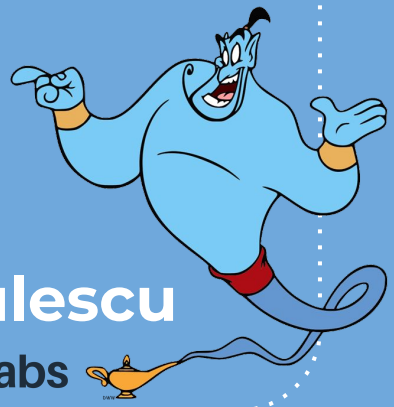
Linear-Map Vector Commitments

and applications

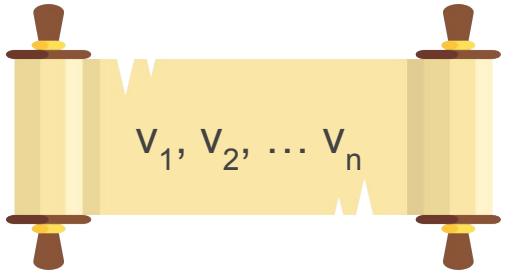


with Matteo Campanelli, Carla Ràfols, Alexandros Zacharakis, Arantxa Zapico

Anca Nitulescu

Protocol Labs

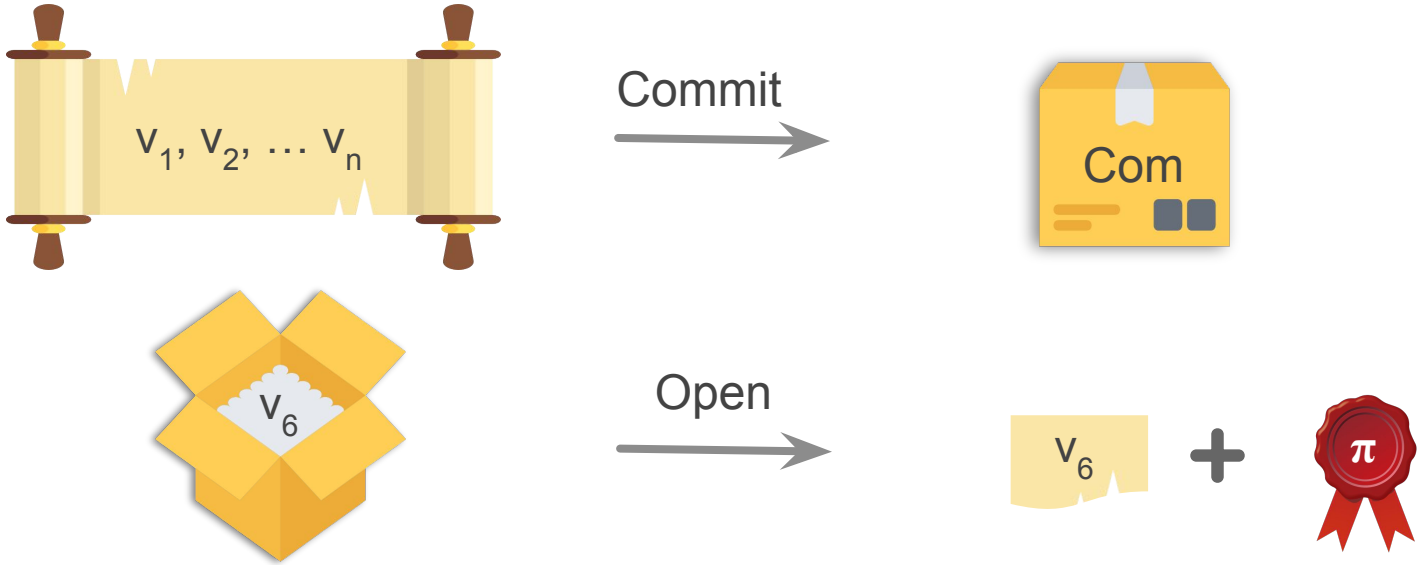
Vector Commitments



Commit

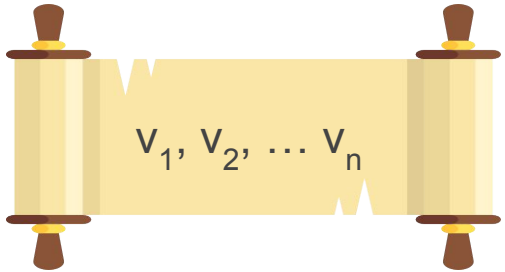
Vector Commitments and their Applications, by Dario Catalano and Dario Fiore

Vector Commitments

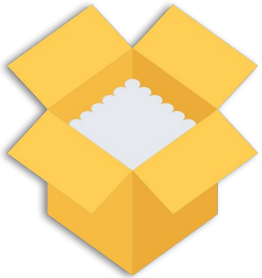


Vector Commitments and their Applications, by Dario Catalano and Dario Fiore

SubVector Commitment

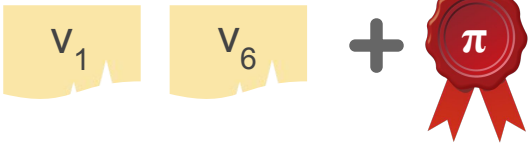


Commit

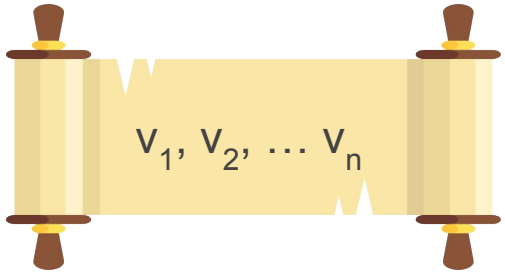


Open

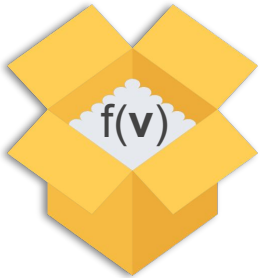
subvector



Functional VC

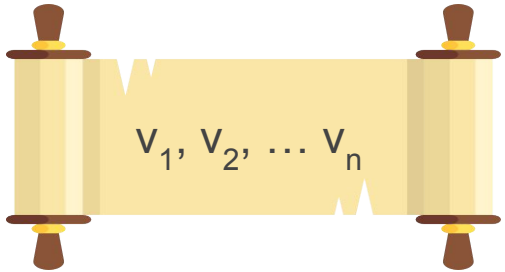


Commit

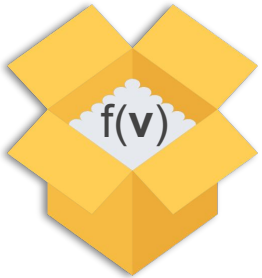


Open

Linear-Map VC



Commit



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

Open

$f : \mathbb{F}^n \to \mathbb{F}^\ell$

Inner Product VC

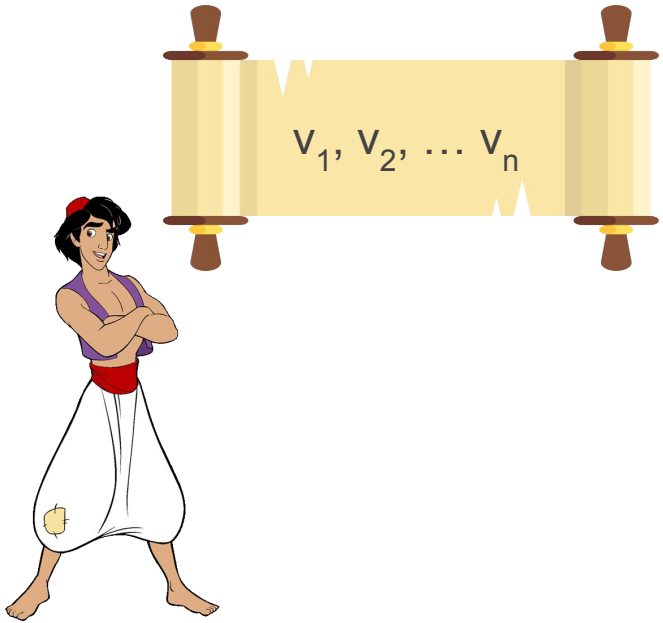
Inner Product of a committed vector with a public vector

$f_b(a) = \langle a, b \rangle$

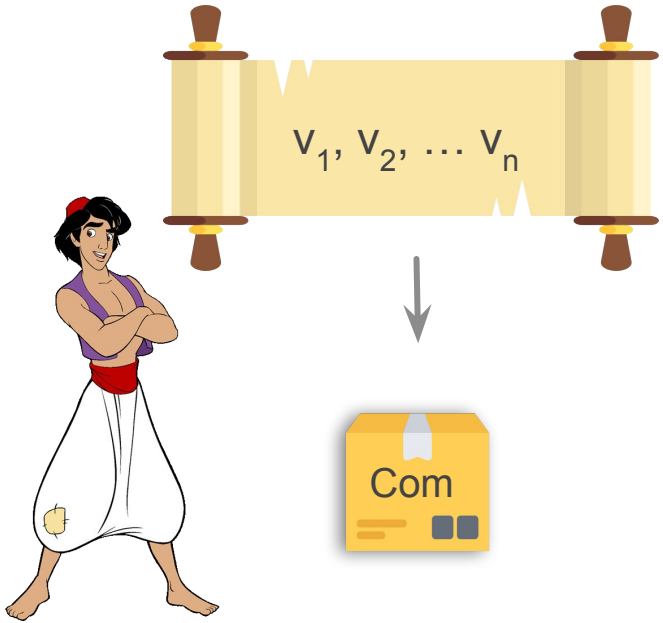
$a, b \in \mathbb{F}^n$

Motivation

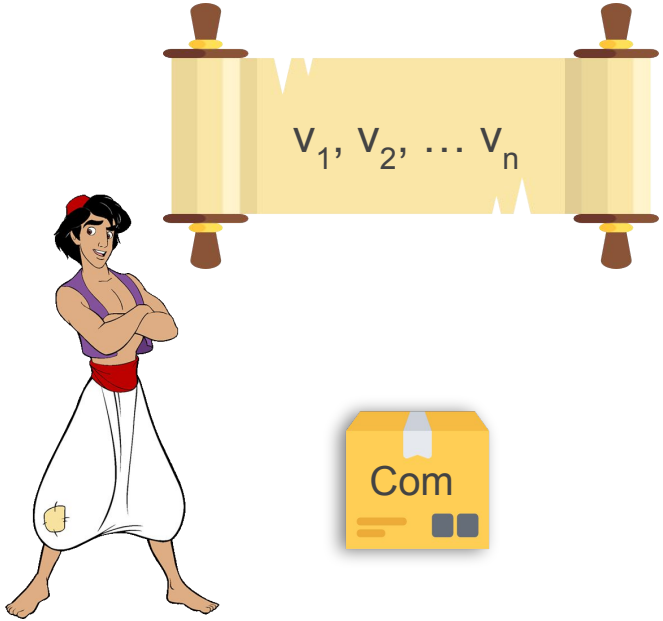
Storage Delegation



Storage Delegation

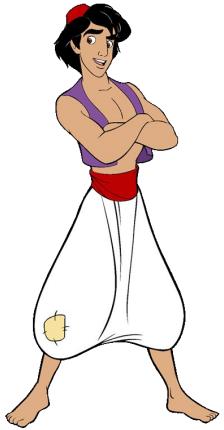


Storage Delegation

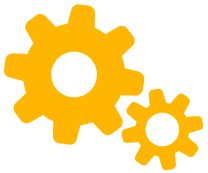


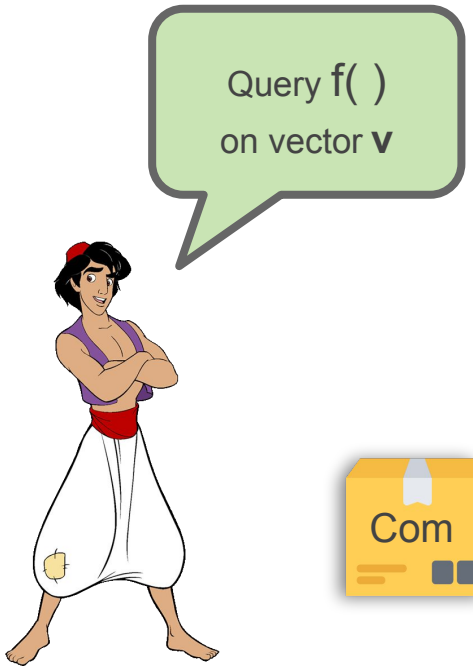
v

Storage Delegation

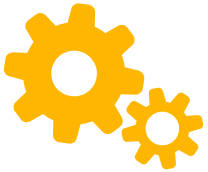


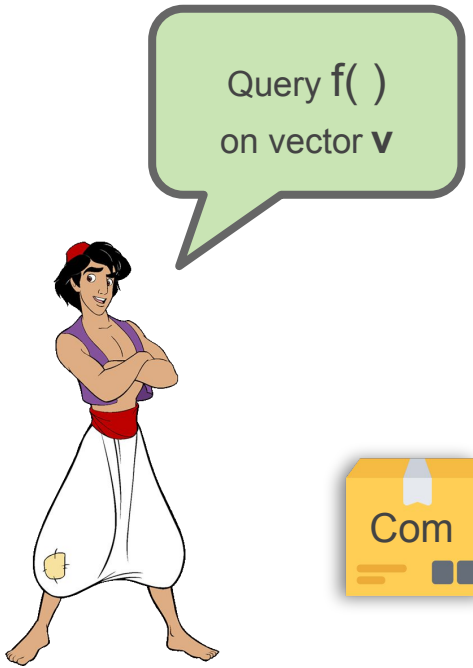
Storage Delegation

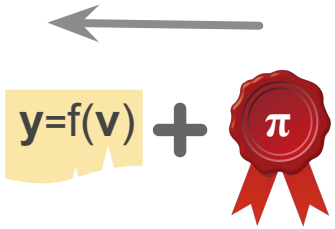




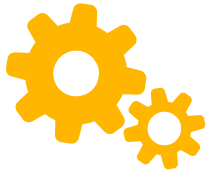
Storage Delegation

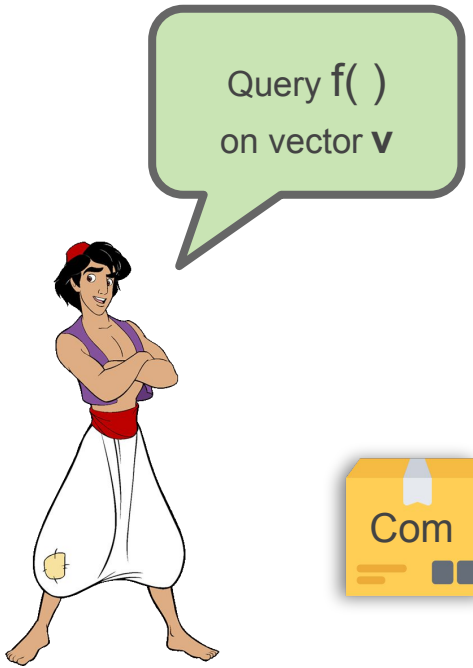


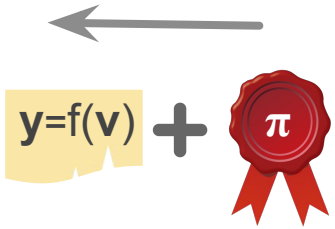




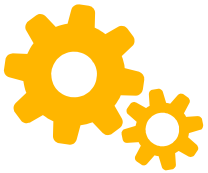
Storage Delegation

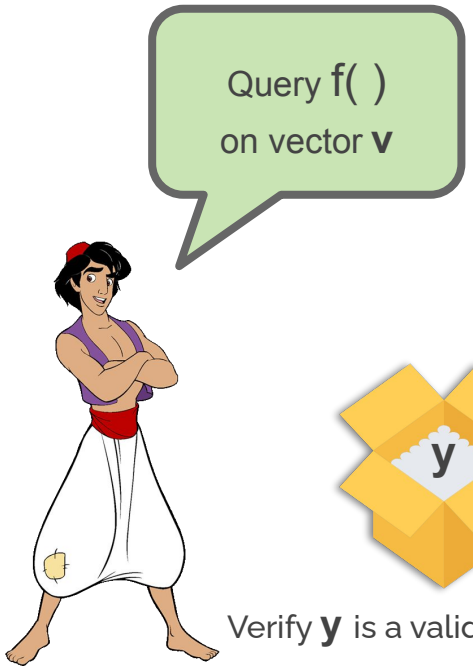






Storage Delegation





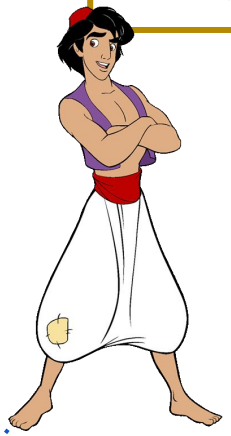
Verify **y** is a valid opening wrt Com

Stateless Cryptocurrency

Distributed Ledger

New Block

A → B, **5 Coins**
C → D, **10 Coins**
A → C, **5 Coins**



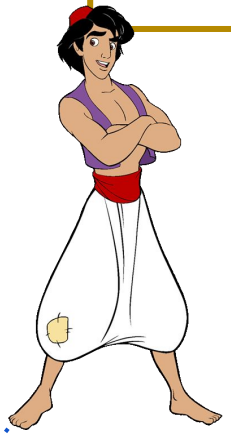
Distributed Ledger

New Block

A → B, **5 Coins**
C → D, **10 Coins**
A → C, **5 Coins**

previous state

A: 20 Coins
B: 55 Coins
C: 30 Coins
D: 0 Coins
E: 90 Coins
F: 10 Coins
...



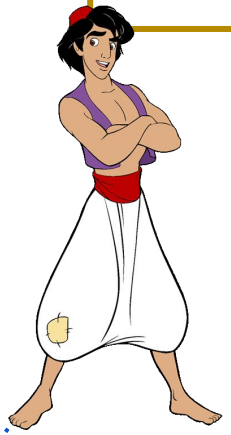
Distributed Ledger

New Block

A → B, **5 Coins**
C → D, **10 Coins**
A → C, **5 Coins**

previous state

Validate Block

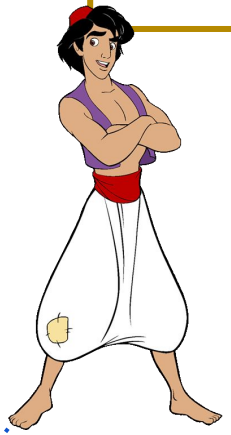


A: 20 Coins
B: 55 Coins
C: 30 Coins
D: 0 Coins
E: 90 Coins
F: 10 Coins
...

Distributed Ledger

New Block

A → B, **5 Coins**
C → D, **10 Coins**
A → C, **5 Coins**



previous state

A: 20 Coins
B: 55 Coins
C: 30 Coins
D: 0 Coins
E: 90 Coins
F: 10 Coins
...

new state

A: 10 Coins
B: 60 Coins
C: 25 Coins
D: 10 Coins
E: 90 Coins
F: 10 Coins
...

Grows over time

High storage & download cost

Distributed Ledger

New Block

A → B, **5 Co**
C → D, **10 C**
A → C, **5 Co**

Solution:

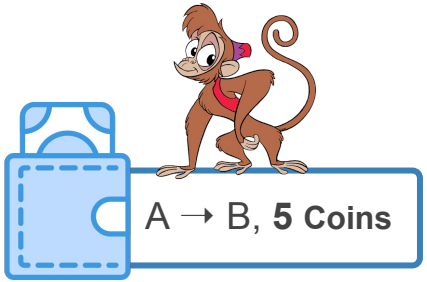
VC

Grows over time

High storage & download cost

Transaction validation

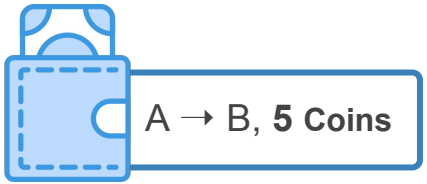
current state



A: 20 Coins
B: 55 Coins
C: 30 Coins
D: 0 Coins
E: 90 Coins
F: 10 Coins
...

Transaction validation

current state



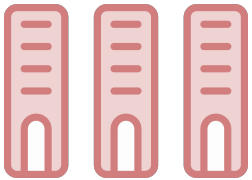
Validate Transaction

A: 20 Coins
B: 55 Coins
C: 30 Coins
D: 0 Coins
E: 90 Coins
F: 10 Coins
...

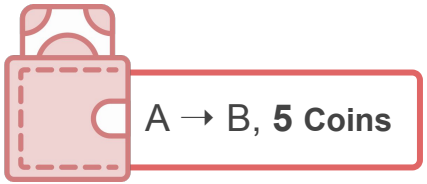
Check that

A has > **5** Coins

Stateful validation

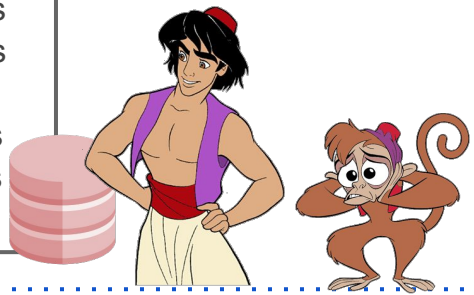


current state



Validate Transaction

A: 20 Coins
B: 55 Coins
C: 30 Coins
D: 0 Coins
E: 90 Coins
F: 10 Coins
...



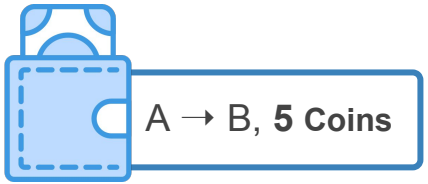
Stateless validation

State Commitment
(vector of balances)

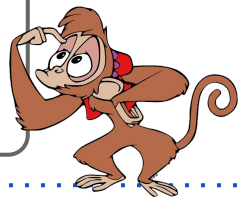
A: 20 Coins
B: 55 Coins
C: 30 Coins
D: 0 Coins
E: 90 Coins
F: 10 Coins
...

Stateless validation

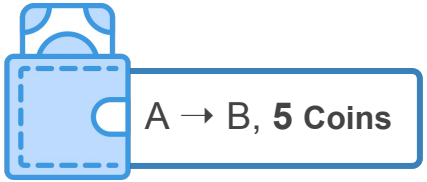
State Commitment
(vector of balances)



A: 20 Coins
B: 55 Coins
C: 30 Coins
D: 0 Coins
E: 90 Coins
F: 10 Coins
...



Stateless validation



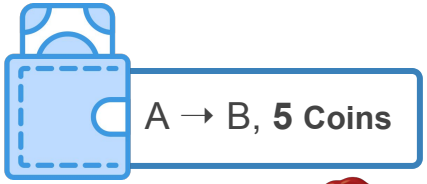
State Commitment
(vector of balances)

A: 20 Coins
B: 55 Coins
C: 30 Coins
D: 0 Coins
E: 90 Coins
F: 10 Coins
...

π_A
π_B
π_C
...

Stateless validation

State Commitment
(vector of balances)



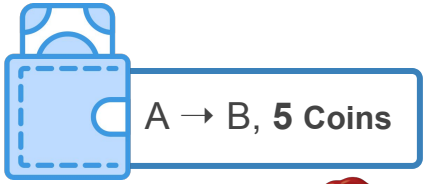
A: 20

A: 20 Coins
B: 55 Coins
C: 30 Coins
D: 0 Coins
E: 90 Coins
F: 10 Coins
...

π_A
π_B
π_C
...

Stateless validation

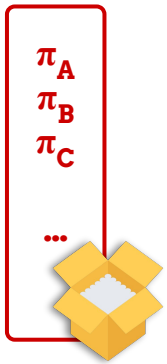
State Commitment
(vector of balances)



A: 20

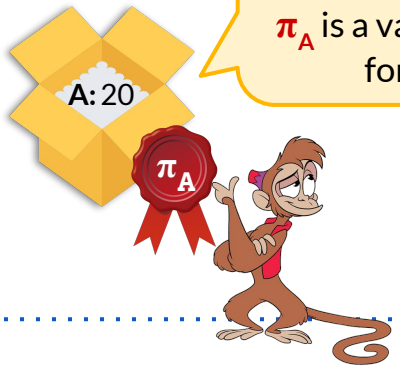
A: 20 Coins
B: 55 Coins
C: 30 Coins
D: 0 Coins
E: 90 Coins
F: 10 Coins
...

π_A
π_B
π_C
...



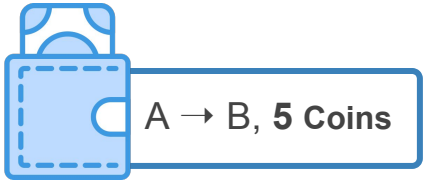
Check that

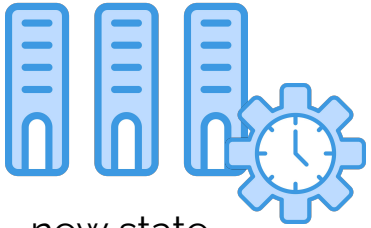
π_A is a valid opening for **Com**



More Requirements

State Commitment
(vector of balances)

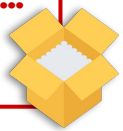




new state

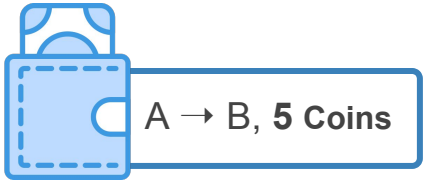
A: 20 Coins
B: 55 Coins
C: 30 Coins
D: 0 Coins
E: 90 Coins
F: 10 Coins
...

π_A
π_B
π_C
...



A: 15 Coins
B: 60 Coins
C: 30 Coins
D: 0 Coins
E: 90 Coins
F: 10 Coins
...

Updatability for Com



State Commitment
(vector of balances)

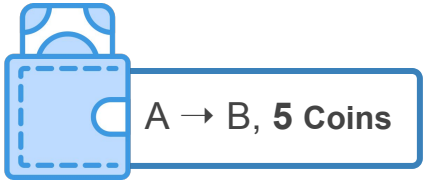
Update

A: 20 Coins
B: 55 Coins
C: 30 Coins
D: 0 Coins
E: 90 Coins
F: 10 Coins
...

π_A
π_B
π_C
...

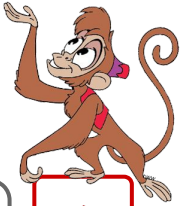
A: 15 Coins
B: 60 Coins
C: 30 Coins
D: 0 Coins
E: 90 Coins
F: 10 Coins
...

Updatability for Proofs



State Commitment
(vector of balances)

Update

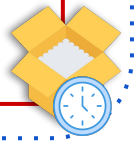


A: 20 Coins
B: 55 Coins
C: 30 Coins
D: 0 Coins
E: 90 Coins
F: 10 Coins
...

π_A
π_B
π_C
...

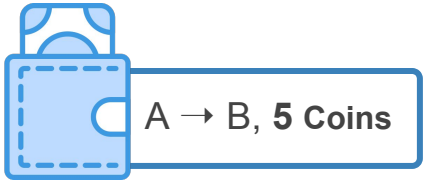
A: 15 Coins
B: 60 Coins
C: 30 Coins
D: 0 Coins
E: 90 Coins
F: 10 Coins
...

π'_A
π'_B
π'_C
...

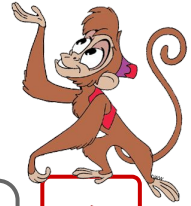


Updatability for Proofs

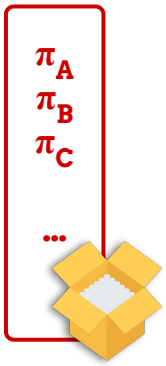
State Commitment
(vector of balances)



Update



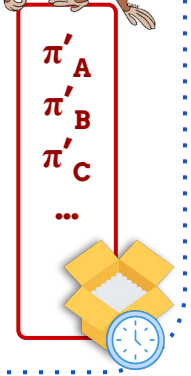
A: 20 Coins
B: 55 Coins
C: 30 Coins
D: 0 Coins
E: 90 Coins
F: 10 Coins
...



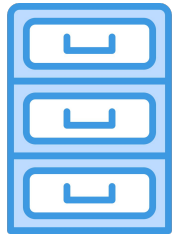
Updates types

- **hint**: needs opening
- **key**: needs static key
- **key-less**: no key

A: 15 Coins
B: 60 Coins
C: 30 Coins
D: 0 Coins
E: 90 Coins
F: 10 Coins
...

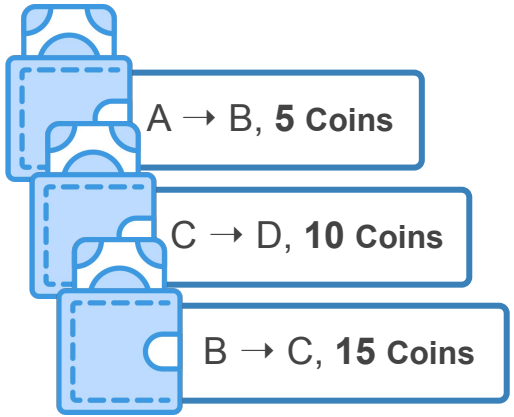


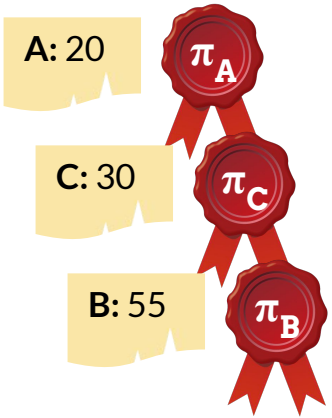
Aggregation



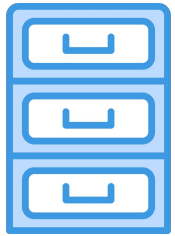
New Block

A → B, **5 Coins**
C → D, **10 Coins**
B → C, **15 Coins**



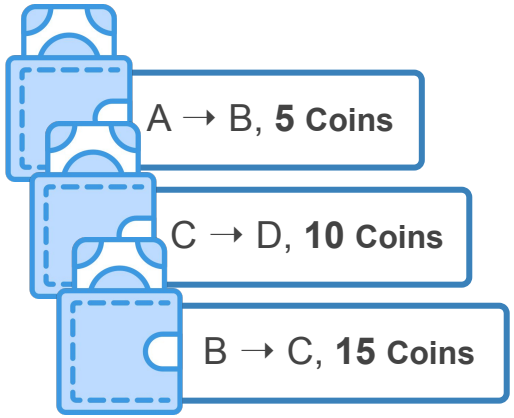


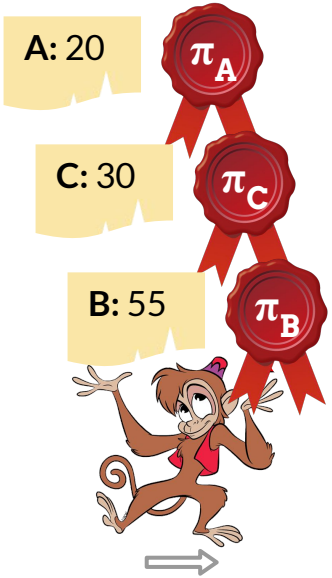
Aggregation

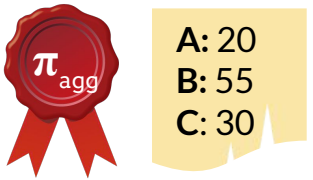


New Block

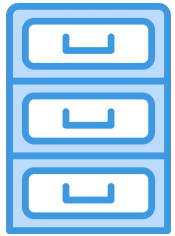
A → B, **5 Coins**
C → D, **10 Coins**
B → C, **15 Coins**





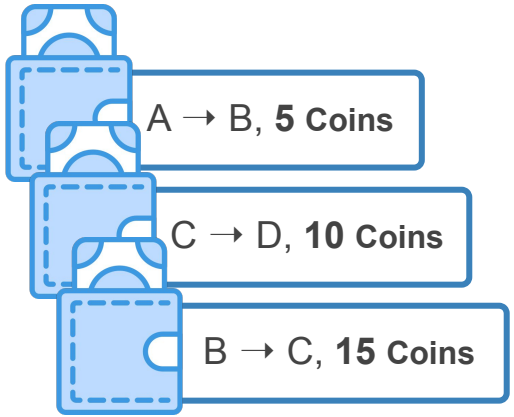


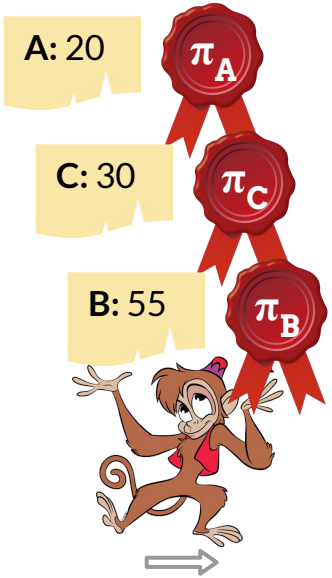
Aggregation

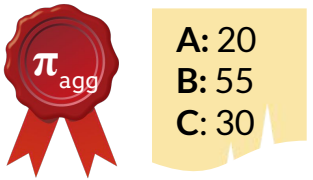


New Block

A → B, **5 Coins**
C → D, **10 Coins**
B → C, **15 Coins**



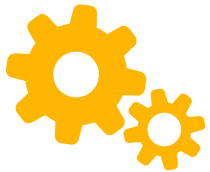




- same-commitment/**cross-commitment**
- one-hop/**unbounded**
- **incremental**: can disaggregate

Our Constructions

Contributions

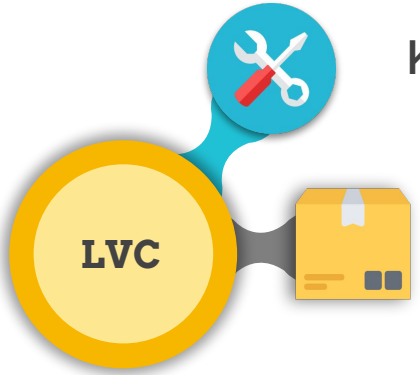


- Framework to add Properties from Homomorphic LVCs
- Unbounded Aggregation & Updatability with static keys
- New simple constructions for Inner-Product Vector Commitments
- Special Subset Openings and Subset opening with Efficient Verifier
- Maintainable Tree-based construction from any LVC
- Trade-offs for the prover with flexible Space and Time savings
- All this with reusable Setup “powers of tau”

LVC Algorithms

KeyGen(λ, n) → (ck, vk)

LVC Algorithms



KeyGen(λ, n) → (ck, vk)

Commit(ck, **v**) → (C, aux)

LVC Algorithms

KeyGen(λ, n) → (ck, vk)

Commit(ck, **v**) → (C, aux)

Open(ck, aux, f, **y**) → π

LVC Algorithms

KeyGen(λ, n) → (ck, vk)

Commit(ck, **v**) → (C, aux)

Open(ck, aux, f, **y**) → π $\mathcal{R}$: f(**v**) = **y**

LVC Algorithms

KeyGen(λ, n) → (ck, vk)

Commit(ck, **v**) → (C, aux)

Open(ck, aux, f, **y**) → π $\mathcal{R}$: f(**v**) = **y**

Verify(vk, C, f, **y**, π) → 0/1

Inner Product VC

Inner Product of a committed vector with a public vector

$f_b(a) = \langle a, b \rangle$

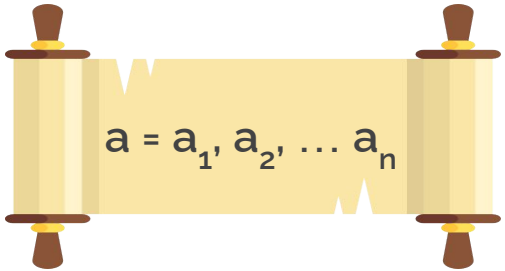
$a, b \in \mathbb{F}^n$

Inner Product VC

Inner Product of a committed vector with a public vector

$f_b(a) = \langle a, b \rangle$

$a, b \in \mathbb{F}^n$



Polynomial A(X)

Monomial basis

$$M_i(X) = X^i$$

Lagrange basis

$$L_i(X) = \prod_{j \neq i} \frac{X - j}{i - j}$$

Monomial basis

$$M_i(X) = X^i$$

$a = a_1, a_2, \ldots a_n$

$A(X) = a_1 + a_2X + a_3X^2 + \ldots a_nX^{n-1}$

Lagrange basis

$$L_i(X) = \prod_{j \neq i} \frac{X - j}{i - j}$$

Monomial basis

$$M_i(X) = X^i$$

$a = a_1, a_2, \ldots a_n$

$A(X) = a_1 + a_2X + a_3X^2 + \ldots a_nX^{n-1}$

Lagrange basis

$$L_i(X) = \prod_{j \neq i} \frac{X - j}{i - j}$$

$a = a_1, a_2, \ldots a_n$

$A(X) = a_1L_1(X) + a_2L_2(X) + \ldots a_nL_n(X)$

Monomial basis

$$M_i(X) = X^i$$

$a = a_1, a_2, \ldots a_n$

$A(X) = a_1 + a_2X + a_3X^2 + \ldots a_nX^{n-1}$

Lagrange basis

$$L_i(X) = \prod_{j \neq i} \frac{X - j}{i - j}$$

$a = a_1, a_2, \ldots a_n$

$A(X) = a_1L_1(X) + a_2L_2(X) + \ldots a_nL_n(X)$

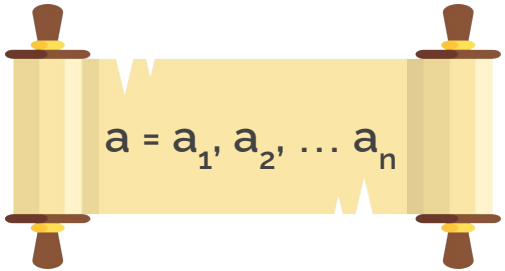
$A(1) = a_1,\ A(2) = a_2,\ \ldots\ A(n) = a_n$

Inner Product VC

Inner Product of a committed vector with a public vector

$f_b(a) = \langle a, b \rangle$

$a, b \in \mathbb{F}^n$



$A(X) = a_1 + a_2X + a_3X^2 + \ldots a_nX^{n-1}$

Background

Bilinear Groups

$$[a]_1 \in \mathbf{G}_1, [b]_2 \in \mathbf{G}_2$$

$$e : \mathbf{G}_1 \times \mathbf{G}_2 \rightarrow \mathbf{G}_T$$

$$e([a]_1, [b]_2) = [ab]_T$$

Inner Product VC

KeyGen(λ, n) → ck: $[1]_1, [\tau]_1, [\tau^2]_1, \ldots [\tau^{n-1}]_1$

Inner Product VC

KeyGen(λ, n) → ck: $[1]_1, [\tau]_1, [\tau^2]_1, \ldots [\tau^{n-1}]_1$

Commit(ck, **a**) → $C = [A(\tau)]_1$

Inner Product VC

KeyGen(λ, n) → ck: $[1]_1, [\tau]_1, [\tau^2]_1, \ldots [\tau^{n-1}]_1$

Commit(ck, **a**) → $C = [A(\tau)]_1$

$A(X) = a_1 + a_2X + a_3X^2 + \ldots a_nX^{n-1}$

Inner Product VC

KeyGen(λ, n) → ck: $[1]_1, [\tau]_1, [\tau^2]_1, \ldots [\tau^{n-1}]_1$

Commit(ck, **a**) → $C = [A(\tau)]_1$

$A(X) = a_1 + a_2X + a_3X^2 + \ldots a_nX^{n-1}$

Open(ck, **a**, **b**, y=<**a**, **b**>) → π : $[H(\tau)]_1, [R(\tau)]_1, [\check{R}(\tau)]_1$

Inner Product VC

KeyGen(λ, n) → ck: $[1]_1, [\tau]_1, [\tau^2]_1, \ldots [\tau^{n-1}]_1$

Commit(ck, **a**) → $C = [A(\tau)]_1$

$A(X) = a_1 + a_2X + a_3X^2 + \ldots a_nX^{n-1}$

Open(ck, **a**, **b**, y=<**a**, **b**>) → π :

$[H(\tau)]_1, [R(\tau)]_1, [\check{R}(\tau)]_1$

$B(X) = b_n + b_{n-1}X + b_{n-2}X^2 + \ldots b_1X^{n-1}$

Inner Product VC

KeyGen(λ, n) → ck: $[1]_1, [\tau]_1, [\tau^2]_1, \ldots [\tau^{n-1}]_1$

Commit(ck, **a**) → $C = [A(\tau)]_1$

Open(ck, **a**, **b**, y=<**a**, **b**>) → π : $[H(\tau)]_1, [R(\tau)]_1, [\check{R}(\tau)]_1$

$B(X) = b_n + b_{n-1}X + b_{n-2}X^2 + \ldots b_1X^{n-1}$

$A(X) = a_1 + a_2X + a_3X^2 + \ldots a_nX^{n-1}$

$A(X)\,B(X) =$

Inner Product VC

KeyGen(λ, n) $\rightarrow$ ck: $[1]_1, [\tau]_1, [\tau^2]_1, \ldots [\tau^{n-1}]_1$

Commit(ck, **a**) $\rightarrow$ $C = [A(\tau)]_1$

Open(ck, **a**, **b**, y=<**a**, **b**>) $\rightarrow$ π : $[H(\tau)]_1, [R(\tau)]_1, [\check{R}(\tau)]_1$

$B(X) = b_n + b_{n-1}X + b_{n-2}X^2 + \ldots b_1X^{n-1}$

$A(X) = a_1 + a_2X + a_3X^2 + \ldots a_nX^{n-1}$

$A(X)\,B(X) = \langle \mathbf{a}, \mathbf{b} \rangle X^{n-1}$

Inner Product VC

KeyGen(λ, n) $\rightarrow$ ck: $[1]_1, [\tau]_1, [\tau^2]_1, \ldots [\tau^{n-1}]_1$

Commit(ck, **a**) $\rightarrow$ $C = [A(\tau)]_1$

Open(ck, **a**, **b**, y=<**a**, **b**>) $\rightarrow$ π : $[H(\tau)]_1, [R(\tau)]_1, [\check{R}(\tau)]_1$

$B(X) = b_n + b_{n-1}X + b_{n-2}X^2 + \ldots b_1X^{n-1}$

$A(X) = a_1 + a_2X + a_3X^2 + \ldots a_nX^{n-1}$

$A(X)\,B(X) = \langle \mathbf{a}, \mathbf{b} \rangle X^{n-1} + H(X)\,X^n$

Inner Product VC

KeyGen(λ, n) $\rightarrow$ ck: $[1]_1, [\tau]_1, [\tau^2]_1, \ldots [\tau^{n-1}]_1$

Commit(ck, **a**) $\rightarrow$ $C = [A(\tau)]_1$

Open(ck, **a**, **b**, y=<**a**, **b**>) $\rightarrow$ π: $[H(\tau)]_1, [R(\tau)]_1, [\check{R}(\tau)]_1$

$B(X) = b_n + b_{n-1}X + b_{n-2}X^2 + \ldots b_1X^{n-1}$

$A(X) = a_1 + a_2X + a_3X^2 + \ldots a_nX^{n-1}$

$A(X)\,B(X) = \langle \mathbf{a}, \mathbf{b} \rangle X^{n-1} + H(X)\,X^n + R(X)$

Inner Product VC

KeyGen(λ, n) $\rightarrow$ ck: $[1]_1, [\tau]_1, [\tau^2]_1, \ldots [\tau^{n-1}]_1$

Commit(ck, **a**) $\rightarrow$ $C = [A(\tau)]_1$

Open(ck, **a**, **b**, y=<**a**, **b**>) $\rightarrow$ π: $[H(\tau)]_1, [R(\tau)]_1, [\check{R}(\tau)]_1$

$B(X) = b_n + b_{n-1}X + b_{n-2}X^2 + \ldots b_1X^{n-1}$

$A(X) = a_1 + a_2X + a_3X^2 + \ldots a_nX^{n-1}$

$A(X)\,B(X) = \langle \mathbf{a}, \mathbf{b} \rangle X^{n-1} + H(X)\,X^n + R(X)$

$\check{R}(X) = XR(X)$

Inner Product VC

KeyGen(λ, n) $\rightarrow$ ck: $[1]_1, [\tau]_1, [\tau^2]_1, \ldots [\tau^{n-1}]_1$

Commit(ck, **a**) $\rightarrow$ $C = [A(\tau)]_1$

Open(ck, **a**, **b**, $y=\langle \mathbf{a}, \mathbf{b} \rangle$) $\rightarrow$ π: $[H(\tau)]_1, [R(\tau)]_1, [\check{R}(\tau)]_1$

Verify(vk, C, **b**, y, π): $A(X)\,B(X) = y\,X^{n-1} + H(X)\,X^n + R(X)$ $\quad \check{R}(X) = XR(X)$

Inner Product VC

KeyGen(λ, n) → ck: $[1]_1, [\tau]_1, [\tau^2]_1, \ldots [\tau^{n-1}]_1$

Commit(ck, **a**) → $C = [A(\tau)]_1$

Open(ck, **a**, **b**, y=<**a**, **b**>) → π : $[H(\tau)]_1, [R(\tau)]_1, [\check{R}(\tau)]_1$

Verify(vk, C, **b**, y, π): $e([A(\tau)]_1, [B(\tau)]_2) = e([y\,\tau^{n-1}]_1, [1]_2)\, e([H(\tau)]_1, [\tau^n]_2)\, e([R(\tau)]_1, [1]_2)$

Inner Product VC

KeyGen(λ, n) → ck: $[1]_1, [\tau]_1, [\tau^2]_1, \ldots [\tau^{n-1}]_1$ vk: $[1]_2, [\tau]_2, \ldots [\tau^n]_2$

Commit(ck, **a**) → $C = [A(\tau)]_1$

Open(ck, **a**, **b**, y=<**a**, **b**>) → π : $[H(\tau)]_1, [R(\tau)]_1, [\check{R}(\tau)]_1$

Verify(vk, C, **b**, y, π): $e([A(\tau)]_1, [B(\tau)]_2) = e([y\,\tau^{n-1}]_1, [1]_2)\, e([H(\tau)]_1, [\tau^n]_2)\, e([R(\tau)]_1, [1]_2)$

Inner Product VC

KeyGen(λ, n) → ck: $[1]_1, [\tau]_1, [\tau^2]_1, \ldots [\tau^{n-1}]_1$ vk: $[1]_2, [\tau]_2, \ldots [\tau^n]_2$

Commit(ck, **a**) → $C = [A(\tau)]_1$

Open(ck, **a**, **b**, y=<**a**, **b**>) → π : $[H(\tau)]_1, [R(\tau)]_1, [\check{R}(\tau)]_1$

Verify(vk, C, **b**, y, π): $e([\check{R}(\tau)]_1, [1]_2) = e([R(\tau)]_1, [\tau]_2)$ $\check{R}(X) = XR(X)$

Linear-Map VC from IP

$f : \mathbb{F}^n \to \mathbb{F}^\ell$

KeyGen(λ, n) ⟶ ck: $[1]_1, [\tau]_1, [\tau^2]_1, \ldots [\tau^{n-1}]_1$ vk: $[1]_2, [\tau]_2, \ldots [\tau^n]_2$

Commit(ck, **a**) ⟶ $C = [A(\tau)]_1$

Open(ck, **a**, f: **B**=(**b**$_i$), **y**) ⟶ $\pi_1, \pi_2, \ldots \pi_\ell$ ⟶ π_{agg}

Verify(vk, C, f , **y**, π*)

Range Subset Opening

$a = a_1, a_2, a_3, \dots a_n$ → $A(X) = a_1 + a_2 X + a_3 X^2 + \dots a_n X^{n-1}$

$ck = [1]_1, [\tau]_1, [\tau^2]_1, \dots [\tau^{n-1}]_1$ → $C = [A(\tau)]_1 = a_1 + a_2 \tau + a_3 \tau^2 + \dots a_n \tau^{n-1}$

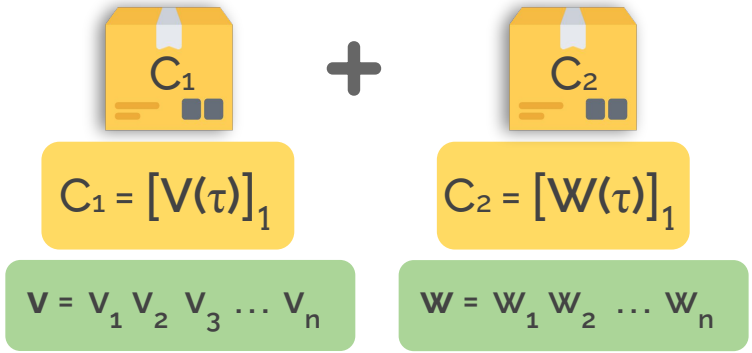
$B(X) = X^{n-k}$ → $A(X)\,B(X) = a_k X^{n-1} + a_{k+1} X^n + \dots + a_{k+m} X^{n+m-1} + H(X)\,X^{n+m} + R(X)$

Open $a_k \rightarrow a_{k+m}$ → $[H(\tau)]_1, [R(\tau)]_1$

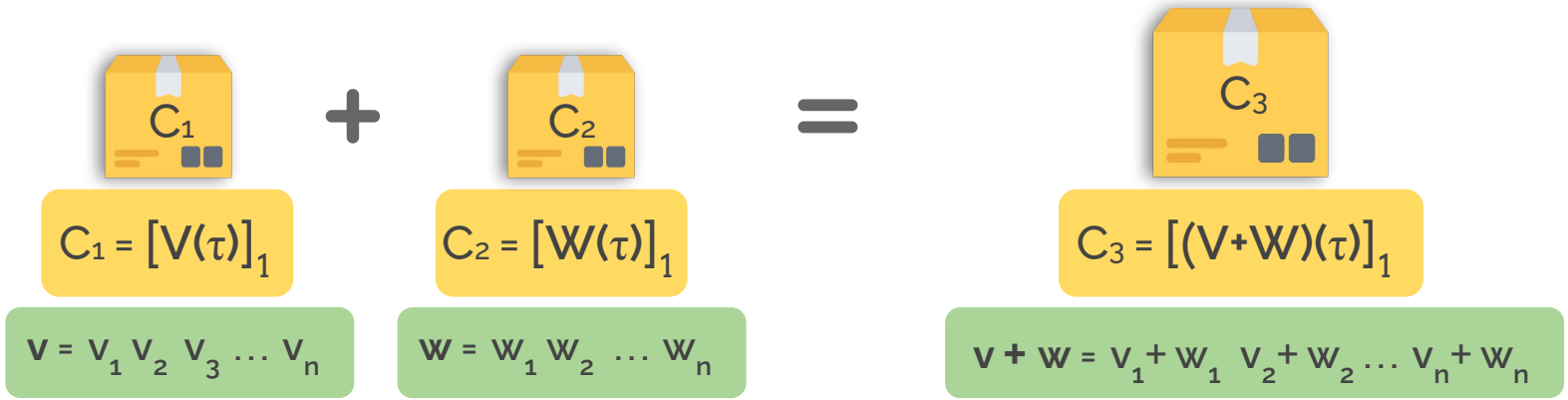
$deg(R(X)) < n-1$

Properties of LVC

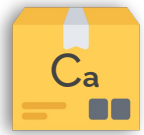
Homomorphism for Com



Homomorphism for Com



Homomorphism for Proof

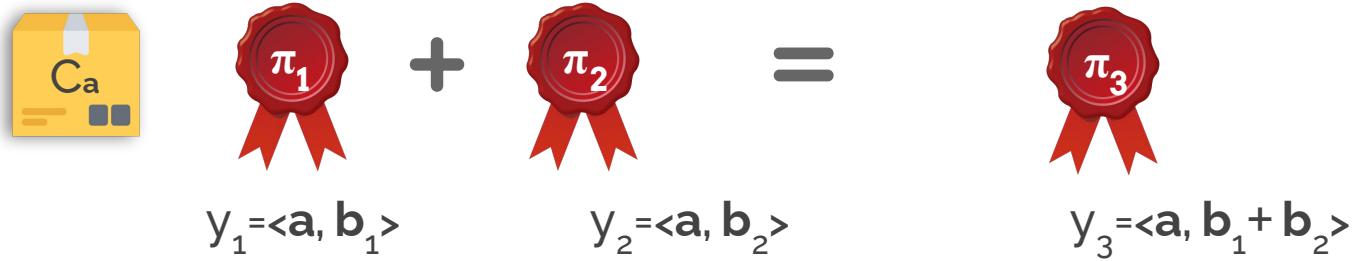


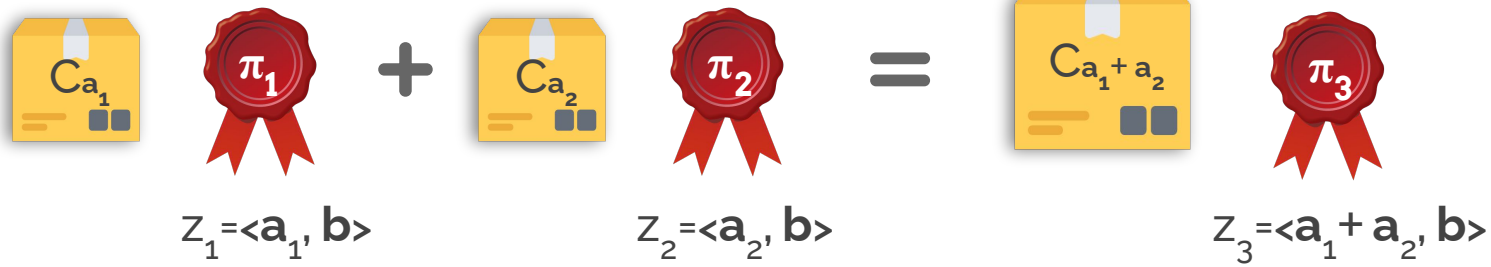
+

=

$y_1 = \langle a, b_1 \rangle$ $y_2 = \langle a, b_2 \rangle$ $y_3 = \langle a, b_1 + b_2 \rangle$

Homomorphism for Proof





Updatability for Com

$a' = a_1\ a_2 \ldots a_i + \delta \ldots a_n$

upk: $[1]_1, [\tau]_1, [\tau^2]_1, \ldots [\tau^{n-1}]_1$

$C' = [A(\tau)]_1 + [\delta\tau^i]_1$

Updatability for Proof

$a' = a_1\ a_2 \dots a_i + \delta \dots a_n$

upk: $[1]_1, [\tau]_1, [\tau^2]_1, \dots [\tau^{n-1}]_1$

$C' = [A(\tau)]_1 + [\delta\tau^i]_1$

$y' = \langle \mathbf{a} + \delta \mathbf{e}_i, \mathbf{b} \rangle = y_1 = \langle \mathbf{a}, \mathbf{b} \rangle + \delta \langle \mathbf{e}_i, \mathbf{b} \rangle$

Updatability for Proof

$\mathbf{a'} = a_1\ a_2 \dots a_i + \delta \dots a_n$

upk: $[1]_1, [\tau]_1, [\tau^2]_1, \dots [\tau^{n-1}]_1$

$C' = [A(\tau)]_1 + [\delta\tau^i]_1$

$A(X)\,B(X) = \langle \mathbf{a}, \mathbf{b} \rangle X^{n-1} + H(X)\,X^n + R(X)$

$y' = \langle \mathbf{a} + \delta \mathbf{e}_i, \mathbf{b} \rangle = y_1 = \langle \mathbf{a}, \mathbf{b} \rangle + \delta \langle \mathbf{e}_i, \mathbf{b} \rangle$

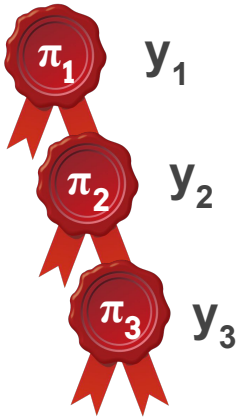
$\pi_{ij}: \langle \mathbf{e}_i, \mathbf{e}_j \rangle = 0$

$H(X)/R(X) = X^{i+n-j}$

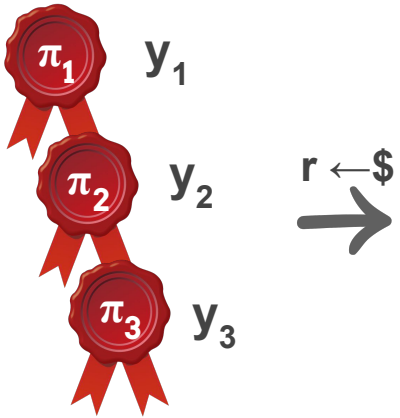
$\pi_{ii}: \langle \mathbf{e}_i, \mathbf{e}_i \rangle = 1$

$H(X)/R(X) = 0$

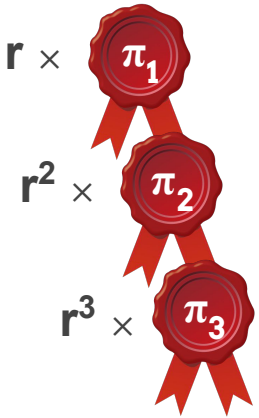
Aggregation

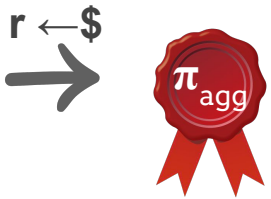


Aggregation



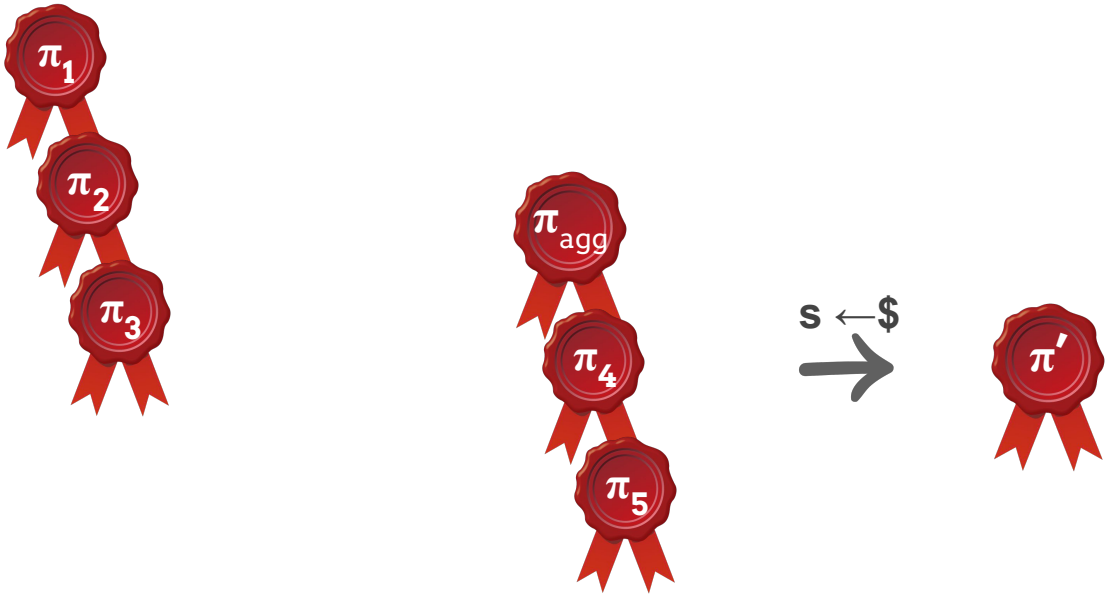
Aggregation



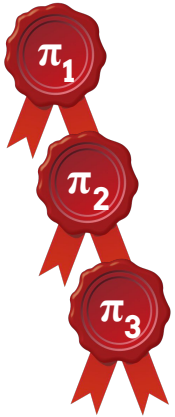


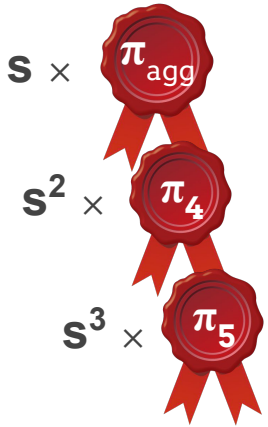
$$\pi_{agg} = \sum r^i \pi_i$$

Aggregation

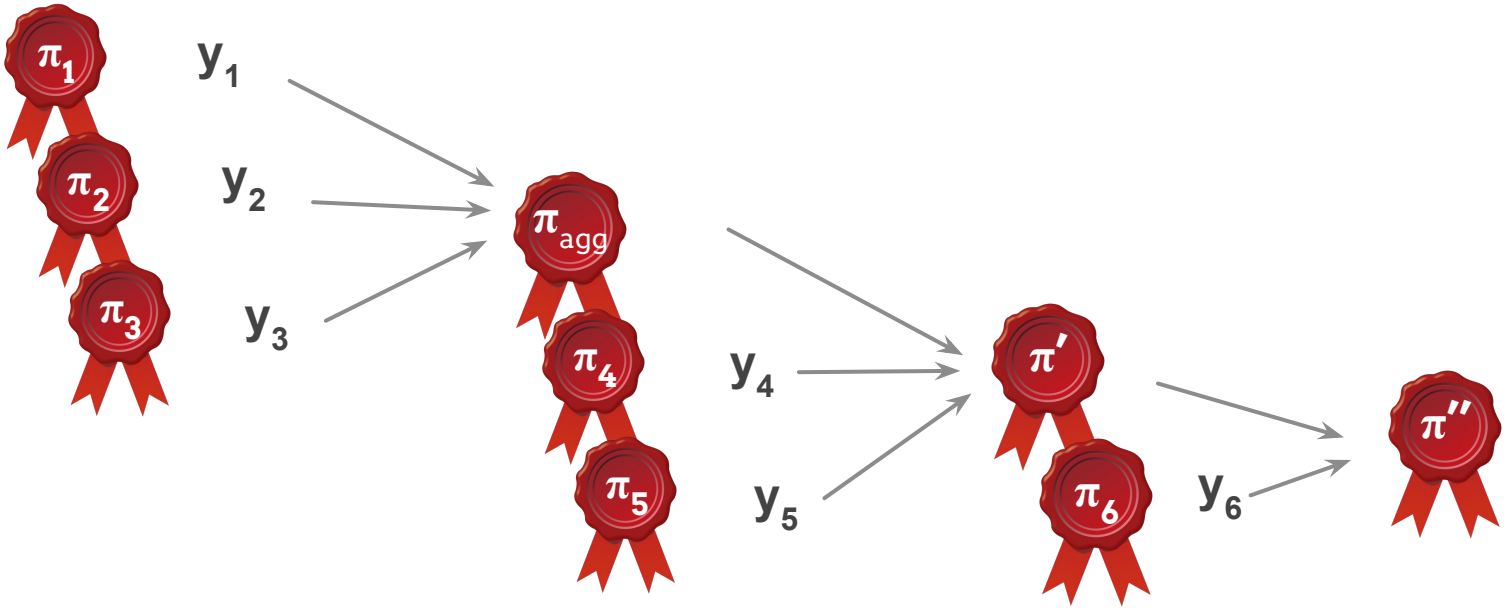


Aggregation





Aggregation

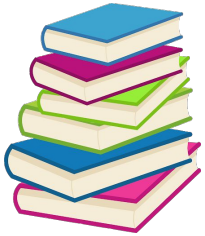


LVC comparison

$f : \mathbb{F}^n \to \mathbb{F}^\ell$

Schemes		|ck|	|vk|	|upk|	Updates	Aggregation	SVC	LVC
[LRY16]		n	n	—	✗	✗	✗	✓
[LM19]	SVC	n^2	n	n	key	✗	✓	✗
	LVC	$n\ell$	n	n	key	✗	✓	✓
Pointproofs		n	n	n	key	cross & one-hop	✓	✗
[CFG+20]		1	1	1	hint	same & incremental	✓	✗
Our Monomial		**n**	**n**	—	**key-less**	**cross & unbounded**	✓	✓
Our Lagrange		**n**	ℓ	**n**	**key**	**cross & unbounded**	✓	✓

References



[CF13] **Vector Commitments and their Applications**, by D. Catalano, D. Fiore, in PKC'13, https://eprint.iacr.org/2011/495

[CFG+20] **Vector Commitment Techniques and Applications to Verifiable Decentralized Storage**, by M. Campanelli, D. Fiore, N. Greco, D. Kolonelos, L. Nizzardo, in ASIACRYPT'20, https://eprint.iacr.org/2020/149

[GRWZ20] **Pointproofs: Aggregating Proofs for Multiple Vector Commitments**, by S. Gorbunov, L. Reyzin, H. Wee, Z. Zhang, in CCS'20, https://eprint.iacr.org/2020/419

[KZG10] **Constant-Size Commitments to Polynomials and Their Applications**, by A. Kate, G. Zaverucha, I. Goldberg, in ASIACRYPT'10,

[LRY10] **Functional commitment schemes: From polynomial commitments to pairing-based accumulators from simple assumptions**, by B. Libert, S. Ramanna and M. Yung, in ICALP'16

[LM19] **Subvector Commitments with Application to Succinct Arguments**, by R. W. F. Lai, G. Malavolta, in CRYPTO'19, https://eprint.iacr.org/2018/705

Maintainable Trees

Pairing-based VC

Pairing-Based VC

[KZG10], [CF13], [LM19],[GRWZ20], [TAB+20]

Advantages:

- constant-size openings
- constant-time verification
- **aggregatable**

Downsides:

- linear prover: no tradeoffs
- trusted setup
- linear parameters = bound on vector size
- updates: hard to maintain proofs

Tree-based solutions

[Merkle Trees, Hyperproofs, Verkle Trees]

Downsides:

- log-size openings
- log-time verification
- non-aggregatable

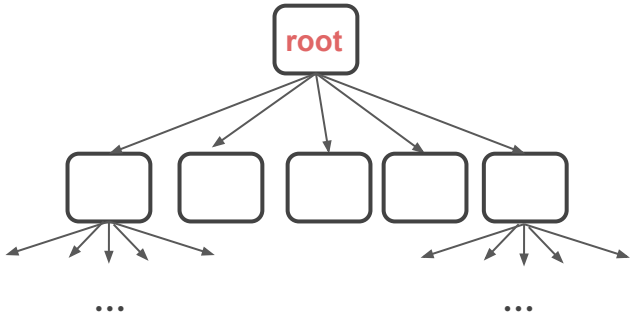
Advantages:

- linear prover: **tradeoffs**
- transparent setup
- independent parameters
- maintainability for proofs

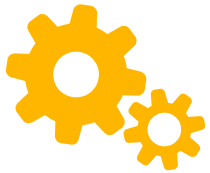
Tree-based Techniques

Ideas:

- tree with maintainability for proofs
- generic for any VC
- any arity **k** vs. binary
    - tune the tradeoff
    - dimension **n = mk**:
        - store **m** elements
        - pay **log m** in size / prove time
        - update all in **log m**
- reusable trusted setup (Groth16)
- from multivariate polynomial to univariate



Hyperproofs Trees



[SCP+22] Hyperproofs: Aggregating and Maintaining Proofs in Vector Commitments
Srinivasan, Chepurnoy, Papamanthou, Tomescu, Zhang, USENIX22

- Using Multi-Linear Trees

[PST13] **Signatures of Correct Computation**, Papamanthou, Shi, Tamassia, TCC13

+ Homomorphic
+ Efficient updates
+ **IPA**-friendly

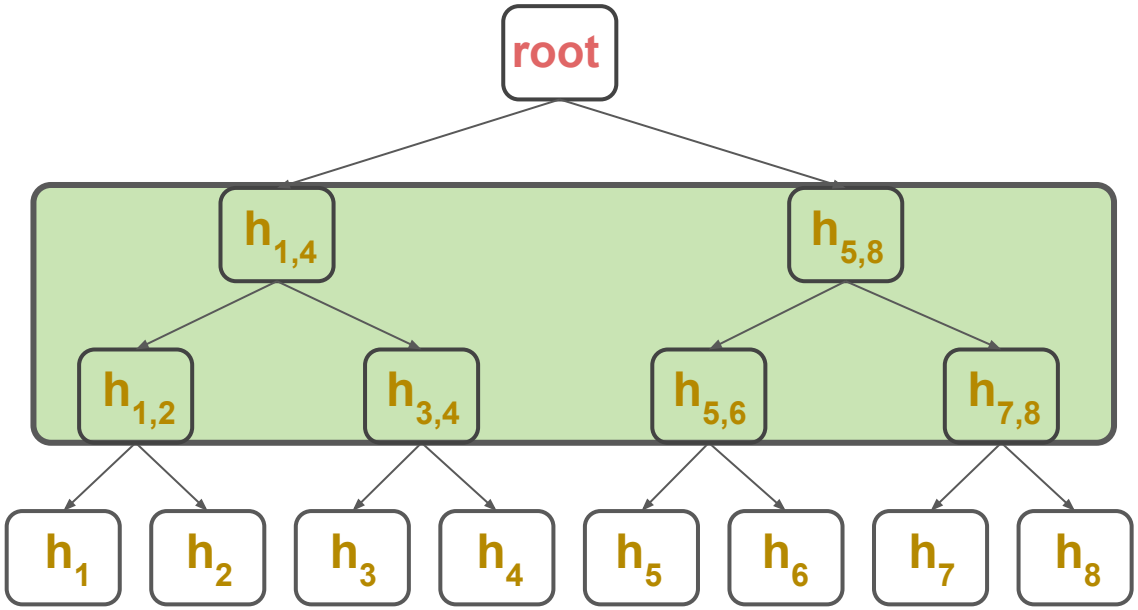
[BMM+] **Proofs for Inner Pairing Products and Applications**, Bünz, Maller, Mishra, Tyagi, Vesely

  - Subvector proofs
  - Proof aggregation

- Trusted setup
- Linear size public parameters
- Subvector proofs are 54 KiB

Operation	**Cost**
Commit	$\mathbf{n}\ \mathbb{G}_1$
Compute **all** proofs	$\mathbf{n} \log(\log \mathbf{n})\ \mathbb{G}_1$
Update digest	$1\ \mathbb{G}_1$
Update **all** proofs	$\log \mathbf{n}\ \mathbb{G}_1$
Verify 1 proof	$\log \mathbf{n}\ \mathbb{P}$
Aggregate **b** proofs / Verify	$\mathbf{b} \log \mathbf{n}$ $(\mathbb{G}_1+\mathbb{G}_2+\mathbb{P})$

Space Trade-off



Future Directions

What's next?

- Transparent Setup Vector Commitments
- Functional Vector Commitments for larger classes of functions
- Post-Quantum Vector Commitments with same properties
- Argument of Knowledge of Openings - in progress
- Structure-Preserving Vector Commitments
- Recursivity for the tree-based construction

Thanks!

Vector Commitments Project

cryptonet.org

Credits

Special thanks to all those who made and released these resources for free:

- Presentation template by SlidesCarnival
- Illustrations by Iconfinder