One Tree to Rule Them All

Optimizing GGM Trees and OWFs for Post-Quantum Signatures

Carsten Baum[1,2] Ward Beullens[3] Cyprien de Saint Guilhem[4] Shibam Mukherjee[5]
Emmanuela Orsini[6] Sebastian Ramacher[7] Christian Rechberger[5] Lawrence Roy[1] Peter Scholl[1]

[1] Aarhus University [2] Technical University of Denmark
[3] IBM Research Zurich [4] COSIC KU Leuven [5] TU Graz
[6] Bocconi University [7] AIT Austrian Institute of Technology

team@faest.info

Thank you to Lennart Braun for many slides.

FAEST

FAEST

After Picnic, BBQ, and Banquet . . .

FAEST

After Picnic, BBQ, and Banquet . . .

. . . welcome to FAEST! — the VOLE-in-the-Head Post Quantum Signature Scheme.

FAEST

After Picnic, BBQ, and Banquet . . .
. . . welcome to FAEST! — the VOLE-in-the-Head Post Quantum Signature Scheme.

Ingredients:

- $1\times$ Zero-knowledge proof for "I know $k \in \{0,1\}^{\lambda}$ such that $\mathrm{AES}_k(x) = y$"

FAEST

After Picnic, BBQ, and Banquet …

…welcome to FAEST! — the VOLE-in-the-Head Post Quantum Signature Scheme.

Ingredients:

- 1× Zero-knowledge proof for “I know $k \in \{0,1\}^\lambda$ such that $\text{AES}_k(x) = y$”
- 1× fresh VOLE (in-the-Head)

FAEST

After Picnic, BBQ, and Banquet . . .

. . . welcome to FAEST! — the VOLE-in-the-Head Post Quantum Signature Scheme.

Ingredients:

- 1× Zero-knowledge proof for "I know $k \in \{0,1\}^\lambda$ such that $\mathrm{AES}_k(x) = y$"
- 1× fresh VOLE (in-the-Head)
- A pinch of Fiat-Shamir

FAEST

After Picnic, BBQ, and Banquet ...

...welcome to FAEST! — the VOLE-in-the-Head Post Quantum Signature Scheme.

Ingredients:

- 1× Zero-knowledge proof for "I know $k \in \{0,1\}^\lambda$ such that $\mathrm{AES}_k(x) = y$"
- 1× fresh VOLE (in-the-Head)
- A pinch of Fiat-Shamir

$\implies$ Delicious Digital Signature Scheme with secret key $k \in \{0,1\}^\lambda$ and public key (x, y).

FAEST

After Picnic, BBQ, and Banquet . . .

. . . welcome to FAEST! — the VOLE-in-the-Head Post Quantum Signature Scheme.

Ingredients:

- 1× Zero-knowledge proof for "I know $k \in \{0,1\}^\lambda$ such that $\mathrm{AES}_k(x) = y$"
- 1× fresh VOLE (in-the-Head)
- A pinch of Fiat-Shamir

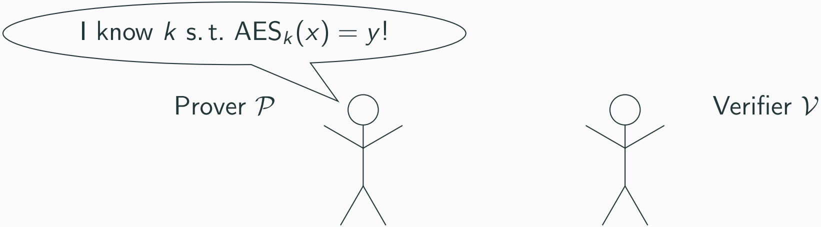
$\implies$ Delicious Digital Signature Scheme with secret key $k \in \{0,1\}^\lambda$ and public key (x, y).

Our submission to the NIST Call for Post Quantum Signatures.

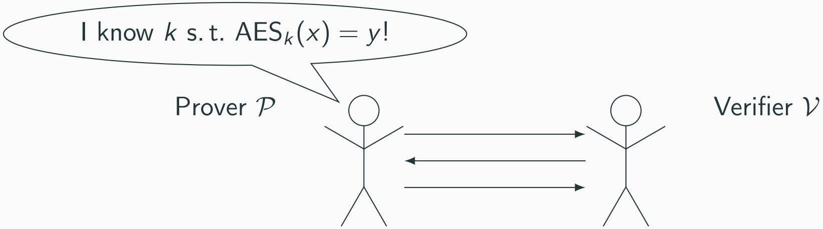
Chefs: Carsten Baum, Lennart Braun, Cyprien Delpech de Saint Guilhem, Michael Klooß, Christian Majenz, Shibam Mukherjee, Emmanuela Orsini, Sebastian Ramacher, Christian Rechberger, Lawrence Roy, Peter Scholl.

Identification Schemes Based on Zero-Knowledge Proofs

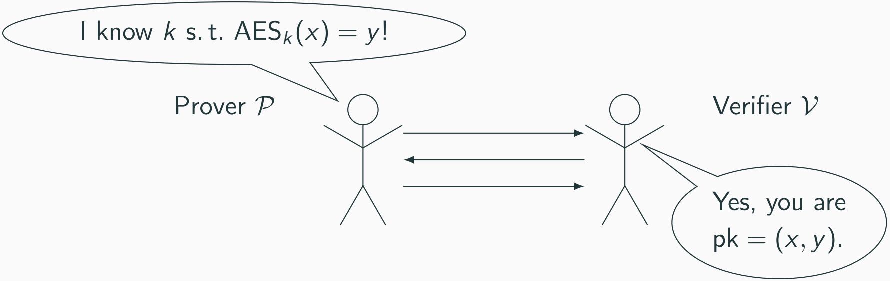
Identification Schemes Based on Zero-Knowledge Proofs



Identification Schemes Based on Zero-Knowledge Proofs



Identification Schemes Based on Zero-Knowledge Proofs



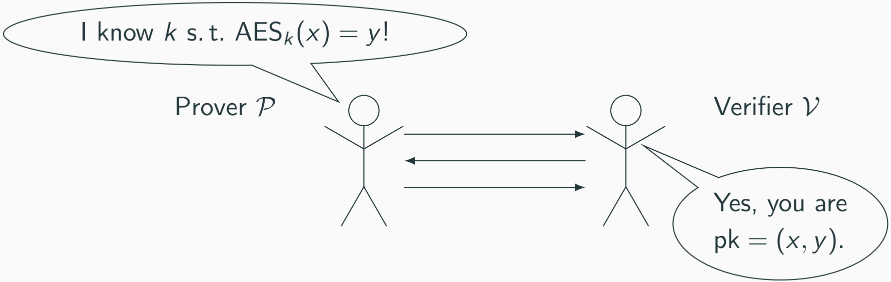
I know k s. t. $\mathsf{AES}_k(x) = y$!

Prover $\mathcal{P}$

Verifier $\mathcal{V}$

Yes, you are $\mathsf{pk} = (x, y)$.

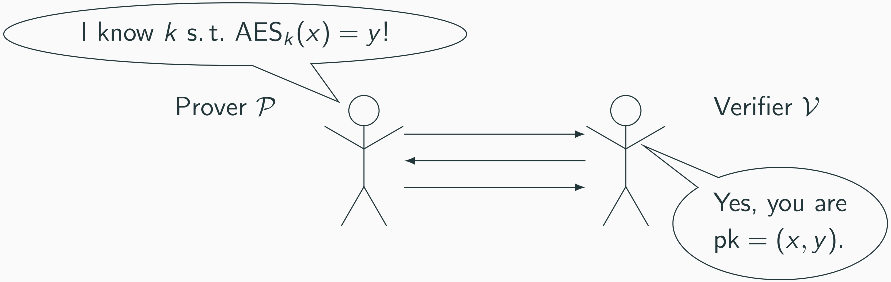
Identification Schemes Based on Zero-Knowledge Proofs


I know k s. t. $\mathsf{AES}_k(x) = y$!
Prover $\mathcal{P}$
Verifier $\mathcal{V}$
Yes, you are $\mathsf{pk} = (x, y)$.

Security Properties

- Soundness: $\mathcal{V}$ cannot be convinced of a false statement
- Zero-Knowledge: $\mathcal{V}$ does not learn anything new from the interaction

Signature Schemes Based on Zero-Knowledge Proofs


I know k s. t. $\mathrm{AES}_k(x) = y$!
Prover $\mathcal{P}$
Verifier $\mathcal{V}$
Yes, you are pk $= (x, y)$.

Security Properties

- Soundness: $\mathcal{V}$ cannot be convinced of a false statement
- Zero-Knowledge: $\mathcal{V}$ does not learn anything new from the interaction

If the verifier has no secrets (i.e., is public-coin), can convert into a signature using Fiat-Shamir.

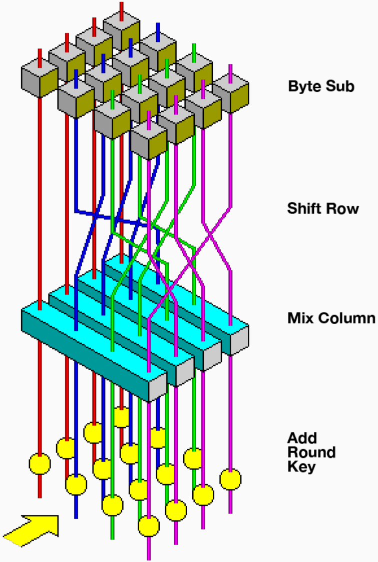
AES as a ZK-friendly Cipher?

- Picnic used LowMC – tailored to MPC, but less well analyzed than AES . . .

AES as a ZK-friendly Cipher?

- Picnic used LowMC – tailored to MPC, but less well analyzed than AES . . .
- AES is $\mathbb{F}_2$-linear except for the S-boxes

$$x \mapsto y = \begin{cases} 0 & \text{if } x = 0 \\ x^{-1} \in \mathbb{F}_{2^8} & \text{otherwise} \end{cases} \qquad (\star)$$



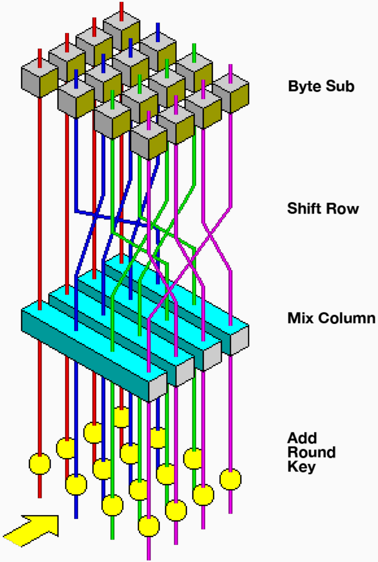
AES as a ZK-friendly Cipher?

- Picnic used LowMC – tailored to MPC, but less well analyzed than AES ...
- AES is $\mathbb{F}_2$-linear except for the S-boxes

$$x \mapsto y = \begin{cases} 0 & \text{if } x = 0 \\ x^{-1} \in \mathbb{F}_{2^8} & \text{otherwise} \end{cases} \qquad (\star)$$

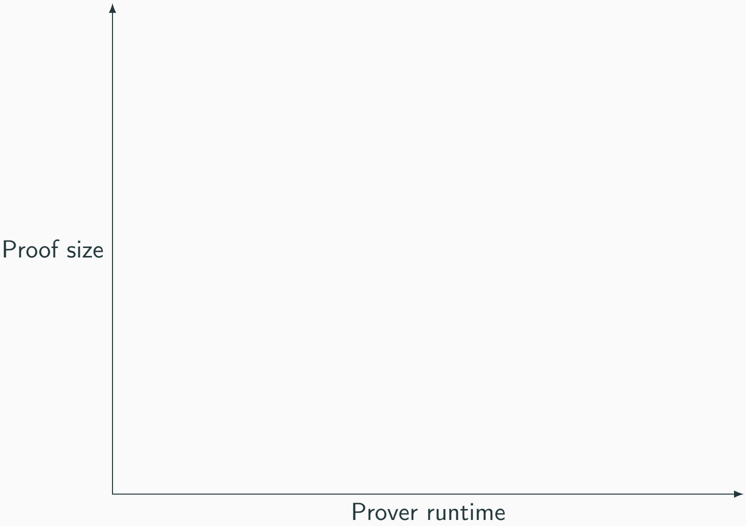
$\Longrightarrow$ Sample keys such that no zeros appear in the S-boxes and just check inversions ($x \cdot y = 1$ over $\mathbb{F}_{2^8}$)

$\rightsquigarrow$ AES-128: 200 quadratic constraints / 1600 bit witness

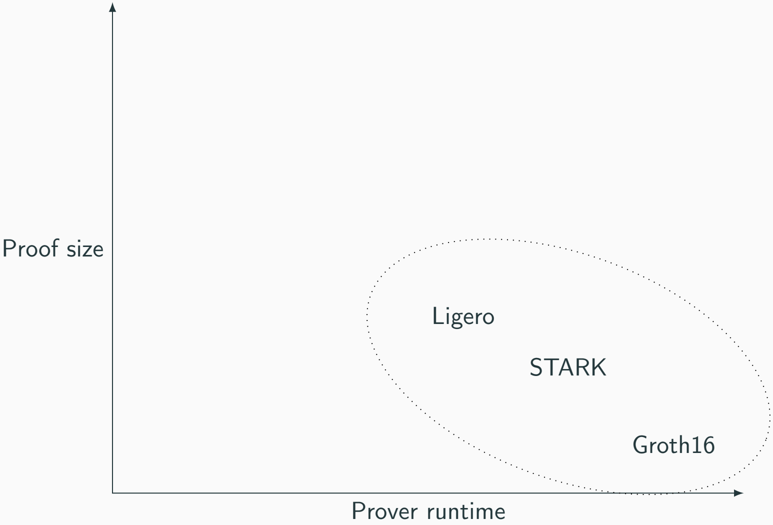


VOLE-based Zero-Knowledge

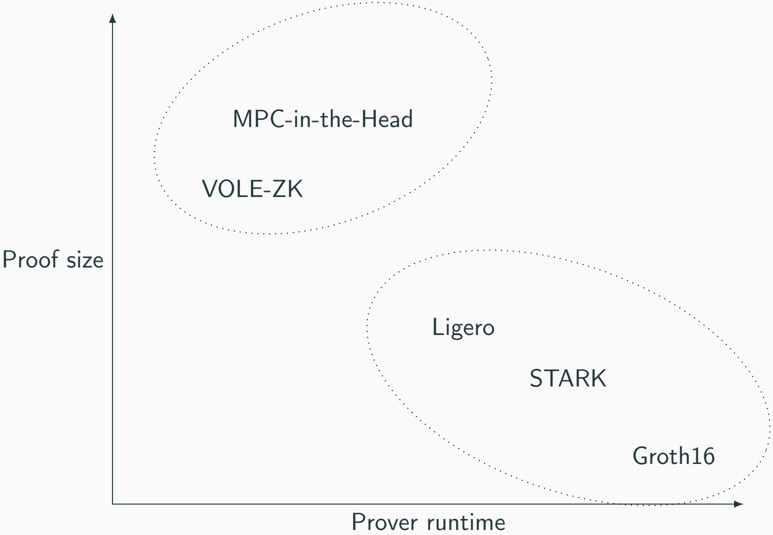
Space of Zero-Knowledge Proofs



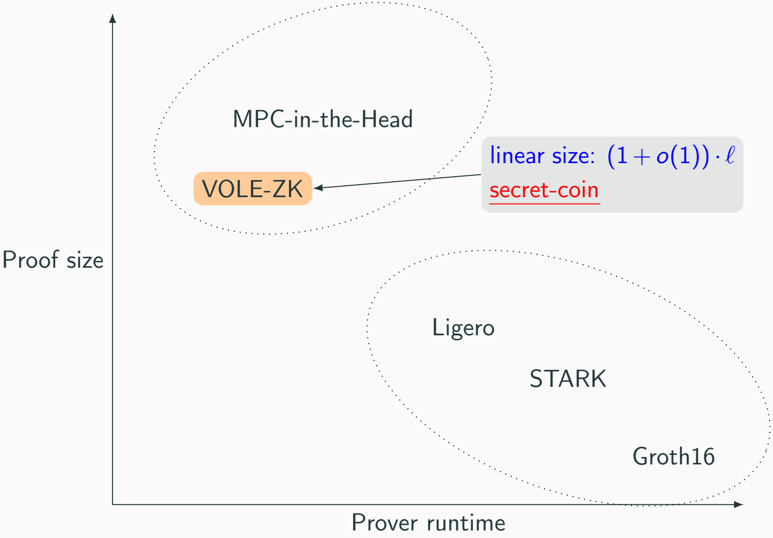
Space of Zero-Knowledge Proofs



Space of Zero-Knowledge Proofs



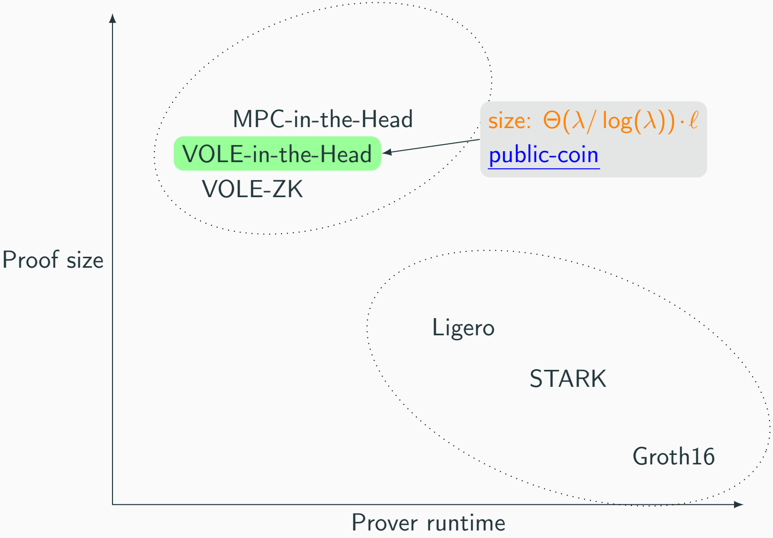
Space of Zero-Knowledge Proofs



linear size: $(1 + o(1)) \cdot \ell$

secret-coin

Space of Zero-Knowledge Proofs



What are VOLEs?

What are VOLEs?

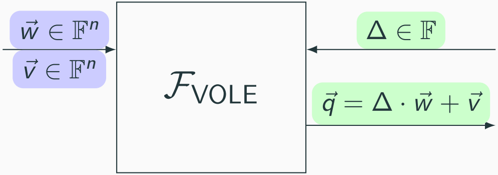
Vector Oblivious Linear Evaluation (VOLE)

Prover $\mathcal{P}$

(VOLE Sender)

Verifier $\mathcal{V}$

(VOLE Receiver)



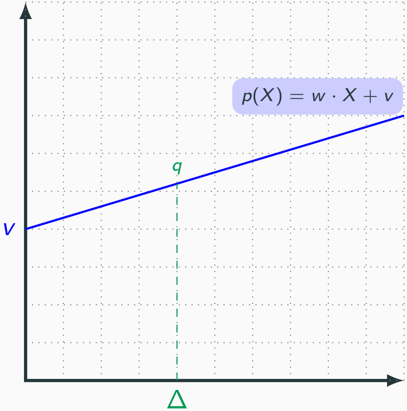
$\vec{w} \in \mathbb{F}^n$

$\vec{v} \in \mathbb{F}^n$

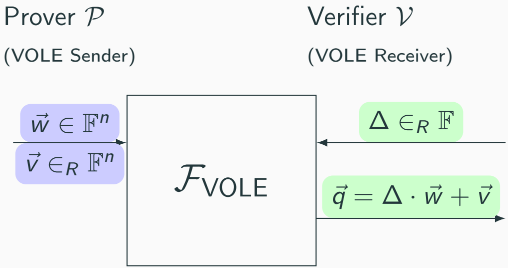
$\mathcal{F}_{\text{VOLE}}$

$\Delta \in \mathbb{F}$

$\vec{q} = \Delta \cdot \vec{w} + \vec{v}$



Vector Oblivious Linear Evaluation (VOLE) as Homomorphic Commitments



Prover $\mathcal{P}$

(VOLE Sender)

Verifier $\mathcal{V}$

(VOLE Receiver)

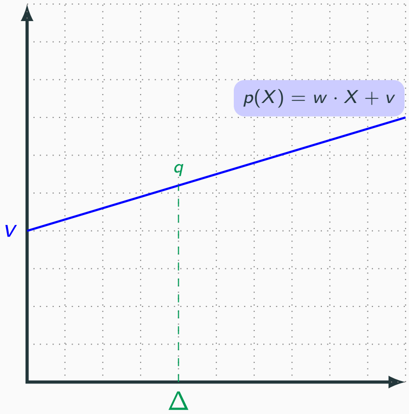
$\vec{w} \in \mathbb{F}^n$

$\vec{v} \in_R \mathbb{F}^n$

$\mathcal{F}_{\mathsf{VOLE}}$

$\Delta \in_R \mathbb{F}$

$\vec{q} = \Delta \cdot \vec{w} + \vec{v}$



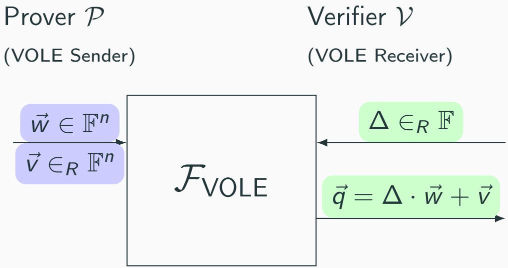
Linearly Homomorphic Commitments

use $q_i = w_i \cdot \Delta + v_i$ as information-theoretic MAC on w_i

- hiding since v_i is random
- breaking binding $\implies$ guessing Δ $\implies$ prob. $1/|\mathbb{F}|$

(cf. **EC:CatFio13 [EC:CatFio13]**, **EC:BDOZ11 [EC:BDOZ11]**)

Vector Oblivious Linear Evaluation (VOLE) as Homomorphic Commitments



Prover $\mathcal{P}$

(VOLE Sender)

Verifier $\mathcal{V}$

(VOLE Receiver)

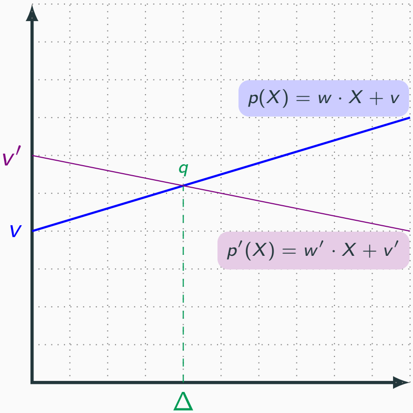
$\vec{w} \in \mathbb{F}^n$

$\vec{v} \in_R \mathbb{F}^n$

$\mathcal{F}_{\mathsf{VOLE}}$

$\Delta \in_R \mathbb{F}$

$\vec{q} = \Delta \cdot \vec{w} + \vec{v}$



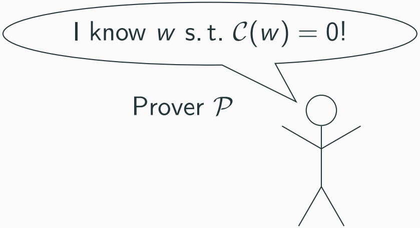
Linearly Homomorphic Commitments

use $q_i = w_i \cdot \Delta + v_i$ as information-theoretic MAC on w_i

- hiding since v_i is random
- breaking binding $\implies$ guessing Δ $\implies$ prob. $1/|\mathbb{F}|$

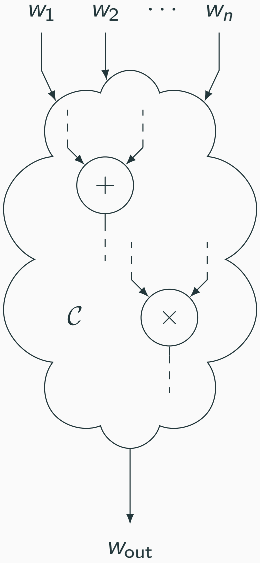
(cf. **EC:CatFio13 [EC:CatFio13]**, **EC:BDOZ11 [EC:BDOZ11]**)

Commit & Prove Zero-Knowledge

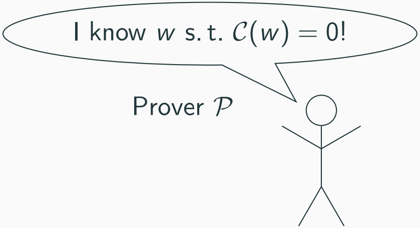


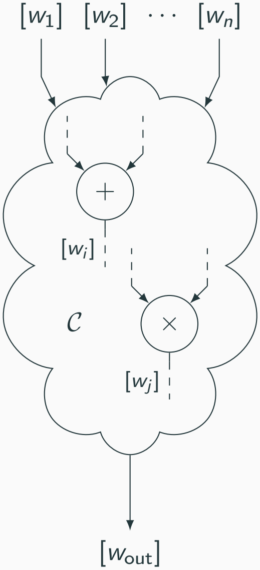
I know w s. t. $\mathcal{C}(w) = 0$!

Prover $\mathcal{P}$



Commit & Prove Zero-Knowledge

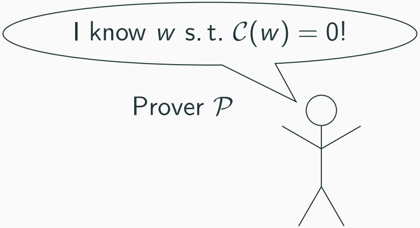




Ingredients:

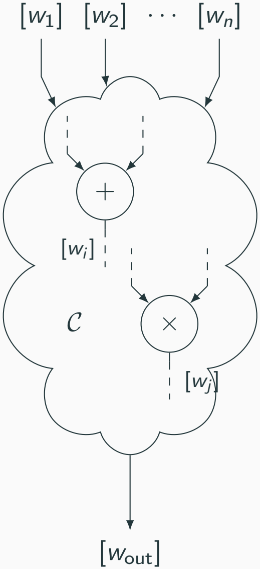
1. linearly homomorphic commitments $[\,\cdot\,]$
   - can compute $[z] \leftarrow a \cdot [x] + [y] + b$ ✅

Commit & Prove Zero-Knowledge



Ingredients:

1. linearly homomorphic commitments $[\,\cdot\,]$
   - can compute $[z] \leftarrow a \cdot [x] + [y] + b$ ✅
2. multiplication check
   - given $([a],[b],[c])$, verify $a \cdot b \overset{?}{=} c$



Verifying Multiplications – LPZK [ITC:DIO21], QuickSilver [CCS:YSWW21]

Goal: Given $([a], [b], [c])$, verify that $a \cdot b = c$ in $\mathbb{F}$

Verifying Multiplications – LPZK [ITC:DIO21], QuickSilver [CCS:YSWW21]

Goal: Given $([a],[b],[c])$, verify that $a \cdot b = c$ in $\mathbb{F}$

QuickSilver Check: Convert the three MAC equations $q_x = v_x + x \cdot \Delta$ for $x \in \{a, b, c\}$ into a polynomial in Δ:

$$\underbrace{\Delta \cdot q_c - q_a \cdot q_b}_{\text{known by } \mathcal{V}} = \underbrace{(-v_a \cdot v_b)}_{\text{known by } \mathcal{P}} + \underbrace{(v_c - a \cdot v_b - b \cdot v_a)}_{\text{known by } \mathcal{P}} \cdot \Delta + \underbrace{(c - a \cdot b)}_{= 0 \text{ if } \mathcal{P} \text{ honest}} \cdot \Delta^2$$

Verifying Multiplications – LPZK [ITC:DIO21], QuickSilver [CCS:YSWW21]

Goal: Given $([a],[b],[c])$, verify that $a \cdot b = c$ in $\mathbb{F}$

QuickSilver Check: Convert the three MAC equations $q_x = v_x + x \cdot \Delta$ for $x \in \{a,b,c\}$ into a polynomial in Δ:

$$\underbrace{\Delta \cdot q_c - q_a \cdot q_b}_{\text{known by } \mathcal{V}} = \underbrace{(-v_a \cdot v_b)}_{\text{known by } \mathcal{P}} + \underbrace{(v_c - a \cdot v_b - b \cdot v_a)}_{\text{known by } \mathcal{P}} \cdot \Delta + \underbrace{(c - a \cdot b)}_{=0 \text{ if } \mathcal{P} \text{ honest}} \cdot \Delta^2$$

use a random linear combination to verify many multiplications

Verifying Multiplications – LPZK [ITC:DIO21], QuickSilver [CCS:YSWW21]

Goal: Given $([a],[b],[c])$, verify that $a \cdot b = c$ in $\mathbb{F}$

QuickSilver Check: Convert the three MAC equations $q_x = v_x + x \cdot \Delta$ for $x \in \{a,b,c\}$ into a polynomial in Δ:

$$\underbrace{\Delta \cdot q_c - q_a \cdot q_b}_{\text{known by } \mathcal{V}} = \underbrace{(-v_a \cdot v_b)}_{\text{known by } \mathcal{P}} + \underbrace{(v_c - a \cdot v_b - b \cdot v_a)}_{\text{known by } \mathcal{P}} \cdot \Delta + \underbrace{(c - a \cdot b)}_{= 0 \text{ if } \mathcal{P} \text{ honest}} \cdot \Delta^2$$

Soundness: cheating $\mathcal{P}$ needs to come up with $p(X) = e_0 + e_1 \cdot X + e \cdot X^2$ such that

Verifying Multiplications – LPZK [ITC:DIO21], QuickSilver [CCS:YSWW21]

Goal: Given $([a], [b], [c])$, verify that $a \cdot b = c$ in $\mathbb{F}$

QuickSilver Check: Convert the three MAC equations $q_x = v_x + x \cdot \Delta$ for $x \in \{a, b, c\}$ into a polynomial in Δ:

$$\underbrace{\Delta \cdot q_c - q_a \cdot q_b}_{\text{known by } \mathcal{V}} = \underbrace{(-v_a \cdot v_b)}_{\text{known by } \mathcal{P}} + \underbrace{(v_c - a \cdot v_b - b \cdot v_a)}_{\text{known by } \mathcal{P}} \cdot \Delta + \underbrace{(c - a \cdot b)}_{= 0 \text{ if } \mathcal{P} \text{ honest}} \cdot \Delta^2$$

Soundness: cheating $\mathcal{P}$ needs to come up with $p(X) = e_0 + e_1 \cdot X + e \cdot X^2$ such that

- $p(\Delta) = 0$, and

Verifying Multiplications – LPZK [ITC:DIO21], QuickSilver [CCS:YSWW21]

Goal: Given $([a],[b],[c])$, verify that $a \cdot b = c$ in $\mathbb{F}$

QuickSilver Check: Convert the three MAC equations $q_x = v_x + x \cdot \Delta$ for $x \in \{a,b,c\}$ into a polynomial in Δ:

$$\underbrace{\Delta \cdot q_c - q_a \cdot q_b}_{\text{known by } \mathcal{V}} = \underbrace{(-v_a \cdot v_b)}_{\text{known by } \mathcal{P}} + \underbrace{(v_c - a \cdot v_b - b \cdot v_a)}_{\text{known by } \mathcal{P}} \cdot \Delta + \underbrace{(c - a \cdot b)}_{= 0 \text{ if } \mathcal{P} \text{ honest}} \cdot \Delta^2$$

Soundness: cheating $\mathcal{P}$ needs to come up with $p(X) = e_0 + e_1 \cdot X + e \cdot X^2$ such that

- $p(\Delta) = 0$, and
- $e := c - a \cdot b \neq 0$

Verifying Multiplications – LPZK [ITC:DIO21], QuickSilver [CCS:YSWW21]

Goal: Given $([a],[b],[c])$, verify that $a \cdot b = c$ in $\mathbb{F}$

QuickSilver Check: Convert the three MAC equations $q_x = v_x + x \cdot \Delta$ for $x \in \{a, b, c\}$ into a polynomial in Δ:

$$\underbrace{\Delta \cdot q_c - q_a \cdot q_b}_{\text{known by } \mathcal{V}} = \underbrace{(-v_a \cdot v_b)}_{\text{known by } \mathcal{P}} + \underbrace{(v_c - a \cdot v_b - b \cdot v_a)}_{\text{known by } \mathcal{P}} \cdot \Delta + \underbrace{(c - a \cdot b)}_{= 0 \text{ if } \mathcal{P} \text{ honest}} \cdot \Delta^2$$

Soundness: cheating $\mathcal{P}$ needs to come up with $p(X) = e_0 + e_1 \cdot X + e \cdot X^2$ such that

- $p(\Delta) = 0$, and
- $e := c - a \cdot b \neq 0$

$\implies$ p has degree 2 $\implies$ p has at most 2 roots $\implies$ soundness error $^2/_{|\mathbb{F}|}$

VOLE-in-the-Head

VOLE-in-the-Head – The Idea

Observation: Why are VOLE-ZK protocols not public coin?

VOLE-in-the-Head – The Idea

Observation: Why are VOLE-ZK protocols not public coin?

- If the prover $\mathcal{P}$ knows Δ, the commitments are no longer binding!

VOLE-in-the-Head – The Idea

Observation: Why are VOLE-ZK protocols not public coin?

- If the prover $\mathcal{P}$ knows Δ, the commitments are no longer binding!
- But: At the end of the protocol, it is fine for $\mathcal{P}$ to learn Δ.

VOLE-in-the-Head – The Idea

Observation: Why are VOLE-ZK protocols not public coin?

- If the prover $\mathcal{P}$ knows Δ, the commitments are no longer binding!
- But: At the end of the protocol, it is fine for $\mathcal{P}$ to learn Δ.

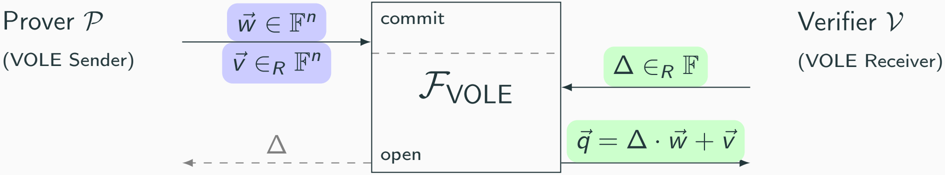
$\implies$ commit $\mathcal{P}$ to its messages and delay $\mathcal{V}$'s choice Δ to the end of the protocol

VOLE-in-the-Head – The Idea

Observation: Why are VOLE-ZK protocols not public coin?

- If the prover $\mathcal{P}$ knows Δ, the commitments are no longer binding!
- But: At the end of the protocol, it is fine for $\mathcal{P}$ to learn Δ.

$\implies$ commit $\mathcal{P}$ to its messages and delay $\mathcal{V}$'s choice Δ to the end of the protocol

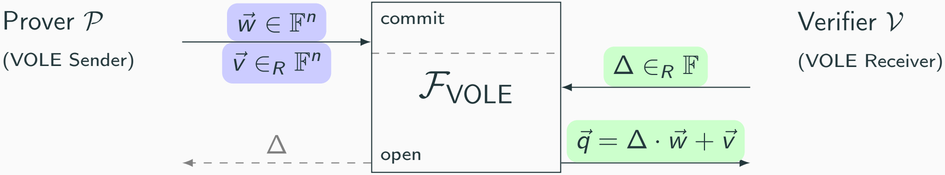

Prover $\mathcal{P}$
(VOLE Sender)
$\vec{w} \in \mathbb{F}^n$
$\vec{v} \in_R \mathbb{F}^n$
Δ
commit
$\mathcal{F}_{\text{VOLE}}$
open
$\Delta \in_R \mathbb{F}$
$\vec{q} = \Delta \cdot \vec{w} + \vec{v}$
Verifier $\mathcal{V}$
(VOLE Receiver)

VOLE-in-the-Head – The Idea

Observation: Why are VOLE-ZK protocols not public coin?

- If the prover $\mathcal{P}$ knows Δ, the commitments are no longer binding!
- But: At the end of the protocol, it is fine for $\mathcal{P}$ to learn Δ.

$\implies$ commit $\mathcal{P}$ to its messages and delay $\mathcal{V}$'s choice Δ to the end of the protocol



Prover $\mathcal{P}$
(VOLE Sender)
$\vec{w} \in \mathbb{F}^n$
$\vec{v} \in_R \mathbb{F}^n$
Δ
commit
$\mathcal{F}_{\text{VOLE}}$
open
$\Delta \in_R \mathbb{F}$
$\vec{q} = \Delta \cdot \vec{w} + \vec{v}$
Verifier $\mathcal{V}$
(VOLE Receiver)

Implement $\mathcal{F}_{\text{VOLE}}$ with SoftSpoken VOLE [**C:Roy22**].

Small-Field SoftSpoken VOLE

Input: An $\binom{N}{N-1}$-OT, for $N = 2^k \leq \text{poly}(\lambda)$:

$\mathcal{P}$ has seeds sd_x for all $x \in \mathbb{F}_{2^k}$. $\mathcal{V}$ has $\Delta \in \mathbb{F}_{2^k}$ and all seeds except sd_Δ.

Small-Field SoftSpoken VOLE

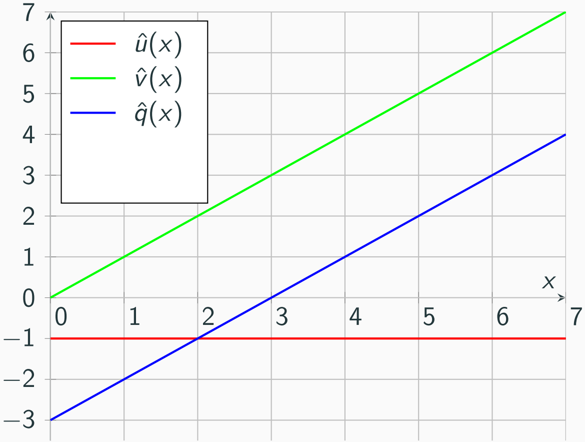
Input: An $\binom{N}{N-1}$-OT, for $N = 2^k \leq \mathrm{poly}(\lambda)$:
$\mathcal{P}$ has seeds sd_x for all $x \in \mathbb{F}_{2^k}$. $\mathcal{V}$ has $\Delta \in \mathbb{F}_{2^k}$ and all seeds except sd_Δ.

$$\vec{u} = -\sum_{x \in \mathbb{F}_{2^k}} G(\mathrm{sd}_x) = \sum_{x \in \mathbb{F}_{2^k}} \hat{u}(x) G(\mathrm{sd}_x)$$

$$\vec{v} = \sum_{x \in \mathbb{F}_{2^k}} x G(\mathrm{sd}_x) = \sum_{x \in \mathbb{F}_{2^k}} \hat{v}(x) G(\mathrm{sd}_x)$$

$$\vec{q} = \sum_{x \in \mathbb{F}_{2^k}} (x - \Delta) G(\mathrm{sd}_x)$$

$$= \sum_{x \in \mathbb{F}_{2^k}} \hat{q}(x) G(\mathrm{sd}_x)$$



Small-Field SoftSpoken VOLE

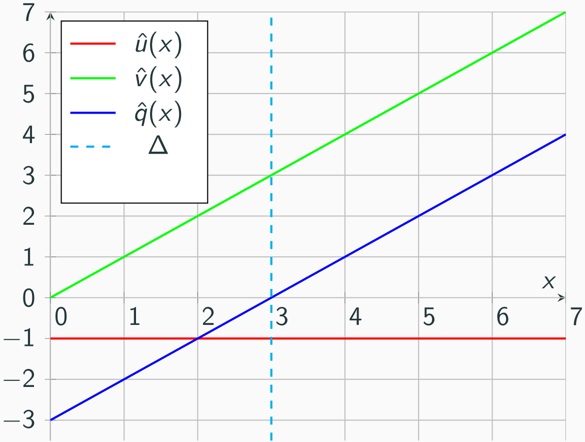
Input: An $\binom{N}{N-1}$-OT, for $N = 2^k \leq \mathrm{poly}(\lambda)$:
$\mathcal{P}$ has seeds sd_x for all $x \in \mathbb{F}_{2^k}$. $\mathcal{V}$ has $\Delta \in \mathbb{F}_{2^k}$ and all seeds except sd_Δ.

$$\vec{u} = -\sum_{x\in\mathbb{F}_{2^k}} G(\mathrm{sd}_x) = \sum_{x\in\mathbb{F}_{2^k}} \hat{u}(x) G(\mathrm{sd}_x)$$

$$\vec{v} = \sum_{x\in\mathbb{F}_{2^k}} x G(\mathrm{sd}_x) = \sum_{x\in\mathbb{F}_{2^k}} \hat{v}(x) G(\mathrm{sd}_x)$$

$$\vec{q} = \sum_{x\in\mathbb{F}_{2^k}} (x - \Delta) G(\mathrm{sd}_x)$$

$$= \sum_{x\in\mathbb{F}_{2^k}} \hat{q}(x) G(\mathrm{sd}_x)$$



Small-Field SoftSpoken VOLE

Input: An $\binom{N}{N-1}$-OT, for $N = 2^k \leq \mathrm{poly}(\lambda)$:
$\mathcal{P}$ has seeds sd_x for all $x \in \mathbb{F}_{2^k}$. $\mathcal{V}$ has $\Delta \in \mathbb{F}_{2^k}$ and all seeds except sd_Δ.

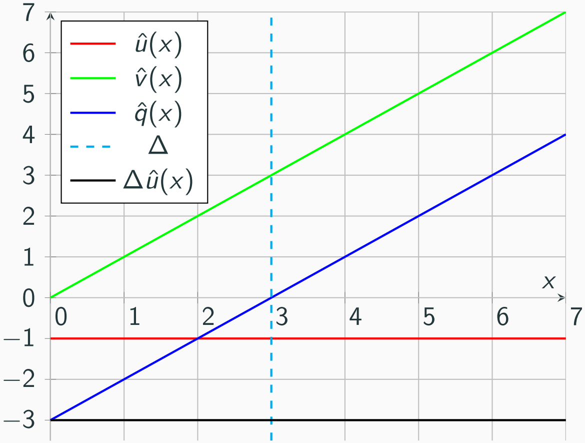
$$\vec{u} = -\sum_{x\in\mathbb{F}_{2^k}} G(\mathrm{sd}_x) = \sum_{x\in\mathbb{F}_{2^k}} \hat{u}(x) G(\mathrm{sd}_x)$$

$$\vec{v} = \sum_{x\in\mathbb{F}_{2^k}} x G(\mathrm{sd}_x) = \sum_{x\in\mathbb{F}_{2^k}} \hat{v}(x) G(\mathrm{sd}_x)$$

$$\vec{q} = \sum_{x\in\mathbb{F}_{2^k}} (x - \Delta) G(\mathrm{sd}_x)$$

$$= \sum_{x\in\mathbb{F}_{2^k}} \hat{q}(x) G(\mathrm{sd}_x)$$

$$\vec{q} - \vec{v} = \sum_{x\in\mathbb{F}_{2^k}} (-\Delta) G(\mathrm{sd}_x) = \Delta \vec{u}$$



Small-Field SoftSpoken VOLE

Input: An $\binom{N}{N-1}$-OT, for $N = 2^k \leq \mathrm{poly}(\lambda)$:
$\mathcal{P}$ has seeds sd_x for all $x \in \mathbb{F}_{2^k}$. $\mathcal{V}$ has $\Delta \in \mathbb{F}_{2^k}$ and all seeds except sd_Δ.

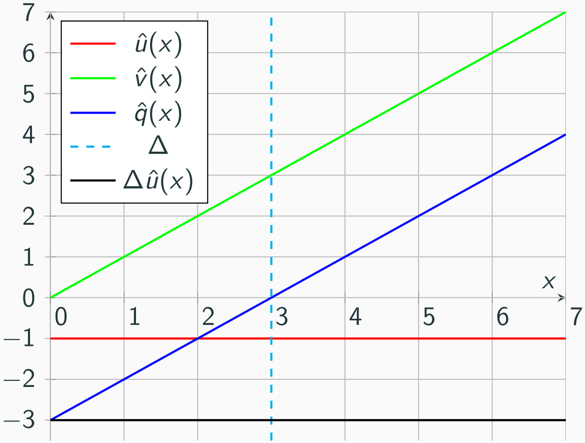
$$\vec{u} = -\sum_{x \in \mathbb{F}_{2^k}} G(\mathrm{sd}_x) = \sum_{x \in \mathbb{F}_{2^k}} \hat{u}(x) G(\mathrm{sd}_x)$$

$$\vec{v} = \sum_{x \in \mathbb{F}_{2^k}} x G(\mathrm{sd}_x) = \sum_{x \in \mathbb{F}_{2^k}} \hat{v}(x) G(\mathrm{sd}_x)$$

$$\vec{q} = \sum_{x \in \mathbb{F}_{2^k}} (x - \Delta) G(\mathrm{sd}_x)$$

$$= \sum_{x \in \mathbb{F}_{2^k}} \hat{q}(x) G(\mathrm{sd}_x)$$

$$\vec{q} - \vec{v} = \sum_{x \in \mathbb{F}_{2^k}} (-\Delta) G(\mathrm{sd}_x) = \Delta \vec{u}$$



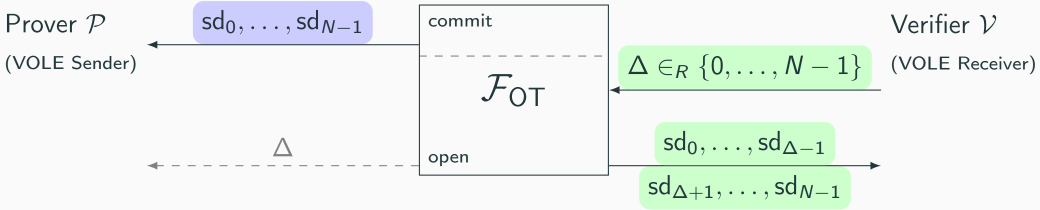
Derandomization: $\mathcal{P}$ sends $\vec{d} = \vec{w} - \vec{u}$. $\mathcal{V}$ updates $\vec{q}' = \vec{q} + \Delta \vec{d}$.

OT-in-the-Head: Commitments

How to get an $\binom{N}{N-1}$-OT for the VOLE?

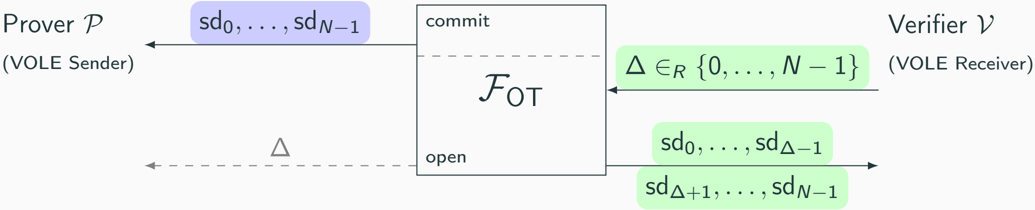
OT-in-the-Head: Commitments

How to get an $\binom{N}{N-1}$-OT for the VOLE?


Prover $\mathcal{P}$
(VOLE Sender)
$\mathsf{sd}_0, \ldots, \mathsf{sd}_{N-1}$
Δ
commit
$\mathcal{F}_{\mathsf{OT}}$
open
Verifier $\mathcal{V}$
(VOLE Receiver)
$\Delta \in_R \{0, \ldots, N-1\}$
$\mathsf{sd}_0, \ldots, \mathsf{sd}_{\Delta-1}$
$\mathsf{sd}_{\Delta+1}, \ldots, \mathsf{sd}_{N-1}$

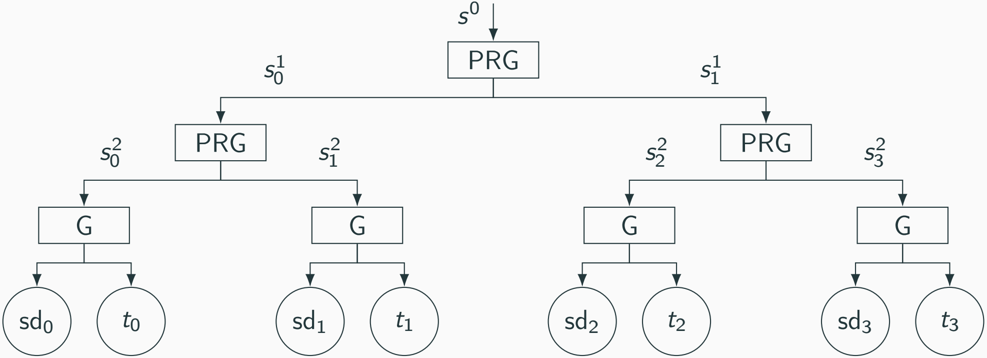
OT-in-the-Head: Commitments

How to get an $\binom{N}{N-1}$-OT for the VOLE?

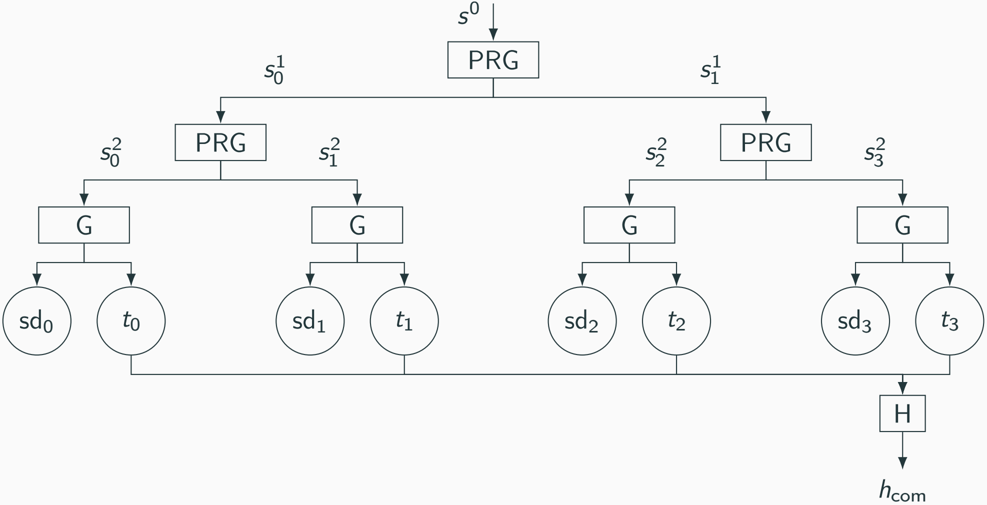

Prover $\mathcal{P}$
(VOLE Sender)
$\mathsf{sd}_0, \ldots, \mathsf{sd}_{N-1}$
commit
$\mathcal{F}_{\mathsf{OT}}$
open
Δ
Verifier $\mathcal{V}$
(VOLE Receiver)
$\Delta \in_R \{0, \ldots, N-1\}$
$\mathsf{sd}_0, \ldots, \mathsf{sd}_{\Delta-1}$
$\mathsf{sd}_{\Delta+1}, \ldots, \mathsf{sd}_{N-1}$

This is just a commitment scheme!

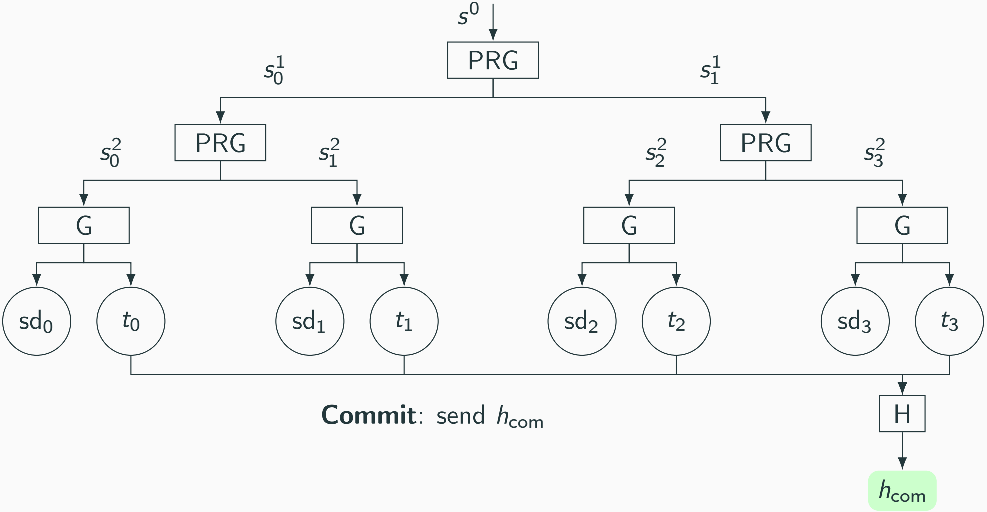
All-but-one Random Vector Commitments



All-but-one Random Vector Commitments

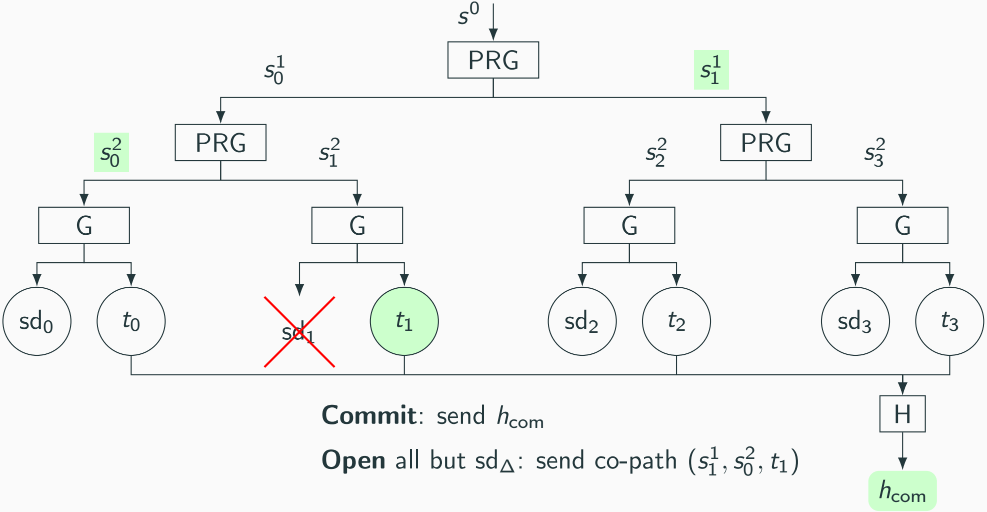


All-but-one Random Vector Commitments



Commit: send h_{com}

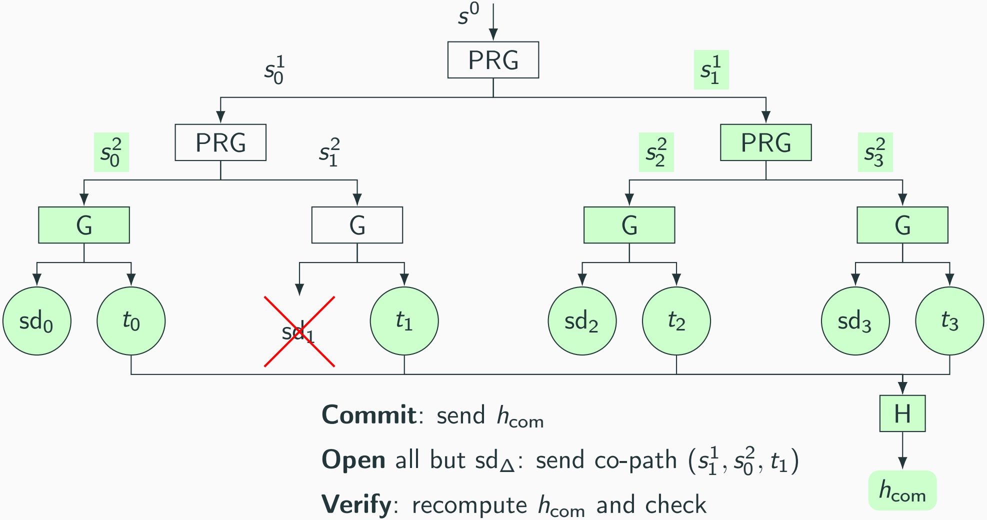
All-but-one Random Vector Commitments



Commit: send h_{com}

Open all but sd_Δ: send co-path (s_1^1, s_0^2, t_1)

All-but-one Random Vector Commitments



Commit: send h_{com}

Open all but sd_Δ: send co-path (s_1^1, s_0^2, t_1)

Verify: recompute h_{com} and check

Small VOLE to Big VOLE

Small VOLE costs $\mathcal{O}(N)$ work, but gives only soundness $\frac{1}{N}$!

$\implies$ need VOLE over a big field $\mathbb{F}_{2^\lambda}$ and Δ from large set.

Small VOLE to Big VOLE

Small VOLE costs $\mathcal{O}(N)$ work, but gives only soundness $\frac{1}{N}$!

$\implies$ need VOLE over a big field $\mathbb{F}_{2^\lambda}$ and Δ from large set.

$\rightsquigarrow$ combine τ small VOLEs into one big VOLE, where $N^\tau = 2^\lambda$. Proof size: $\tau \times$ witness size.

Small VOLE to Big VOLE

Small VOLE costs $\mathcal{O}(N)$ work, but gives only soundness $\frac{1}{N}$!

$\implies$ need VOLE over a big field $\mathbb{F}_{2^\lambda}$ and Δ from large set.

$\rightsquigarrow$ combine τ small VOLEs into one big VOLE, where $N^\tau = 2^\lambda$. Proof size: $\tau \times$ witness size.

$$\begin{aligned} \vec{q}_0 &= \vec{w} \cdot \Delta_0 + \vec{v}_0 \\ &\vdots \\ \vec{q}_{\tau-1} &= \vec{w} \cdot \Delta_{\tau-1} + \vec{v}_{\tau-1} \end{aligned}$$

Small VOLE to Big VOLE

Small VOLE costs $\mathcal{O}(N)$ work, but gives only soundness $\frac{1}{N}$!

$\implies$ need VOLE over a big field $\mathbb{F}_{2^\lambda}$ and Δ from large set.

$\rightsquigarrow$ combine τ small VOLEs into one big VOLE, where $N^\tau = 2^\lambda$. Proof size: $\tau \times$ witness size.

$$\vec{q}_0 = \vec{w} \cdot \Delta_0 + \vec{v}_0$$

$$\vdots$$

$$\vec{q}_{\tau-1} = \vec{w} \cdot \Delta_{\tau-1} + \vec{v}_{\tau-1}$$

$$\Downarrow$$

$$\underbrace{\sum_{i\in[\tau]} \vec{q}_i \cdot X^i}_{\vec{q}\in\mathbb{F}^\ell_{q^\tau}} = \vec{w} \cdot \underbrace{\sum_{i\in[\tau]} \Delta_i \cdot X^i}_{\Delta\in\mathbb{F}_{q^\tau}} + \underbrace{\sum_{i\in[\tau]} \vec{v}_i \cdot X^i}_{\vec{v}\in\mathbb{F}^\ell_{q^\tau}}$$

Small VOLE to Big VOLE

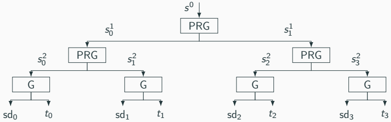
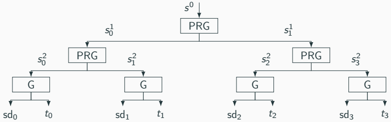
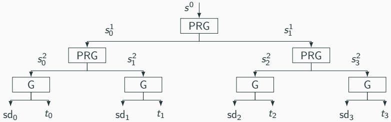
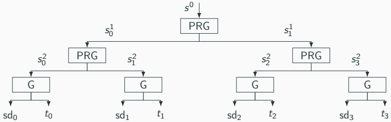
Small VOLE costs $\mathcal{O}(N)$ work, but gives only soundness $\frac{1}{N}$!

$\implies$ need VOLE over a big field $\mathbb{F}_{2^\lambda}$ and Δ from large set.
$\rightsquigarrow$ combine τ small VOLEs into one big VOLE, where $N^\tau = 2^\lambda$. Proof size: $\tau \times$ witness size.

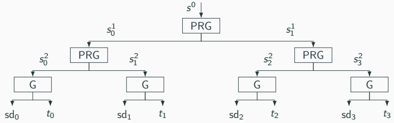
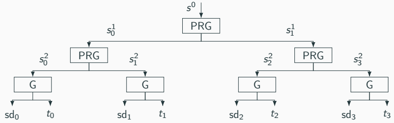
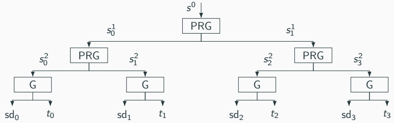
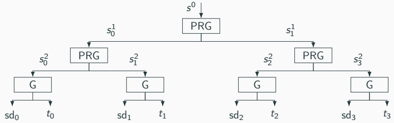
$$\begin{aligned}
\vec{q}_0 &= \vec{w} \cdot \Delta_0 + \vec{v}_0 \\
&\vdots \\
\vec{q}_{\tau-1} &= \vec{w} \cdot \Delta_{\tau-1} + \vec{v}_{\tau-1} \\
&\Downarrow \\
\underbrace{\sum_{i \in [\tau]} \vec{q}_i \cdot X^i}_{\vec{q} \in \mathbb{F}_{q^\tau}^\ell} &= \vec{w} \cdot \underbrace{\sum_{i \in [\tau]} \Delta_i \cdot X^i}_{\Delta \in \mathbb{F}_{q^\tau}} + \underbrace{\sum_{i \in [\tau]} \vec{v}_i \cdot X^i}_{\vec{v} \in \mathbb{F}_{q^\tau}^\ell}
\end{aligned}$$

Use a consistency check to verify that the same $\vec{w}$ was used in every VOLE.

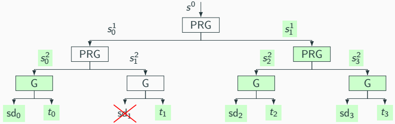
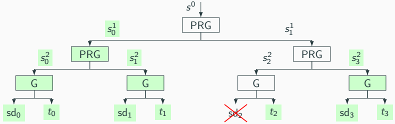
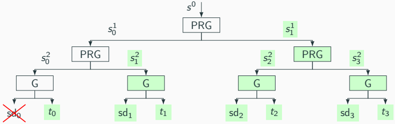
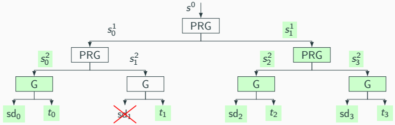
All-but-some Random Vector Commitments

		Field element $x \in [0, 2^k)$				Commitment
		0	1	2	3	
Repetition $i \in [0, \tau)$	0	$\mathrm{sd}_{0,0}$	$\mathrm{sd}_{0,1}$	$\mathrm{sd}_{0,2}$	$\mathrm{sd}_{0,3}$	 s^0 PRG s^1_0 s^1_1 s^2_0 PRG s^2_1 s^2_2 PRG s^2_3 G G G G sd_0 t_0 sd_1 t_1 sd_2 t_2 sd_3 t_3
	1	$\mathrm{sd}_{1,0}$	$\mathrm{sd}_{1,1}$	$\mathrm{sd}_{1,2}$	$\mathrm{sd}_{1,3}$	 s^0 PRG s^1_0 s^1_1 s^2_0 PRG s^2_1 s^2_2 PRG s^2_3 G G G G sd_0 t_0 sd_1 t_1 sd_2 t_2 sd_3 t_3
	2	$\mathrm{sd}_{2,0}$	$\mathrm{sd}_{2,1}$	$\mathrm{sd}_{2,2}$	$\mathrm{sd}_{2,3}$	 s^0 PRG s^1_0 s^1_1 s^2_0 PRG s^2_1 s^2_2 PRG s^2_3 G G G G sd_0 t_0 sd_1 t_1 sd_2 t_2 sd_3 t_3
	3	$\mathrm{sd}_{3,0}$	$\mathrm{sd}_{3,1}$	$\mathrm{sd}_{3,2}$	$\mathrm{sd}_{3,3}$	 s^0 PRG s^1_0 s^1_1 s^2_0 PRG s^2_1 s^2_2 PRG s^2_3 G G G G sd_0 t_0 sd_1 t_1 sd_2 t_2 sd_3 t_3

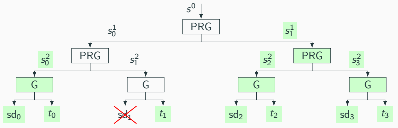
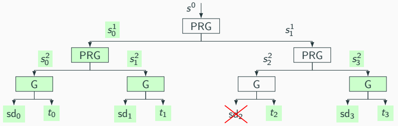
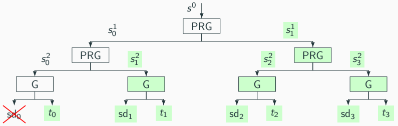
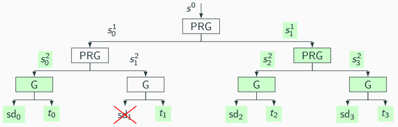
All-but-some Random Vector Commitments

		Field element $x \in [0, 2^k)$				Commitment
		0	1	2	3	
Repetition $i \in [0, \tau)$	0	$\mathrm{sd}_{0,0}$	$\xcancel{\mathrm{sd}_{0,1}}$	$\mathrm{sd}_{0,2}$	$\mathrm{sd}_{0,3}$	 s^0 PRG s_0^1 s_1^1 PRG PRG s_0^2 s_1^2 s_2^2 s_3^2 G G G G sd_0 t_0 sd_1 t_1 sd_2 t_2 sd_3 t_3
	1	$\mathrm{sd}_{1,0}$	$\mathrm{sd}_{1,1}$	$\xcancel{\mathrm{sd}_{1,2}}$	$\mathrm{sd}_{1,3}$	 s^0 PRG s_0^1 s_1^1 PRG PRG s_0^2 s_1^2 s_2^2 s_3^2 G G G G sd_0 t_0 sd_1 t_1 sd_2 t_2 sd_3 t_3
	2	$\xcancel{\mathrm{sd}_{2,0}}$	$\mathrm{sd}_{2,1}$	$\mathrm{sd}_{2,2}$	$\mathrm{sd}_{2,3}$	 s^0 PRG s_0^1 s_1^1 PRG PRG s_0^2 s_1^2 s_2^2 s_3^2 G G G G sd_0 t_0 sd_1 t_1 sd_2 t_2 sd_3 t_3
	3	$\mathrm{sd}_{3,0}$	$\xcancel{\mathrm{sd}_{3,1}}$	$\mathrm{sd}_{3,2}$	$\mathrm{sd}_{3,3}$	 s^0 PRG s_0^1 s_1^1 PRG PRG s_0^2 s_1^2 s_2^2 s_3^2 G G G G sd_0 t_0 sd_1 t_1 sd_2 t_2 sd_3 t_3

All-but-some Random Vector Commitments

		Field element $x \in [0, 2^k)$				Commitment
		0	1	2	3	
Repetition $i \in [0, \tau)$	0	$\mathsf{sd}_{0,0}$	$\xcancel{\mathsf{sd}_{0,1}}$	$\mathsf{sd}_{0,2}$	$\mathsf{sd}_{0,3}$	 s^0 PRG s^1_0 s^1_1 PRG PRG s^2_0 s^2_1 s^2_2 s^2_3 G G G G sd_0 t_0 $\xcancel{\mathsf{sd}_1}$ t_1 sd_2 t_2 sd_3 t_3
	1	$\mathsf{sd}_{1,0}$	$\mathsf{sd}_{1,1}$	$\xcancel{\mathsf{sd}_{1,2}}$	$\mathsf{sd}_{1,3}$	 s^0 PRG s^1_0 s^1_1 PRG PRG s^2_0 s^2_1 s^2_2 s^2_3 G G G G sd_0 t_0 sd_1 t_1 $\xcancel{\mathsf{sd}_2}$ t_2 sd_3 t_3
	2	$\xcancel{\mathsf{sd}_{2,0}}$	$\mathsf{sd}_{2,1}$	$\mathsf{sd}_{2,2}$	$\mathsf{sd}_{2,3}$	 s^0 PRG s^1_0 s^1_1 PRG PRG s^2_0 s^2_1 s^2_2 s^2_3 G G G G $\xcancel{\mathsf{sd}_0}$ t_0 sd_1 t_1 sd_2 t_2 sd_3 t_3
	3	$\mathsf{sd}_{3,0}$	$\xcancel{\mathsf{sd}_{3,1}}$	$\mathsf{sd}_{3,2}$	$\mathsf{sd}_{3,3}$	 s^0 PRG s^1_0 s^1_1 PRG PRG s^2_0 s^2_1 s^2_2 s^2_3 G G G G sd_0 t_0 $\xcancel{\mathsf{sd}_1}$ t_1 sd_2 t_2 sd_3 t_3

All-but-some Random Vector Commitments

		Field element $x \in [0, 2^k)$				Commitment
		0	1	2	3	
Repetition $i \in [0, \tau)$	0	$\mathsf{sd}_{0,0}$	$\xcancel{\mathsf{sd}_{0,1}}$	$\mathsf{sd}_{0,2}$	$\mathsf{sd}_{0,3}$	 s^0 PRG s^1_0 s^1_1 PRG PRG s^2_0 s^2_1 s^2_2 s^2_3 G G G G sd_0 t_0 sd_1 t_1 sd_2 t_2 sd_3 t_3
	1	$\mathsf{sd}_{1,0}$	$\mathsf{sd}_{1,1}$	$\xcancel{\mathsf{sd}_{1,2}}$	$\mathsf{sd}_{1,3}$	 s^0 PRG s^1_0 s^1_1 PRG PRG s^2_0 s^2_1 s^2_2 s^2_3 G G G G sd_0 t_0 sd_1 t_1 sd_2 t_2 sd_3 t_3
	2	$\xcancel{\mathsf{sd}_{2,0}}$	$\mathsf{sd}_{2,1}$	$\mathsf{sd}_{2,2}$	$\mathsf{sd}_{2,3}$	 s^0 PRG s^1_0 s^1_1 PRG PRG s^2_0 s^2_1 s^2_2 s^2_3 G G G G sd_0 t_0 sd_1 t_1 sd_2 t_2 sd_3 t_3
	3	$\mathsf{sd}_{3,0}$	$\xcancel{\mathsf{sd}_{3,1}}$	$\mathsf{sd}_{3,2}$	$\mathsf{sd}_{3,3}$	 s^0 PRG s^1_0 s^1_1 PRG PRG s^2_0 s^2_1 s^2_2 s^2_3 G G G G sd_0 t_0 sd_1 t_1 sd_2 t_2 sd_3 t_3

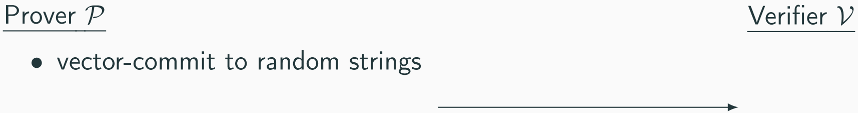
Because $N^\tau = 2^\lambda$, the co-paths always have λ nodes, so opening costs roughly λ^2 bits.

FAEST Rounds

Prover $\mathcal{P}$

Verifier $\mathcal{V}$

FAEST Rounds



Prover $\mathcal{P}$

Verifier $\mathcal{V}$

- vector-commit to random strings

FAEST Rounds

Prover $\mathcal{P}$

- vector-commit to random strings
- expand small VOLEs $\longrightarrow$

Verifier $\mathcal{V}$

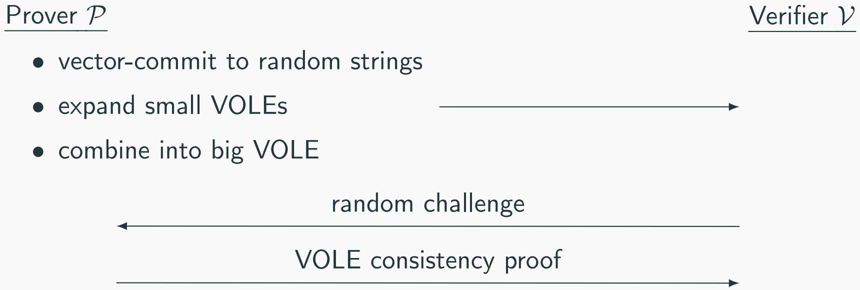
FAEST Rounds

Prover $\mathcal{P}$

Verifier $\mathcal{V}$

- vector-commit to random strings
- expand small VOLEs
- combine into big VOLE

FAEST Rounds



Prover $\mathcal{P}$

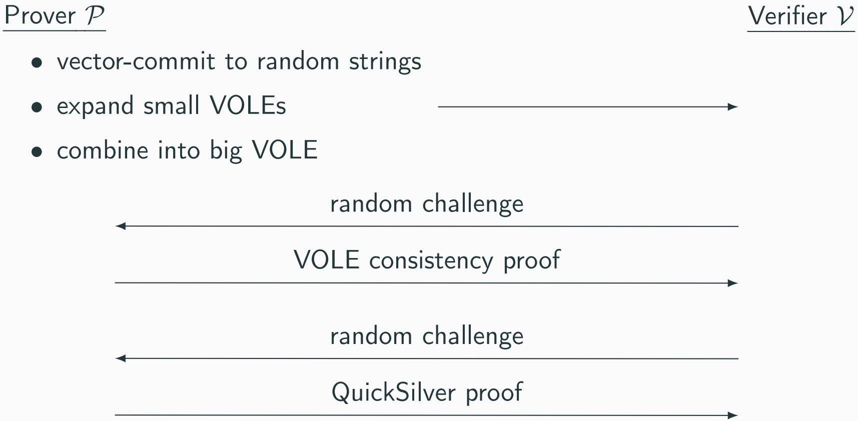
Verifier $\mathcal{V}$

- vector-commit to random strings
- expand small VOLEs
- combine into big VOLE

random challenge

VOLE consistency proof

FAEST Rounds



Prover $\mathcal{P}$

Verifier $\mathcal{V}$

- vector-commit to random strings
- expand small VOLEs
- combine into big VOLE

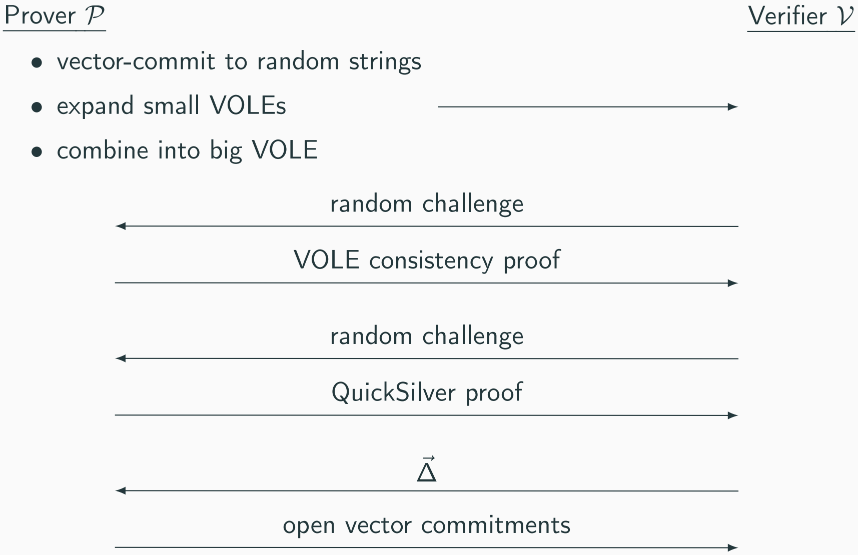
random challenge

VOLE consistency proof

random challenge

QuickSilver proof

FAEST Rounds



Prover $\mathcal{P}$

Verifier $\mathcal{V}$

- vector-commit to random strings
- expand small VOLEs
- combine into big VOLE

random challenge

VOLE consistency proof

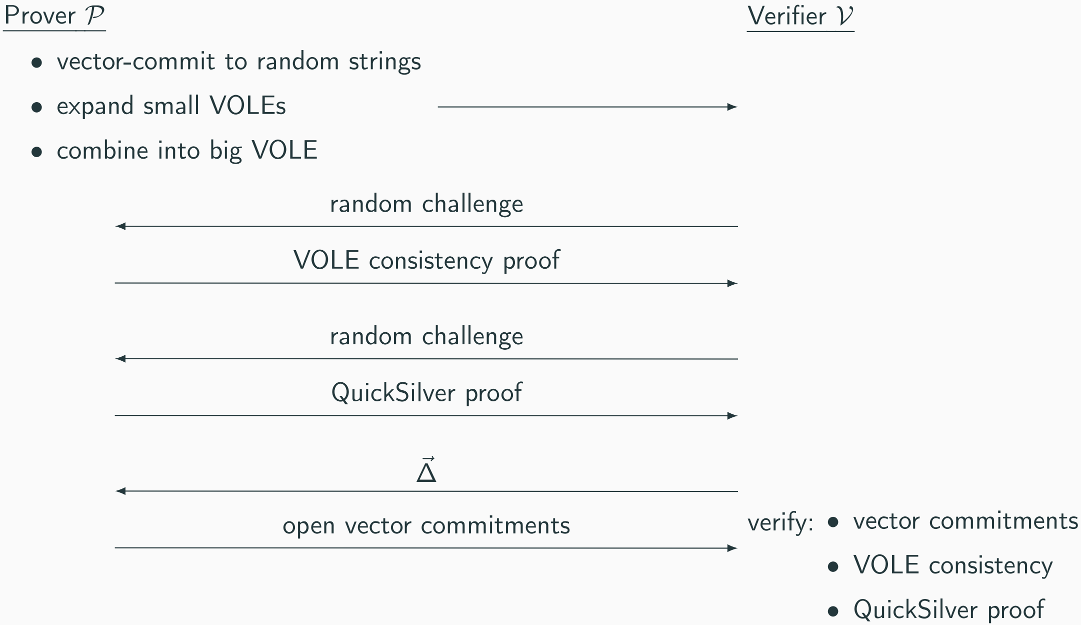
random challenge

QuickSilver proof

$\vec{\Delta}$

open vector commitments

FAEST Rounds



Prover $\mathcal{P}$

Verifier $\mathcal{V}$

- vector-commit to random strings
- expand small VOLEs
- combine into big VOLE

random challenge

VOLE consistency proof

random challenge

QuickSilver proof

$\vec{\Delta}$

open vector commitments

verify:
- vector commitments
- VOLE consistency
- QuickSilver proof

Grinding

Grinding: Overview

Mismatch: cost of generating a proof $\gg$ per-trial attack cost lowerbound in the security argument.

Grinding: Overview

Mismatch: cost of generating a proof $\gg$ per-trial attack cost lowerbound in the security argument.

- Prover must generate $\Theta(\tau 2^k \ell)$ PRG bits and run $\Theta(\tau 2^k)$ hashes.
- Lower bound is based on a single Fiat-Shamir hash evaluation.

Grinding: Overview

Mismatch: cost of generating a proof $\gg$ per-trial attack cost lowerbound in the security argument.

- Prover must generate $\Theta(\tau 2^k \ell)$ PRG bits and run $\Theta(\tau 2^k)$ hashes.
- Lower bound is based on a single Fiat-Shamir hash evaluation.

Fix: make the Fiat-Shamir hash 2^w times more expensive (Grinding). Only need to target $2^{\lambda - w}$ security level.

Grinding: Overview

Mismatch: cost of generating a proof $\gg$ per-trial attack cost lowerbound in the security argument.

- Prover must generate $\Theta(\tau 2^k \ell)$ PRG bits and run $\Theta(\tau 2^k)$ hashes.
- Lower bound is based on a single Fiat-Shamir hash evaluation.

Fix: make the Fiat-Shamir hash 2^w times more expensive (Grinding). Only need to target $2^{\lambda - w}$ security level.

- This allows for smaller signatures by reducing τ.

Grinding: Overview

Mismatch: cost of generating a proof $\gg$ per-trial attack cost lowerbound in the security argument.

- Prover must generate $\Theta(\tau 2^k \ell)$ PRG bits and run $\Theta(\tau 2^k)$ hashes.
- Lower bound is based on a single Fiat-Shamir hash evaluation.

Fix: make the Fiat-Shamir hash 2^w times more expensive (Grinding). Only need to target $2^{\lambda - w}$ security level.

- This allows for smaller signatures by reducing τ.
- Counter-intuitively, this can also make signing faster — k can be reduced while preserving security.

Grinding: Correlation

$$\vec{q}_0 = \vec{w} \cdot \Delta_0 + \vec{v}_0$$

$$\vdots$$

$$\vec{q}_{\tau-2} = \vec{w} \cdot \Delta_{\tau-2} + \vec{v}_{\tau-2}$$

$$\vec{q}_{\tau-1} = \vec{w} \cdot \Delta_{\tau-1} + \vec{v}_{\tau-1}$$

$$\Downarrow$$

$$\underbrace{\sum_{i\in[\tau]} \vec{q}_i \cdot X^i}_{\vec{q}\in\mathbb{F}^{\ell}_{q^\tau}} = \vec{w} \cdot \underbrace{\sum_{i\in[\tau]} \Delta_i \cdot X^i}_{\Delta\in\mathbb{F}_{q^\tau}} + \underbrace{\sum_{i\in[\tau]} \vec{v}_i \cdot X^i}_{\vec{v}\in\mathbb{F}^{\ell}_{q^\tau}}$$

Grinding: Correlation

What if $\Delta_{\tau-1} = 0$?

$$\begin{aligned}
\vec{q}_0 &= \vec{w} \cdot \Delta_0 + \vec{v}_0 \\
&\vdots \\
\vec{q}_{\tau-2} &= \vec{w} \cdot \Delta_{\tau-2} + \vec{v}_{\tau-2} \\
\vec{q}_{\tau-1} &= \vec{w} \cdot \Delta_{\tau-1} + \vec{v}_{\tau-1} \\
&\Downarrow \\
\underbrace{\sum_{i \in [\tau]} \vec{q}_i \cdot X^i}_{\vec{q} \in \mathbb{F}_{q^\tau}^{\ell}} &= \vec{w} \cdot \underbrace{\sum_{i \in [\tau]} \Delta_i \cdot X^i}_{\Delta \in \mathbb{F}_{q^\tau}} + \underbrace{\sum_{i \in [\tau]} \vec{v}_i \cdot X^i}_{\vec{v} \in \mathbb{F}_{q^\tau}^{\ell}}
\end{aligned}$$

Grinding: Correlation

What if $\Delta_{\tau-1} = 0$?

$$\begin{aligned}
\vec{q}_0 &= \vec{w} \cdot \Delta_0 + \vec{v}_0 \\
&\vdots \\
\vec{q}_{\tau-2} &= \vec{w} \cdot \Delta_{\tau-2} + \vec{v}_{\tau-2} \\
\vec{q}_{\tau-1} &= \vec{w} \cdot 0 + \vec{v}_{\tau-1} \\
&\Downarrow \\
\underbrace{\sum_{i\in[\tau]} \vec{q}_i \cdot X^i}_{\vec{q}\in\mathbb{F}^{\ell}_{q^\tau}} &= \vec{w} \cdot \underbrace{\sum_{i\in[\tau]} \Delta_i \cdot X^i}_{\Delta\in\mathbb{F}_{q^\tau}} + \underbrace{\sum_{i\in[\tau]} \vec{v}_i \cdot X^i}_{\vec{v}\in\mathbb{F}^{\ell}_{q^\tau}}
\end{aligned}$$

Grinding: Correlation

What if $\Delta_{\tau-1} = 0$?

The last small vole correlation is now trivial,

$$\begin{aligned} \vec{q}_0 &= \vec{w} \cdot \Delta_0 + \vec{v}_0 \\ &\vdots \\ \vec{q}_{\tau-2} &= \vec{w} \cdot \Delta_{\tau-2} + \vec{v}_{\tau-2} \\ 0 &= \vec{w} \cdot 0 + 0 \\ &\Downarrow \end{aligned}$$

$$\underbrace{\sum_{i\in[\tau]} \vec{q}_i \cdot X^i}_{\vec{q}\in\mathbb{F}_{q^\tau}^{\ell}} = \vec{w} \cdot \underbrace{\sum_{i\in[\tau]} \Delta_i \cdot X^i}_{\Delta\in\mathbb{F}_{q^\tau}} + \underbrace{\sum_{i\in[\tau]} \vec{v}_i \cdot X^i}_{\vec{v}\in\mathbb{F}_{q^\tau}^{\ell}}$$

Grinding: Correlation

What if $\Delta_{\tau-1} = 0$?
The last small vole correlation is now trivial, and can be removed to save communication.

$$\vec{q}_0 = \vec{w} \cdot \Delta_0 + \vec{v}_0$$

$$\vdots$$

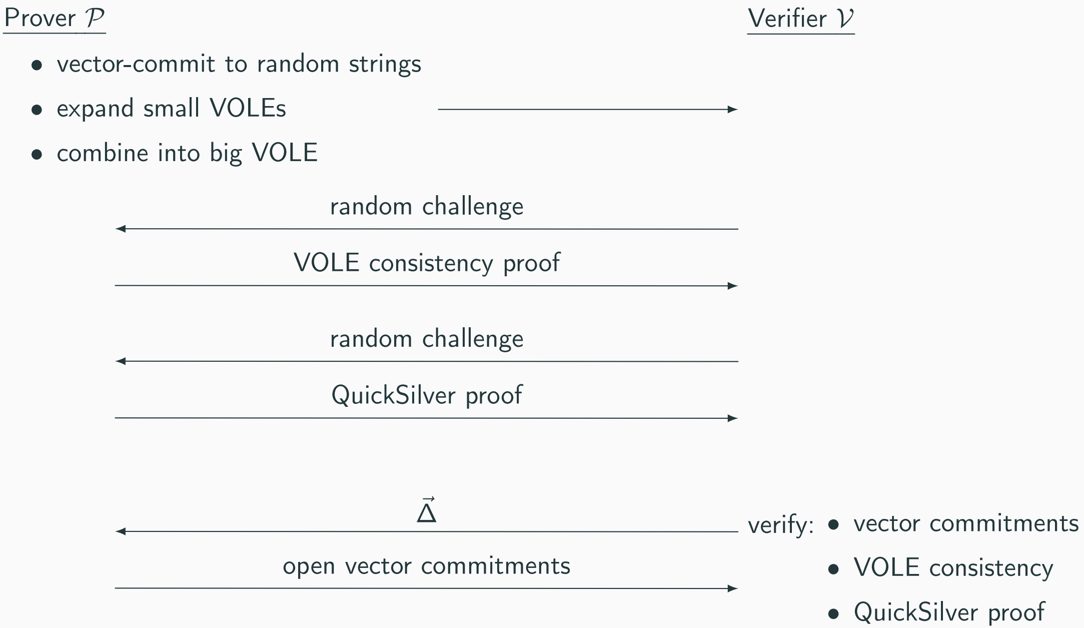
$$\vec{q}_{\tau-2} = \vec{w} \cdot \Delta_{\tau-2} + \vec{v}_{\tau-2}$$

$$0 = \vec{w} \cdot 0 + 0$$

$$\Downarrow$$

$$\underbrace{\sum_{i \in [\tau-1]} \vec{q}_i \cdot X^i}_{\vec{q} \in \mathbb{F}^{\ell}_{q^\tau}} = \vec{w} \cdot \underbrace{\sum_{i \in [\tau-1]} \Delta_i \cdot X^i}_{\Delta \in \mathbb{F}_{q^\tau}} + \underbrace{\sum_{i \in [\tau-1]} \vec{v}_i \cdot X^i}_{\vec{v} \in \mathbb{F}^{\ell}_{q^\tau}}$$

Grinding: Rounds



Prover $\mathcal{P}$

Verifier $\mathcal{V}$

- vector-commit to random strings
- expand small VOLEs →
- combine into big VOLE

← random challenge

VOLE consistency proof →

← random challenge

QuickSilver proof →

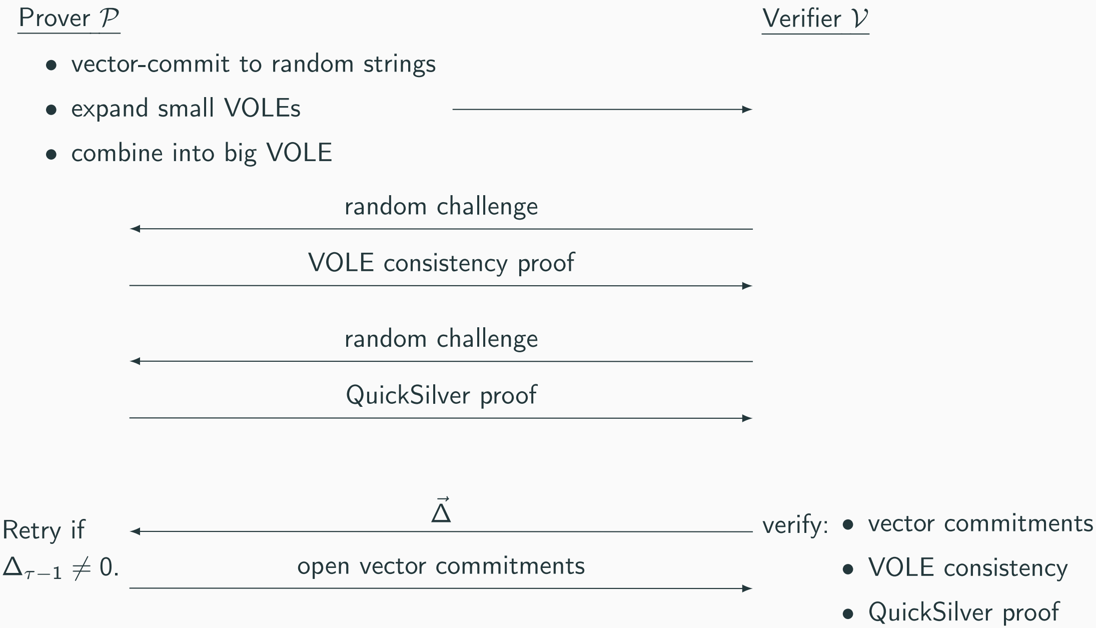
← $\vec{\Delta}$

open vector commitments →

verify:

- vector commitments
- VOLE consistency
- QuickSilver proof

Grinding: Rounds



Prover $\mathcal{P}$

Verifier $\mathcal{V}$

- vector-commit to random strings
- expand small VOLEs
- combine into big VOLE

random challenge

VOLE consistency proof

random challenge

QuickSilver proof

$\vec{\Delta}$

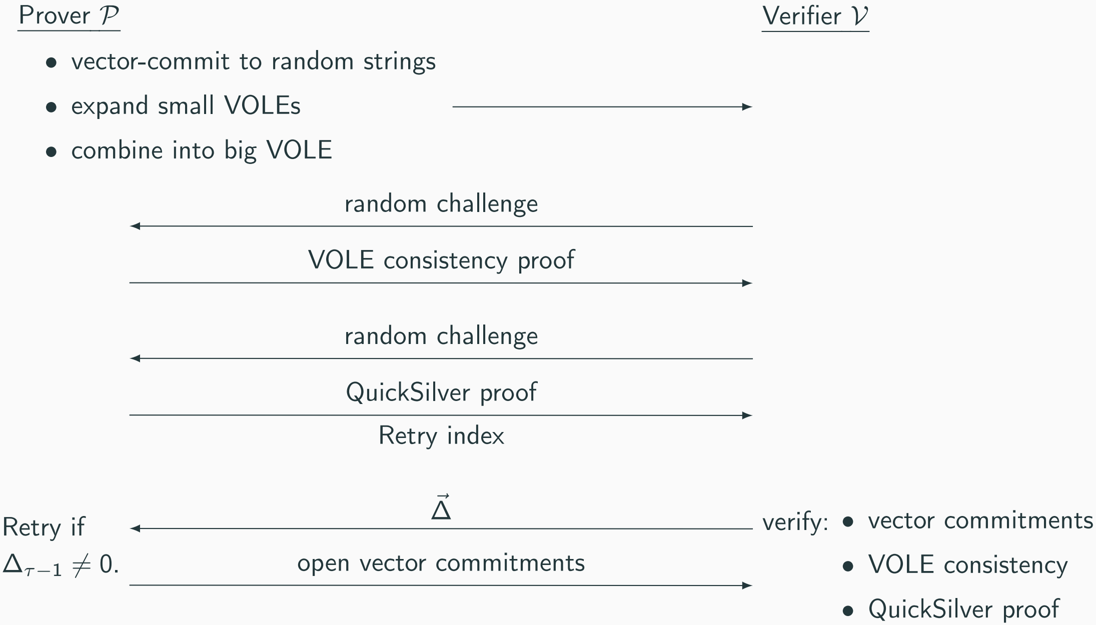
Retry if $\Delta_{\tau-1} \neq 0$.

open vector commitments

verify:

- vector commitments
- VOLE consistency
- QuickSilver proof

Grinding: Rounds



Prover $\mathcal{P}$

- vector-commit to random strings
- expand small VOLEs
- combine into big VOLE

Verifier $\mathcal{V}$

random challenge

VOLE consistency proof

random challenge

QuickSilver proof

Retry index

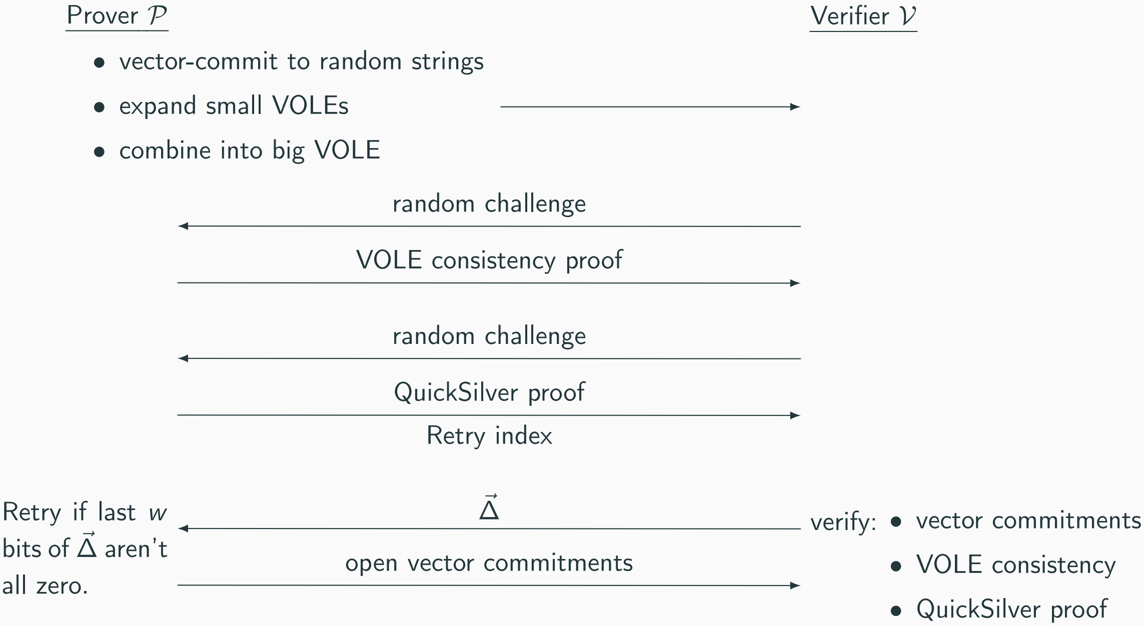
$\vec{\Delta}$

Retry if $\Delta_{\tau-1} \neq 0$.

open vector commitments

verify:
- vector commitments
- VOLE consistency
- QuickSilver proof

Grinding: Rounds



Prover $\mathcal{P}$

Verifier $\mathcal{V}$

- vector-commit to random strings
- expand small VOLEs
- combine into big VOLE

random challenge

VOLE consistency proof

random challenge

QuickSilver proof

Retry index

Retry if last w bits of $\vec{\Delta}$ aren't all zero.

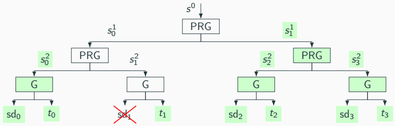
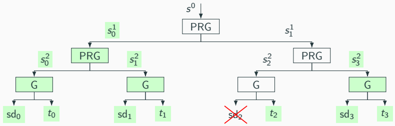
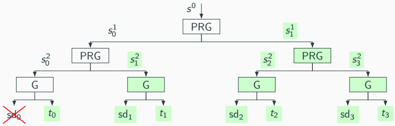
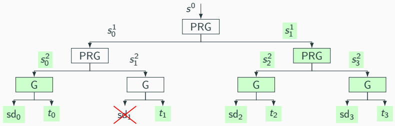
$\vec{\Delta}$

open vector commitments

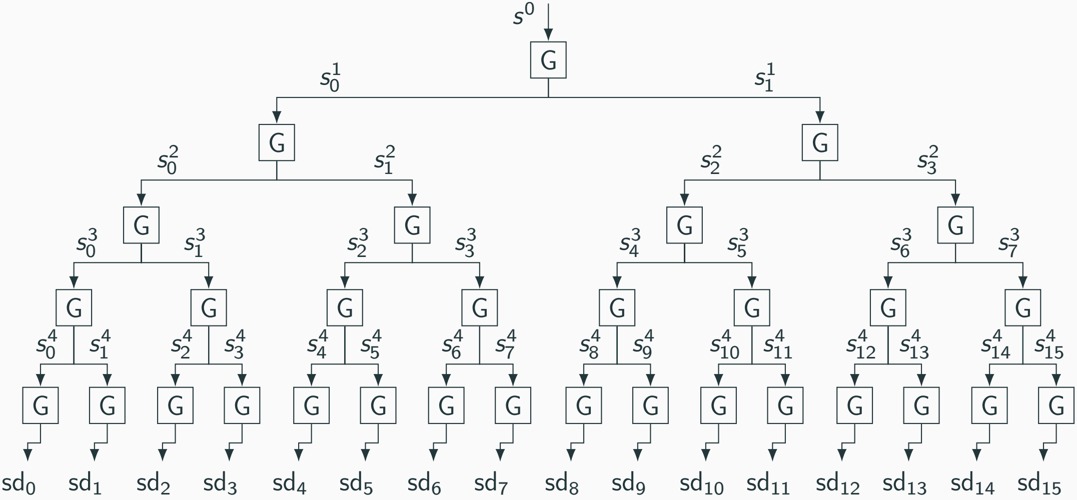
verify:
- vector commitments
- VOLE consistency
- QuickSilver proof

One Tree to Rule Them All

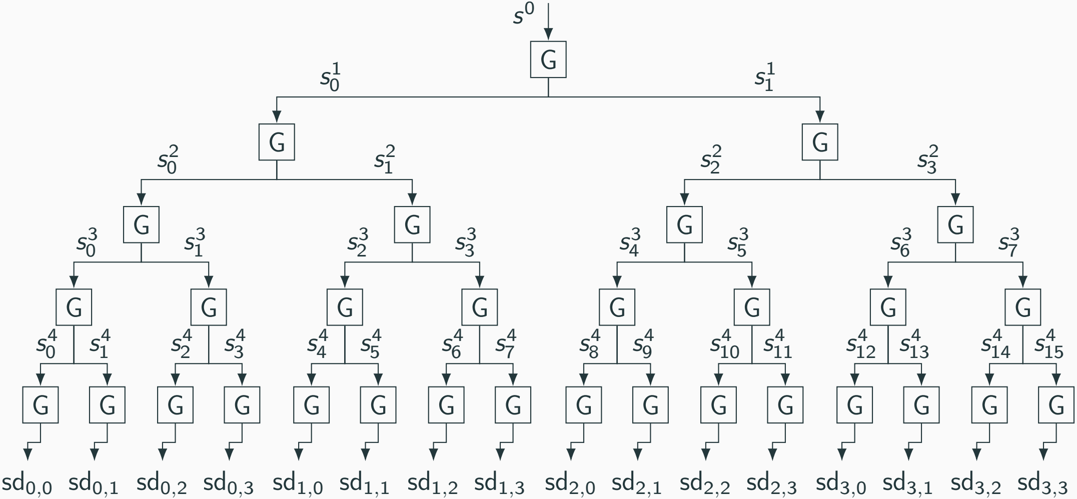
All-but-some Random Vector Commitments

		Field element $x \in [0, 2^k)$				Commitment
		0	1	2	3	
Repetition $i \in [0, \tau)$	0	$\mathsf{sd}_{0,0}$	~~$\mathsf{sd}_{0,1}$~~	$\mathsf{sd}_{0,2}$	$\mathsf{sd}_{0,3}$	
	1	$\mathsf{sd}_{1,0}$	$\mathsf{sd}_{1,1}$	~~$\mathsf{sd}_{1,2}$~~	$\mathsf{sd}_{1,3}$	
	2	~~$\mathsf{sd}_{2,0}$~~	$\mathsf{sd}_{2,1}$	$\mathsf{sd}_{2,2}$	$\mathsf{sd}_{2,3}$	
	3	$\mathsf{sd}_{3,0}$	~~$\mathsf{sd}_{3,1}$~~	$\mathsf{sd}_{3,2}$	$\mathsf{sd}_{3,3}$	

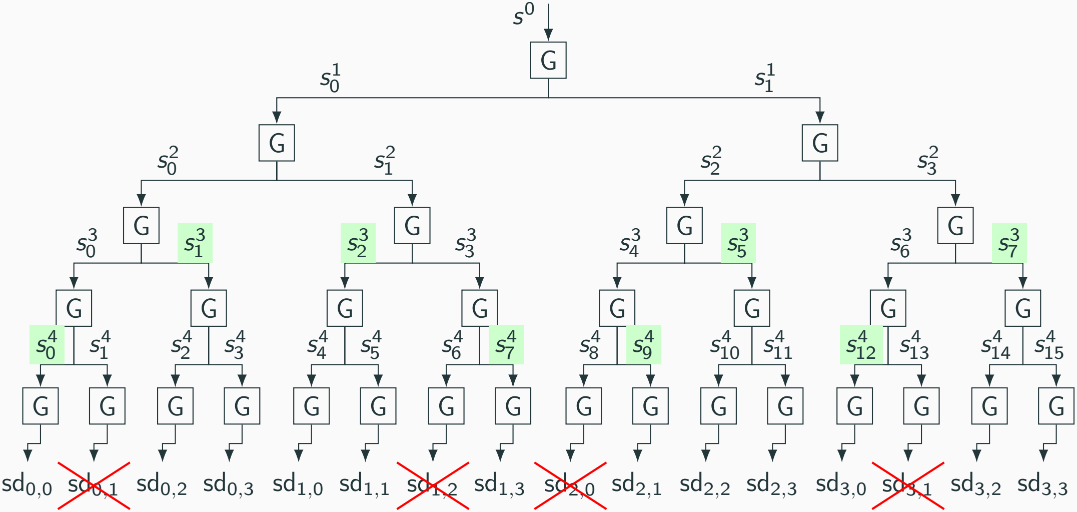
One Tree to Bind Them



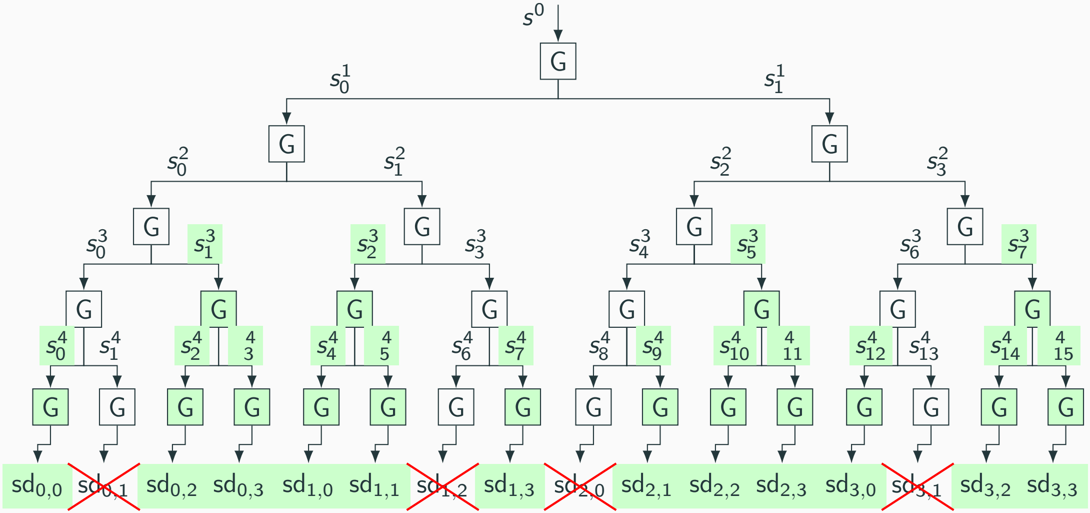
One Tree to Bind Them



One Tree to Bind Them



One Tree to Bind Them



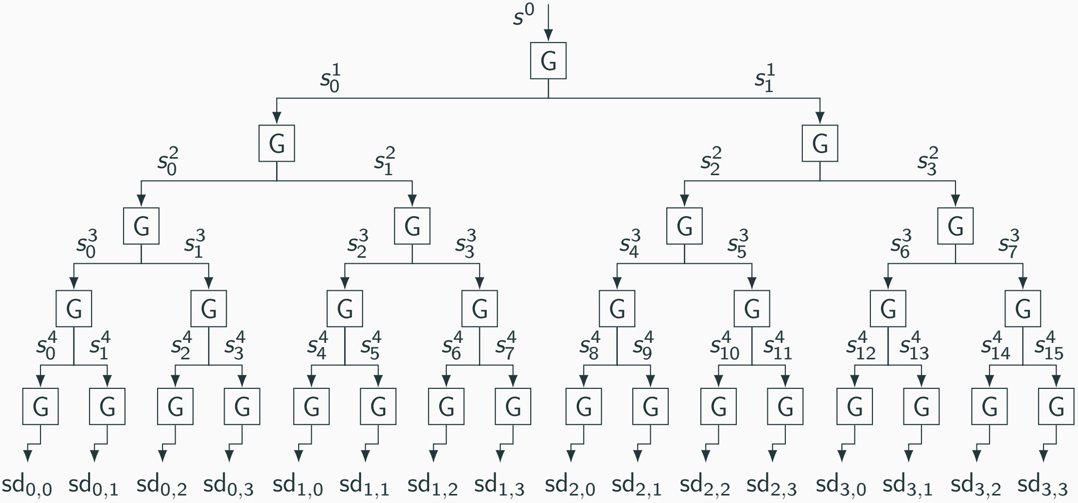
All-but-some Random Vector Commitments

		Field element $x \in [0, 2^k)$			
		0	1	2	3
Repetition $i \in [0, \tau)$	0	$\mathrm{sd}_{0,0}$	$\xcancel{\mathrm{sd}_{0,1}}$	$\mathrm{sd}_{0,2}$	$\mathrm{sd}_{0,3}$
	1	$\mathrm{sd}_{1,0}$	$\mathrm{sd}_{1,1}$	$\xcancel{\mathrm{sd}_{1,2}}$	$\mathrm{sd}_{1,3}$
	2	$\xcancel{\mathrm{sd}_{2,0}}$	$\mathrm{sd}_{2,1}$	$\mathrm{sd}_{2,2}$	$\mathrm{sd}_{2,3}$
	3	$\mathrm{sd}_{3,0}$	$\xcancel{\mathrm{sd}_{3,1}}$	$\mathrm{sd}_{3,2}$	$\mathrm{sd}_{3,3}$

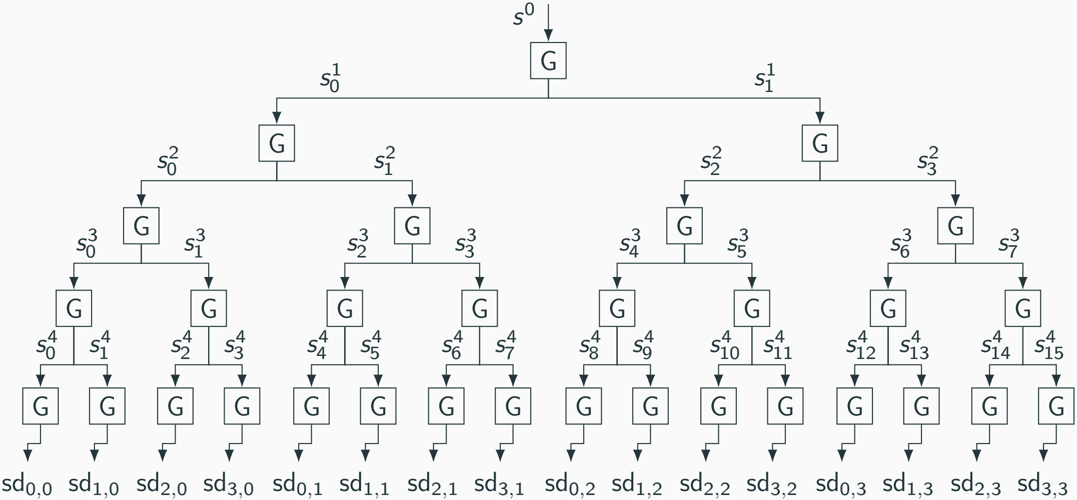
All-but-some Random Vector Commitments

		Repetition $i \in [0, \tau)$			
		0	1	2	3
Field element $x \in [0, 2^k)$	0	$\mathsf{sd}_{0,0}$	$\mathsf{sd}_{1,0}$	$\cancel{\mathsf{sd}_{2,0}}$	$\mathsf{sd}_{3,0}$
	1	$\cancel{\mathsf{sd}_{0,1}}$	$\mathsf{sd}_{1,1}$	$\mathsf{sd}_{2,1}$	$\cancel{\mathsf{sd}_{3,1}}$
	2	$\mathsf{sd}_{0,2}$	$\cancel{\mathsf{sd}_{1,2}}$	$\mathsf{sd}_{2,2}$	$\mathsf{sd}_{3,2}$
	3	$\mathsf{sd}_{0,3}$	$\mathsf{sd}_{1,3}$	$\mathsf{sd}_{2,3}$	$\mathsf{sd}_{3,3}$

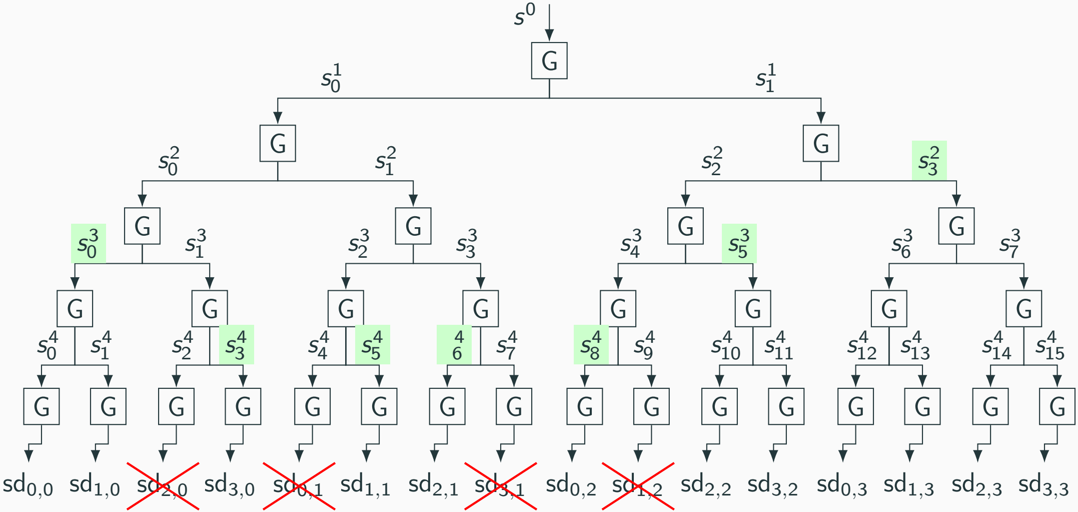
One Tree to Bind Them



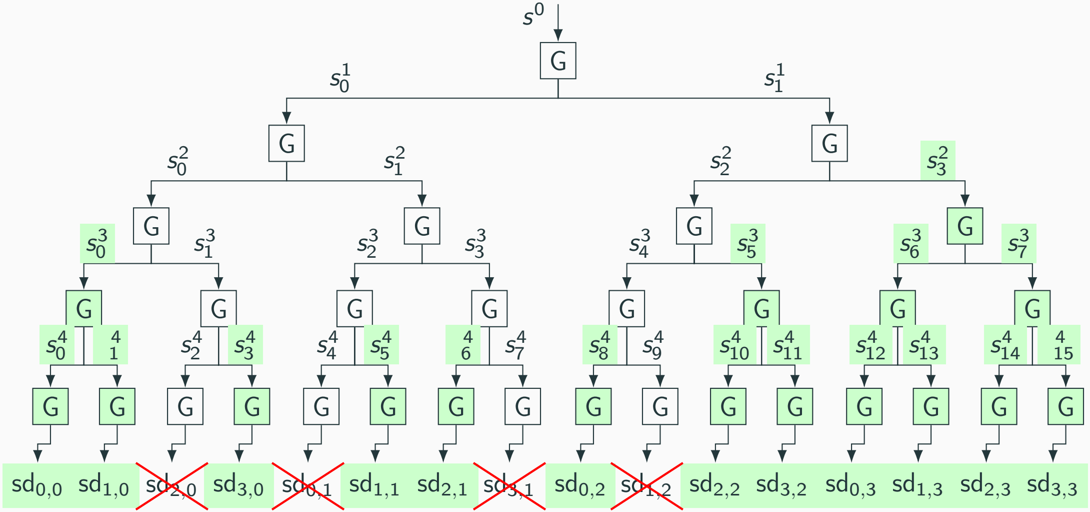
One Tree to Bind Them



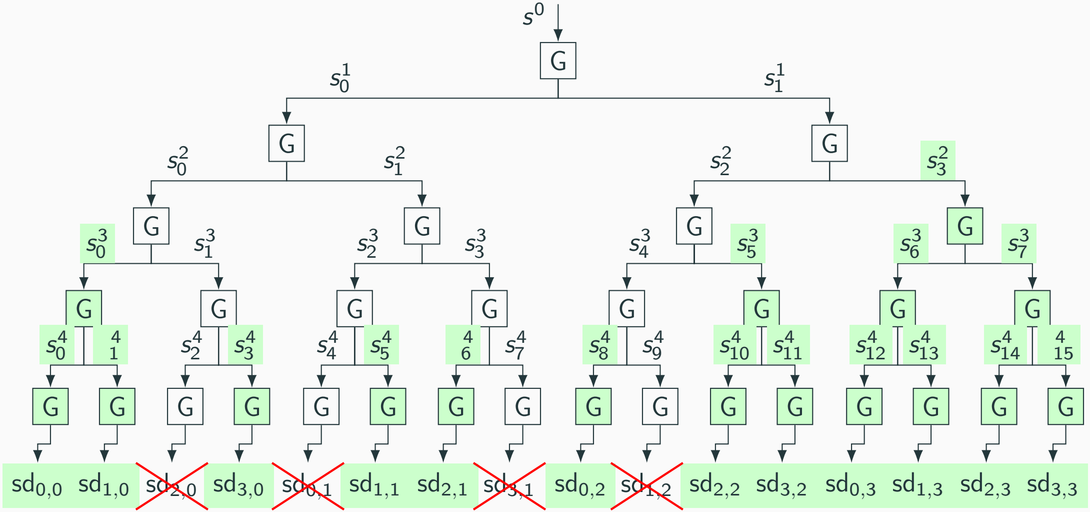
One Tree to Bind Them



One Tree to Bind Them

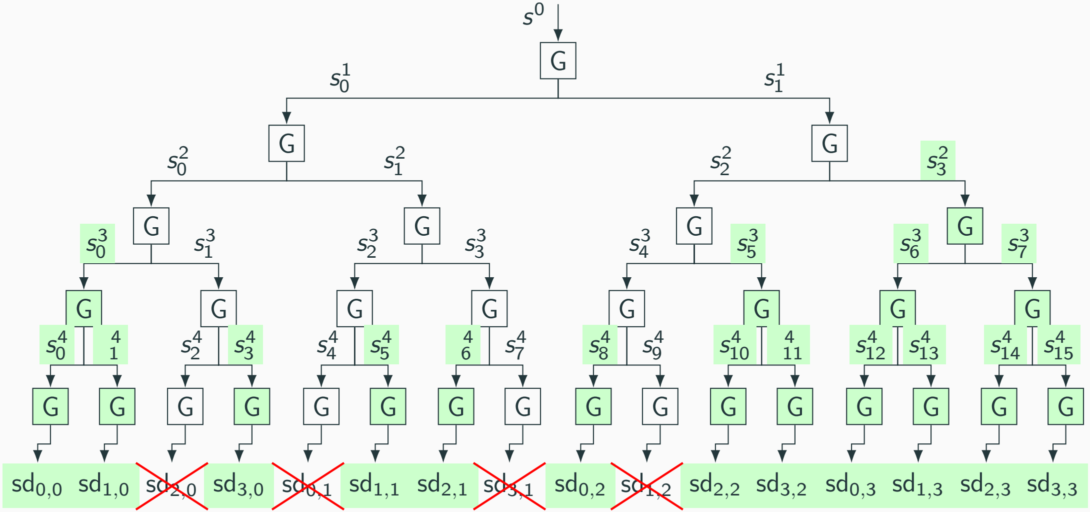


One Tree to Bind Them



Note: only 7 seeds to open, not 8.

One Tree to Bind Them



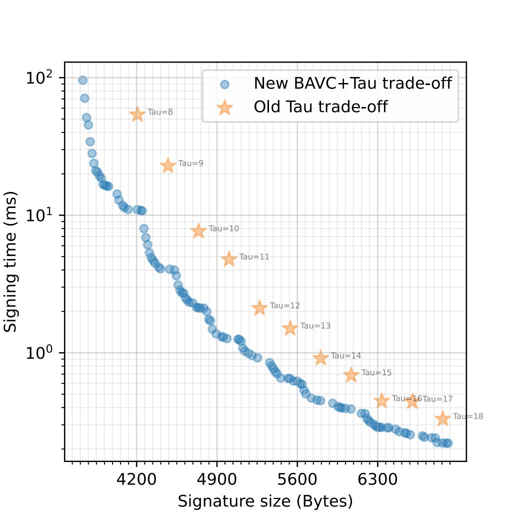
Note: only 7 seeds to open, not 8.

In general, the opening size depends on Δ.

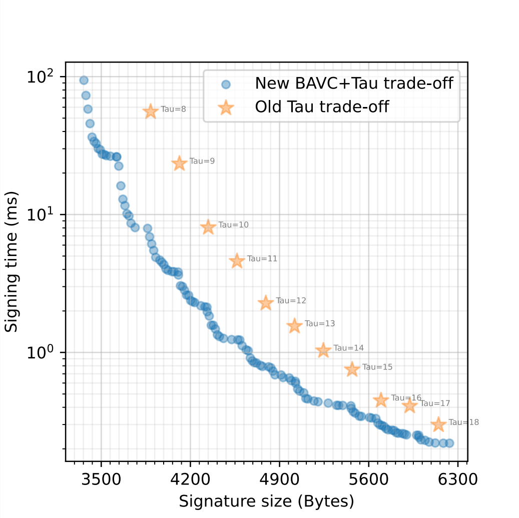
$\rightsquigarrow$ Set a limit T_{open} on seeds in the opening, and retry if it's exceeded.

FAESTER

Size-time Tradeoff



(a) FAESTER-128.



(b) FAESTER-EM-128.

Parameter Choices

Signature Scheme	OWF $E_{sk}(x)$	l	w	$\mathrm{T}_{\mathrm{open}}$	τ	τ_0	τ_1	k_0	k_1	sk size	pk size	sig. size
FAEST-128$_\mathsf{s}$	AES128$_{sk}(x)$	1600	–	–	11	7	4	12	11	16	32	5006
FAEST-128$_\mathsf{f}$	AES128$_{sk}(x)$	1600	–	–	16	0	16	8	8	16	32	6336
FAEST-EM-128$_\mathsf{s}$	AES128$_x(sk) \oplus sk$	1280	–	–	11	7	4	12	11	16	32	4566
FAEST-EM-128$_\mathsf{f}$	AES128$_x(sk) \oplus sk$	1280	–	–	16	0	16	8	8	16	32	5696
FAESTER-128$_\mathsf{s}$	AES128$_{sk}(x)$	1600	7	102	11	0	11	11	11	16	32	4594
FAESTER-128$_\mathsf{f}$	AES128$_{sk}(x)$	1600	8	110	16	8	8	8	7	16	32	6052
FAESTER-EM-128$_\mathsf{s}$	AES128$_x(sk) \oplus sk$	1280	7	103	11	0	11	11	11	16	32	4170
FAESTER-EM-128$_\mathsf{f}$	AES128$_x(sk) \oplus sk$	1280	8	112	16	8	8	8	7	16	32	5444

Performance Comparison

Scheme	Runtime in ms			Size in bytes		
	Keygen	Sign	Verify	sk	pk	Signature
FAEST-128$_s$	0.0006	4.381	4.102	16	32	5006
FAEST-128$_f$	0.0005	0.404	0.395	16	32	6336
FAEST-EM-128$_s$	0.0005	4.151	4.415	16	32	4566
FAEST-EM-128$_f$	0.0005	0.446	0.474	16	32	5696
FAESTER-128$_s$	0.0006	3.282	4.467	16	32	4594
FAESTER-128$_f$	0.0005	0.433	0.610	16	32	6052
FAESTER-EM-128$_s$	0.0005	3.005	4.386	16	32	4170
FAESTER-EM-128$_f$	0.0005	0.422	0.609	16	32	5444

Signing time (ms), verification time (ms), and signature size (bytes).

Performance Comparison

Scheme	Runtime in ms			Size in bytes		
	Keygen	Sign	Verify	sk	pk	Signature
FAEST-128_s	0.0006	4.381	4.102	16	32	5006
FAEST-128_f	0.0005	0.404	0.395	16	32	6336
FAEST-EM-128_s	0.0005	4.151	4.415	16	32	4566
FAEST-EM-128_f	0.0005	0.446	0.474	16	32	5696
FAESTER-128_s	0.0006	3.282	4.467	16	32	4594
FAESTER-128_f	0.0005	0.433	0.610	16	32	6052
FAESTER-EM-128_s	0.0005	3.005	4.386	16	32	4170
FAESTER-EM-128_f	0.0005	0.422	0.609	16	32	5444

Signing time (ms), verification time (ms), and signature size (bytes).

Benchmarking system: AMD Ryzen 9 7900X 12-Core CPU running Ubuntu 22.04.

Zeroes in S-boxes

- AES S-boxes:

$$x \mapsto y = \begin{cases} 0 & \text{if } x = 0 \\ x^{-1} \in \mathbb{F}_{2^8} & \text{otherwise} \end{cases} \tag{$\star$}$$

- Constraint: $x \cdot y = 1$. This requires $x \neq 0$.

Zeroes in S-boxes

- AES S-boxes:

$$x \mapsto y = \begin{cases} 0 & \text{if } x = 0 \\ x^{-1} \in \mathbb{F}_{2^8} & \text{otherwise} \end{cases} \tag{$\star$}$$

- Constraint: $x \cdot y = 1$. This requires $x \neq 0$.

- $(\star) \iff x^2 \cdot y = x \ \wedge \ x \cdot y^2 = y$
  - observe that $x \mapsto x^2$ is $\mathbb{F}_2$-linear.
  - $\rightsquigarrow$ 2 quadratic constraints per S-box.

Zeroes in S-boxes

- AES S-boxes:

$$x \mapsto y = \begin{cases} 0 & \text{if } x = 0 \\ x^{-1} \in \mathbb{F}_{2^8} & \text{otherwise} \end{cases} \qquad (\star)$$

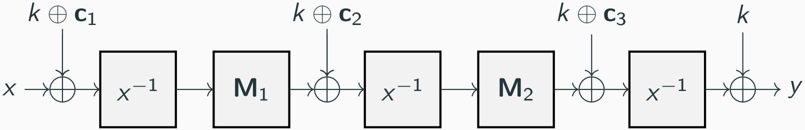
- Constraint: $x \cdot y = 1$. This requires $x \neq 0$.

- $(\star) \iff x^2 \cdot y = x \;\wedge\; x \cdot y^2 = y$
  - observe that $x \mapsto x^2$ is $\mathbb{F}_2$-linear.
  - $\rightsquigarrow$ 2 quadratic constraints per S-box.

- Can use any AES key! No rejection sampling.

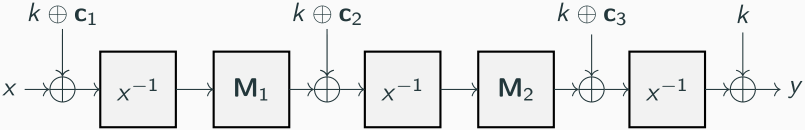
MandaRain

Rain Cipher



3-round Rain.

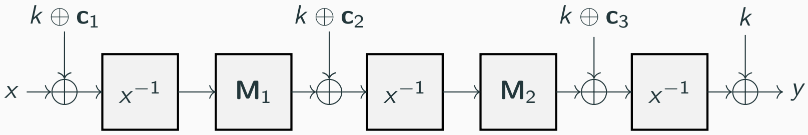
Rain Cipher



3-round Rain.

- $x, k, y \in \mathbb{F}_{2^\lambda}$.
- M_i is a $\mathbb{F}_2$-linear transformations.

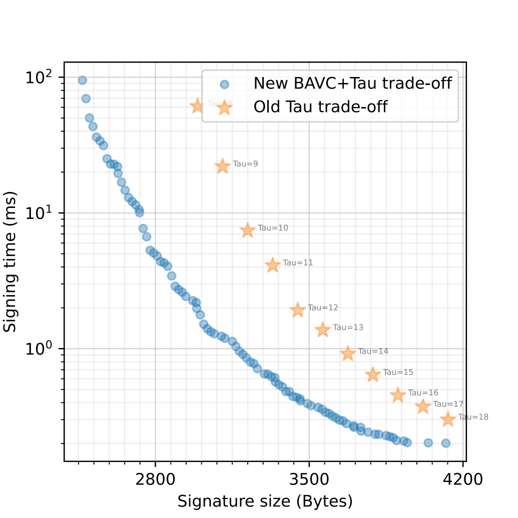
Rain Cipher



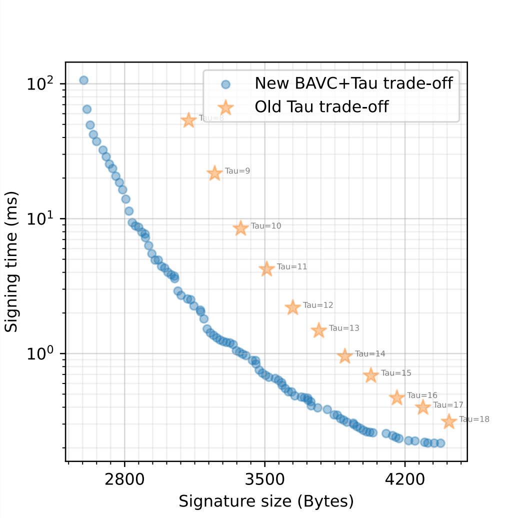
3-round Rain.

- $x, k, y \in \mathbb{F}_{2^\lambda}$.
- M_i is a $\mathbb{F}_2$-linear transformations.
- Fewer rounds $\implies$ smaller witness.

Size-time Tradeoff



(a) MandaRain-3-128.



(b) MandaRain-4-128.

Performance Comparison

Scheme	Runtime in ms			Size in bytes		
	Keygen	Sign	Verify	sk	pk	Signature
FAEST-128$_s$	0.0006	4.381	4.102	16	32	5006
FAEST-128$_f$	0.0005	0.404	0.395	16	32	6336
FAEST-EM-128$_s$	0.0005	4.151	4.415	16	32	4566
FAEST-EM-128$_f$	0.0005	0.446	0.474	16	32	5696
FAESTER-128$_s$	0.0006	3.282	4.467	16	32	4594
FAESTER-128$_f$	0.0005	0.433	0.610	16	32	6052
FAESTER-EM-128$_s$	0.0005	3.005	4.386	16	32	4170
FAESTER-EM-128$_f$	0.0005	0.422	0.609	16	32	5444
MandaRain-3-128$_s$	0.0018	2.800	5.895	16	32	2890
MandaRain-3-128$_f$	0.0018	0.346	0.807	16	32	3588
MandaRain-4-128$_s$	0.0026	2.876	6.298	16	32	3052
MandaRain-4-128$_f$	0.0026	0.371	0.817	16	32	3876

Signing time (ms), verification time (ms), and signature size (bytes).

KuMQuat

Unstructured Multivariate-Quadratic

Sample $\mathrm{A}_i \in \mathbb{F}_q^{n \times n}$, $b_i \in \mathbb{F}_q^n$, and $x \in \mathbb{F}_q^n$.
Public key: seeds for A and b, and $y \in \mathbb{F}_q^n$ where

$$y_i = x^{\mathsf{T}} \mathrm{A}_i \, x + b_i^{\mathsf{T}} x_j$$

Unstructured Multivariate-Quadratic

Sample $\mathrm{A}_i \in \mathbb{F}_q^{n \times n}$, $b_i \in \mathbb{F}_q^n$, and $x \in \mathbb{F}_q^n$.
Public key: seeds for A and b, and $y \in \mathbb{F}_q^n$ where

$$y_i = x^{\mathsf{T}} \mathrm{A}_i \, x + b_i^{\mathsf{T}} x_j$$

Witness: $x \in \mathbb{F}_q^n$
Constraints:

$$y_i = \sum_{jk} \mathrm{A}_{ijk} \, x_j x_k + \sum_{j} b_{ij} \, x_j - y_i \quad \forall i \in [n]$$

Unstructured Multivariate-Quadratic

Sample $\mathrm{A}_i \in \mathbb{F}_q^{n \times n}$, $b_i \in \mathbb{F}_q^n$, and $x \in \mathbb{F}_q^n$.
Public key: seeds for A and b, and $y \in \mathbb{F}_q^n$ where

$$y_i = x^{\mathsf{T}} \mathrm{A}_i \, x + b_i^{\mathsf{T}} x_j$$

Witness: $x \in \mathbb{F}_q^n$
Constraints:

$$y_i = \sum_{jk} \mathrm{A}_{ijk} \, x_j x_k + \sum_j b_{ij} \, x_j - y_i \quad \forall i \in [n]$$

- Witness size is minimal (assuming only quadratic constraints).

Unstructured Multivariate-Quadratic

Sample $\mathsf{A}_i \in \mathbb{F}_q^{n \times n}$, $b_i \in \mathbb{F}_q^n$, and $x \in \mathbb{F}_q^n$.
Public key: seeds for A and b, and $y \in \mathbb{F}_q^n$ where

$$y_i = x^{\mathsf{T}} \mathsf{A}_i \, x + b_i^{\mathsf{T}} x_j$$

Witness: $x \in \mathbb{F}_q^n$
Constraints:

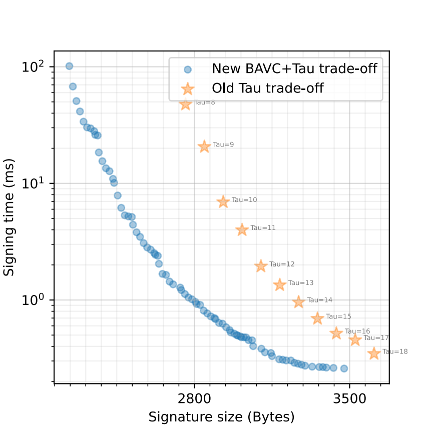
$$y_i = \sum_{jk} \mathsf{A}_{ijk} \, x_j x_k + \sum_j b_{ij} \, x_j - y_i \quad \forall i \in [n]$$

- Witness size is minimal (assuming only quadratic constraints).
- Optimization: pack multiple $\mathbb{F}_q$ constraints together into a $\mathbb{F}_{2^\lambda}$ constraint.

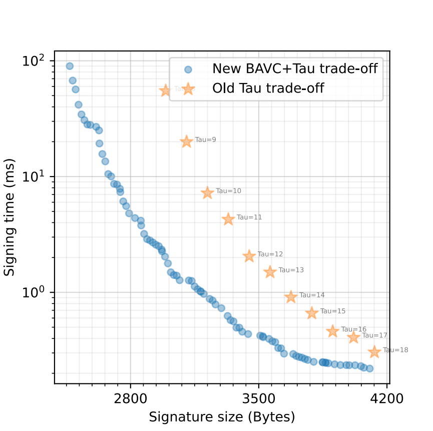
MQ Parameters

Instance	Security Level	$\mathbb{F}_q$	n
$\texttt{MQ-2}^1\texttt{-L1}$	L1	$\mathbb{F}_{2^1}$	152
$\texttt{MQ-2}^8\texttt{-L1}$	L1	$\mathbb{F}_{2^8}$	48
$\texttt{MQ-2}^1\texttt{-L3}$	L3	$\mathbb{F}_{2^1}$	224
$\texttt{MQ-2}^8\texttt{-L3}$	L3	$\mathbb{F}_{2^8}$	72
$\texttt{MQ-2}^1\texttt{-L5}$	L5	$\mathbb{F}_{2^1}$	320
$\texttt{MQ-2}^8\texttt{-L5}$	L5	$\mathbb{F}_{2^8}$	96

Size-time Tradeoff



(a) KuMQuat-2^1-L1.



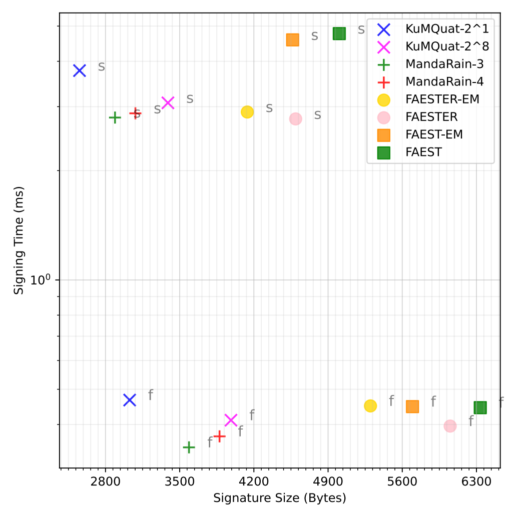
(b) KuMQuat-2^8-L1.

Performance Comparison

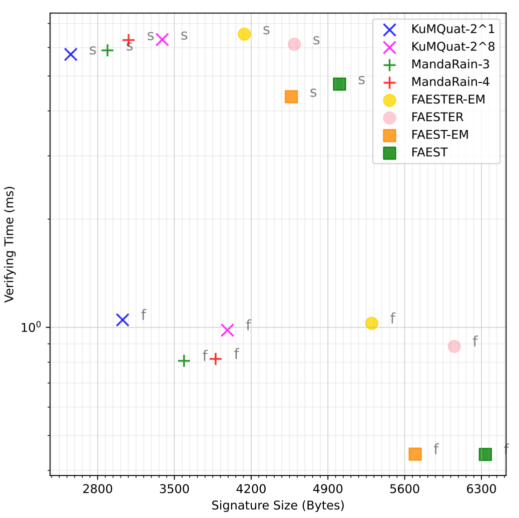
Scheme	Runtime in ms			Size in bytes		
	Keygen	Sign	Verify	sk	pk	Signature
FAEST-128$_s$	0.0006	4.381	4.102	16	32	5006
FAEST-128$_f$	0.0005	0.404	0.395	16	32	6336
FAEST-EM-128$_s$	0.0005	4.151	4.415	16	32	4566
FAEST-EM-128$_f$	0.0005	0.446	0.474	16	32	5696
FAESTER-128$_s$	0.0006	3.282	4.467	16	32	4594
FAESTER-128$_f$	0.0005	0.433	0.610	16	32	6052
FAESTER-EM-128$_s$	0.0005	3.005	4.386	16	32	4170
FAESTER-EM-128$_f$	0.0005	0.422	0.609	16	32	5444
MandaRain-3-128$_s$	0.0018	2.800	5.895	16	32	2890
MandaRain-3-128$_f$	0.0018	0.346	0.807	16	32	3588
MandaRain-4-128$_s$	0.0026	2.876	6.298	16	32	3052
MandaRain-4-128$_f$	0.0026	0.371	0.817	16	32	3876
KuMQuat-2^1-L1$_s$	0.173	4.305	4.107	19	35	2555
KuMQuat-2^1-L1$_f$	0.172	0.539	0.736	19	35	3028
KuMQuat-2^8-L1$_s$	0.174	3.599	4.053	48	64	2890
KuMQuat-2^8-L1$_f$	0.172	0.400	0.623	48	64	3588

Signing time (ms), verification time (ms), and signature size (bytes).

Performance Graph

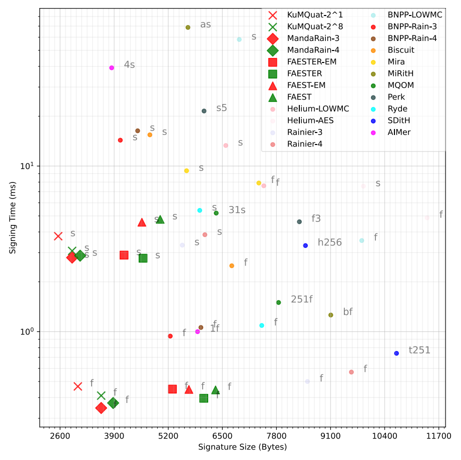


(a) Signing time - signature size trade-off.

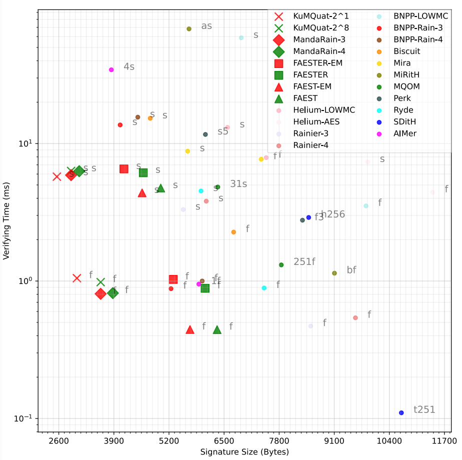


(b) Verification time - signature size trade-off.

Additional Graphs



(a) L1 Signing.



(b) L1 Verify.