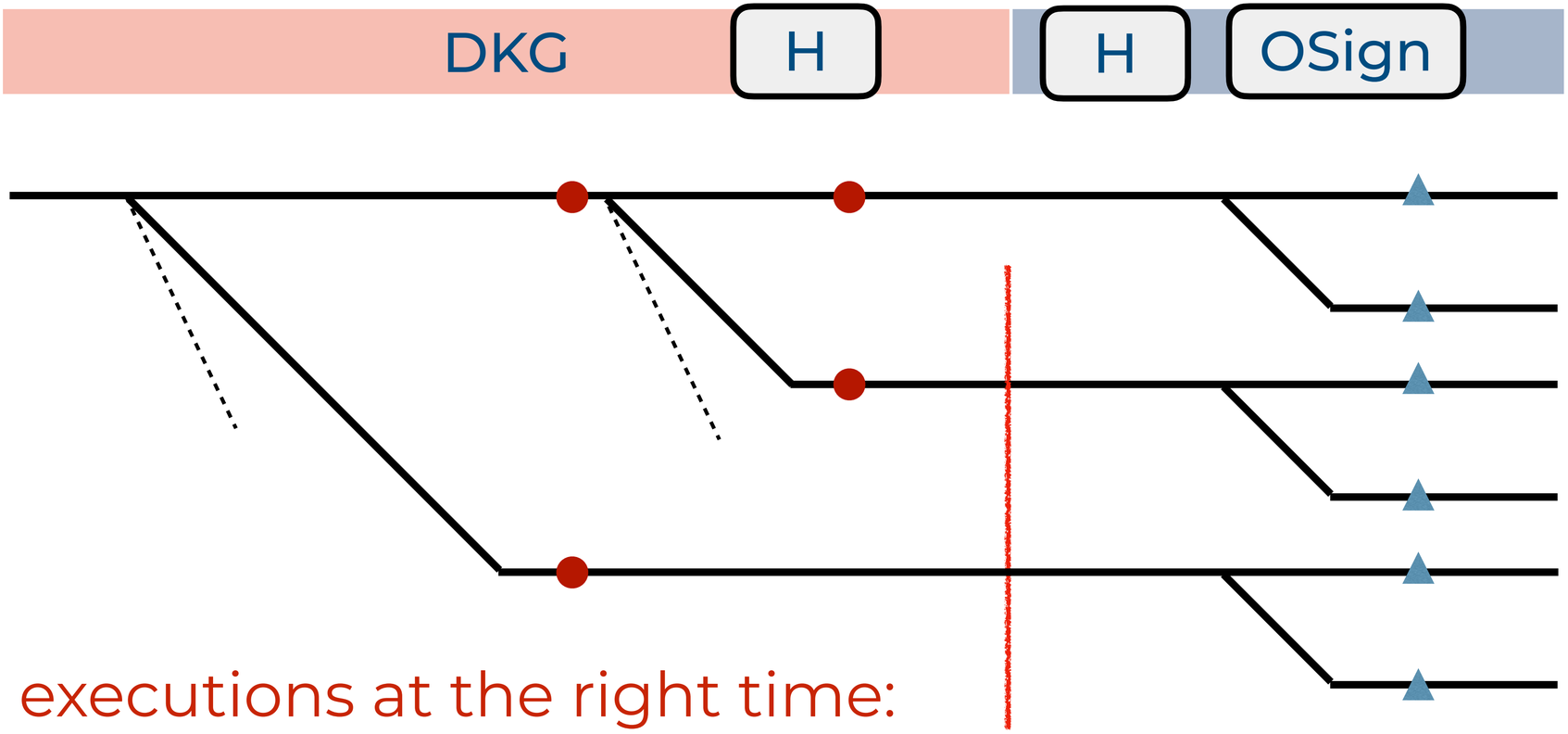
$\mathscr{D}\left(\text{Fork}^{\mathscr{C}}\right)$ [BCJ08] $\longrightarrow$ $x_1, \ldots, x_{t-1}$

Blockstream ChaAC FAU

We only have enough DLog queries for two executions of signing.

Blockstream ChaAC FAU

Reducing Number of DL Queries



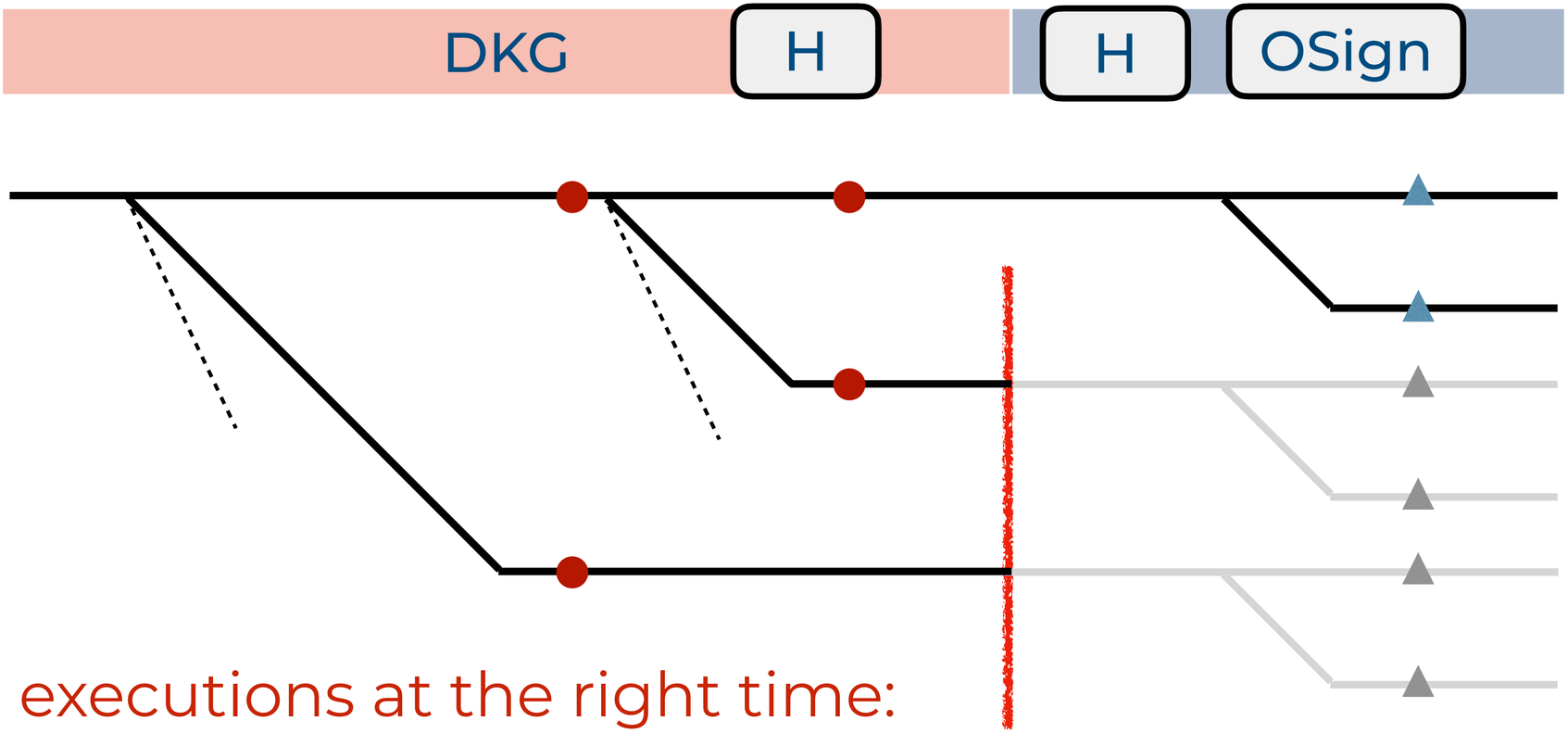
➡ Pruning executions at the right time:

right after the DKG completion

then $\mathscr{D}' \approx \mathscr{D}$ but only **two executions** of $\mathscr{A}$ needs signing simulation

Blockstream ChaAC FAU

Reducing Number of DL Queries



➡ Pruning executions at the right time:

right after the DKG completion

then $\mathscr{D}' \approx \mathscr{D}$ but only **two executions** of $\mathscr{A}$ needs signing simulation

Blockstream ChaAC FAU

Our Conclusion

- We proved the unforgeability of FROST3 with probability $\epsilon' \approx \frac{\epsilon^2}{8q}$ in expected poly(T) time
  - Using a **practical DKG** (Simplified Pedersen DKG + PoPs)
  - In **AOMDL + ROM**

Blockstream ChaAC FAU

Our Work:

Practical Schnorr Threshold Signatures without the Algebraic Group Model

Hien Chu, Paul Gerhart, Tim Ruffing, Dominique Schröder

CRYPTO 2023

For Robustness, see:

ROAST: Robust Asynchronous Schnorr Threshold Signatures

Tim Ruffing, Viktoria Ronge, Elliott Jin, Jonas Schneider-Bensch, and Dominique Schröder

CCS 2023