NYU CRYPTOSEC SEMINAR | OCT. 13, 2026

$$\frac{1}{X+i}\cdot\frac{1}{X+j}=\frac{1}{j-i}\left(\frac{1}{X+i}-\frac{1}{X+j}\right)$$

$$\mathscr{C}=\begin{bmatrix}\frac{1}{a_1-b_1} & \frac{1}{a_1-b_2} & \cdots & \frac{1}{a_1-b_n}\\ \frac{1}{a_2-b_1} & \frac{1}{a_2-b_2} & \cdots & \frac{1}{a_2-b_n}\\ \vdots & \vdots & \ddots & \vdots\\ \frac{1}{a_m-b_1} & \frac{1}{a_m-b_2} & \cdots & \frac{1}{a_m-b_n}\end{bmatrix}$$

Partial Fraction Techniques For Cryptography

exploiting the algebra of rational functions to do cool crypto

$$\frac{1}{(X+k)^2}$$

Rohit Nema (Stanford)

joint works with



Dan Boneh
(Stanford)



Charanjit Jutla
(IBM Research)



Arnab Roy
(Mysten Labs)



Ertem Nusret Tas
(a16z Crypto Research)

$$\sum\frac{1}{X+k}$$

ia.cr/2025/2081 (EUROCRYPT'26) and ia.cr/2026/674 (CRYPTO'26)

Contents

Contents

1 The Core Primitive

Contents

1 The Core Primitive

2 Key-Value Commitments

Contents

1 The Core Primitive

2 Key-Value Commitments

2.1 Using Partial Fraction Decomposition (PFD)

Contents

1 The Core Primitive

2 Key-Value Commitments

2.1 Using Partial Fraction Decomposition (PFD)

3 Batch Encryption

Contents

1 The Core Primitive

2 Key-Value Commitments

2.1 Using Partial Fraction Decomposition (PFD)

3 Batch Encryption

3.1 A Relaxation Using PFD

3.2 Using *Linear Independence*

Contents

1 The Core Primitive

2 Key-Value Commitments

2.1 Using Partial Fraction Decomposition (PFD)

3 Batch Encryption

3.1 A Relaxation Using PFD

3.2 Using *Linear Independence*

4 Takeaways & Future Work

The Core Primitive: *Partial Fractions*

The Core Primitive: *Partial Fractions*

Main Idea

Encode cryptographic objects (e.g. keys, ciphertexts) using rational functions

The Core Primitive: *Partial Fractions*

> **Main Idea**
> Encode cryptographic objects (e.g. keys, ciphertexts) using rational functions

Use *simple* rational functions or "partial fractions" as first-class primitives

$$\frac{1}{X+c},\ c \in \mathbb{F}_p$$

finite field of prime order p

The Core Primitive: *Partial Fractions*

> **Main Idea**
> Encode cryptographic objects (e.g. keys, ciphertexts) using rational functions

Use *simple* rational functions or "partial fractions" as first-class primitives

$$\frac{1}{X+c},\ c \in \mathbb{F}_p$$

finite field of prime order p

Instantiated in a bilinear pairing group $\mathcal{G} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e)$

$\frac{1}{X+c}$ **maps to** $g_i^{\frac{1}{x+c}}$ **or** $\left[\frac{1}{x+c}\right]_i$ **for** $x \xleftarrow{\$} \mathbb{F}_p$

The Core Primitive: *Partial Fractions*

> **Main Idea**
> Encode cryptographic objects (e.g. keys, ciphertexts) using rational functions

Use *simple* rational functions or "partial fractions" as first-class primitives

$$\frac{1}{X+c}, c \in \mathbb{F}_p$$

finite field of prime order p

> **NOTE**
> Not the first to use rational functions (e.g. [BB04b,DY05,JL09,Wee16, Hab22,GV22])

The Core Primitive: *Partial Fractions*

Main Idea

Encode cryptographic objects (e.g. keys, ciphertexts) using rational functions

Use *simple* rational functions or "partial fractions" as first-class primitives

$$\frac{1}{X+c}, c \in \mathbb{F}_p$$

finite field of prime order p

NOTE

Not the first to use rational functions (e.g. [BB04b,DY05,JL09,Wee16, Hab22,GV22])

BUT…

First to systematically study their properties to build crypto

The Core Primitive: *Partial Fractions*

> **Main Idea**
> Encode cryptographic objects (e.g. keys, ciphertexts) using rational functions

Why? Rational functions have useful algebraic properties

The Core Primitive: *Partial Fractions*

> **Main Idea**
> Encode cryptographic objects (e.g. keys, ciphertexts) using rational functions

Why? Rational functions have useful algebraic properties

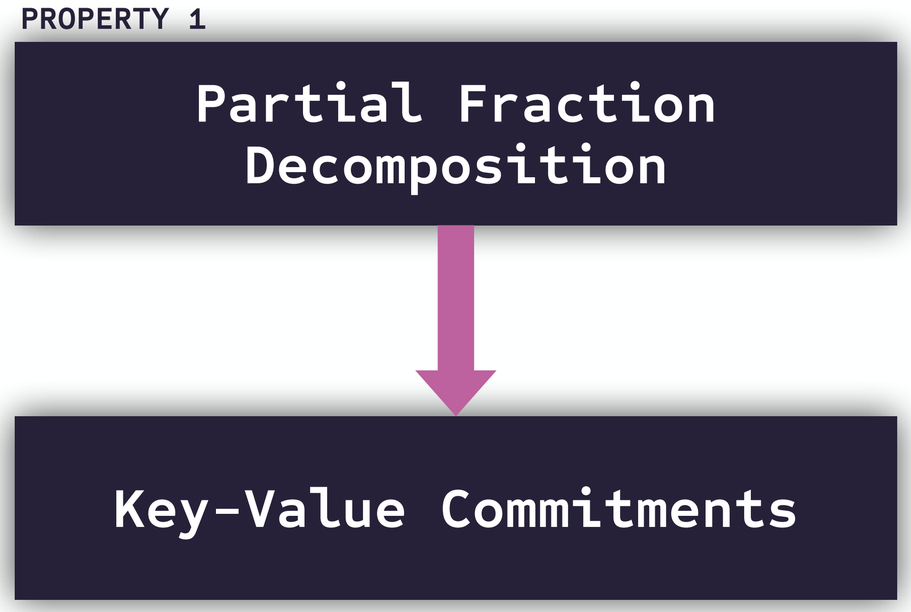
PROPERTY 1

Partial Fraction Decomposition

The Core Primitive: *Partial Fractions*

> **Main Idea**
> Encode cryptographic objects (e.g. keys, ciphertexts) using rational functions

Why? Rational functions have useful algebraic properties



The Core Primitive: *Partial Fractions*

> **Main Idea**
> Encode cryptographic objects (e.g. keys, ciphertexts) using rational functions

Why? Rational functions have useful algebraic properties

PROPERTY 1

Partial Fraction Decomposition



Key-Value Commitments

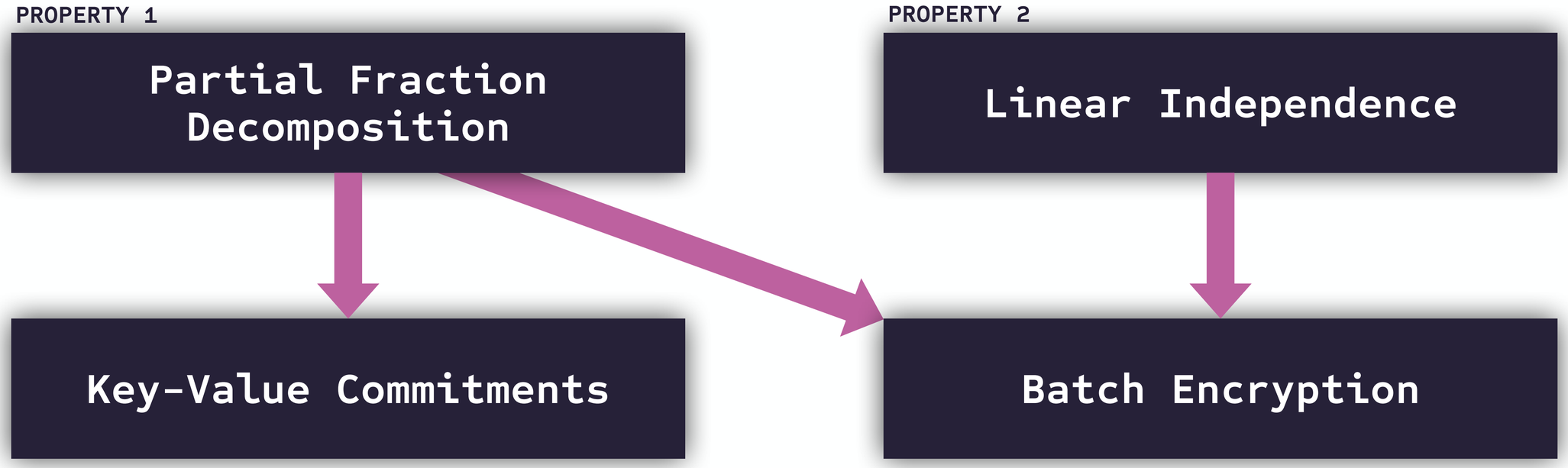
PROPERTY 2

Linear Independence

The Core Primitive: *Partial Fractions*

> **Main Idea**
> Encode cryptographic objects (e.g. keys, ciphertexts) using rational functions

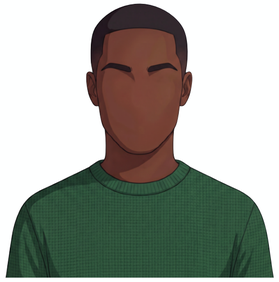
Why? Rational functions have useful algebraic properties

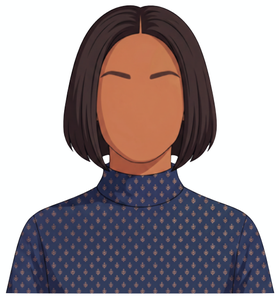


Key-Value Commitments

introduced by [AR20]

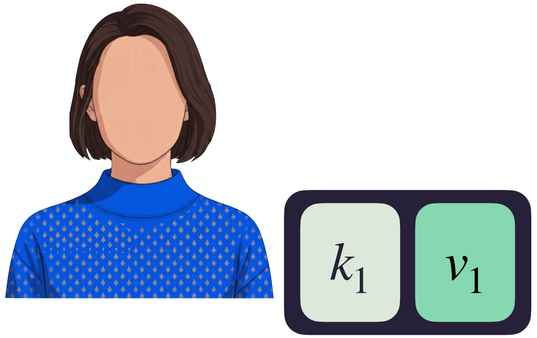


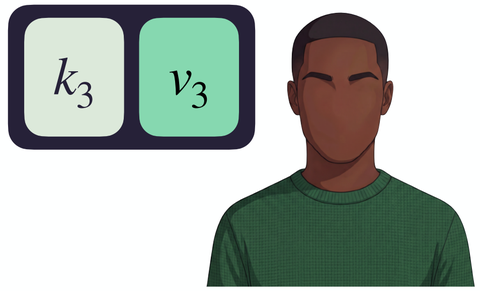


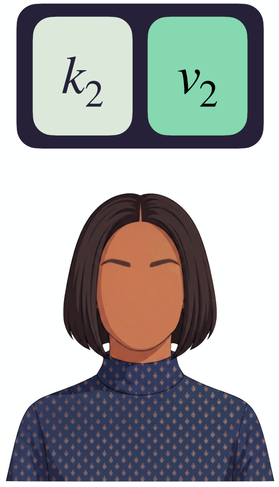


Key-Value Commitments

introduced by [AR20]

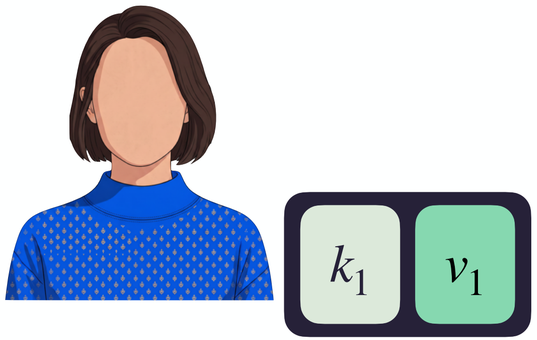






Key-Value Commitments

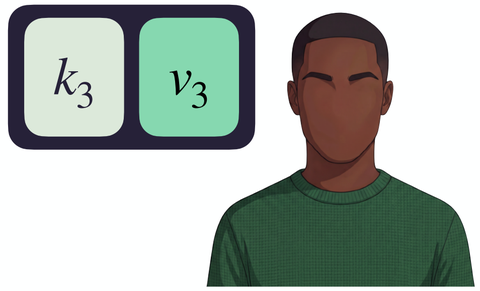
introduced by [AR20]

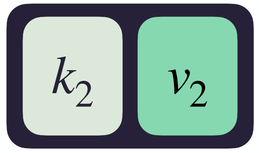


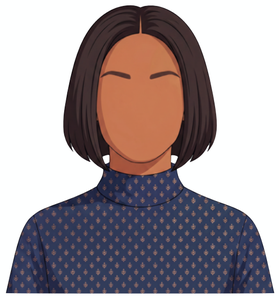
commitment of $D = \{\}$

com









Key-Value Commitments

introduced by [AR20]



insert key-value pair

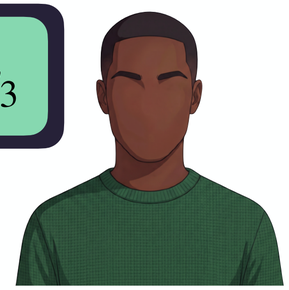
k_1 v_1

commitment of $D = \{\}$

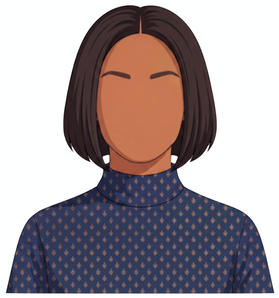
com



k_3 v_3



k_2 v_2

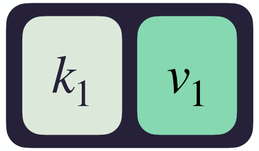


Key-Value Commitments

introduced by [AR20]



insert key-value pair

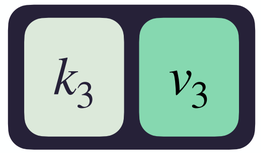


$\mathsf{Ins}\left(\mathsf{com}, k_1, v_1\right)$

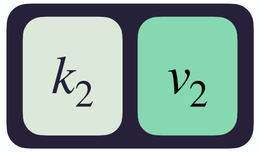
commitment of $D = \{\}$

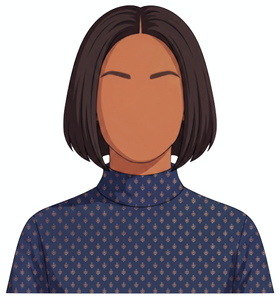
com









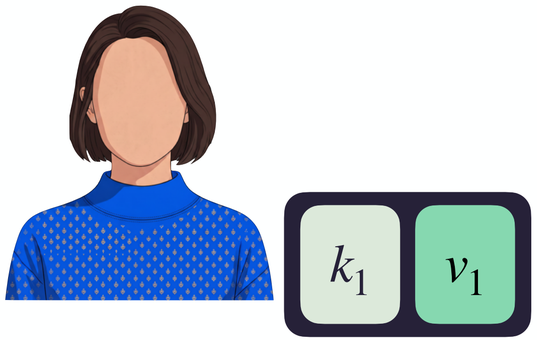


Key-Value Commitments

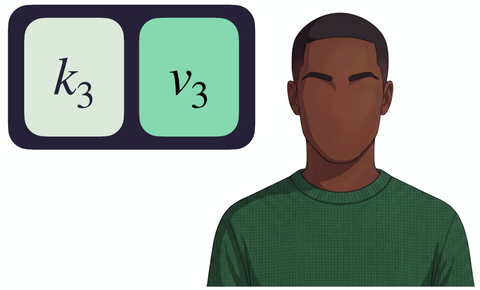
introduced by [AR20]

commitment of $D = \left\{ (k_1, v_1) \right\}$

com'



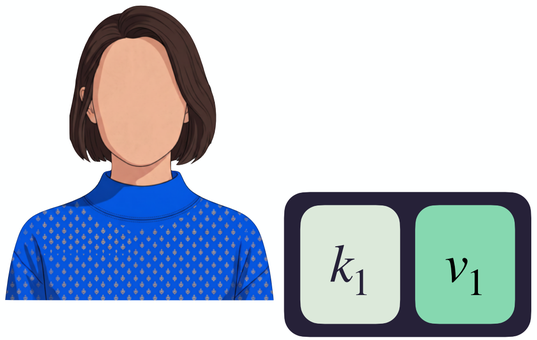




Key-Value Commitments

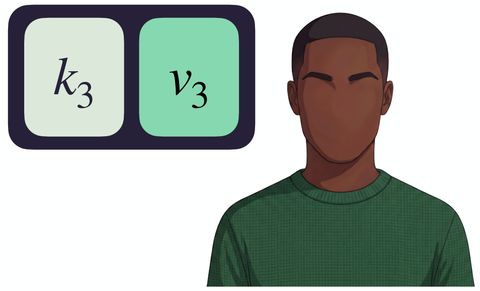
introduced by [AR20]

commitment of $D = \left\{ (k_1, v_1) \right\}$

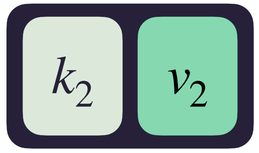


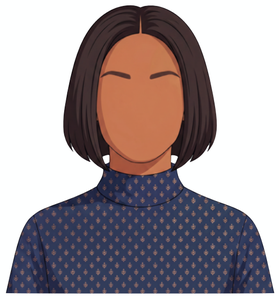
com′





$\mathsf{Ins}\left(\mathsf{com}', k_2, v_2\right)$



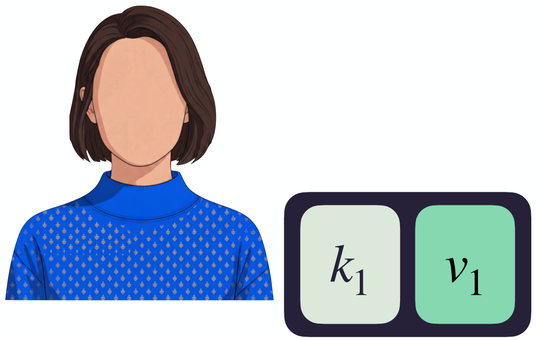


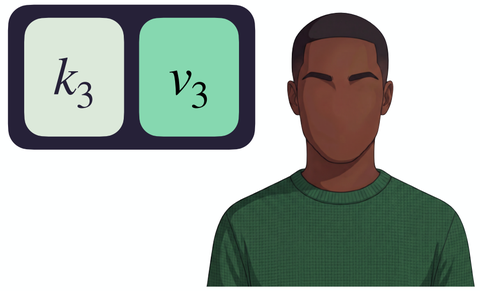
Key-Value Commitments

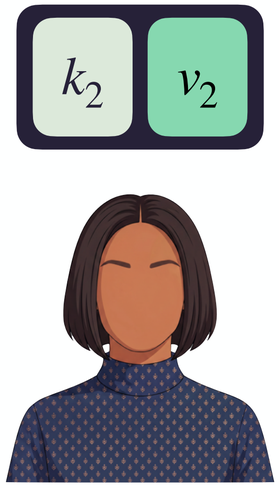
introduced by [AR20]

commitment of $D = \left\{ \left(k_i, v_i\right) \right\}_i$

com''



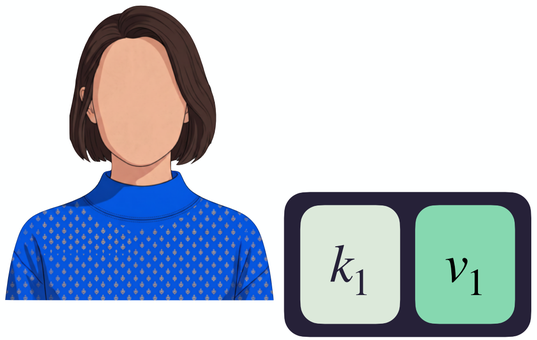




Key-Value Commitments

introduced by [AR20]

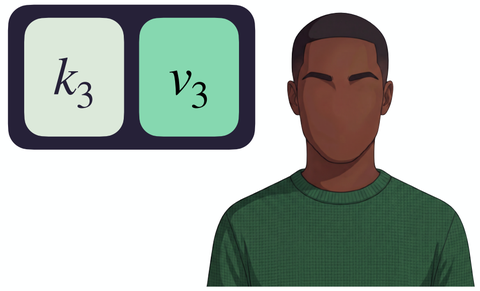
commitment of $D = \left\{ \left(k_i, v_i\right) \right\}_i$

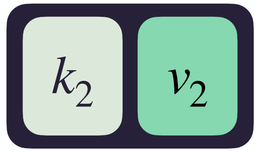


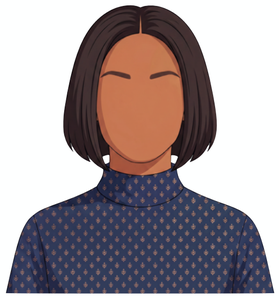
com''



$\mathsf{Ins}\left(\mathsf{com}'', k_3, v_3\right)$







Key-Value Commitments

introduced by [AR20]



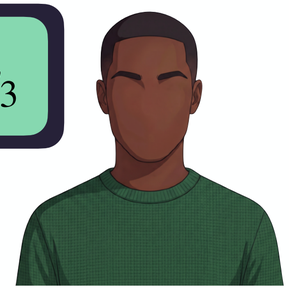
k_1 v_1

commitment of $D = \left\{ \left(k_i, v_i\right) \right\}_i$

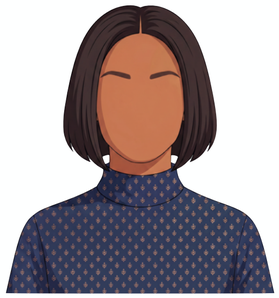
com'''



k_3 v_3



k_2 v_2



supports (non-)membership proofs

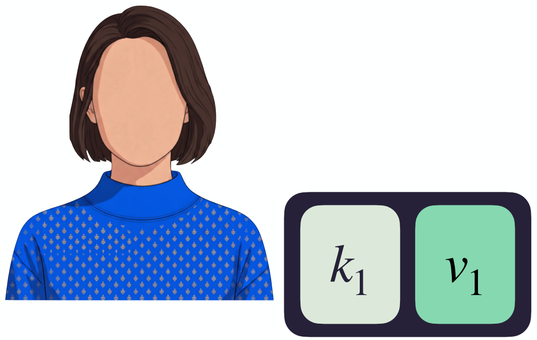
$\left(k_i, v_i\right) \stackrel{?}{\in} D$

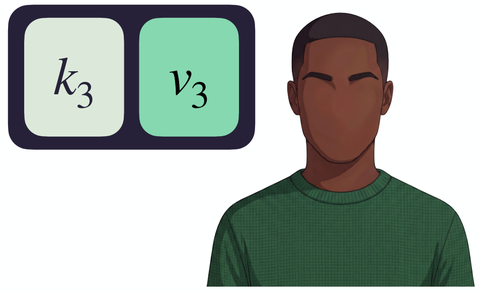
Key-Value Commitments

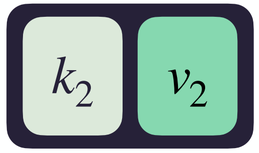
introduced by [AR20]

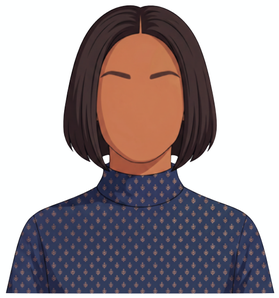
commitment of $D = \left\{ \left(k_i, v_i\right) \right\}_i$

com'''









supports (non-)membership proofs

$\left(k_i, v_i\right) \overset{?}{\in} D$

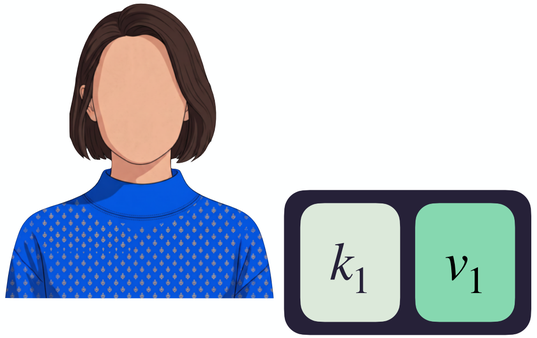
$\left(\mathsf{com}''', \pi\right) \leftarrow \mathsf{Ins}\left(\mathsf{com}'', k_3, v_3\right)$

s.t.

$\mathsf{VerifyMem}\left(\mathsf{com}''', k_3, v_3, \pi\right) \rightarrow \mathsf{accept}$

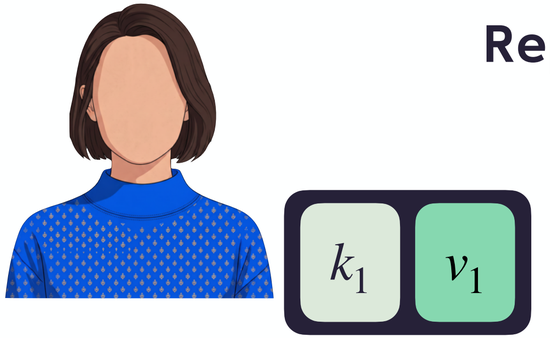
Encoding Key-Value (KV) Pairs

[Jutla-N-Roy26]



Encoding Key-Value (KV) Pairs

[Jutla-N-Roy26]

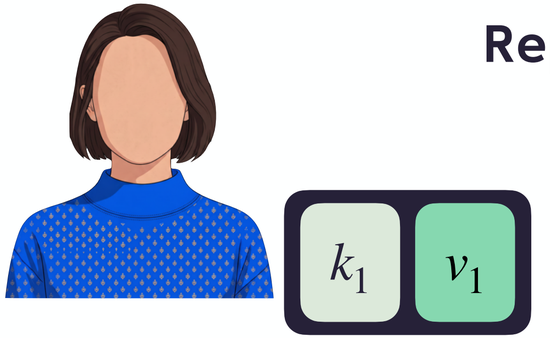


Represent keys using a “partial fraction”

$$k_i \to \frac{1}{X + k_i}$$

Encoding Key-Value (KV) Pairs

[Jutla-N-Roy26]



Represent keys using a “partial fraction”

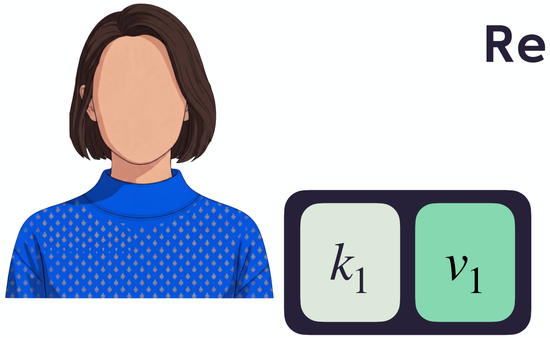
$$k_i \to \frac{1}{X + k_i}$$

cryptographically, this is a group element:

$$\left[\frac{1}{x + k_i}\right]_1 \in \mathbb{G}_1$$

Encoding Key-Value (KV) Pairs

[Jutla-N-Roy26]



Represent keys using a “partial fraction”

$$k_i \to \frac{1}{X + k_i}$$

KV pair is a scaled fraction

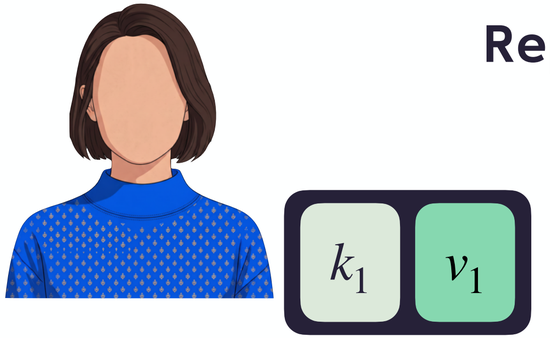
$$(k_i, v_i) \to \frac{v_i}{X + k_i}$$

cryptographically, this is a group element:

$$\left[\frac{1}{x + k_i}\right]_1 \in \mathbb{G}_1$$

Encoding Key-Value (KV) Pairs

[Jutla-N-Roy26]



Represent keys using a “partial fraction”

$$k_i \to \frac{1}{X + k_i}$$

KV pair is a scaled fraction

$$(k_i, v_i) \to \frac{v_i}{X + k_i}$$

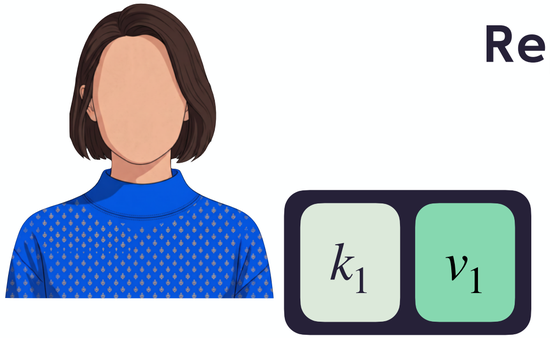
cryptographically, this is a group element:

$$\left[\frac{1}{x + k_i}\right]_1 \in \mathbb{G}_1$$

$$v_i \cdot \left[\frac{1}{x + k_i}\right]_1 \in \mathbb{G}_1$$

Encoding Key-Value (KV) Pairs

[Jutla-N-Roy26]



Represent keys using a "partial fraction"

$$k_i \to \frac{1}{X + k_i}$$

KV pair is a scaled fraction

$$(k_i, v_i) \to \frac{v_i}{X + k_i}$$

cryptographically, this is a group element:

$$\left[\frac{1}{x + k_i}\right]_1 \in \mathbb{G}_1$$

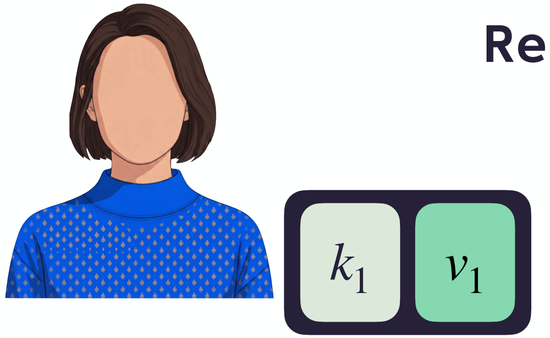
$$v_i \cdot \left[\frac{1}{x + k_i}\right]_1 \in \mathbb{G}_1$$

commitment of $D = \left\{(k_i, v_i)\right\}_i$ will be sum of fractions

$$\text{com} : C_D(X) = \sum_i \frac{v_i}{X + k_i}$$

Encoding Key-Value (KV) Pairs

[Jutla-N-Roy26]



Represent keys using a "partial fraction"

$$k_i \to \frac{1}{X + k_i}$$

cryptographically, this is a group element:

$$\left[\frac{1}{x + k_i}\right]_1 \in \mathbb{G}_1$$

KV pair is a scaled fraction

$$(k_i, v_i) \to \frac{v_i}{X + k_i}$$

$$v_i \cdot \left[\frac{1}{x + k_i}\right]_1 \in \mathbb{G}_1$$

commitment of $D = \left\{(k_i, v_i)\right\}_i$ will be sum of fractions

$$\text{com} : C_D(X) = \sum_i \frac{v_i}{X + k_i}$$

What do we do with this?

Partial Fraction Decomposition (PFD)

Partial Fraction Decomposition (PFD)

We can **rewrite product of rational functions as a sum** (with easily computable coefficients)

Partial Fraction Decomposition (PFD)

We can **rewrite product of rational functions as a sum** (with easily computable coefficients)

$$\text{e.g. } \frac{1}{X+3}\cdot\frac{1}{X+5}=\frac{1}{2}\left(\frac{1}{X+3}-\frac{1}{X+5}\right);$$

Partial Fraction Decomposition (PFD)

We can **rewrite product of rational functions as a sum** (with easily computable coefficients)

e.g. $$\frac{1}{X+3}\cdot\frac{1}{X+5}=\frac{1}{2}\left(\frac{1}{X+3}-\frac{1}{X+5}\right);$$

in general: $$\frac{1}{X+i}\cdot\frac{1}{X+j}=\frac{1}{j-i}\left(\frac{1}{X+i}-\frac{1}{X+j}\right)$$ (*)

Partial Fraction Decomposition (PFD)

We can **rewrite product of rational functions as a sum** (with easily computable coefficients)

e.g. $$\frac{1}{X+3}\cdot\frac{1}{X+5}=\frac{1}{2}\left(\frac{1}{X+3}-\frac{1}{X+5}\right);$$

in general: $$\frac{1}{X+i}\cdot\frac{1}{X+j}=\frac{1}{j-i}\left(\frac{1}{X+i}-\frac{1}{X+j}\right)$$ (*)

Partial Fraction Decomposition (PFD)

We can **rewrite product of rational functions as a sum** (with easily computable coefficients)

e.g. $$\frac{1}{X+3}\cdot\frac{1}{X+5}=\frac{1}{2}\left(\frac{1}{X+3}-\frac{1}{X+5}\right);$$

in general: $$\frac{1}{X+i}\cdot\frac{1}{X+j}=\frac{1}{j-i}\left(\frac{1}{X+i}-\frac{1}{X+j}\right)$$ (*)

Product with itself cannot be decomposed into linear fractions

Partial Fraction Decomposition (PFD)

We can **rewrite product of rational functions as a sum** (with easily computable coefficients)

$$\text{e.g. } \frac{1}{X+3}\cdot\frac{1}{X+5}=\frac{1}{2}\left(\frac{1}{X+3}-\frac{1}{X+5}\right);$$

$$\text{in general: } \frac{1}{X+i}\cdot\frac{1}{X+j}=\frac{1}{j-i}\left(\frac{1}{X+i}-\frac{1}{X+j}\right) \quad (*)$$

Product with itself cannot be decomposed into linear fractions

$$\text{e.g. } \frac{1}{X+3}\cdot\frac{1}{X+3}=\frac{1}{(X+3)^2}$$

PFD only simplifies *cross-terms*

PFD only simplifies *cross-terms*

Poles	$-i$	$-j$	$-k$
$-i$	$\frac{1}{X+i} \cdot \frac{1}{X+i}$	$\frac{1}{X+i} \cdot \frac{1}{X+j}$	$\frac{1}{X+i} \cdot \frac{1}{X+k}$
$-j$	$\frac{1}{X+j} \cdot \frac{1}{X+i}$	$\frac{1}{X+j} \cdot \frac{1}{X+j}$	$\frac{1}{X+j} \cdot \frac{1}{X+k}$
$-k$	$\frac{1}{X+k} \cdot \frac{1}{X+i}$	$\frac{1}{X+k} \cdot \frac{1}{X+j}$	$\frac{1}{X+k} \cdot \frac{1}{X+k}$

Matrix of pairwise products

PFD only simplifies *cross-terms*

Poles	$-i$	$-j$	$-k$
$-i$	$\frac{1}{(X+i)^2}$	$\frac{1}{X+i} \cdot \frac{1}{X+j}$	$\frac{1}{X+i} \cdot \frac{1}{X+k}$
$-j$	$\frac{1}{X+j} \cdot \frac{1}{X+i}$	$\frac{1}{(X+j)^2}$	$\frac{1}{X+j} \cdot \frac{1}{X+k}$
$-k$	$\frac{1}{X+k} \cdot \frac{1}{X+i}$	$\frac{1}{X+k} \cdot \frac{1}{X+j}$	$\frac{1}{(X+k)^2}$

PFD only simplifies *cross-terms*

Poles	$-i$	$-j$	$-k$
$-i$	$\frac{1}{(X+i)^2}$	$\frac{1}{X+i} \cdot \frac{1}{X+j}$	$\frac{1}{X+i} \cdot \frac{1}{X+k}$
$-j$	$\frac{1}{X+j} \cdot \frac{1}{X+i}$	$\frac{1}{(X+j)^2}$	$\frac{1}{X+j} \cdot \frac{1}{X+k}$
$-k$	$\frac{1}{X+k} \cdot \frac{1}{X+i}$	$\frac{1}{X+k} \cdot \frac{1}{X+j}$	$\frac{1}{(X+k)^2}$

Products on the *diagonal* do not decompose. Remain *quadratic*.

PFD only simplifies *cross-terms*

Poles	$-i$	$-j$	$-k$
$-i$	$\frac{1}{(X+i)^2}$	$\frac{1}{j-i}\left(\frac{1}{X+i}-\frac{1}{X+j}\right)$	$\frac{1}{k-i}\left(\frac{1}{X+i}-\frac{1}{X+k}\right)$
$-j$	$\frac{1}{j-i}\left(\frac{1}{X+i}-\frac{1}{X+j}\right)$	$\frac{1}{(X+j)^2}$	$\frac{1}{k-j}\left(\frac{1}{X+k}-\frac{1}{X+j}\right)$
$-k$	$\frac{1}{k-i}\left(\frac{1}{X+i}-\frac{1}{X+k}\right)$	$\frac{1}{k-j}\left(\frac{1}{X+k}-\frac{1}{X+j}\right)$	$\frac{1}{(X+k)^2}$

PFD only simplifies *cross-terms*

Poles	$-i$	$-j$	$-k$
$-i$	$\frac{1}{(X+i)^2}$	$\frac{1}{j-i}\left(\frac{1}{X+i}-\frac{1}{X+j}\right)$	$\frac{1}{k-i}\left(\frac{1}{X+i}-\frac{1}{X+k}\right)$
$-j$	$\frac{1}{j-i}\left(\frac{1}{X+i}-\frac{1}{X+j}\right)$	$\frac{1}{(X+j)^2}$	$\frac{1}{k-j}\left(\frac{1}{X+k}-\frac{1}{X+j}\right)$
$-k$	$\frac{1}{k-i}\left(\frac{1}{X+i}-\frac{1}{X+k}\right)$	$\frac{1}{k-j}\left(\frac{1}{X+k}-\frac{1}{X+j}\right)$	$\frac{1}{(X+k)^2}$

Off-diagonal/cross-terms *linearize*!

(Non-)Membership Testing Using PFD

[Jutla-N-Roy26]

(Non-)Membership Testing Using PFD

[Jutla-N-Roy26]

com **is a sum of fractions**

$$C_D(X) = \sum_{j \in [n]} \frac{v_j}{X + k_j}$$

(Non-)Membership Testing Using PFD

[Jutla-N-Roy26]

com **is a sum of fractions**

$$C_D(X) = \sum_{j \in [n]} \frac{v_j}{X + k_j}$$

$$(k_i, v) \notin D$$

(Non-)Membership Testing Using PFD

[Jutla-N-Roy26]

com **is a sum of fractions**

$$C_D(X) = \sum_{j \in [n]} \frac{v_j}{X + k_j}$$

$$\frac{(k_i, v) \notin D}{\left(C_D(X) - \frac{v}{X + k_i}\right)\left(\frac{1}{X + k_i}\right)}$$

has a **quadratic term**

$$\frac{v_i - v}{(X + k_i)^2},$$

(Non-)Membership Testing Using PFD

[Jutla-N-Roy26]

com **is a sum of fractions**

$$C_D(X) = \sum_{j \in [n]} \frac{v_j}{X + k_j}$$

$$\frac{(k_i, v) \notin D}{\left(C_D(X) - \frac{v}{X + k_i}\right)\left(\frac{1}{X + k_i}\right)}$$

has a **quadratic term**

$$\frac{v_i - v}{(X + k_i)^2},$$

and thus **cannot be decomposed into linear fractions**

(Non-)Membership Testing Using PFD

[Jutla-N-Roy26]

com **is a sum of fractions**

$$C_D(X) = \sum_{j \in [n]} \frac{v_j}{X + k_j}$$

$$\frac{(k_i, v) \notin D}{\left(C_D(X) - \frac{v}{X + k_i}\right)\left(\frac{1}{X + k_i}\right)}$$

has a **quadratic term**

$$\frac{v_i - v}{(X + k_i)^2},$$

and thus **cannot be decomposed into linear fractions**

$$\frac{(k_i, v_i) \in D}{}$$

(Non-)Membership Testing Using PFD

[Jutla-N-Roy26]

com **is a sum of fractions**

$$C_D(X) = \sum_{j \in [n]} \frac{v_j}{X + k_j}$$

$$\frac{(k_i, v) \notin D}{\left(C_D(X) - \frac{v}{X + k_i}\right)\left(\frac{1}{X + k_i}\right)}$$

has a **quadratic term**

$$\frac{v_i - v}{(X + k_i)^2},$$

and thus **cannot be decomposed into linear fractions**

$$\frac{(k_i, v_i) \in D}{\left(C_D(X) - \frac{v_i}{X + k_i}\right) \text{ has } \textbf{no } \frac{1}{X + k_i} \text{ term}}$$

(Non-)Membership Testing Using PFD

[Jutla-N-Roy26]

com **is a sum of fractions**

$$C_D(X) = \sum_{j \in [n]} \frac{v_j}{X + k_j}$$

$$\frac{(k_i, v) \notin D}{\left(C_D(X) - \frac{v}{X + k_i}\right)\left(\frac{1}{X + k_i}\right)}$$

has a **quadratic term**

$$\frac{v_i - v}{(X + k_i)^2},$$

and thus **cannot be decomposed into linear fractions**

$$\frac{(k_i, v_i) \in D}{\left(C_D(X) - \frac{v_i}{X + k_i}\right) \text{ has } \textbf{no } \frac{1}{X + k_i} \text{ term}}$$

and so, $\left(C_D(X) - \frac{v_i}{X + k_i}\right)\left(\frac{1}{X + k_i}\right)$ can be

decomposed into linear fractions

(Non-)Membership Testing Using PFD

[Jutla-N-Roy26]

com **is a sum of fractions**

$$C_D(X) = \sum_{j \in [n]} \frac{v_j}{X + k_j}$$

$$\frac{(k_i, v) \notin D}{\left(C_D(X) - \frac{v}{X + k_i}\right)\left(\frac{1}{X + k_i}\right)}$$

has a **quadratic term**

$$\frac{v_i - v}{(X + k_i)^2},$$

and thus **cannot be decomposed into linear fractions**

$$\frac{(k_i, v_i) \in D}{\left(C_D(X) - \frac{v_i}{X + k_i}\right) \text{ has } \textbf{no } \frac{1}{X + k_i} \text{ term}}$$

and so, $\left(C_D(X) - \frac{v_i}{X + k_i}\right)\left(\frac{1}{X + k_i}\right)$ can be

decomposed into linear fractions

$$\left(C_D(X) - \frac{v_i}{X + k_i}\right)\left(\frac{1}{X + k_i}\right) = \sum_{j \neq i} \frac{v_j}{k_j - k_i}\left(\frac{1}{X + k_i} - \frac{1}{X + k_j}\right)$$

(Non-)Membership Testing Using PFD

[Jutla-N-Roy26]

com **is a sum of fractions**

$$C_D(X) = \sum_{j \in [n]} \frac{v_j}{X + k_j}$$

(*) $\frac{1}{X+i} \cdot \frac{1}{X+j} = \frac{1}{j-i}\left(\frac{1}{X+i} - \frac{1}{X+j}\right)$

$$\frac{(k_i, v) \notin D}{\left(C_D(X) - \frac{v}{X+k_i}\right)\left(\frac{1}{X+k_i}\right)}$$

has a **quadratic term**

$$\frac{v_i - v}{(X+k_i)^2},$$

and thus **cannot be decomposed into linear fractions**

$$\frac{(k_i, v_i) \in D}{\left(C_D(X) - \frac{v_i}{X+k_i}\right) \text{ has } \textbf{no } \frac{1}{X+k_i} \text{ term}}$$

and so, $\left(C_D(X) - \frac{v_i}{X+k_i}\right)\left(\frac{1}{X+k_i}\right)$ can be

decomposed into linear fractions

$$\left(C_D(X) - \frac{v_i}{X+k_i}\right)\left(\frac{1}{X+k_i}\right) = \sum_{j \neq i} \frac{v_j}{k_j - k_i}\left(\frac{1}{X+k_i} - \frac{1}{X+k_j}\right)$$

applying (*) to each product

Toward A Commitment Scheme

[Jutla-N-Roy26]

com **is a sum of fractions**

$$C_D(X) = \sum_{j \in [n]} \frac{v_j}{X + k_j}$$

$$\left(C_D(X) - \frac{v_i}{X + k_i} \right) \left(\frac{1}{X + k_i} \right) = \sum_{j \neq i} \frac{v_j}{k_j - k_i} \left(\frac{1}{X + k_i} - \frac{1}{X + k_j} \right)$$

Toward A Commitment Scheme

[Jutla-N-Roy26]

com **is a sum of fractions**

$$C_D(X) = \sum_{j \in [n]} \frac{v_j}{X + k_j}$$

$$\left(C_D(X) - \frac{v_i}{X + k_i} \right) \left(\frac{1}{X + k_i} \right) = \sum_{j \neq i} \frac{v_j}{k_j - k_i} \left(\frac{1}{X + k_i} - \frac{1}{X + k_j} \right)$$

commitment

Toward A Commitment Scheme

[Jutla-N-Roy26]

com **is a sum of fractions**

$$C_D(X) = \sum_{j \in [n]} \frac{v_j}{X + k_j}$$

$$\left(C_D(X) - \frac{v_i}{X + k_i} \right) \left(\frac{1}{X + k_i} \right) = \sum_{j \neq i} \frac{v_j}{k_j - k_i} \left(\frac{1}{X + k_i} - \frac{1}{X + k_j} \right)$$

commitment

public

Toward A Commitment Scheme

[Jutla-N-Roy26]

com **is a sum of fractions**

$$C_D(X) = \sum_{j \in [n]} \frac{v_j}{X + k_j}$$

$$\left(C_D(X) - \frac{v_i}{X + k_i} \right) \left(\frac{1}{X + k_i} \right) = \sum_{j \neq i} \frac{v_j}{k_j - k_i} \left(\frac{1}{X + k_i} - \frac{1}{X + k_j} \right)$$

commitment

public

membership proof

Toward A Commitment Scheme

[Jutla-N-Roy26]

com **is a sum of fractions**

$$C_D(X) = \sum_{j \in [n]} \frac{v_j}{X + k_j}$$

$$\left(C_D(X) - \frac{v_i}{X + k_i} \right) \left(\frac{1}{X + k_i} \right) = \sum_{j \neq i} \frac{v_j}{k_j - k_i} \left(\frac{1}{X + k_i} - \frac{1}{X + k_j} \right)$$

commitment

public

membership proof

compute product using group pairing

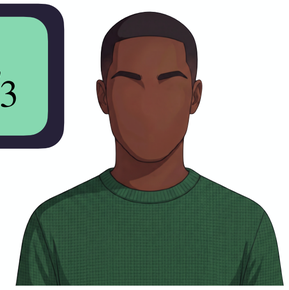
An Example

An Example

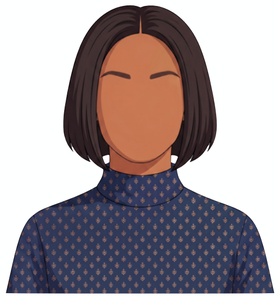


k_1 v_1

k_3 v_3



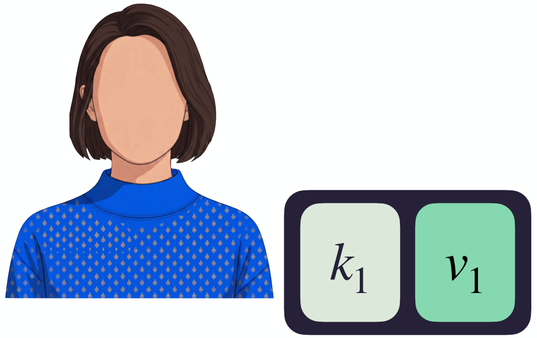
k_2 v_2

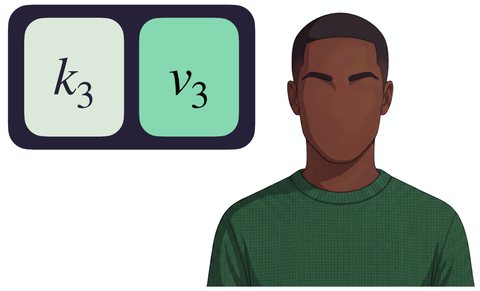


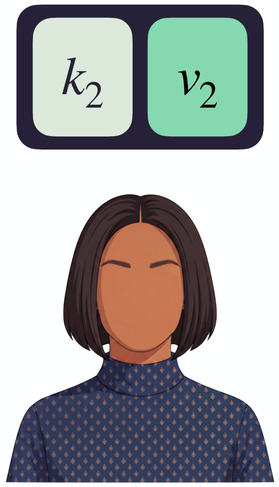
An Example

$$\texttt{commitment of } D = \left\{ \left(k_i, v_i\right) \right\}_i$$

$$\text{com} = C_D(X) = \frac{v_1}{X + k_1} + \frac{v_2}{X + k_2} + \frac{v_3}{X + k_3}$$



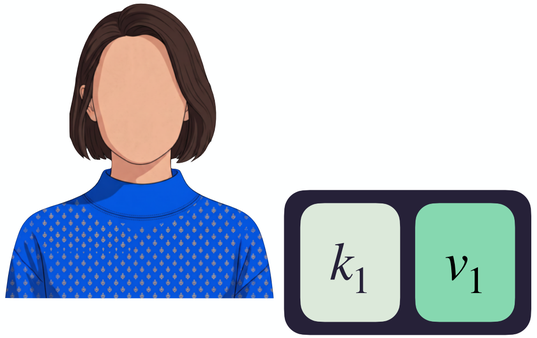




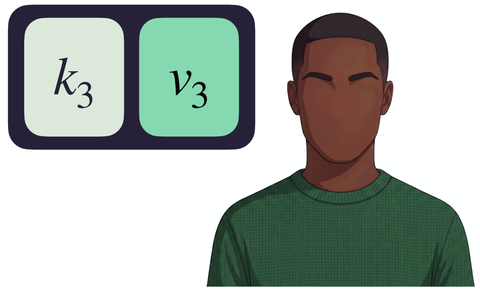
An Example

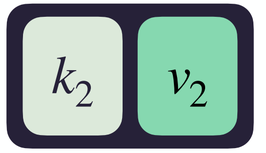
commitment of $D = \left\{ (k_i, v_i) \right\}_i$

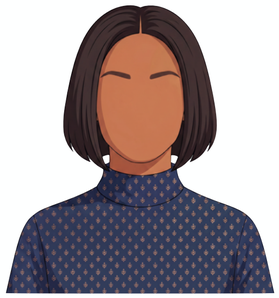
$$\text{com} = C_D(X) = \frac{v_1}{X + k_1} + \frac{v_2}{X + k_2} + \frac{v_3}{X + k_3}$$



wants to test membership of her KV pair



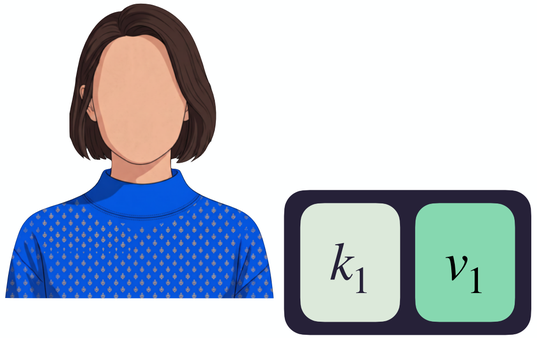




An Example

commitment of $D = \left\{ \left(k_i, v_i\right) \right\}_i$

$$\text{com} = C_D(X) = \frac{v_1}{X + k_1} + \frac{v_2}{X + k_2} + \frac{v_3}{X + k_3}$$



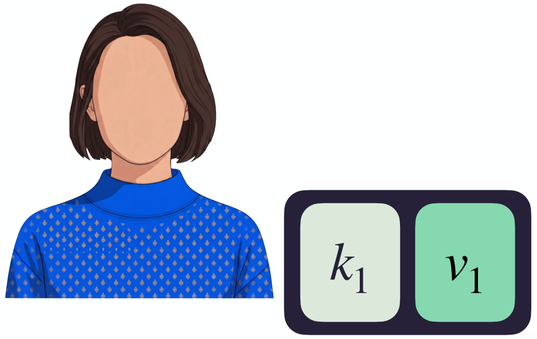
wants to test membership of her KV pair

$$\left(C_D(X) - \frac{v_1}{X + k_1} \right) \left(\frac{1}{X + k_1} \right)$$

An Example

commitment of $D = \left\{ (k_i, v_i) \right\}_i$

$$\text{com} = C_D(X) = \frac{v_1}{X + k_1} + \frac{v_2}{X + k_2} + \frac{v_3}{X + k_3}$$



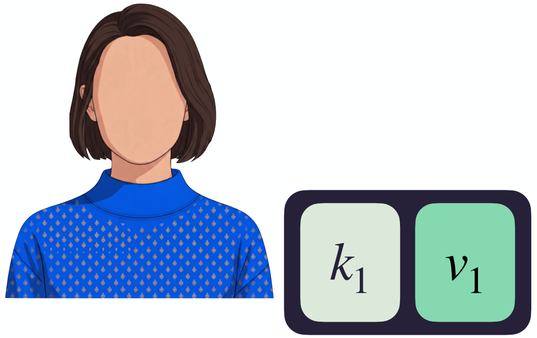
wants to test membership of her KV pair

$$\left(C_D(X) - \frac{v_1}{X + k_1} \right) \left(\frac{1}{X + k_1} \right) = \left(\frac{v_2}{X + k_2} + \frac{v_3}{X + k_3} \right) \left(\frac{1}{X + k_1} \right)$$

An Example

commitment of $D = \left\{ (k_i, v_i) \right\}_i$

$$\text{com} = C_D(X) = \frac{v_1}{X + k_1} + \frac{v_2}{X + k_2} + \frac{v_3}{X + k_3}$$



wants to test membership of her KV pair

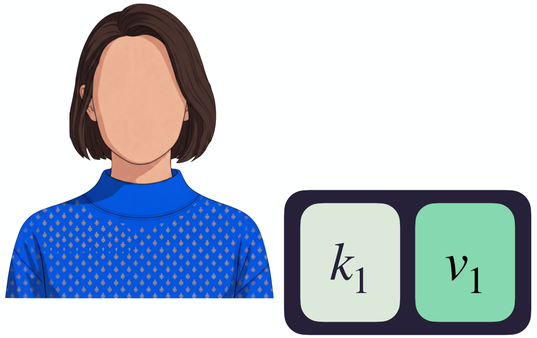
$$\left(C_D(X) - \frac{v_1}{X + k_1} \right) \left(\frac{1}{X + k_1} \right) = \left(\frac{v_2}{X + k_2} + \frac{v_3}{X + k_3} \right) \left(\frac{1}{X + k_1} \right)$$

$$= \left(\frac{v_2}{X + k_2} \right) \left(\frac{1}{X + k_1} \right)$$

An Example

commitment of $D = \left\{ (k_i, v_i) \right\}_i$

$$\text{com} = C_D(X) = \frac{v_1}{X + k_1} + \frac{v_2}{X + k_2} + \frac{v_3}{X + k_3}$$



wants to test membership of her KV pair

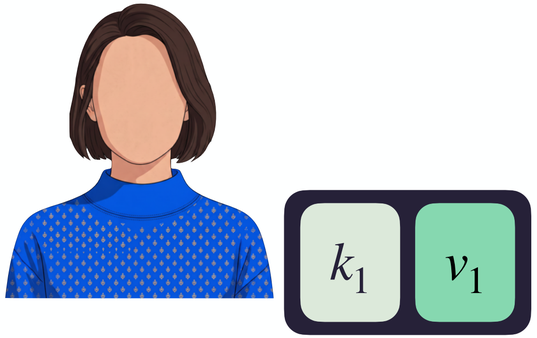
$$\left(C_D(X) - \frac{v_1}{X + k_1} \right) \left(\frac{1}{X + k_1} \right) = \left(\frac{v_2}{X + k_2} + \frac{v_3}{X + k_3} \right) \left(\frac{1}{X + k_1} \right)$$

$$= \left(\frac{v_2}{X + k_2} \right) \left(\frac{1}{X + k_1} \right) + \left(\frac{v_3}{X + k_3} \right) \left(\frac{1}{X + k_1} \right)$$

An Example

commitment of $D = \left\{ (k_i, v_i) \right\}_i$

$$\text{com} = C_D(X) = \frac{v_1}{X + k_1} + \frac{v_2}{X + k_2} + \frac{v_3}{X + k_3}$$



wants to test membership of her KV pair

$$\left(C_D(X) - \frac{v_1}{X + k_1} \right) \left(\frac{1}{X + k_1} \right) = \left(\frac{v_2}{X + k_2} + \frac{v_3}{X + k_3} \right) \left(\frac{1}{X + k_1} \right)$$

$$= \left(\frac{v_2}{X + k_2} \right) \left(\frac{1}{X + k_1} \right) + \left(\frac{v_3}{X + k_3} \right) \left(\frac{1}{X + k_1} \right)$$

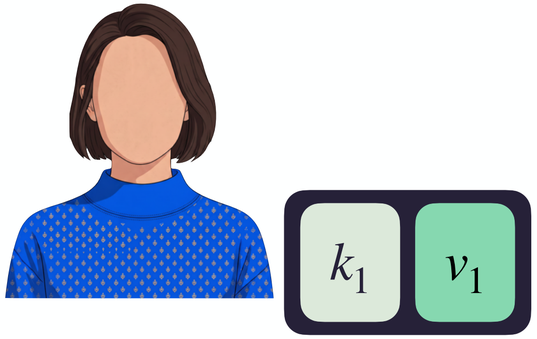
(*) $\frac{1}{X+i} \cdot \frac{1}{X+j} = \frac{1}{j-i} \left(\frac{1}{X+i} - \frac{1}{X+j} \right)$

$$= \left(\frac{v_2}{k_1 - k_2} \right) \left(\frac{1}{X + k_2} - \frac{1}{X + k_1} \right)$$

An Example

commitment of $D = \left\{ (k_i, v_i) \right\}_i$

$$\text{com} = C_D(X) = \frac{v_1}{X+k_1} + \frac{v_2}{X+k_2} + \frac{v_3}{X+k_3}$$



wants to test membership of her KV pair

(*) $\frac{1}{X+i} \cdot \frac{1}{X+j} = \frac{1}{j-i}\left(\frac{1}{X+i} - \frac{1}{X+j}\right)$

$$\left(C_D(X) - \frac{v_1}{X+k_1}\right)\left(\frac{1}{X+k_1}\right) = \left(\frac{v_2}{X+k_2} + \frac{v_3}{X+k_3}\right)\left(\frac{1}{X+k_1}\right)$$

$$= \left(\frac{v_2}{X+k_2}\right)\left(\frac{1}{X+k_1}\right) + \left(\frac{v_3}{X+k_3}\right)\left(\frac{1}{X+k_1}\right)$$

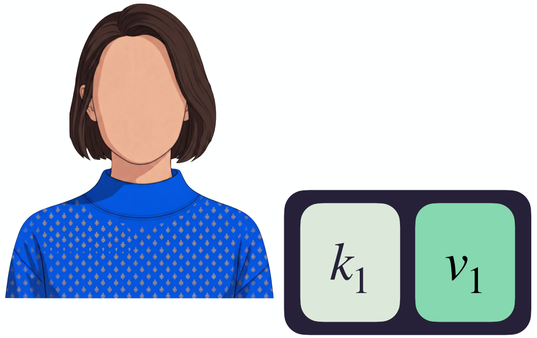
(*)

$$= \left(\frac{v_2}{k_1-k_2}\right)\left(\frac{1}{X+k_2} - \frac{1}{X+k_1}\right) + \left(\frac{v_3}{k_1-k_3}\right)\left(\frac{1}{X+k_3} - \frac{1}{X+k_1}\right)$$

An Example

commitment of $D = \left\{ (k_i, v_i) \right\}_i$

$$\text{com} = C_D(X) = \frac{v_1}{X+k_1} + \frac{v_2}{X+k_2} + \frac{v_3}{X+k_3}$$



wants to test membership of her KV pair

(*) $\frac{1}{X+i} \cdot \frac{1}{X+j} = \frac{1}{j-i}\left(\frac{1}{X+i} - \frac{1}{X+j}\right)$

$$\left(C_D(X) - \frac{v_1}{X+k_1}\right)\left(\frac{1}{X+k_1}\right) = \left(\frac{v_2}{X+k_2} + \frac{v_3}{X+k_3}\right)\left(\frac{1}{X+k_1}\right)$$

$$= \left(\frac{v_2}{X+k_2}\right)\left(\frac{1}{X+k_1}\right) + \left(\frac{v_3}{X+k_3}\right)\left(\frac{1}{X+k_1}\right)$$

(*)

$$= \left(\frac{v_2}{k_1-k_2}\right)\left(\frac{1}{X+k_2} - \frac{1}{X+k_1}\right) + \left(\frac{v_3}{k_1-k_3}\right)\left(\frac{1}{X+k_3} - \frac{1}{X+k_1}\right)$$

membership proof

ParFrac KVC Construction Sketch

[Jutla-N-Roy26]

ParFrac KVC Construction Sketch

[Jutla-N-Roy26]

Recall (asymmetric) bilinear group: $\mathcal{G} = \left(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e\right)$

additive notation: $[x]_1 = x \cdot g_1, [y]_2 = y \cdot g_2,$ and $[z]_T = z \cdot e(g_1, g_2)$

ParFrac KVC Construction Sketch

[Jutla-N-Roy26]

Recall (asymmetric) bilinear group: $\mathcal{G} = \left(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e\right)$

additive notation: $[x]_1 = x \cdot g_1, [y]_2 = y \cdot g_2,$ and $[z]_T = z \cdot e(g_1, g_2)$

Setup $\left(\text{key space } \mathcal{K} = \{k_i\}_{i \in [n]}\right)$

Sample $x \xleftarrow{\$} \mathbb{Z}_p,$ and

output linear fractions

in $\mathbb{G}_1$: $\left[\frac{1}{x+k_1}\right]_1, \ldots, \left[\frac{1}{x+k_n}\right]_1,$ and in $\mathbb{G}_2$: $\left[\frac{1}{x+k_1}\right]_2, \ldots, \left[\frac{1}{x+k_n}\right]_2,$

and

initial commitment: $\mathsf{com} = [1]_1$

ParFrac KVC Construction Sketch

[Jutla-N-Roy26]

commitment of $D = \{\}$

com



Recall (asymmetric) bilinear group: $\mathcal{G} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e)$

additive notation: $[x]_1 = x \cdot g_1, [y]_2 = y \cdot g_2,$ and $[z]_T = z \cdot e(g_1, g_2)$

Setup $\left(\text{key space } \mathcal{K} = \{k_i\}_{i \in [n]}\right)$

Sample $x \xleftarrow{\$} \mathbb{Z}_p,$ and

output linear fractions

in $\mathbb{G}_1$: $\left[\frac{1}{x+k_1}\right]_1, \ldots, \left[\frac{1}{x+k_n}\right]_1,$ and in $\mathbb{G}_2$: $\left[\frac{1}{x+k_1}\right]_2, \ldots, \left[\frac{1}{x+k_n}\right]_2,$

and

initial commitment: $\text{com} = [1]_1$

ParFrac KVC Construction Sketch

[Jutla-N-Roy26]

commitment of $D = \{\}$

com



Recall (asymmetric) bilinear group: $\mathcal{G} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e)$

additive notation: $[x]_1 = x \cdot g_1, [y]_2 = y \cdot g_2,$ and $[z]_T = z \cdot e(g_1, g_2)$

Setup $\left(\text{key space } \mathcal{K} = \{k_i\}_{i \in [n]}\right)$

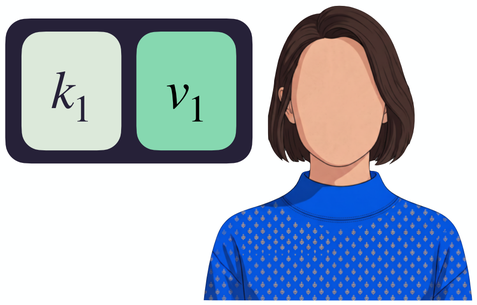
Sample $x \xleftarrow{\$} \mathbb{Z}_p,$ and

output linear fractions

in $\mathbb{G}_1$: $\left[\frac{1}{x+k_1}\right]_1, \ldots, \left[\frac{1}{x+k_n}\right]_1,$ and in $\mathbb{G}_2$: $\left[\frac{1}{x+k_1}\right]_2, \ldots, \left[\frac{1}{x+k_n}\right]_2,$

and

initial commitment: $\text{com} = [1]_1$



ParFrac KVC Construction Sketch

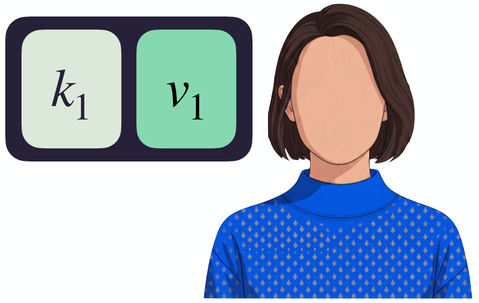
[Jutla-N-Roy26]

commitment of $D = \{\}$

com



$\mathsf{Ins}\left(\mathsf{com}, k_1, v_1\right)$



Recall (asymmetric) bilinear group: $\mathcal{G} = \left(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e\right)$

additive notation: $[x]_1 = x \cdot g_1, [y]_2 = y \cdot g_2,$ and $[z]_T = z \cdot e(g_1, g_2)$

$\mathsf{Setup}\left(\text{key space } \mathcal{K} = \{k_i\}_{i \in [n]}\right)$

Sample $x \xleftarrow{\$} \mathbb{Z}_p,$ and

output linear fractions

$$\text{in } \mathbb{G}_1\text{: } \left[\frac{1}{x+k_1}\right]_1, \ldots, \left[\frac{1}{x+k_n}\right]_1, \text{ and in } \mathbb{G}_2\text{: } \left[\frac{1}{x+k_1}\right]_2, \ldots, \left[\frac{1}{x+k_n}\right]_2,$$

and

initial commitment: $\mathsf{com} = [1]_1$

Insertion is homomorphic

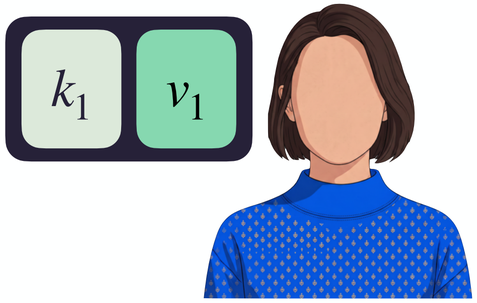
[Jutla-N-Roy26]

commitment of $D = \{\}$

com



$\mathrm{Ins}\left(\mathrm{com}, k_1, v_1\right)$



Recall (asymmetric) bilinear group: $\mathcal{G} = \left(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e\right)$

additive notation: $[x]_1 = x \cdot g_1, [y]_2 = y \cdot g_2,$ and $[z]_T = z \cdot e(g_1, g_2)$

fractions: $\left[\frac{1}{x+k_1}\right]_1, \ldots, \left[\frac{1}{x+k_n}\right]_1, \left[\frac{1}{x+k_1}\right]_2, \ldots, \left[\frac{1}{x+k_n}\right]_2,$

initial commitment: $\mathrm{com} = [1]_1$

Insertion is homomorphic

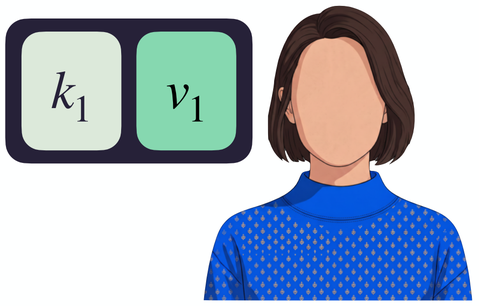
[Jutla-N-Roy26]

commitment of $D = \{\}$

com



$\text{Ins}\left(\text{com}, k_1, v_1\right)$



Recall (asymmetric) bilinear group: $\mathcal{G} = \left(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e\right)$

additive notation: $[x]_1 = x \cdot g_1, [y]_2 = y \cdot g_2,$ and $[z]_T = z \cdot e(g_1, g_2)$

fractions: $\left[\frac{1}{x+k_1}\right]_1, \ldots, \left[\frac{1}{x+k_n}\right]_1, \left[\frac{1}{x+k_1}\right]_2, \ldots, \left[\frac{1}{x+k_n}\right]_2,$

initial commitment: $\text{com} = [1]_1$

Compute **new commitment:**

$$\text{com}' = \text{com} + v_1 \cdot \left[\frac{1}{x+k_1}\right]_1 = \left[\frac{v_1}{x+k_1}\right]_1$$

Insertion is homomorphic

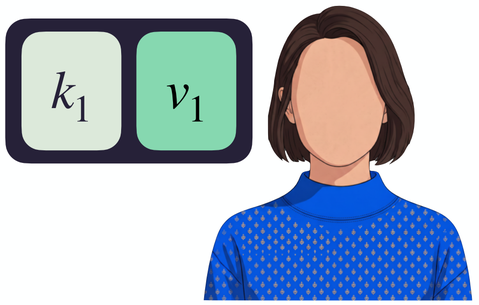
[Jutla-N-Roy26]

commitment of $D = \{\}$

com



$\mathrm{Ins}\left(\mathrm{com}, k_1, v_1\right)$



Recall (asymmetric) bilinear group: $\mathcal{G} = \left(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e\right)$

additive notation: $[x]_1 = x \cdot g_1, [y]_2 = y \cdot g_2,$ and $[z]_T = z \cdot e(g_1, g_2)$

fractions: $\left[\frac{1}{x+k_1}\right]_1, \ldots, \left[\frac{1}{x+k_n}\right]_1, \left[\frac{1}{x+k_1}\right]_2, \ldots, \left[\frac{1}{x+k_n}\right]_2,$

initial commitment: $\mathrm{com} = [1]_1$

Compute **new commitment:**

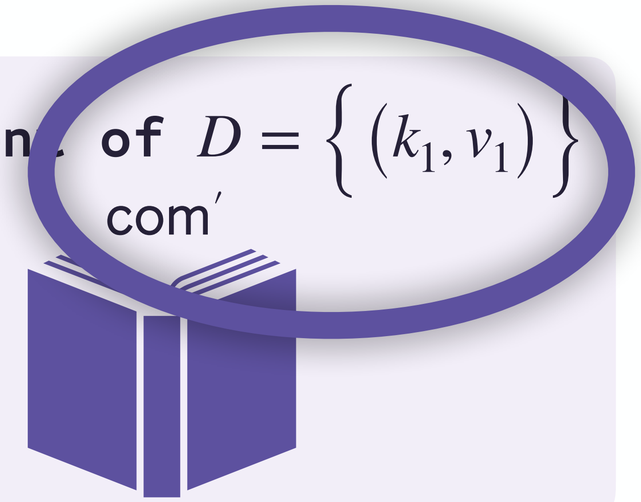
$$\mathrm{com}' = \mathrm{com} + v_1 \cdot \left[\frac{1}{x+k_1}\right]_1 = \left[\frac{v_1}{x+k_1}\right]_1$$

Insertion is homomorphic

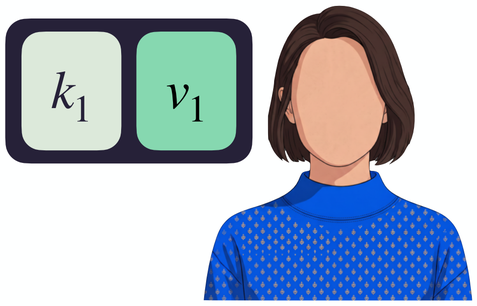
[Jutla-N-Roy26]

commitment of $D = \{(k_1, v_1)\}$

com'



$\mathsf{Ins}(\mathsf{com}, k_1, v_1)$



Recall (asymmetric) bilinear group: $\mathcal{G} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e)$

additive notation: $[x]_1 = x \cdot g_1, [y]_2 = y \cdot g_2,$ and $[z]_T = z \cdot e(g_1, g_2)$

fractions: $\left[\frac{1}{x+k_1}\right]_1, \ldots, \left[\frac{1}{x+k_n}\right]_1, \left[\frac{1}{x+k_1}\right]_2, \ldots, \left[\frac{1}{x+k_n}\right]_2,$

initial commitment: $\mathsf{com} = [1]_1$

Compute **new commitment:**

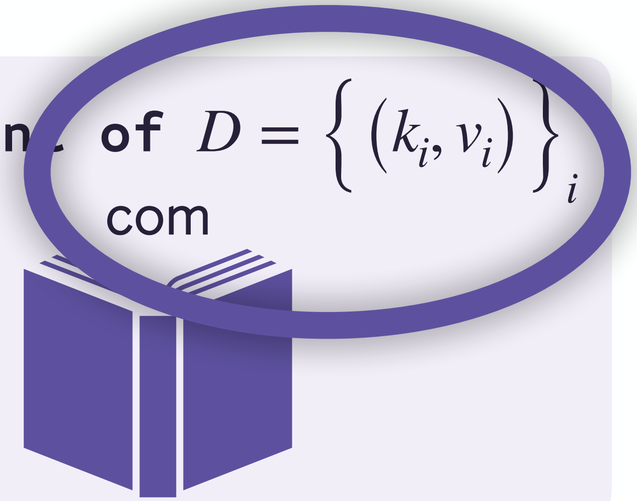
$$\mathsf{com}' = \mathsf{com} + v_1 \cdot \left[\frac{1}{x+k_1}\right]_1 = \left[\frac{v_1}{x+k_1}\right]_1$$

Insertion is homomorphic

[Jutla-N-Roy26]

commitment of $D = \left\{ (k_i, v_i) \right\}_i$

com

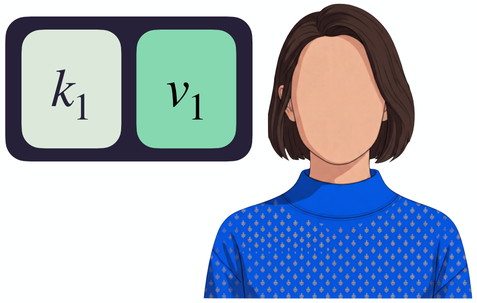


Recall (asymmetric) bilinear group: $\mathcal{G} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e)$

additive notation: $[x]_1 = x \cdot g_1, [y]_2 = y \cdot g_2,$ and $[z]_T = z \cdot e(g_1, g_2)$

fractions: $\left[\frac{1}{x+k_1}\right]_1, \ldots, \left[\frac{1}{x+k_n}\right]_1, \left[\frac{1}{x+k_1}\right]_2, \ldots, \left[\frac{1}{x+k_n}\right]_2,$

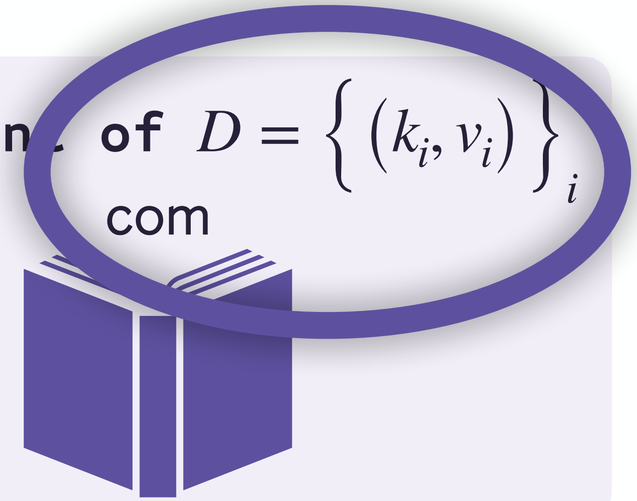
after many inserts...



Insertion is homomorphic

[Jutla-N-Roy26]

commitment of $D = \left\{ (k_i, v_i) \right\}_i$

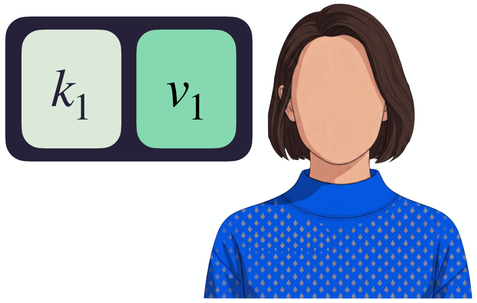


Recall (asymmetric) bilinear group: $\mathcal{G} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e)$

additive notation: $[x]_1 = x \cdot g_1, [y]_2 = y \cdot g_2,$ and $[z]_T = z \cdot e(g_1, g_2)$

fractions: $\left[\frac{1}{x+k_1}\right]_1, \ldots, \left[\frac{1}{x+k_n}\right]_1, \left[\frac{1}{x+k_1}\right]_2, \ldots, \left[\frac{1}{x+k_n}\right]_2,$

after many inserts...



Commitment becomes

$$\text{com} = \left[\frac{v_1}{x+k_1}\right]_1 + \left[\frac{v_2}{x+k_2}\right]_1 + \cdots = \left[\sum_i \frac{v_i}{x+k_i}\right]_1$$

Membership proof from PFD

[Jutla-N-Roy26]

commitment of $D = \left\{ (k_i, v_i) \right\}_i$

com

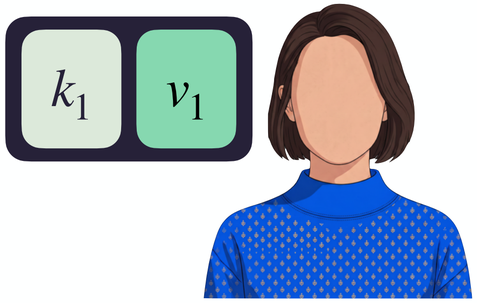


Recall (asymmetric) bilinear group: $\mathcal{G} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e)$

additive notation: $[x]_1 = x \cdot g_1, [y]_2 = y \cdot g_2,$ and $[z]_T = z \cdot e(g_1, g_2)$

fractions: $\left[\frac{1}{x+k_1}\right]_1, \ldots, \left[\frac{1}{x+k_n}\right]_1, \left[\frac{1}{x+k_1}\right]_2, \ldots, \left[\frac{1}{x+k_n}\right]_2,$

$$\text{com} = \left[\sum_i \frac{v_i}{x+k_i}\right]_1$$



Membership proof from PFD

[Jutla-N-Roy26]

commitment of $D = \left\{ (k_i, v_i) \right\}_i$

com



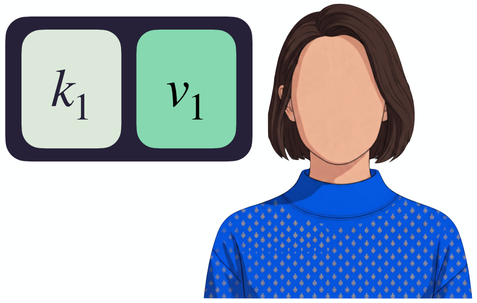
Recall (asymmetric) bilinear group: $\mathcal{G} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e)$

additive notation: $[x]_1 = x \cdot g_1, [y]_2 = y \cdot g_2,$ and $[z]_T = z \cdot e(g_1, g_2)$

fractions: $\left[\frac{1}{x+k_1}\right]_1, \ldots, \left[\frac{1}{x+k_n}\right]_1, \left[\frac{1}{x+k_1}\right]_2, \ldots, \left[\frac{1}{x+k_n}\right]_2,$

$$\text{com} = \left[\sum_i \frac{v_i}{x+k_i}\right]_1$$

wants to prove membership of



Membership proof from PFD

[Jutla-N-Roy26]

commitment of $D = \left\{ (k_i, v_i) \right\}_i$

com



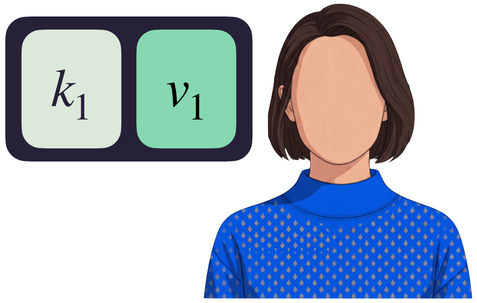
Recall (asymmetric) bilinear group: $\mathcal{G} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e)$

additive notation: $[x]_1 = x \cdot g_1, [y]_2 = y \cdot g_2,$ and $[z]_T = z \cdot e(g_1, g_2)$

fractions: $\left[\frac{1}{x+k_1}\right]_1, \ldots, \left[\frac{1}{x+k_n}\right]_1, \left[\frac{1}{x+k_1}\right]_2, \ldots, \left[\frac{1}{x+k_n}\right]_2,$

$$\text{com} = \left[\sum_i \frac{v_i}{x+k_i}\right]_1$$

wants to prove membership of



Generates membership proof:

$$\pi_{\text{mem}} = \sum_{\substack{(k,v)\in D \\ k \neq k_1}} \frac{v}{k_1 - k} \left(\left[\frac{1}{x+k}\right]_1 - \left[\frac{1}{x+k_1}\right]_1 \right)$$

Membership proof from PFD

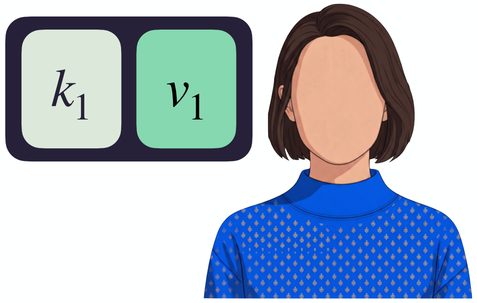
[Jutla-N-Roy26]

commitment of $D = \left\{ (k_i, v_i) \right\}_i$

com



wants to prove membership of



Recall (asymmetric) bilinear group: $\mathcal{G} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e)$

additive notation: $[x]_1 = x \cdot g_1, [y]_2 = y \cdot g_2,$ and $[z]_T = z \cdot e(g_1, g_2)$

fractions: $\left[\frac{1}{x+k_1}\right]_1, \ldots, \left[\frac{1}{x+k_n}\right]_1, \left[\frac{1}{x+k_1}\right]_2, \ldots, \left[\frac{1}{x+k_n}\right]_2,$

$$\text{com} = \left[\sum_i \frac{v_i}{x+k_i}\right]_1$$

Generates membership proof:

$$\pi_{\text{mem}} = \sum_{\substack{(k,v)\in D \\ k \neq k_1}} \frac{v}{k_1 - k} \left(\left[\frac{1}{x+k}\right]_1 - \left[\frac{1}{x+k_1}\right]_1 \right)$$

the proof is the partial fraction decomposition with respect to k_1 fraction

Verification checks PFD

[Jutla-N-Roy26]

commitment of $D = \left\{ (k_i, v_i) \right\}_i$

com

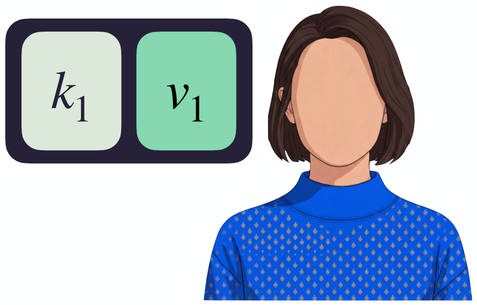


Recall (asymmetric) bilinear group: $\mathcal{G} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e)$

additive notation: $[x]_1 = x \cdot g_1, [y]_2 = y \cdot g_2,$ and $[z]_T = z \cdot e(g_1, g_2)$

fractions: $\left[\frac{1}{x+k_1}\right]_1, \ldots, \left[\frac{1}{x+k_n}\right]_1, \left[\frac{1}{x+k_1}\right]_2, \ldots, \left[\frac{1}{x+k_n}\right]_2,$

$$\text{com} = \left[\sum_i \frac{v_i}{x+k_i}\right]_1 \quad \pi_{\text{mem}} = \sum_{\substack{(k,v)\in D \\ k \neq k_1}} \frac{v}{k_1 - k}\left(\left[\frac{1}{x+k}\right]_1 - \left[\frac{1}{x+k_1}\right]_1\right)$$



Verification checks PFD

[Jutla-N-Roy26]

commitment of $D = \left\{ (k_i, v_i) \right\}_i$

com



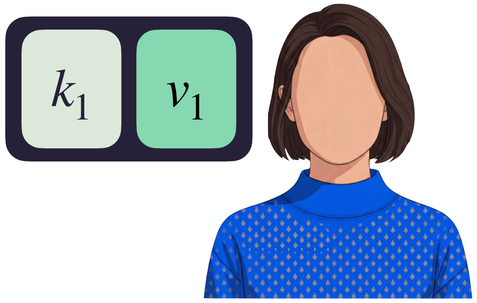
Recall (asymmetric) bilinear group: $\mathcal{G} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e)$

additive notation: $[x]_1 = x \cdot g_1, [y]_2 = y \cdot g_2,$ and $[z]_T = z \cdot e(g_1, g_2)$

fractions: $\left[\frac{1}{x+k_1}\right]_1, \ldots, \left[\frac{1}{x+k_n}\right]_1, \left[\frac{1}{x+k_1}\right]_2, \ldots, \left[\frac{1}{x+k_n}\right]_2,$

$$\text{com} = \left[\sum_i \frac{v_i}{x+k_i}\right]_1 \quad \pi_{\text{mem}} = \sum_{\substack{(k,v)\in D \\ k \neq k_1}} \frac{v}{k_1 - k}\left(\left[\frac{1}{x+k}\right]_1 - \left[\frac{1}{x+k_1}\right]_1\right)$$

Verify by checking decomposition



Verification checks PFD

[Jutla-N-Roy26]

commitment of $D = \left\{ (k_i, v_i) \right\}_i$

com



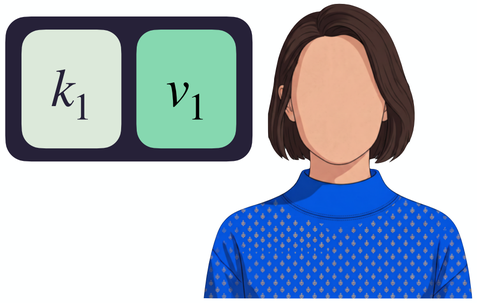
Recall (asymmetric) bilinear group: $\mathcal{G} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e)$

additive notation: $[x]_1 = x \cdot g_1, [y]_2 = y \cdot g_2,$ and $[z]_T = z \cdot e(g_1, g_2)$

fractions: $\left[\frac{1}{x+k_1}\right]_1, \ldots, \left[\frac{1}{x+k_n}\right]_1, \left[\frac{1}{x+k_1}\right]_2, \ldots, \left[\frac{1}{x+k_n}\right]_2,$

$$\text{com} = \left[\sum_i \frac{v_i}{x+k_i}\right]_1 \quad \pi_{\text{mem}} = \sum_{\substack{(k,v)\in D \\ k \neq k_1}} \frac{v}{k_1 - k}\left(\left[\frac{1}{x+k}\right]_1 - \left[\frac{1}{x+k_1}\right]_1\right)$$

Verify by checking decomposition



Check pairing equation:

$$e\left(\text{com} - v_1 \cdot \left[\frac{1}{x+k_1}\right]_1, \left[\frac{1}{x+k_1}\right]_2\right) \stackrel{?}{=} e(\pi_{\text{mem}}, [1]_2)$$

Verification checks PFD

[Jutla-N-Roy26]

commitment of $D = \left\{ (k_i, v_i) \right\}_i$

com



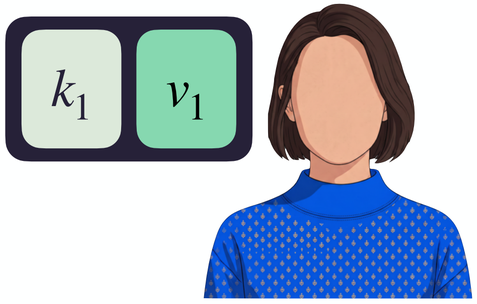
Recall (asymmetric) bilinear group: $\mathcal{G} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e)$

additive notation: $[x]_1 = x \cdot g_1, [y]_2 = y \cdot g_2,$ and $[z]_T = z \cdot e(g_1, g_2)$

fractions: $\left[\frac{1}{x+k_1}\right]_1, \ldots, \left[\frac{1}{x+k_n}\right]_1, \left[\frac{1}{x+k_1}\right]_2, \ldots, \left[\frac{1}{x+k_n}\right]_2,$

$$\text{com} = \left[\sum_i \frac{v_i}{x+k_i}\right]_1 \quad \pi_{\text{mem}} = \sum_{\substack{(k,v)\in D \\ k \neq k_1}} \frac{v}{k_1 - k}\left(\left[\frac{1}{x+k}\right]_1 - \left[\frac{1}{x+k_1}\right]_1\right)$$

Verify by checking decomposition



Check pairing equation:

$$e\left(\text{com} - v_1 \cdot \left[\frac{1}{x+k_1}\right]_1, \left[\frac{1}{x+k_1}\right]_2\right) \stackrel{?}{=} e(\pi_{\text{mem}}, [1]_2)$$

if $(k_1, v_1) \in D$**, then decomposition is valid;**

else no decomposition exists using $\left[\frac{1}{x+k_i}\right]_1$**'s**

(except with negl. prob.)

Verification checks PFD

[Jutla-N-Roy26]

commitment of $D = \left\{ (k_i, v_i) \right\}_i$

com



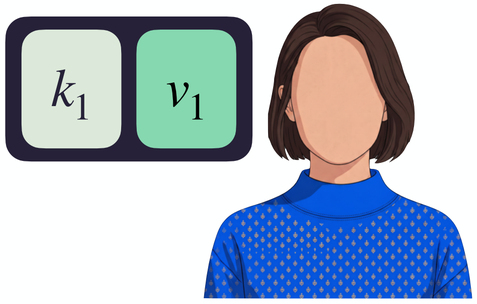
Recall (asymmetric) bilinear group: $\mathcal{G} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e)$

additive notation: $[x]_1 = x \cdot g_1, [y]_2 = y \cdot g_2,$ and $[z]_T = z \cdot e(g_1, g_2)$

fractions: $\left[\frac{1}{x+k_1}\right]_1, \ldots, \left[\frac{1}{x+k_n}\right]_1, \left[\frac{1}{x+k_1}\right]_2, \ldots, \left[\frac{1}{x+k_n}\right]_2,$

$$\text{com} = \left[\sum_i \frac{v_i}{x+k_i}\right]_1 \qquad \pi_{\text{mem}} = \sum_{\substack{(k,v)\in D \\ k \neq k_1}} \frac{v}{k_1 - k} \left(\left[\frac{1}{x+k}\right]_1 - \left[\frac{1}{x+k_1}\right]_1 \right)$$

Verify by checking decomposition



Check pairing equation:

$$e\left(\text{com} - v_1 \cdot \left[\frac{1}{x+k_1}\right]_1, \left[\frac{1}{x+k_1}\right]_2\right) \stackrel{?}{=} e(\pi_{\text{mem}}, [1]_2)$$

if $(k_1, v_1) \in D$**, then decomposition is valid;**

else no decomposition exists using $\left[\frac{1}{x+k_i}\right]_1$**'s**

(except with negl. prob.)

ParFrac KVC Construction Sketch

[Jutla-N-Roy26]

Recall (asymmetric) bilinear group: $\mathcal{G} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e)$

additive notation: $[x]_1 = x \cdot g_1, [y]_2 = y \cdot g_2,$ and $[z]_T = z \cdot e(g_1, g_2)$

Setup $\left(\text{key space } \mathcal{K} = \{k_i\}_i\right)$

Sample $x \xleftarrow{\$} \mathbb{Z}_p$, and

output linear fractions

in $\mathbb{G}_1$: $\left[\frac{1}{x+k_1}\right]_1, \ldots, \left[\frac{1}{x+k_n}\right]_1,$

in $\mathbb{G}_2$: $\left[\frac{1}{x+k_1}\right]_2, \ldots, \left[\frac{1}{x+k_n}\right]_2,$

and

initial commitment:

$\text{com} = [1]_1$

Ins $\left(\text{com}, k^* \in \mathcal{K}, v^*\right)$

Compute **new commitment:**

$$\text{com}' = \text{com} + v \cdot \left[\frac{1}{x+k}\right]_1$$

and **proof of membership**

$$\pi_{\text{mem}} = \sum_{\substack{(k,v)\in D \\ k \neq k^*}} \frac{v}{k^* - k} \left(\left[\frac{1}{x+k}\right]_1 - \left[\frac{1}{x+k^*}\right]_1 \right)$$

the proof is the partial fraction decomposition

VerifyMem $\left(\text{com}, k^*, v^*, \pi\right)$

Check pairing equation:

$$e\left(\text{com} - v^* \cdot \left[\frac{1}{x+k^*}\right]_1, \left[\frac{1}{x+k^*}\right]_2\right) \stackrel{?}{=} e(\pi, [1]_2)$$

if $(k^*, v^*) \in D$, then decomposition is valid;

else no decomposition exists using $\left[\frac{1}{x+k}\right]_1$'s

(except with negl. prob.)

Credential-based KVCs

[Jutla-**N**-Roy26]

$$\pi_{\text{mem}} = \sum_{\substack{(k,v)\in D \\ k \neq k^*}} \frac{v}{k^* - k} \left(\left[\frac{1}{x+k} \right]_1 - \left[\frac{1}{x+k^*} \right]_1 \right)$$

Credential-based KVCs

[Jutla-**N**-Roy26]

$$\pi_{\text{mem}} = \sum_{\substack{(k,v)\in D \\ k \neq k^*}} \frac{v}{k^* - k} \left(\left[\frac{1}{x+k} \right]_1 - \left[\frac{1}{x+k^*} \right]_1 \right)$$

computing membership proof requires knowing D :(

Credential-based KVCs

[Jutla-**N**-Roy26]

$$\pi_{\text{mem}} = \sum_{\substack{(k,v)\in D \\ k \neq k^*}} \frac{v}{k^* - k} \left(\left[\frac{1}{x+k} \right]_1 - \left[\frac{1}{x+k^*} \right]_1 \right)$$

computing membership proof requires knowing D :(

But ideally, we'd like a more *efficient* scheme

Credential-based KVCs

[Jutla-**N**-Roy26]

$$\pi_{\text{mem}} = \sum_{\substack{(k,v)\in D \\ k \neq k^*}} \frac{v}{k^* - k} \left(\left[\frac{1}{x+k} \right]_1 - \left[\frac{1}{x+k^*} \right]_1 \right)$$

computing membership proof requires knowing D :(

But ideally, we'd like a more *efficient* scheme

We introduce a party known as the *Registrar* who registers parties' keys and returns a *credential:*

Credential-based KVCs

[Jutla-**N**-Roy26]

$$\pi_{\text{mem}} = \sum_{\substack{(k,v)\in D \\ k \neq k^*}} \frac{v}{k^* - k} \left(\left[\frac{1}{x+k} \right]_1 - \left[\frac{1}{x+k^*} \right]_1 \right)$$

computing membership proof requires knowing D :(

But ideally, we'd like a more *efficient* scheme

We introduce a party known as the *Registrar* who registers parties' keys and returns a *credential:*

(1) the initial proof **(2) fraction tied to registered key**

Credential-based KVCs

[Jutla-**N**-Roy26]

$$\pi_{\text{mem}} = \sum_{\substack{(k,v)\in D \\ k \neq k^*}} \frac{v}{k^* - k} \left(\left[\frac{1}{x+k} \right]_1 - \left[\frac{1}{x+k^*} \right]_1 \right)$$

computing membership proof requires knowing D :(

But ideally, we'd like a more *efficient* scheme

We introduce a party known as the *Registrar* who registers parties' keys and returns a *credential:*

(1) the initial proof **(2) fraction tied to registered key**

Proof can be updated in a *stateless* homomorphic manner (very efficient)

Credential-based KVCs

[Jutla-**N**-Roy26]

$$\pi_{\text{mem}} = \sum_{\substack{(k,v)\in D \\ k \neq k^*}} \frac{v}{k^* - k} \left(\left[\frac{1}{x+k} \right]_1 - \left[\frac{1}{x+k^*} \right]_1 \right)$$

computing membership proof requires knowing D :(

But ideally, we'd like a more *efficient* scheme

We introduce a party known as the *Registrar* who registers parties' keys and returns a *credential:*

(1) the initial proof **(2) fraction tied to registered key**

Proof can be updated in a *stateless* homomorphic manner (very efficient)

We call this a *Credential-based* KVC

Credential-based KVCs

[Jutla-N-Roy26]

$$\pi_{\text{mem}} = \sum_{\substack{(k,v)\in D \\ k \neq k^*}} \frac{v}{k^* - k} \left(\left[\frac{1}{x+k} \right]_1 - \left[\frac{1}{x+k^*} \right]_1 \right)$$

computing membership proof requires knowing D :(

But ideally, we'd like a more *efficient* scheme

We introduce a party known as the *Registrar* who registers parties' keys and returns a *credential:*

(1) the initial proof **(2) fraction tied to registered key**

Proof can be updated in a *stateless* homomorphic manner (very efficient)

We call this a *Credential-based* KVC

see paper for details

A Quick Note on Security

A Quick Note on Security

a new q-type assumption

A Quick Note on Security

a new q-type assumption

informally, captures the hardness of obtaining

a quadratic fraction $\frac{1}{(x+i)^2}$ in the exponent

given only linear fractions $\frac{1}{x+j}$'s

A Quick Note on Security

a new q-type assumption

informally, captures the hardness of obtaining

a quadratic fraction $\frac{1}{(x+i)^2}$ in the exponent

given only linear fractions $\frac{1}{x+j}$'s



Assumption holds in Generic (Bilinear) Group Model

Comparison With Prior Work

Comparison With Prior Work

with RSA-based “KVaC” [AR20]

✓ Smaller groups

Prime-order instead of composite-order groups. **Much smaller in practice ⟹ less communication**

✓ Computationally efficient

No expensive calculation of Bezout coefficients

Comparison With Prior Work

with RSA-based “KVaC” [AR20]

✓ Smaller groups

Prime-order instead of composite-order groups. **Much smaller in practice ⟹ less communication**

✓ Computationally efficient

No expensive calculation of Bezout coefficients

with Cuckoo Commitments [FKdP23]

✓ Purely algebraic

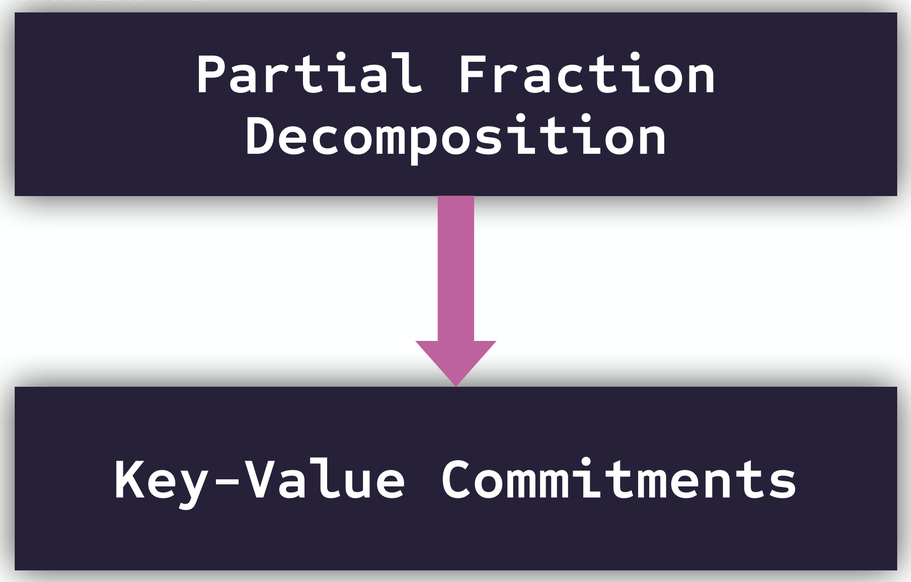
Our construction is purely algebraic **avoiding non-algebraic operations such as cuckoo hashing**

✓ Stateless updates

Supports stateless updates while Cuckoo commitments does not

Summary of ParFrac KVCs

PROPERTY 1



Another useful property

PROPERTY 1

Partial Fraction Decomposition

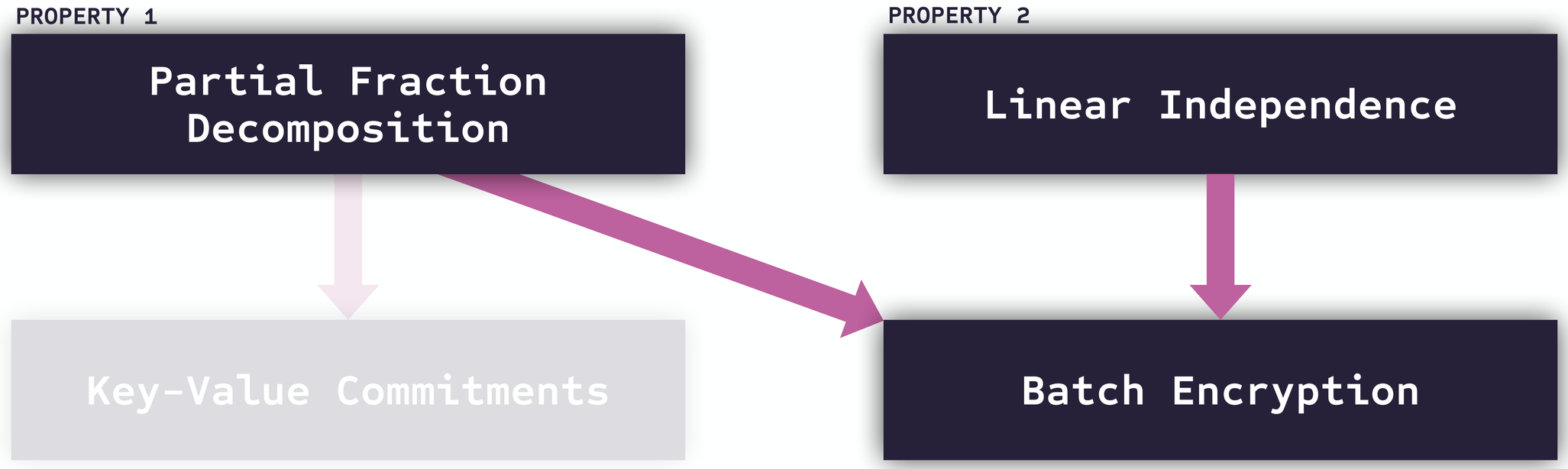
Key-Value Commitments

PROPERTY 2

Linear Independence

Another useful property

[Boneh-N-Roy-Tas26]

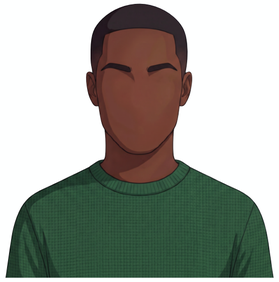


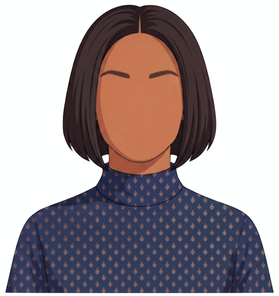
Batch Encryption

introduced by [CGPP24]









Batch Encryption

introduced by [CGPP24]



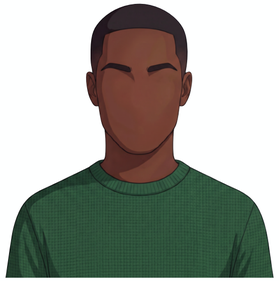


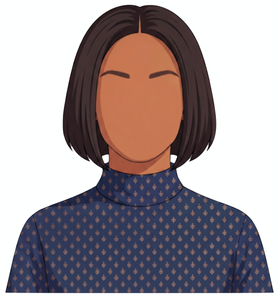
pk





sk





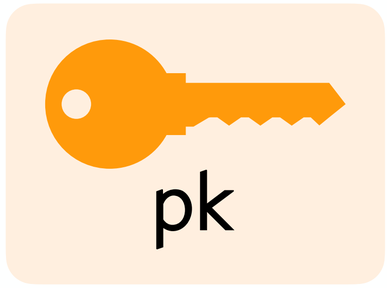
Batch Encryption

introduced by [CGPP24]



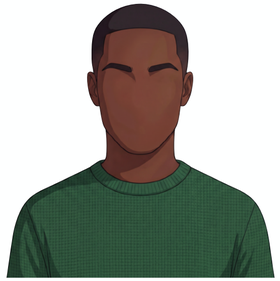
ct_1





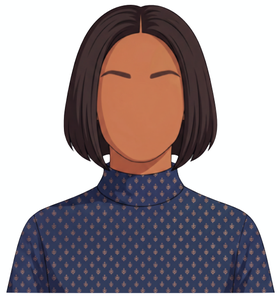






ct_2





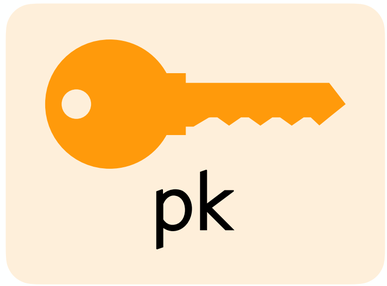
ct_3



Batch Encryption

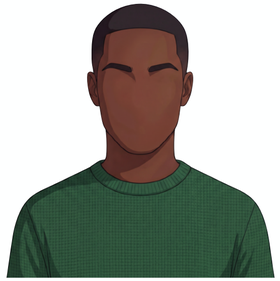
introduced by [CGPP24]



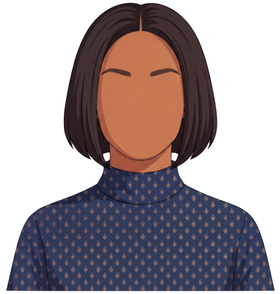








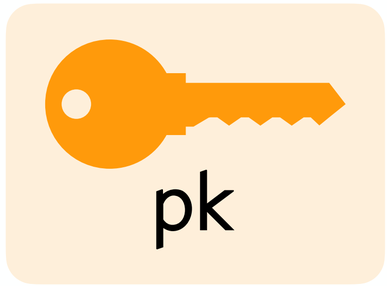




Batch Encryption

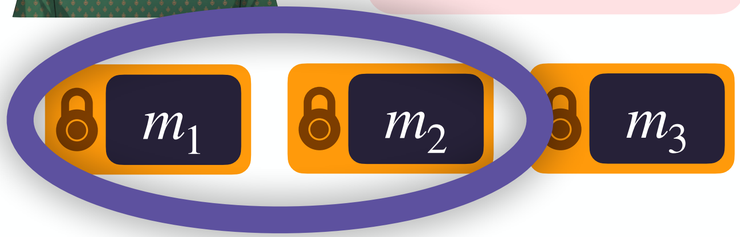
introduced by [CGPP24]

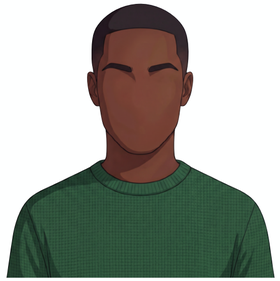


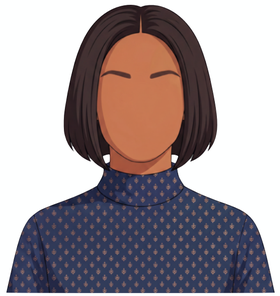








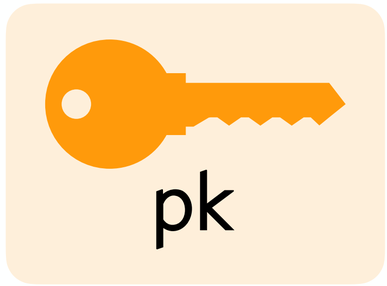


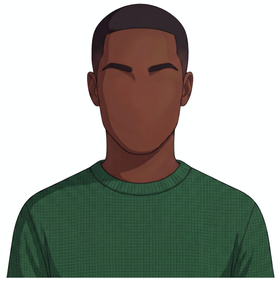


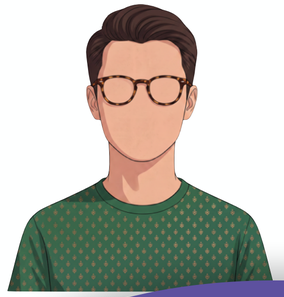
Batch Encryption

introduced by [CGPP24]

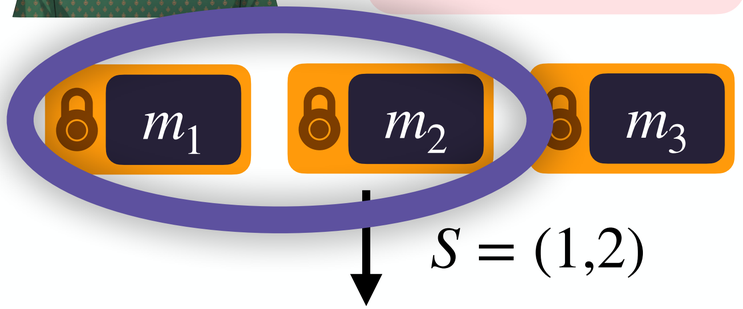


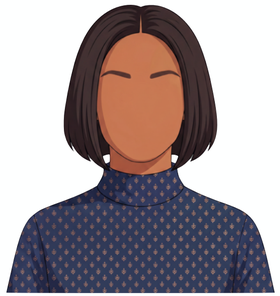








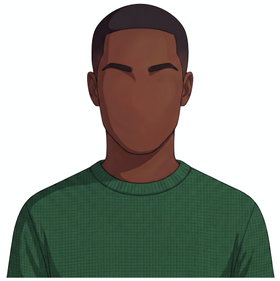


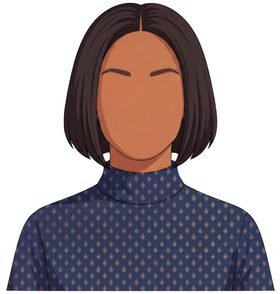


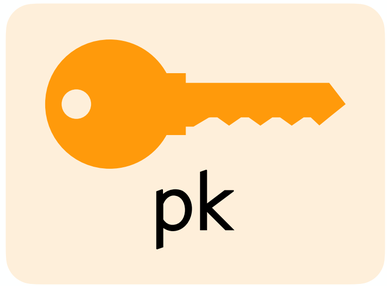
Batch Encryption

introduced by [CGPP24]



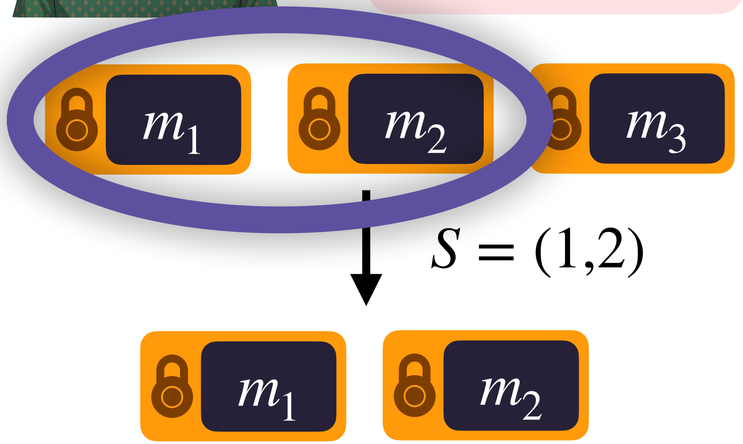








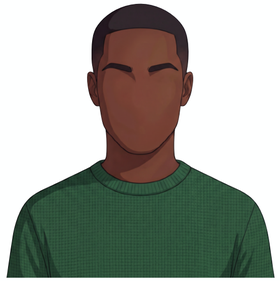


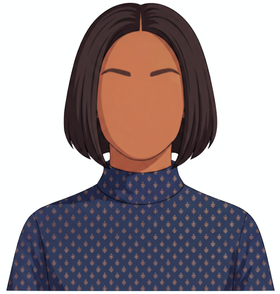


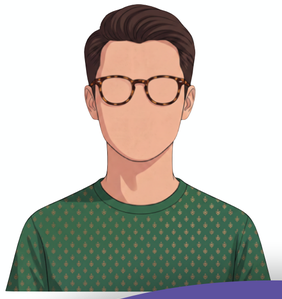
Batch Encryption

introduced by [CGPP24]









pk

sk

m_1 m_2 m_3

$S = (1,2)$

m_1 m_2

derive "pre-decryption" key

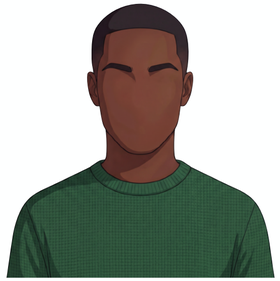
S

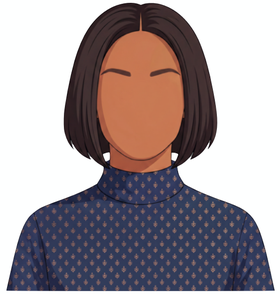
sbk

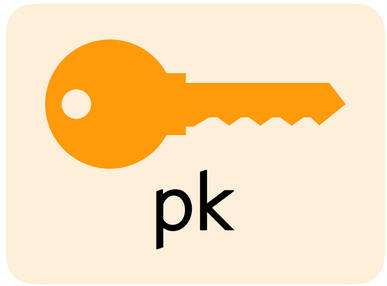
Batch Encryption

introduced by [CGPP24]



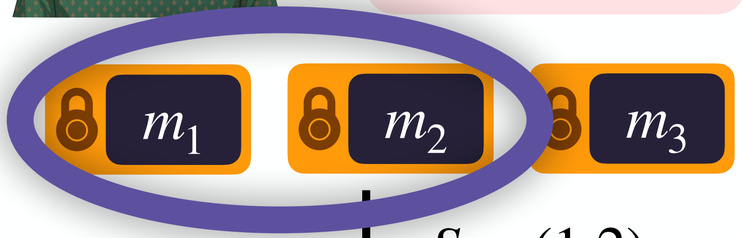


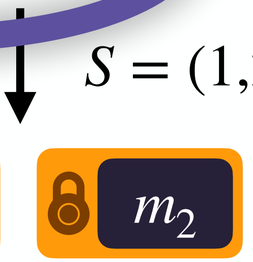












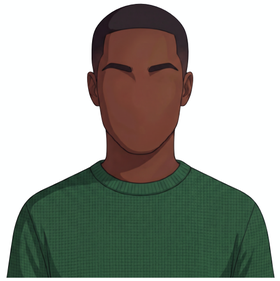


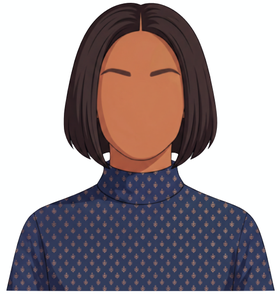


Batch Encryption

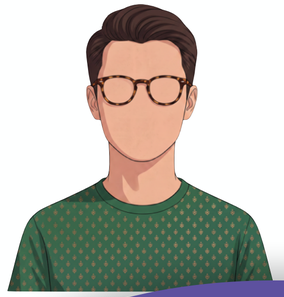
introduced by [CGPP24]















$S = (1,2)$



derive "pre-decryption" key

decrypt



sbk



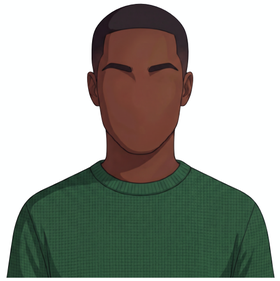


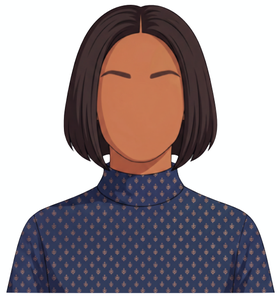
remains semantically secure

Batch Encryption

introduced by [CGPP24]







pk



sk

m_1 m_2 m_3

trivial solution:
Decryptor can simply do the decryption themself and publish the decrypted messages as sbk

decrypt

“pre-decryption” key

m_1 m_2

S

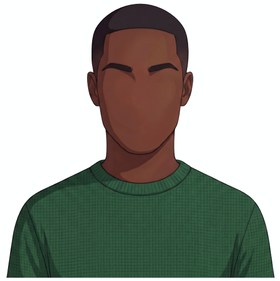
sbk

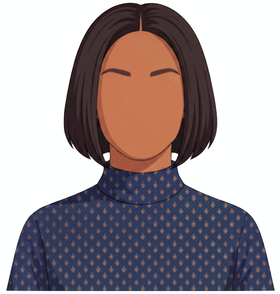
m_3 remains semantically secure

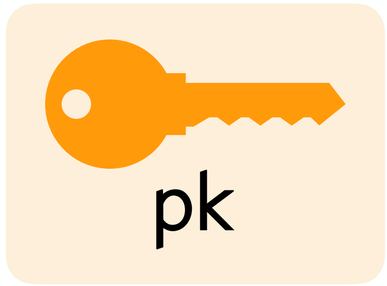
Batch Encryption

introduced by [CGPP24]

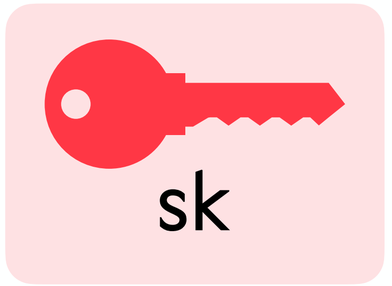








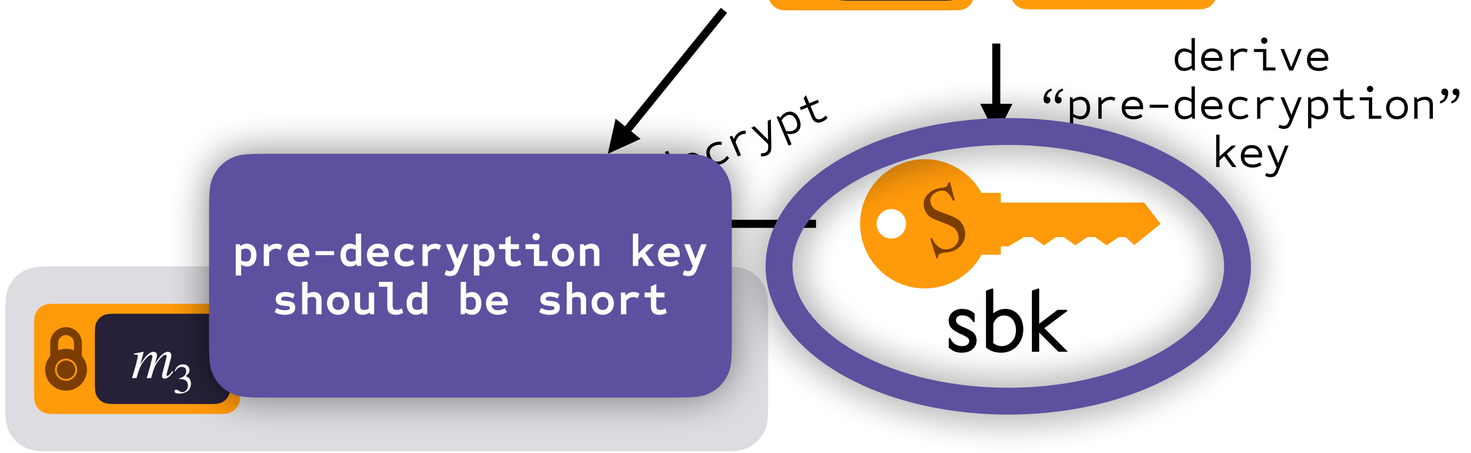








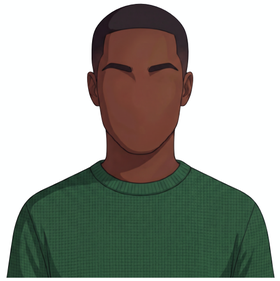


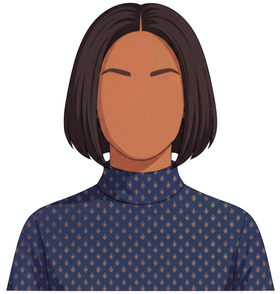


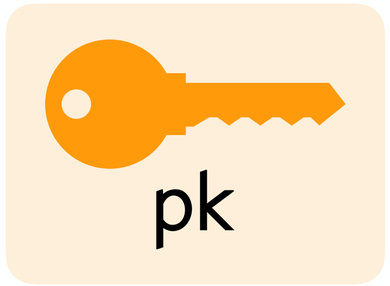
Batch Encryption

introduced by [CGPP24]





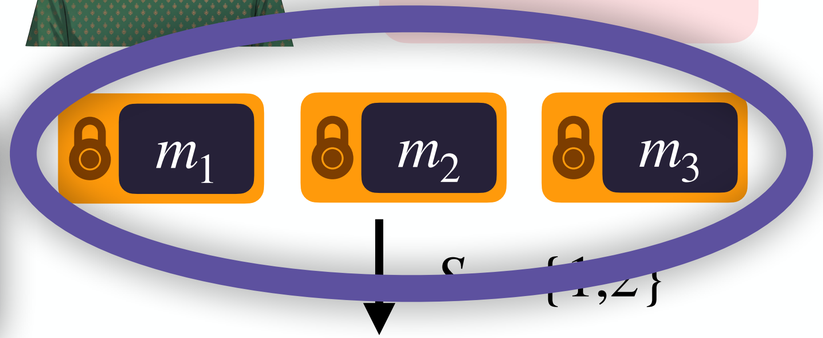








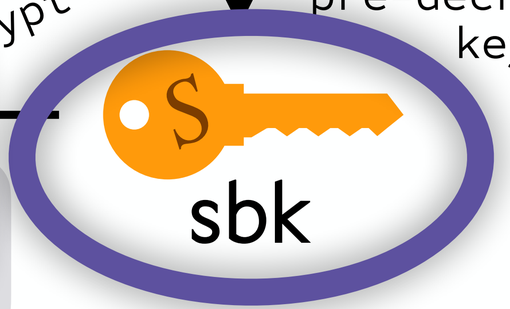
ciphertexts should also be short





derive “pre-decryption” key

pre-decryption key should be short

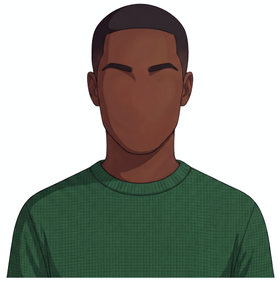


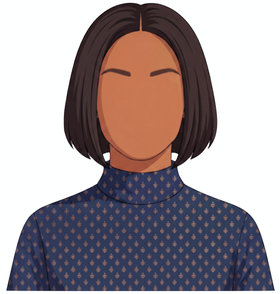


Batch *Threshold* Encryption

introduced by [CGPP24]









pk

sk

can be split between multiple parties with

sk_1 sk_2 • • • sk_n

and threshold t, s.t. at least t parties required to decrypt subset.

Decryption is non-interactive!

m_1

m_3

remains s
secure

Batch Threshold Encryption

Batch Threshold Encryption

Highly active area of research

Loads of work on this in various settings!

See e.g. [SAS24, ABD+25, AFP25, BCF+25, BFO+25, CGP+25, FPT+25, GWW+25, ADG+26, Pol26]

Achieve either **a relaxed notion of the primitive or suffer from a large CRS**. (We will revisit this toward the end…)

Batch Threshold Encryption

Highly active area of research

Loads of work on this in various settings!

See e.g. [SAS24, ABD+25, AFP25, BCF+25, BFO+25, CGP+25, FPT+25, GWW+25, ADG+26, Pol26]

Achieve either **a relaxed notion of the primitive or suffer from a large CRS**. (We will revisit this toward the end…)

Why do we care?

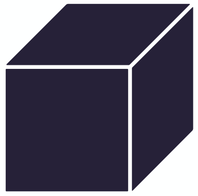
Batch encryption has a **strong application in blockchains**.

It **allows transactions to stay hidden (encrypted)** until the block is confirmed.

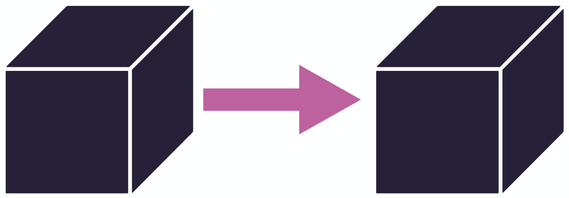
Helps **prevent MEV attacks** (e.g. front-running)

Application: *Encrypted Mempools* for Blockchain

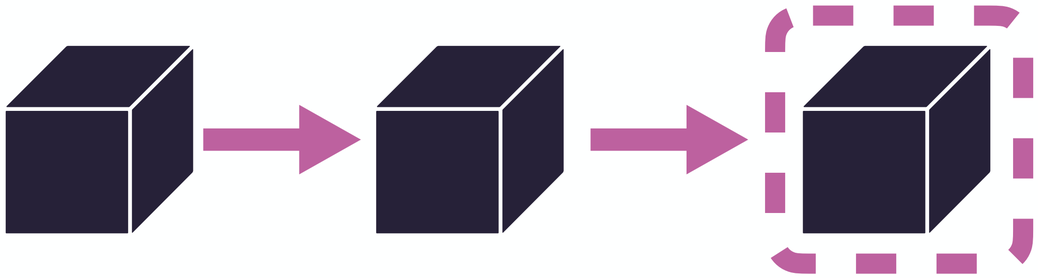
Application: *Encrypted Mempools* for Blockchain



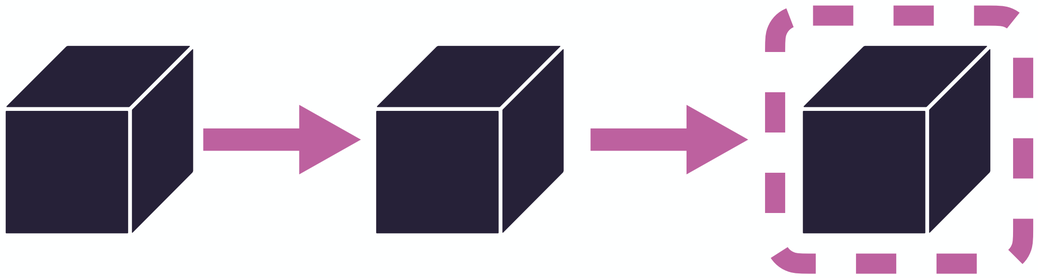
Application: *Encrypted Mempools* for Blockchain



Application: *Encrypted Mempools* for Blockchain



Application: *Encrypted Mempools* for Blockchain



users with transactions



tx_1

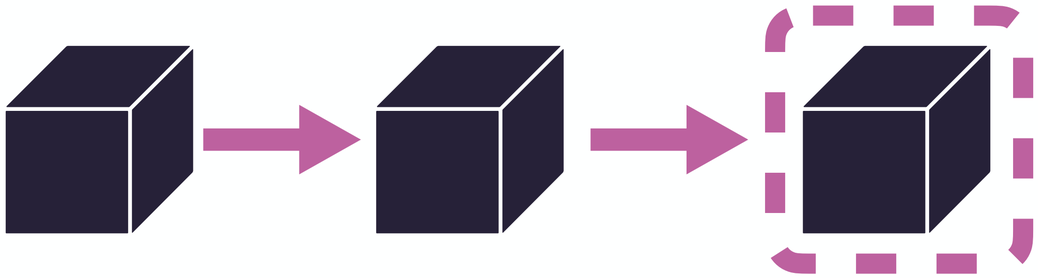


tx_2



tx_3 tx_4

Application: *Encrypted Mempools* for Blockchain



miner



users with transactions



tx_1

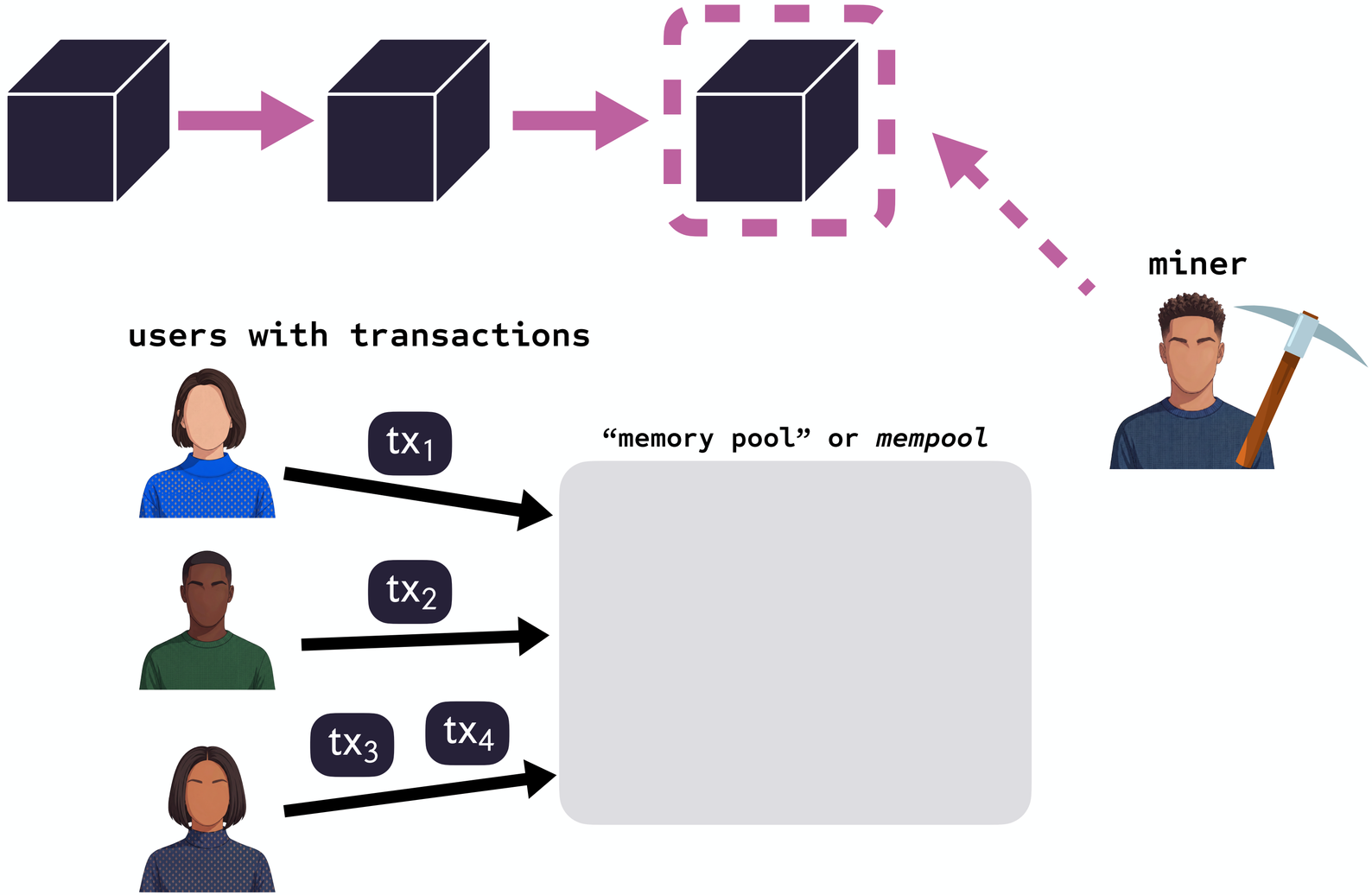


tx_2

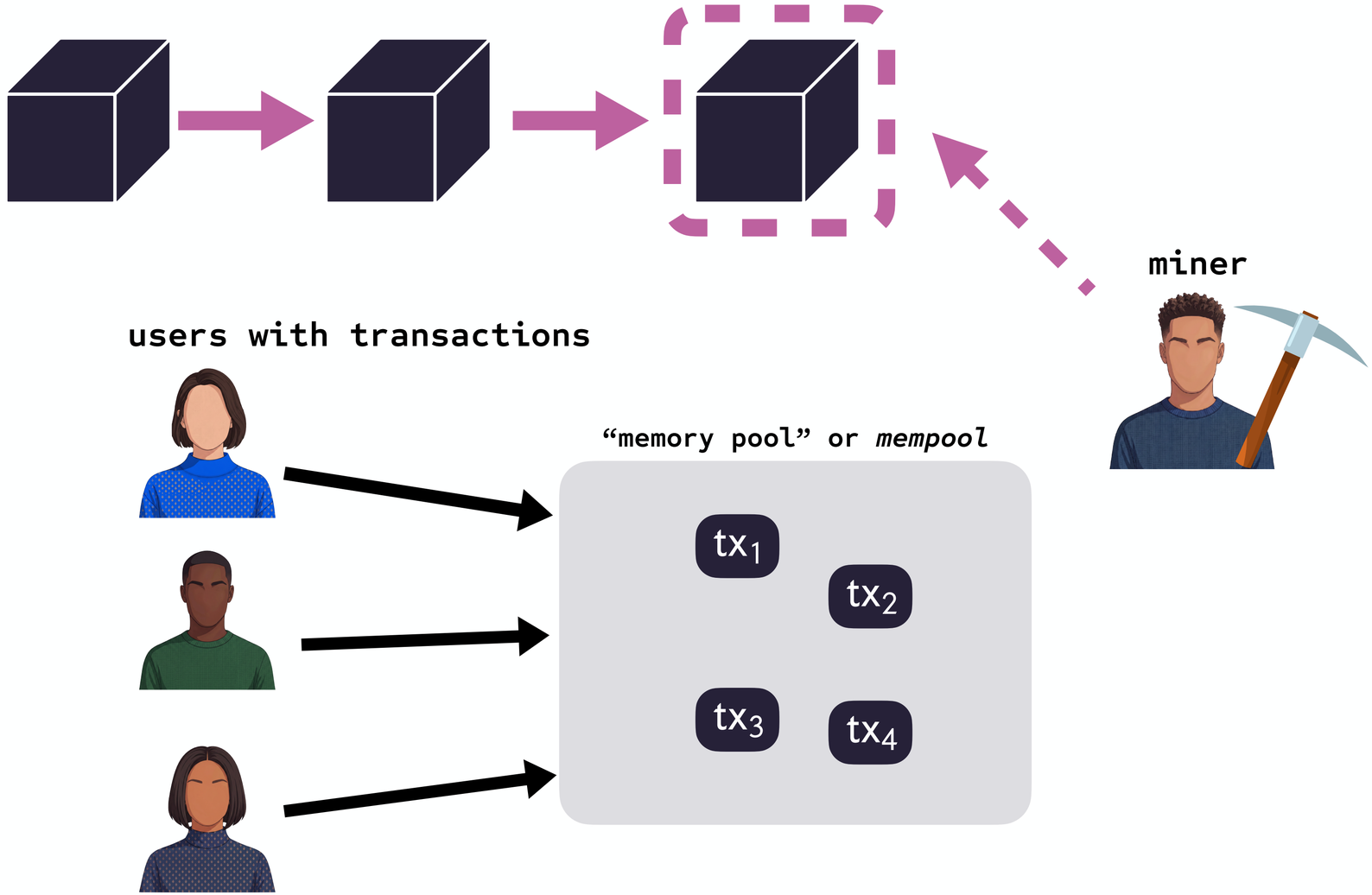
tx_3 tx_4



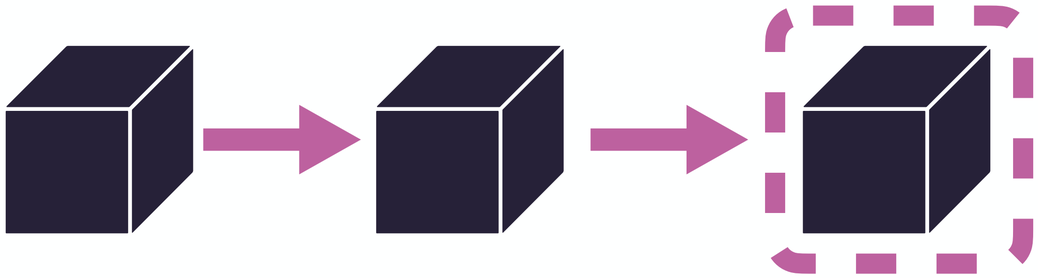
Application: *Encrypted Mempools* for Blockchain

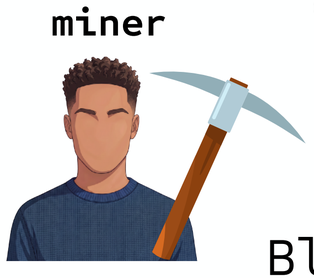


Application: *Encrypted Mempools* for Blockchain



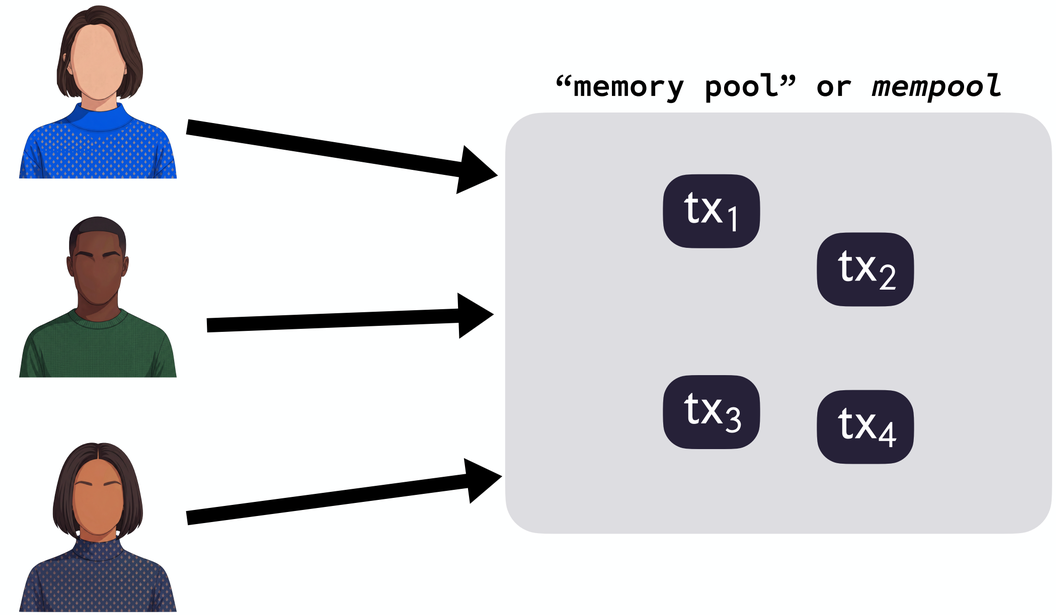
Application: *Encrypted Mempools* for Blockchain





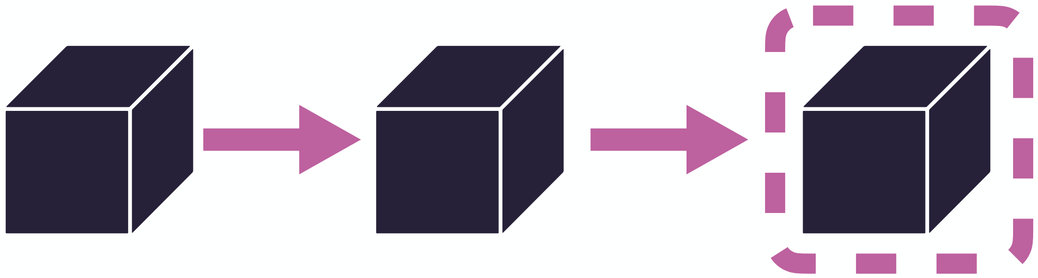
Transactions on block are executed sequentially.

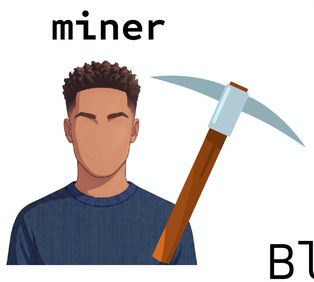
Order matters!



Block proposer can order/ pick in a way that’s beneficial to them (MEV attacks e.g. front-running)

Application: *Encrypted Mempools* for Blockchain





Transactions on block are executed sequentially.

Order matters!

users with transactions







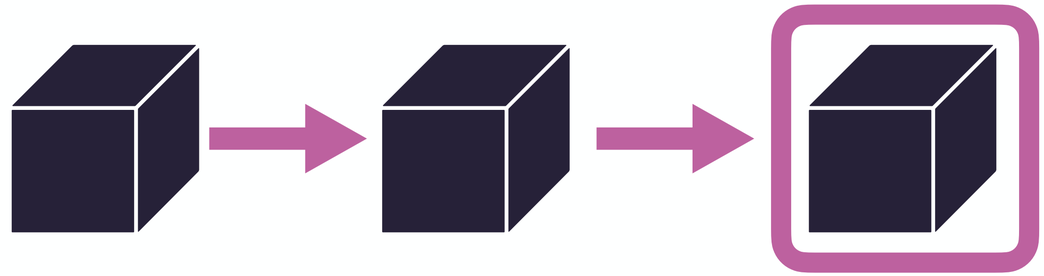
“memory pool” or *mempool*



Block proposer can order/ pick in a way that’s beneficial to them (MEV attacks e.g. front-running)

Can we hide them?
Yes, we could encrypt…

Application: *Encrypted Mempools* for Blockchain



ℓ: max block size
($\ell = 3$)







miner



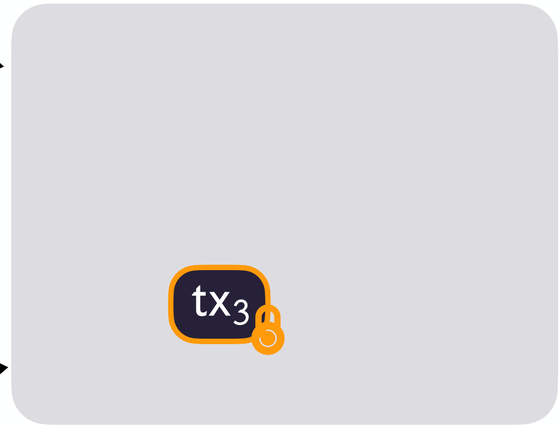
users with transactions



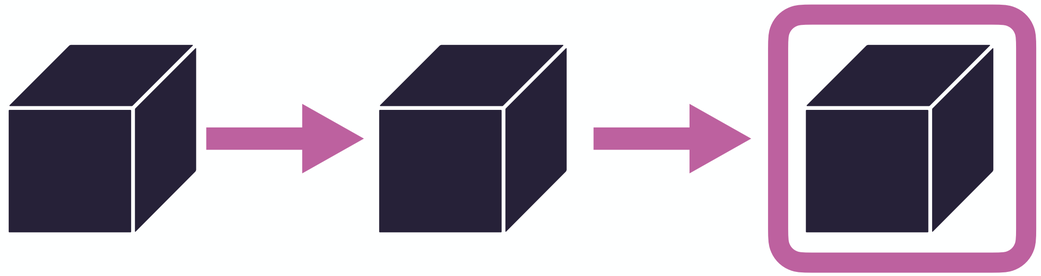




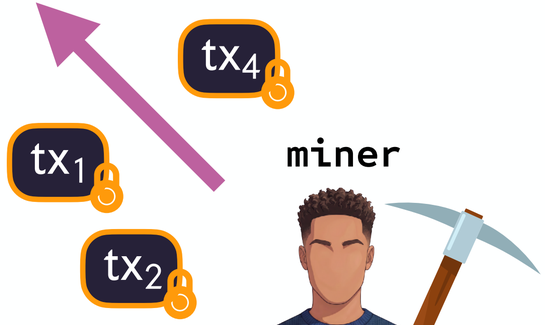
“memory pool” or *mempool*



Application: *Encrypted Mempools* for Blockchain



ℓ: max block size
($\ell = 3$)



miner



Block is confirmed.

How to decrypt?

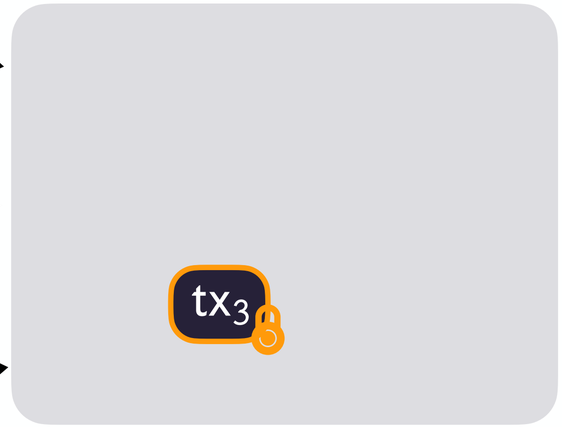
users with transactions



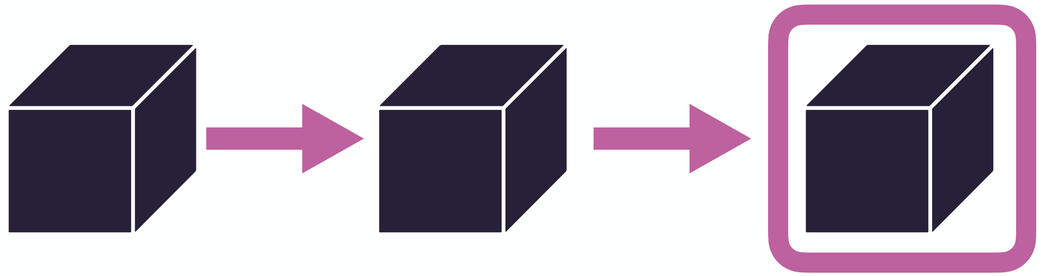




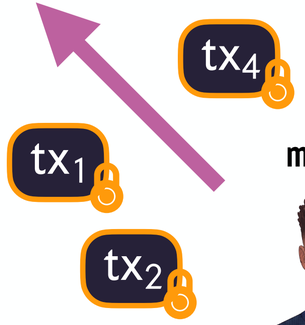
“memory pool” or *mempool*



Application: *Encrypted Mempools* for Blockchain



ℓ: max block size
($\ell = 3$)



miner



Block is confirmed.

How to decrypt?

What about transactions that were **not included**?

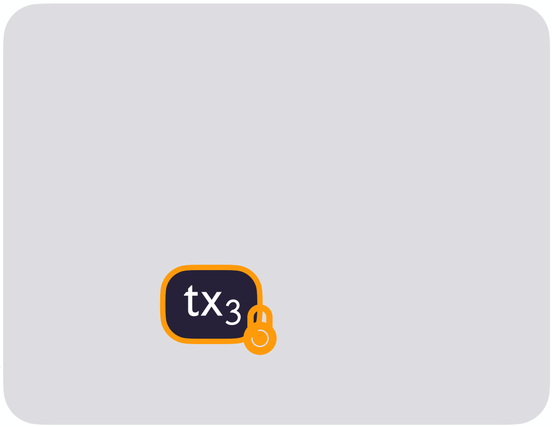
users with transactions



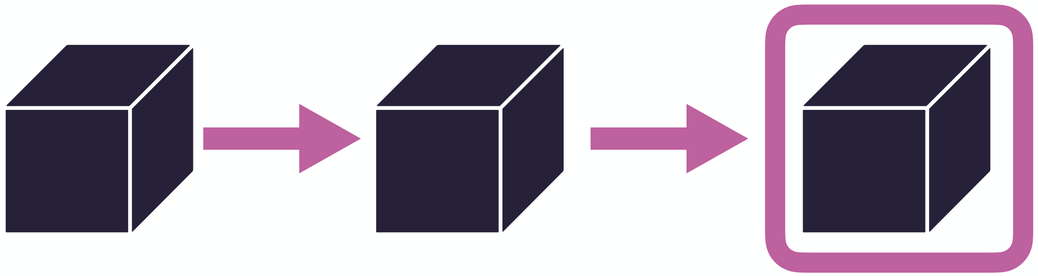




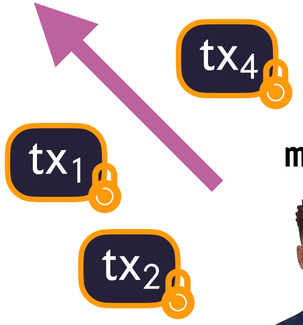
“memory pool” or *mempool*



Application: *Encrypted Mempools* for Blockchain



ℓ: max block size
($\ell = 3$)



miner



Block is confirmed.

How to decrypt?

What about transactions that were **not included**?

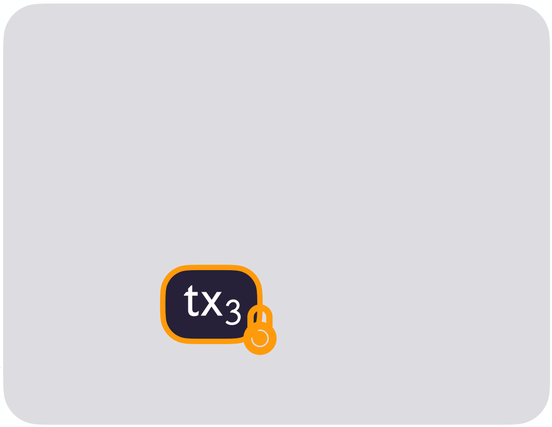
users with transactions







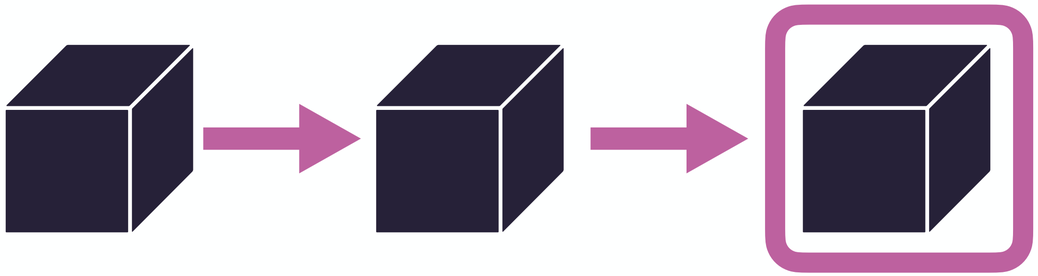
“memory pool” or *mempool*



Batch Encryption enables efficient decryption of *only* included transactions: tx_1, tx_2, tx_4

while keeping excluded transactions (tx_3) secret (semantically secure)

Application: *Encrypted Mempools* for Blockchain



ℓ: max block size
($\ell = 3$)

tx_4

tx_1

tx_2

miner



Block is confirmed.

How to decrypt?

What about transactions that were **not included**?

users with transactions







“memory pool” or *mempool*

tx_3

can now be included in a future epoch without being revealed until confirmed

Batch Encryption enables efficient decryption of *only* included transactions: $\mathsf{tx}_1, \mathsf{tx}_2, \mathsf{tx}_4$

while keeping excluded transactions (tx_3) secret (semantically secure)

Batch Encryption (BE) Syntax

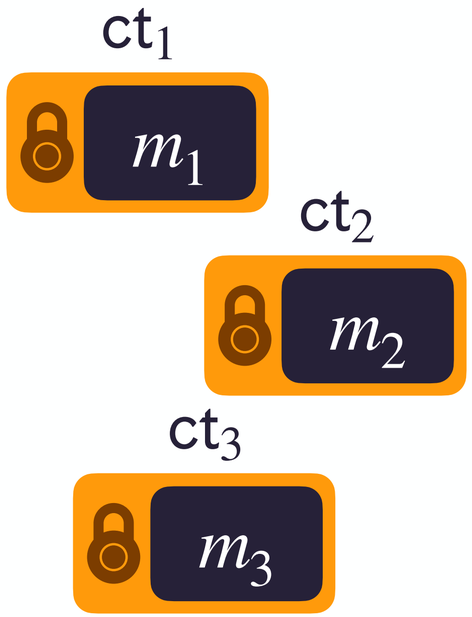
Batch Encryption (BE) Syntax

$$\mathsf{Setup}\left(1^{\lambda}, 1^{\ell}\right) \rightarrow \left(\mathsf{ek}, \mathsf{sk}, \mathsf{dk}\right)$$

Batch Encryption (BE) Syntax

$$\mathsf{Setup}\left(1^{\lambda}, 1^{\ell}\right) \rightarrow \left(\mathsf{ek}, \mathsf{sk}, \mathsf{dk}\right)$$

$$\mathsf{Enc}\left(\mathsf{ek}, m\right) \rightarrow \mathsf{ct}$$

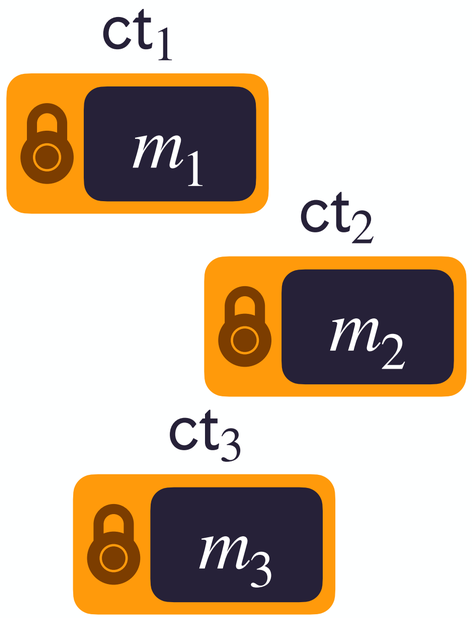


Batch Encryption (BE) Syntax

$$\mathsf{Setup}\left(1^{\lambda}, 1^{\ell}\right) \to \left(\mathsf{ek}, \mathsf{sk}, \mathsf{dk}\right)$$

$$\mathsf{Enc}\left(\mathsf{ek}, m\right) \to \mathsf{ct}$$

$$\mathsf{PreDec}\left(\mathsf{sk}, \left(\mathsf{ct}_i\right)_i\right) \to \mathsf{sbk}$$

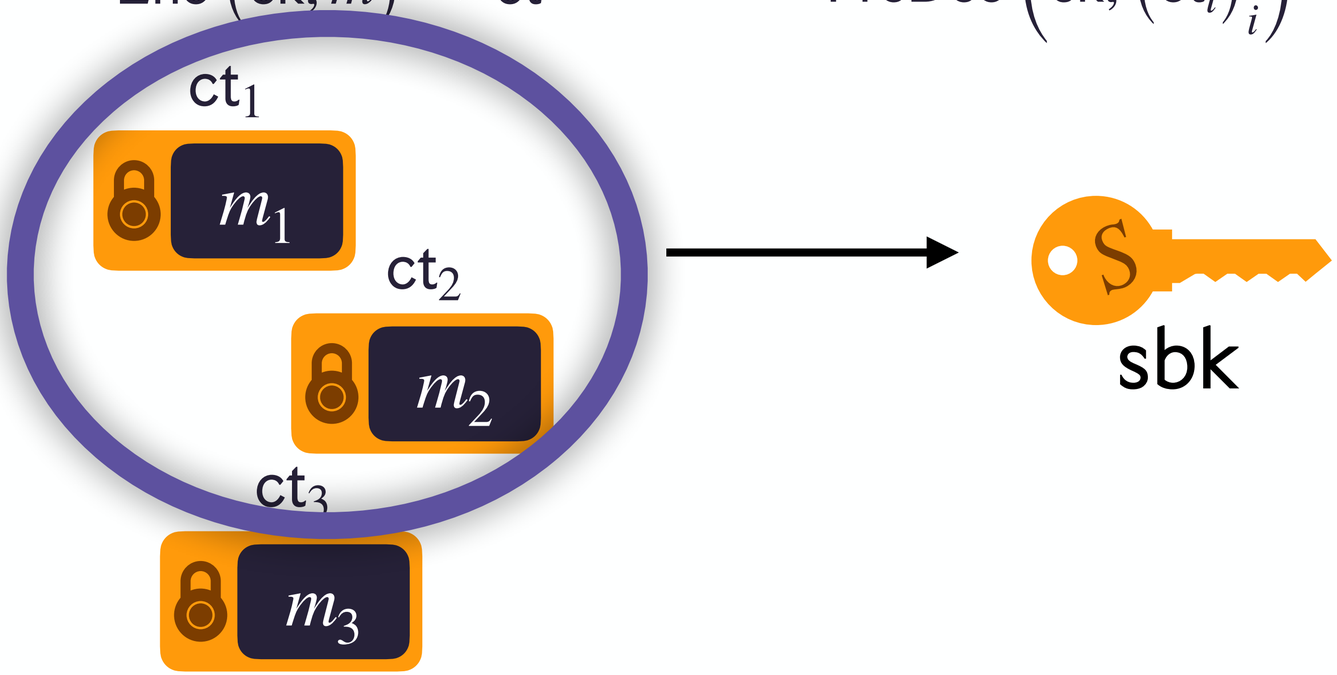


Batch Encryption (BE) Syntax

$\text{Setup}\left(1^{\lambda}, 1^{\ell}\right) \to \left(\text{ek}, \text{sk}, \text{dk}\right)$

$\text{Enc}\left(\text{ek}, m\right) \to \text{ct}$

$\text{PreDec}\left(\text{sk}, \left(\text{ct}_i\right)_i\right) \to \text{sbk}$



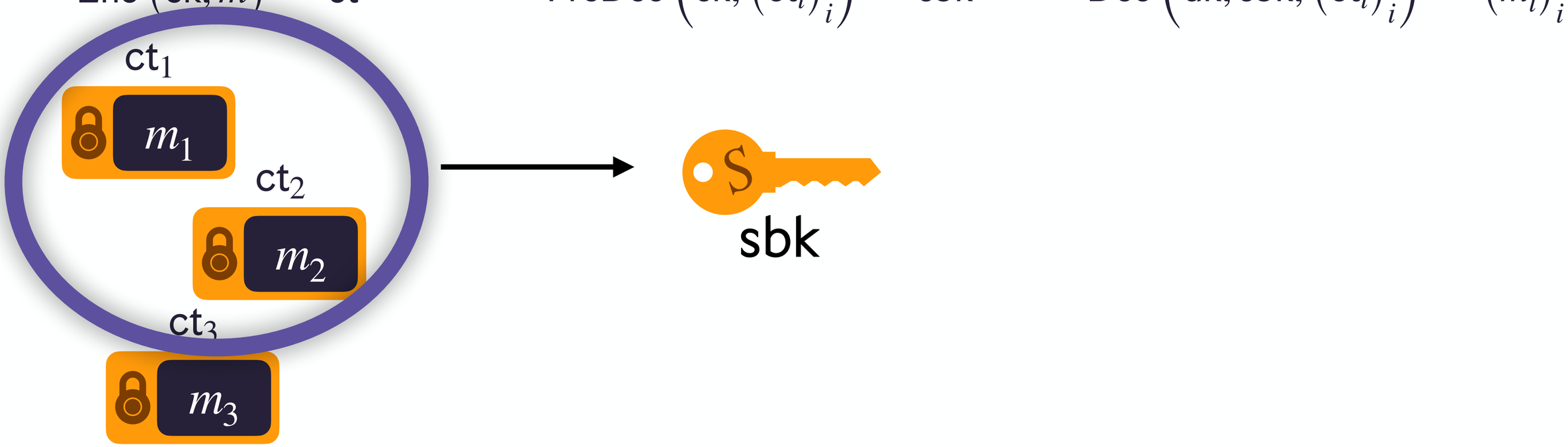
Batch Encryption (BE) Syntax

$$\text{Setup}\left(1^{\lambda}, 1^{\ell}\right) \to \left(\text{ek}, \text{sk}, \text{dk}\right)$$

$$\text{Enc}\left(\text{ek}, m\right) \to \text{ct}$$

$$\text{PreDec}\left(\text{sk}, \left(\text{ct}_i\right)_i\right) \to \text{sbk}$$

$$\text{Dec}\left(\text{dk}, \text{sbk}, \left(\text{ct}_i\right)_i\right) \to \left(m_i\right)_i$$



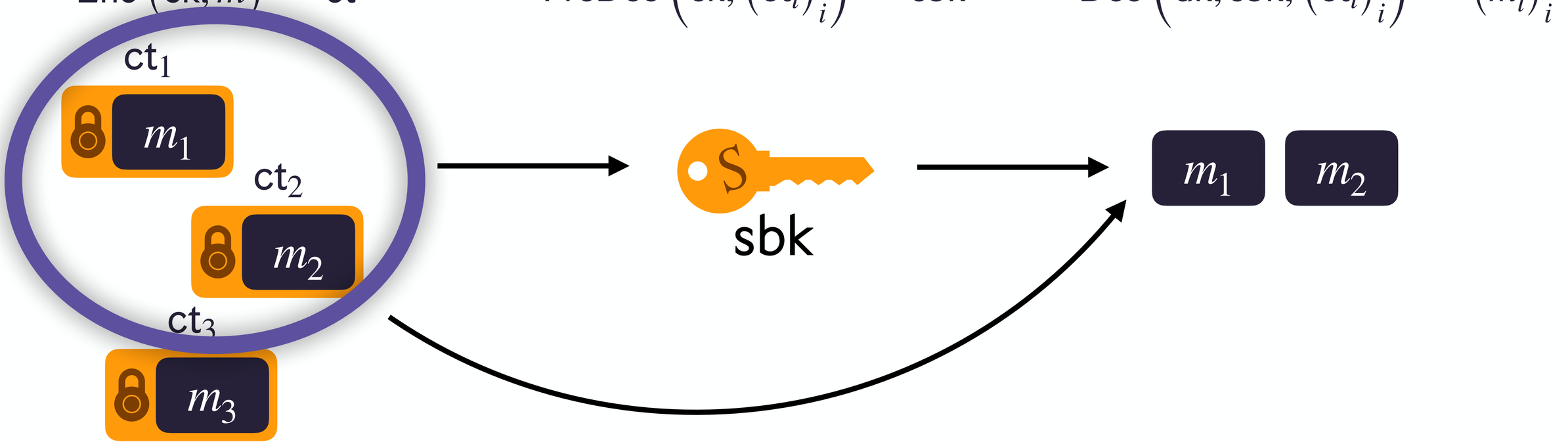
Batch Encryption (BE) Syntax

$$\text{Setup}\left(1^{\lambda}, 1^{\ell}\right) \to \left(\text{ek}, \text{sk}, \text{dk}\right)$$

$$\text{Enc}\left(\text{ek}, m\right) \to \text{ct}$$

$$\text{PreDec}\left(\text{sk}, \left(\text{ct}_i\right)_i\right) \to \text{sbk}$$

$$\text{Dec}\left(\text{dk}, \text{sbk}, \left(\text{ct}_i\right)_i\right) \to \left(m_i\right)_i$$



Our Construction: Roadmap

[Boneh-**N**-Roy-Tas26]

Our Construction: Roadmap

[Boneh-**N**-Roy-Tas26]

Index-dependent
Batch Encryption

Ciphertext is tied to a **fixed position** in the batch **known at encryption time**

Enc(ek, **index** $i \in [\ell], m$)

Our Construction: Roadmap

[Boneh-**N**-Roy-Tas26]

Index-dependent
Batch Encryption

Ciphertext is tied to a **fixed position** in the batch **known at encryption time**

Enc(ek, **index** $i \in [\ell], m$)

+

Broadcast Encryption

Short ciphertext can be decrypted by only authorized secret keys

Dec(**authorized** $\mathsf{sk}_j, \mathsf{ct}) \to m$

Dec(**unauthorized** $\mathsf{sk}_j, \mathsf{ct}) \to \perp$

from [Jutla-**N**-Roy26]

Our Construction: Roadmap

[Boneh-**N**-Roy-Tas26]

Index-dependent
Batch Encryption

Ciphertext is tied to a **fixed position** in the batch **known at encryption time**

$\mathsf{Enc}(\mathsf{ek},\ \textbf{index}\ i \in [\ell], m)$

+

Broadcast Encryption

Short ciphertext can be decrypted by only authorized secret keys

$\mathsf{Dec}(\textbf{authorized}\ \mathsf{sk}_j, \mathsf{ct}) \to m$

$\mathsf{Dec}(\textbf{unauthorized}\ \mathsf{sk}_j, \mathsf{ct}) \to \perp$

from [Jutla-**N**-Roy26]

→

Batch Encryption

Our Construction: Roadmap

[Boneh-**N**-Roy-Tas26]

Index-dependent
Batch Encryption

Ciphertext is tied to a **fixed position** in the batch **known at encryption time**

Enc(ek, **index** $i \in [\ell], m$)

+

Broadcast Encryption

Short ciphertext can be decrypted by only authorized secret keys

Dec(**authorized** $\mathsf{sk}_j, \mathsf{ct}) \to m$

Dec(**unauthorized** $\mathsf{sk}_j, \mathsf{ct}) \to \perp$

from [Jutla-**N**-Roy26]

→

Batch Encryption

Thresholdize using LSS

↓

Batch Threshold Encryption

Our Construction: Roadmap

[Boneh-**N**-Roy-Tas26]

Index-dependent
Batch Encryption

Ciphertext is tied to a **fixed position** in the batch **known at encryption time**

Enc(ek, **index** $i \in [\ell], m$)

We'll first construct this relaxed notion of BE

+

Broadcast Encryption

Short ciphertext can be decrypted by only authorized secret keys

Dec(**authorized** $\mathsf{sk}_j, \mathsf{ct}) \to m$

Dec(**unauthorized** $\mathsf{sk}_j, \mathsf{ct}) \to \perp$

from [Jutla-**N**-Roy26]

→

Batch Encryption

Thresholdize using LSS

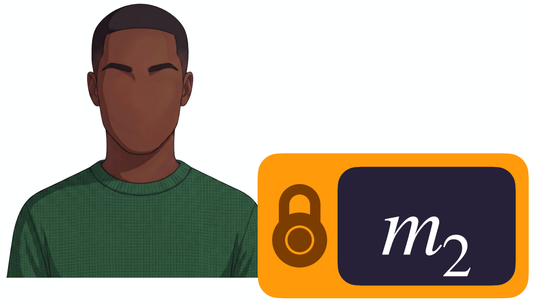
↓

Batch Threshold Encryption

Index Dependence

ℓ: max batch size = 3








pk

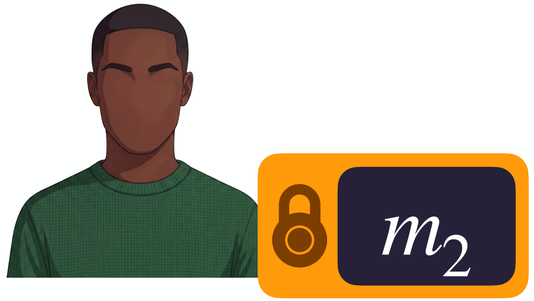



sk

Index Dependence

ℓ: max batch size = 3



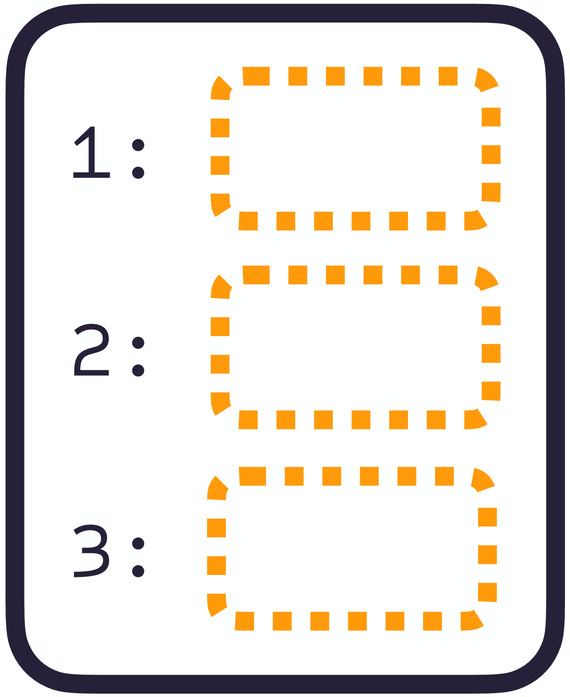





pk




sk

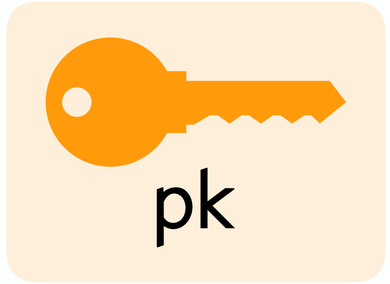


Index Dependence

ℓ: max batch size = 3

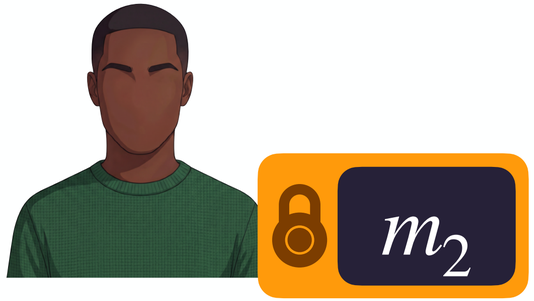


, 1

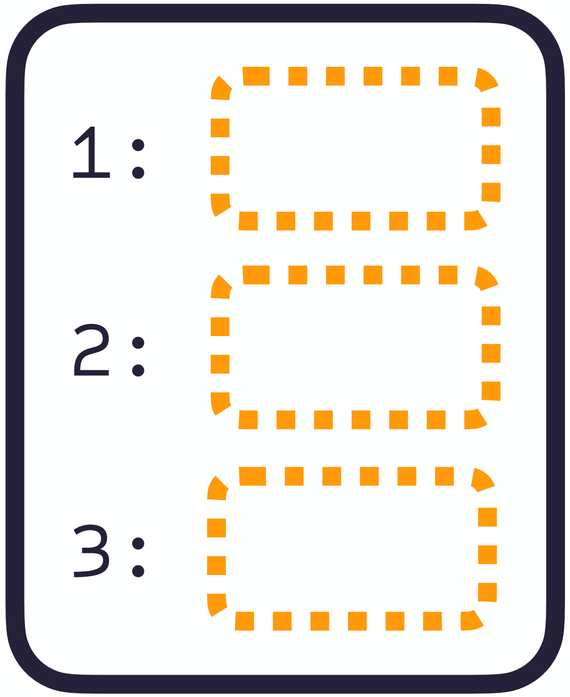










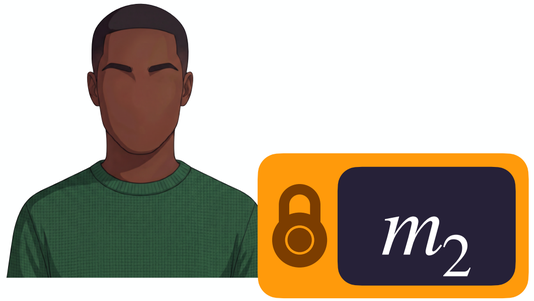


Index Dependence

ℓ: max batch size = 3

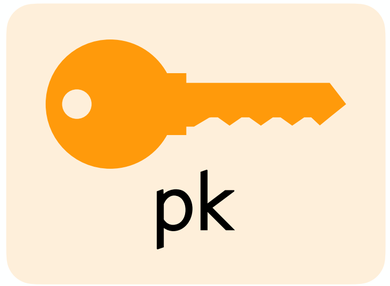


, 1

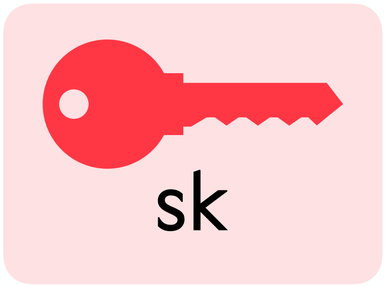


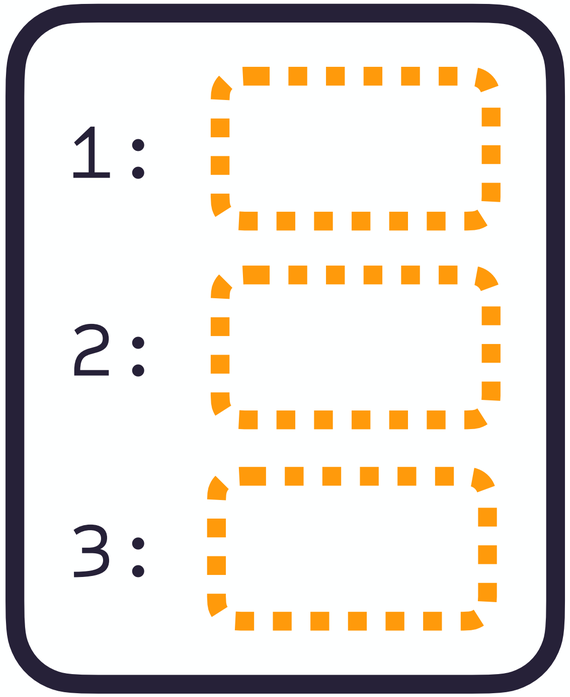
, 2









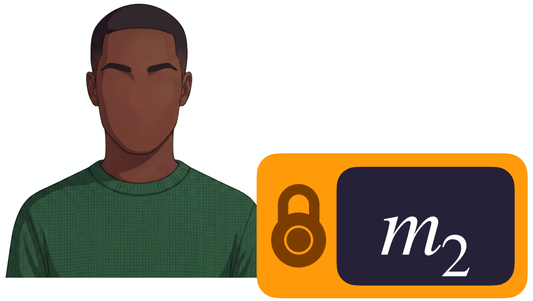


Index Dependence

ℓ: max batch size = 3



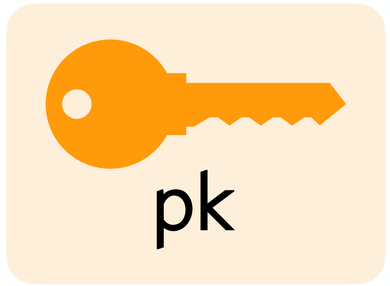
, 1



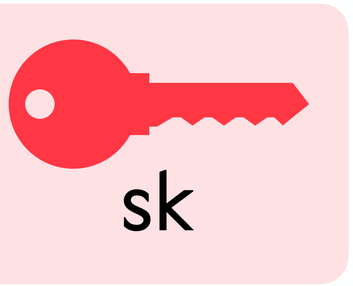
, 2

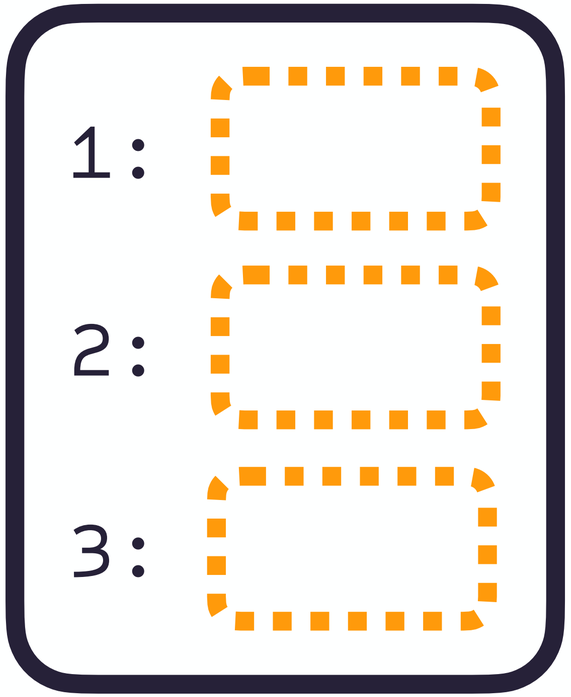


, 3









Index Dependence: Good Case

ℓ: max batch size = 3





m_1 , 1



pk





sk



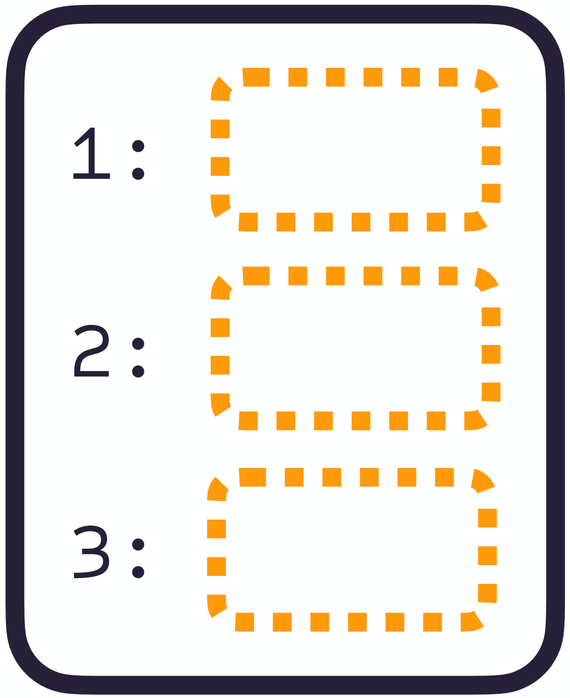


m_2 , 2





m_3 , 3



Index Dependence: Good Case

ℓ: max batch size = 3



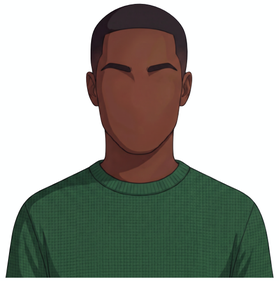


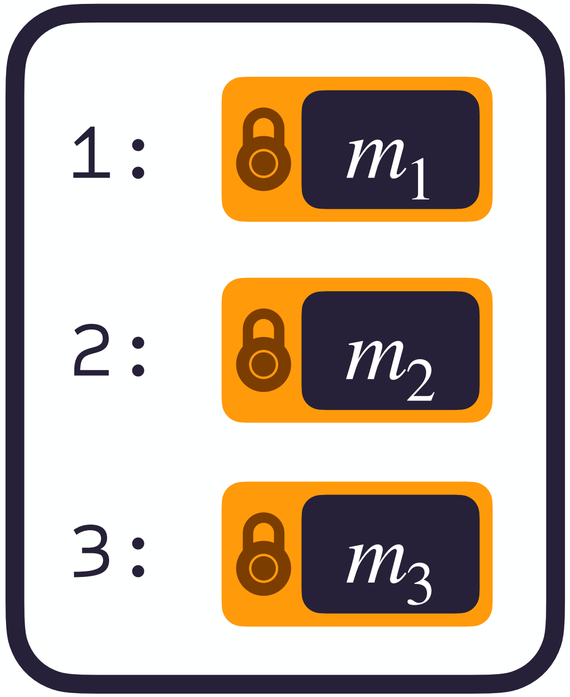
pk

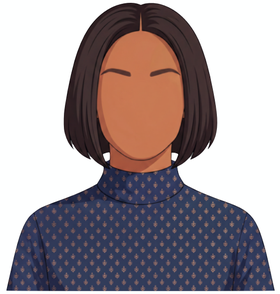




sk



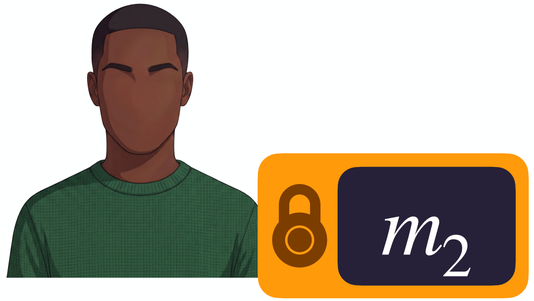




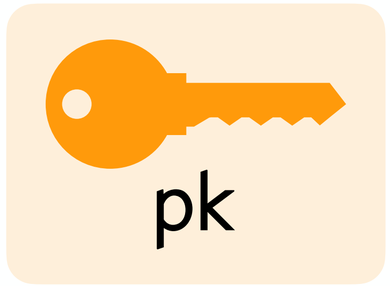
Index Dependence: Bad Case

ℓ: max batch size = 3

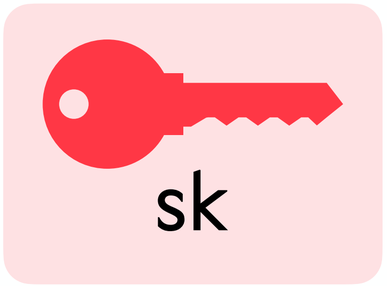


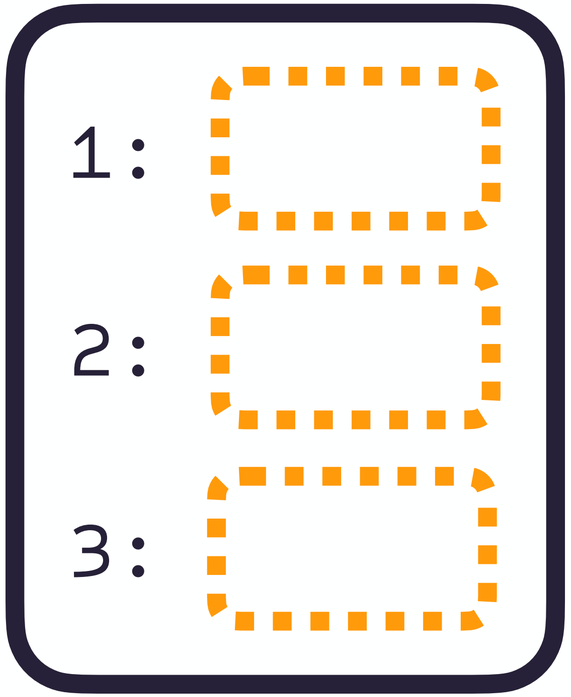










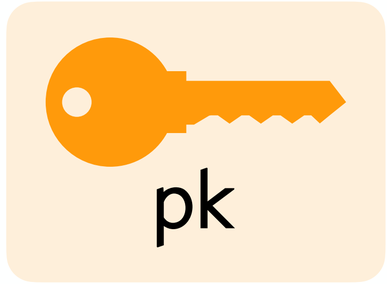


Index Dependence: Bad Case

ℓ: max batch size = 3

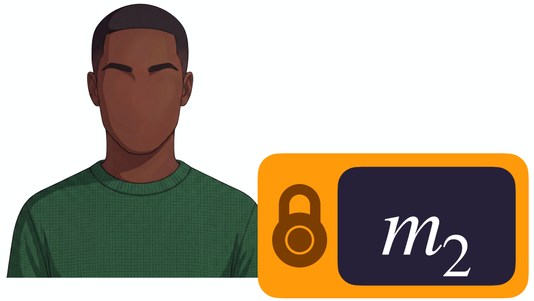


, 1





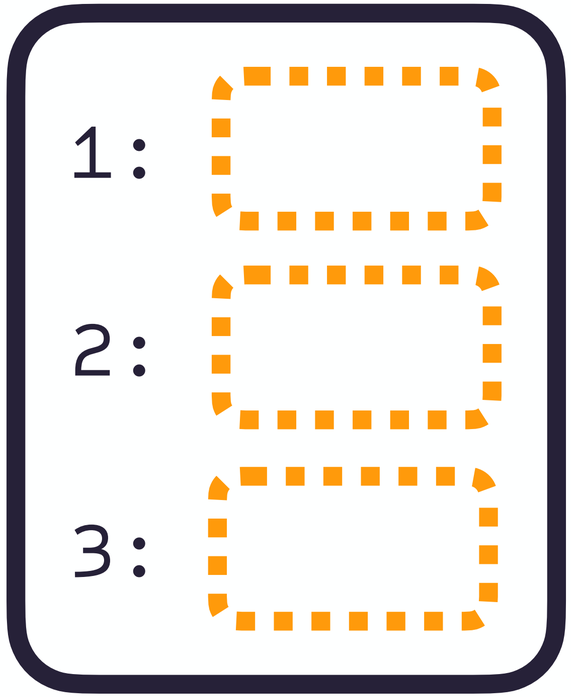




, 2



, 1



Index Dependence: Bad Case

ℓ: max batch size = 3





m_1 , 1

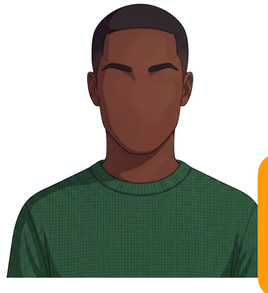


pk





sk





m_2 , 2





m_3 , 1

1:

2:

3:

Index Dependence: Bad Case

ℓ: max batch size = 3





m_1 , 1



pk





sk



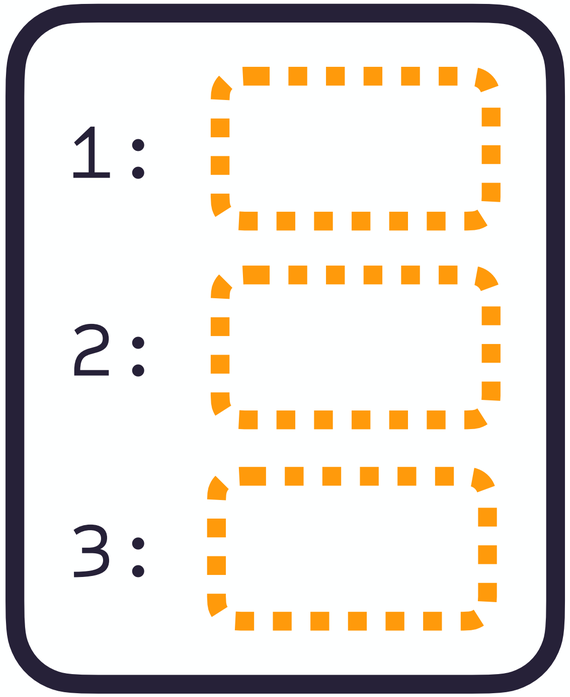


m_2 , 2





m_3 , 1



Index Dependence: Bad Case

ℓ: max batch size = 3





m_1 , 1

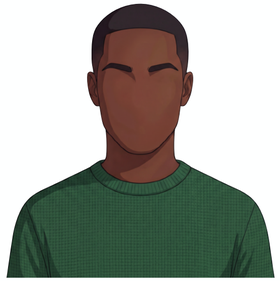


pk





sk





m_3 , 1

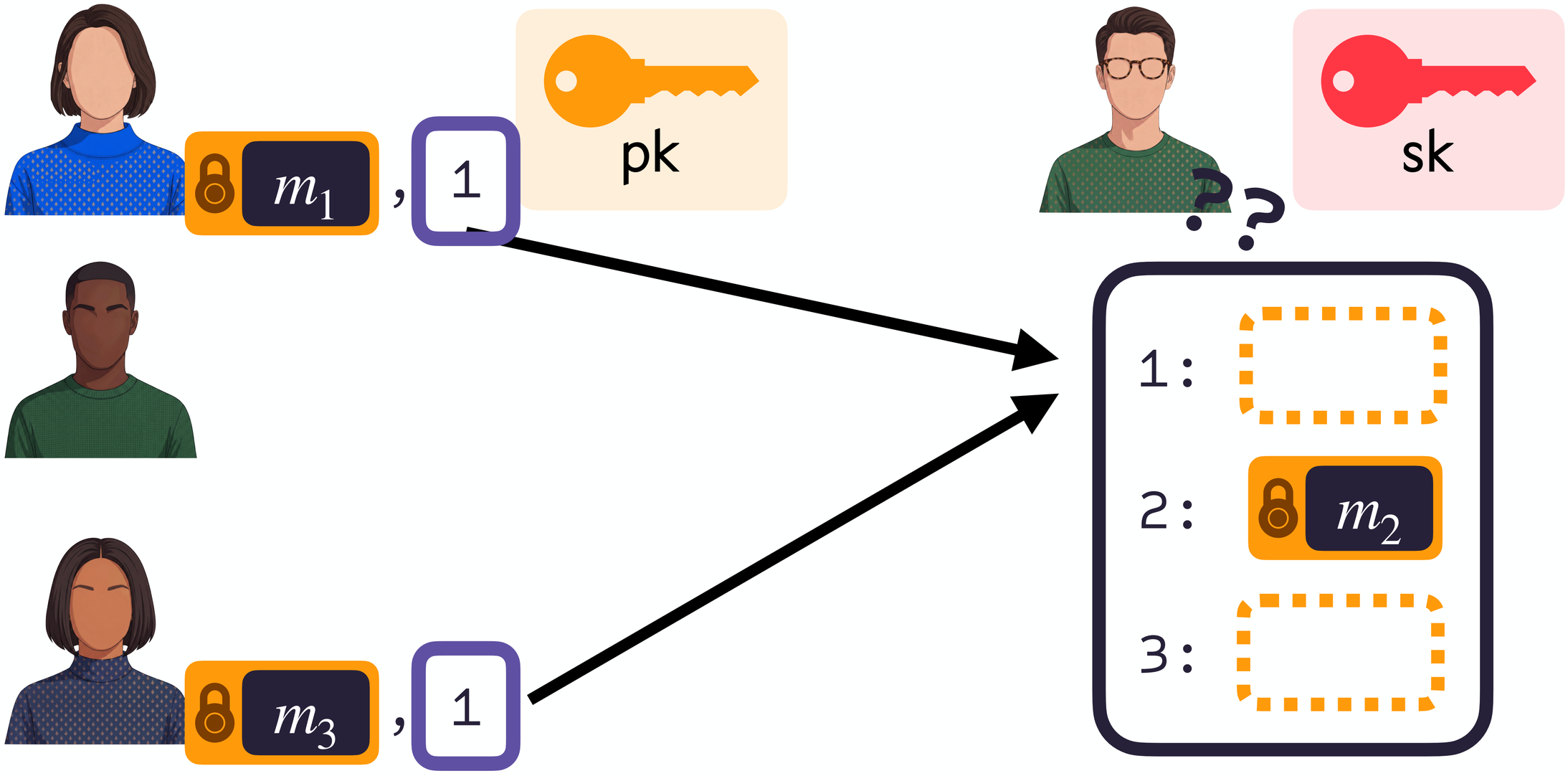
1:

2: m_2

3:

Index Dependence: Bad Case

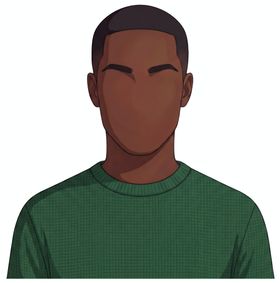
ℓ: max batch size = 3



Index Dependence: Bad Case

ℓ: max batch size = 3

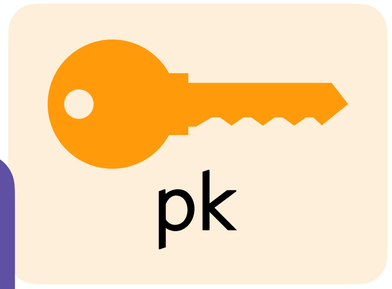








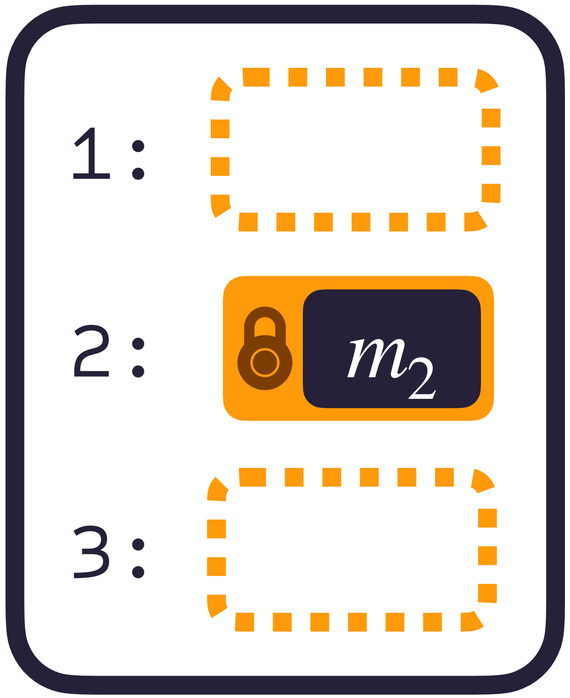








only one of them can be included…

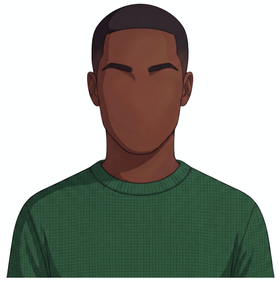


Index Dependence: Bad Case

ℓ: max batch size = 3



m_1 , 1





m_3 , 1



pk





sk

1:

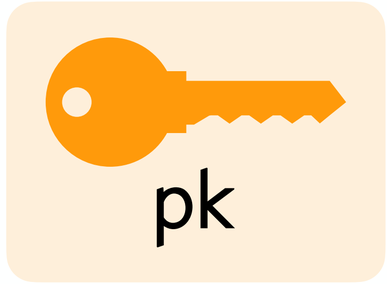
2: m_2

3:

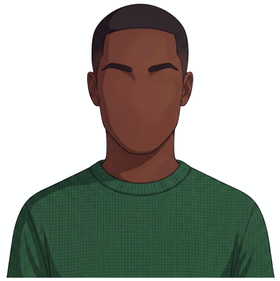
Index Dependence: Bad Case

ℓ: max batch size = 3



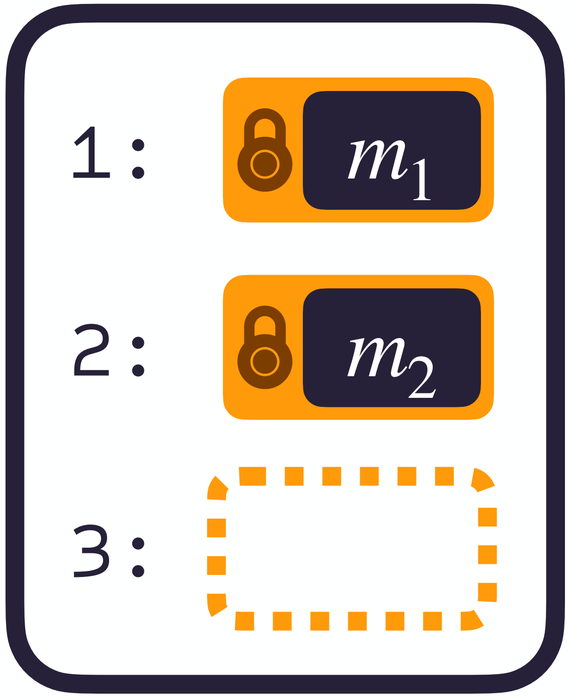








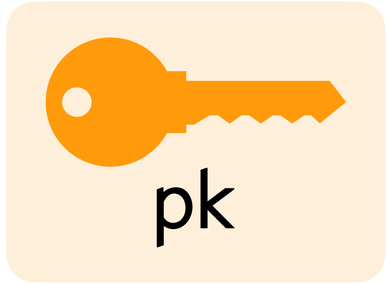
, 1



Index Dependence: Bad Case

ℓ: max batch size = 3





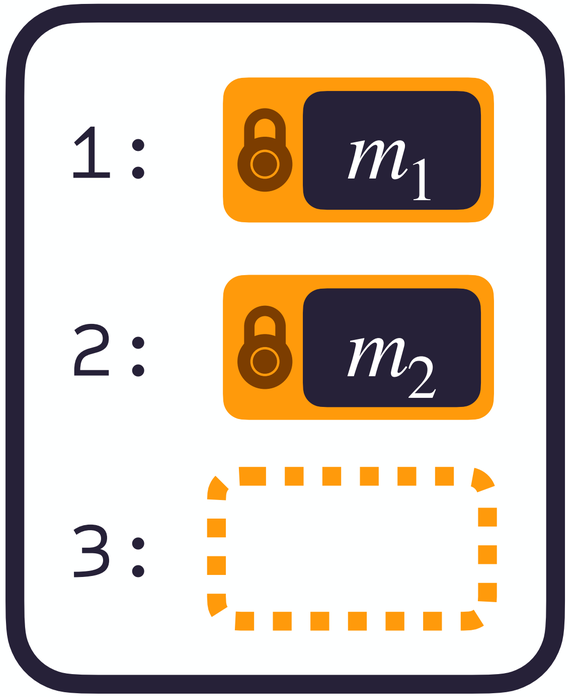








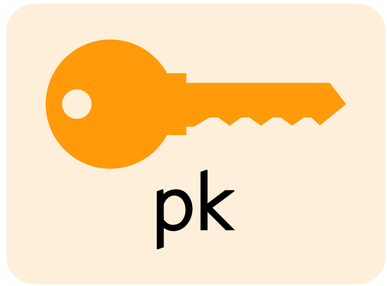
m_3 , 1



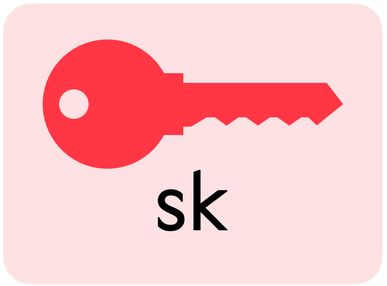
Index Dependence: Bad Case

ℓ: max batch size = 3







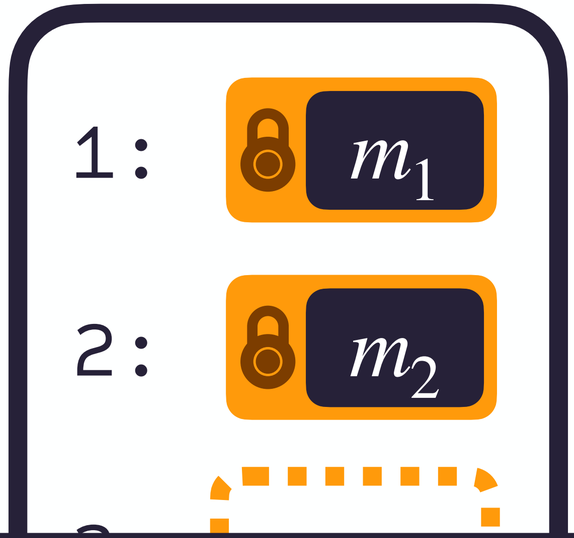




sad… there was space for me too!



m_3 , 1



we will first construct this weaker notion of BE, and then *upgrade* it

Index-Dependent Batch Encryption

[Boneh-**N**-Roy-Tas26]

Property 1 PFD is sufficient to construct the weaker notion

Index-Dependent Batch Encryption

[Boneh-**N**-Roy-Tas26]

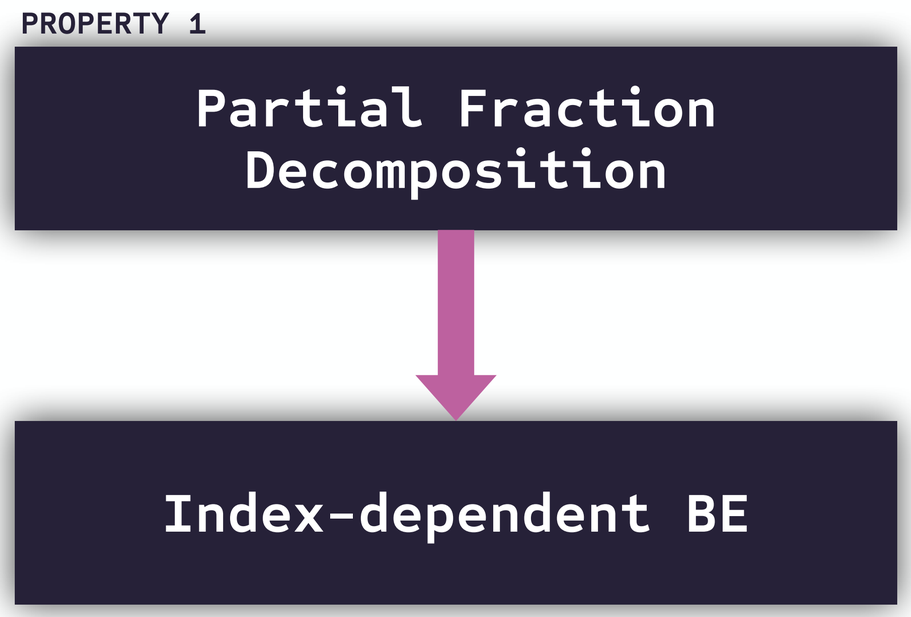
Property 1 PFD is sufficient to construct the weaker notion

sorry, I lied...
Property 2 is coming later

Index-Dependent Batch Encryption

[Boneh-N-Roy-Tas26]

Property 1 PFD is sufficient to construct the weaker notion



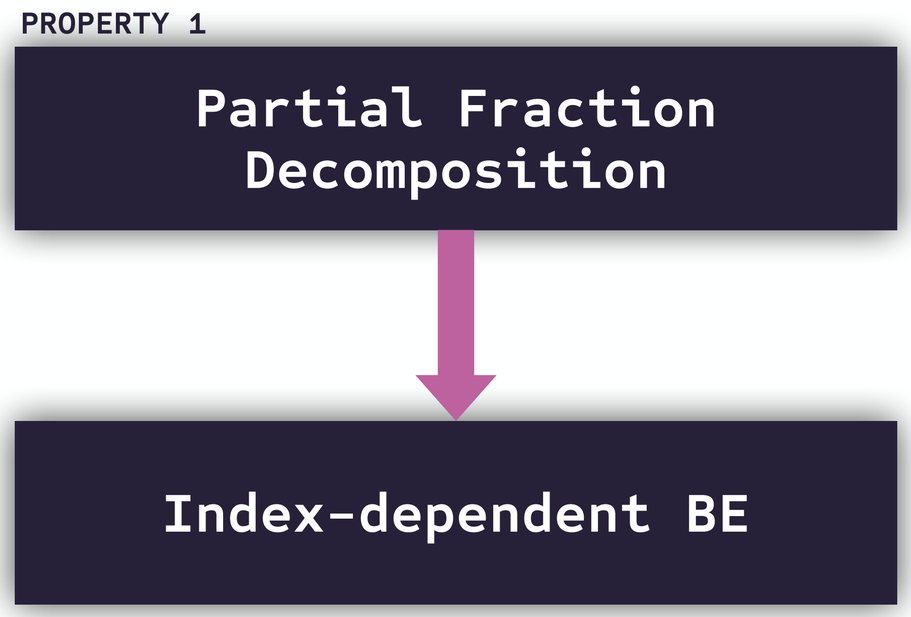
sorry, I lied...
Property 2 is coming later

Index-Dependent Batch Encryption

[Boneh-N-Roy-Tas26]

Property 1 PFD is sufficient to construct the weaker notion

sorry, I lied...
Property 2 is coming later



> **Main Idea**
> Use partial fraction to *one-time pad* the message. Pre-decryptor publishes another partial fraction using sk that recovers it.

Intuition

Poles	$-i$	$-j$	$-k$
$-i$	$\frac{1}{(X+i)^2}$	$\frac{1}{X+i} \cdot \frac{1}{X+j}$	$\frac{1}{X+i} \cdot \frac{1}{X+k}$
$-j$	$\frac{1}{X+j} \cdot \frac{1}{X+i}$	$\frac{1}{(X+j)^2}$	$\frac{1}{X+j} \cdot \frac{1}{X+k}$
$-k$	$\frac{1}{X+k} \cdot \frac{1}{X+i}$	$\frac{1}{X+k} \cdot \frac{1}{X+j}$	$\frac{1}{(X+k)^2}$

Intuition

All parties have *linear* fractions in $\mathbb{G}_2$:

$$\frac{1}{X+j} \ \forall j \in [\ell];$$

and *quadratic* fractions in $\mathbb{G}_T$:

$$\frac{1}{(X+j)^2} \ \forall j \in [\ell].$$

Poles	$-i$	$-j$	$-k$
$-i$	$\frac{1}{(X+i)^2}$	$\frac{1}{X+i} \cdot \frac{1}{X+j}$	$\frac{1}{X+i} \cdot \frac{1}{X+k}$
$-j$	$\frac{1}{X+j} \cdot \frac{1}{X+i}$	$\frac{1}{(X+j)^2}$	$\frac{1}{X+j} \cdot \frac{1}{X+k}$
$-k$	$\frac{1}{X+k} \cdot \frac{1}{X+i}$	$\frac{1}{X+k} \cdot \frac{1}{X+j}$	$\frac{1}{(X+k)^2}$

Intuition

All parties have *linear* fractions in $\mathbb{G}_2$:

$$\frac{1}{X+j} \ \forall j \in [\ell];$$

and *quadratic* fractions in $\mathbb{G}_T$:

$$\frac{1}{(X+j)^2} \ \forall j \in [\ell].$$

Poles	$-i$	$-j$	$-k$
$-i$	$\frac{1}{(X+i)^2}$	$\frac{1}{X+i} \cdot \frac{1}{X+j}$	$\frac{1}{X+i} \cdot \frac{1}{X+k}$
$-j$	$\frac{1}{X+j} \cdot \frac{1}{X+i}$	$\frac{1}{(X+j)^2}$	$\frac{1}{X+j} \cdot \frac{1}{X+k}$
$-k$	$\frac{1}{X+k} \cdot \frac{1}{X+i}$	$\frac{1}{X+k} \cdot \frac{1}{X+j}$	$\frac{1}{(X+k)^2}$

Ciphertext for index $i \in [\ell]$

One-time pad* message using *quadratic fraction *on the diagonal multiplied by a random value:*

$$\mathsf{ct}_i : m + \frac{r_i}{(X+i)^2}$$

Intuition

All parties have *linear* fractions in $\mathbb{G}_2$:

$$\frac{1}{X+j} \quad \forall j \in [\ell];$$

and *quadratic* fractions in $\mathbb{G}_T$:

$$\frac{1}{(X+j)^2} \quad \forall j \in [\ell].$$

Poles	$-i$	$-j$	$-k$
$-i$	$\frac{1}{(X+i)^2}$	$\frac{1}{X+i} \cdot \frac{1}{X+j}$	$\frac{1}{X+i} \cdot \frac{1}{X+k}$
$-j$	$\frac{1}{X+j} \cdot \frac{1}{X+i}$	$\frac{1}{(X+j)^2}$	$\frac{1}{X+j} \cdot \frac{1}{X+k}$
$-k$	$\frac{1}{X+k} \cdot \frac{1}{X+i}$	$\frac{1}{X+k} \cdot \frac{1}{X+j}$	$\frac{1}{(X+k)^2}$

Ciphertext for index $i \in [\ell]$

One-time pad* message using *quadratic fraction *on the diagonal multiplied by a random value:*

$$\mathsf{ct}_i : m + \frac{r_i}{(X+i)^2}$$

Intuition

All parties have *linear* fractions in $\mathbb{G}_2$:

$$\frac{1}{X+j} \ \forall j \in [\ell];$$

and *quadratic* fractions in $\mathbb{G}_T$:

$$\frac{1}{(X+j)^2} \ \forall j \in [\ell].$$

Poles	$-i$	$-j$	$-k$
$-i$	$\frac{1}{(X+i)^2}$	$\frac{1}{X+i} \cdot \frac{1}{X+j}$	$\frac{1}{X+i} \cdot \frac{1}{X+k}$
$-j$	$\frac{1}{X+j} \cdot \frac{1}{X+i}$	$\frac{1}{(X+j)^2}$	$\frac{1}{X+j} \cdot \frac{1}{X+k}$
$-k$	$\frac{1}{X+k} \cdot \frac{1}{X+i}$	$\frac{1}{X+k} \cdot \frac{1}{X+j}$	$\frac{1}{(X+k)^2}$

Ciphertext for index $i \in [\ell]$

One-time pad* message using *quadratic fraction *on the diagonal multiplied by a random value:*

$$\mathsf{ct}_i : m + \frac{r_i}{(X+i)^2}$$

Pre-decrypting batch

The pre-decryptor computes the pre-decryption key:

$$\mathsf{sbk} = \sum_{j \in [\ell]} \frac{r_j}{X+j}$$

Intuition

All parties have *linear* fractions in $\mathbb{G}_2$:

$$\frac{1}{X+j} \quad \forall j \in [\ell];$$

and *quadratic* fractions in $\mathbb{G}_T$:

$$\frac{1}{(X+j)^2} \quad \forall j \in [\ell].$$

Poles	$-i$	$-j$	$-k$
$-i$	$\frac{1}{(X+i)^2}$	$\frac{1}{X+i} \cdot \frac{1}{X+j}$	$\frac{1}{X+i} \cdot \frac{1}{X+k}$
$-j$	$\frac{1}{X+j} \cdot \frac{1}{X+i}$	$\frac{1}{(X+j)^2}$	$\frac{1}{X+j} \cdot \frac{1}{X+k}$
$-k$	$\frac{1}{X+k} \cdot \frac{1}{X+i}$	$\frac{1}{X+k} \cdot \frac{1}{X+j}$	$\frac{1}{(X+k)^2}$

Ciphertext for index $i \in [\ell]$

One-time pad* message using *quadratic fraction *on the diagonal multiplied by a random value:*

$$\mathsf{ct}_i : m + \frac{r_i}{(X+i)^2}$$

Pre-decrypting batch

The pre-decry
computes the
decryption ke

$$\mathsf{sbk} = \sum_{j \in [\ell]} \frac{r_j}{X+j}$$

r_j is used to tie ciphertext to index j

Intuition

All parties have *linear* fractions in $\mathbb{G}_2$:

$$\frac{1}{X+j} \ \forall j \in [\ell];$$

and *quadratic* fractions in $\mathbb{G}_T$:

$$\frac{1}{(X+j)^2} \ \forall j \in [\ell].$$

Poles	$-i$	$-j$	$-k$
$-i$	$\frac{1}{(X+i)^2}$	$\frac{1}{X+i} \cdot \frac{1}{X+j}$	$\frac{1}{X+i} \cdot \frac{1}{X+k}$
$-j$	$\frac{1}{X+j} \cdot \frac{1}{X+i}$	$\frac{1}{(X+j)^2}$	$\frac{1}{X+j} \cdot \frac{1}{X+k}$
$-k$	$\frac{1}{X+k} \cdot \frac{1}{X+i}$	$\frac{1}{X+k} \cdot \frac{1}{X+j}$	$\frac{1}{(X+k)^2}$

Ciphertext for index $i \in [\ell]$

One-time pad **message using** ***quadratic fraction*** *on the diagonal multiplied by a random value:*

$$\mathsf{ct}_i : m + \frac{r_i}{(X+i)^2}$$

Pre-decrypting batch

The pre-decryptor computes the pre-decryption key:

$$\mathsf{sbk} = \sum_{j \in [\ell]} \frac{r_j}{X+j}$$

Decryption for index $i \in [\ell]$

Compute $\mathsf{sbk} \cdot \left(\frac{1}{X+i}\right)$

$$= \frac{r_i}{(X+i)^2} + \sum_{j \neq i} \frac{r_i}{(X+j)} \cdot \frac{1}{(X+i)}$$

Intuition

All parties have *linear* fractions in $\mathbb{G}_2$:

$$\frac{1}{X+j} \ \forall j \in [\ell];$$

and *quadratic* fractions in $\mathbb{G}_T$:

$$\frac{1}{(X+j)^2} \ \forall j \in [\ell].$$

Poles	$-i$	$-j$	$-k$
$-i$	$\frac{1}{(X+i)^2}$	$\frac{1}{j-i}\left(\frac{1}{X+i}-\frac{1}{X+j}\right)$	$\frac{1}{k-i}\left(\frac{1}{X+i}-\frac{1}{X+k}\right)$
$-j$	$\frac{1}{j-i}\left(\frac{1}{X+i}-\frac{1}{X+j}\right)$	$\frac{1}{(X+j)^2}$	$\frac{1}{k-j}\left(\frac{1}{X+k}-\frac{1}{X+j}\right)$
$-k$	$\frac{1}{k-i}\left(\frac{1}{X+i}-\frac{1}{X+k}\right)$	$\frac{1}{k-j}\left(\frac{1}{X+k}-\frac{1}{X+j}\right)$	$\frac{1}{(X+k)^2}$

Ciphertext for index $i \in [\ell]$

One-time pad* message using *quadratic fraction *on the diagonal multiplied by a random value:*

$$m + \frac{r_i}{(X+i)^2}$$

Pre-decrypting batch

The pre-decryptor computes the pre-decryption key:

$$\mathsf{sbk} = \sum_{j \in [\ell]} \frac{r_j}{X+j}$$

Decryption for index $i \in [\ell]$

Compute $\mathsf{sbk} \cdot \left(\frac{1}{X+i}\right)$

$$= \frac{r_i}{(X+i)^2} + \sum_{j \neq i} \frac{r_i}{(X+j)} \cdot \frac{1}{(X+i)}$$

Intuition

All parties have *linear* fractions in $\mathbb{G}_2$:

$$\frac{1}{X+j} \quad \forall j \in [\ell];$$

and *quadratic* fractions in $\mathbb{G}_T$:

$$\frac{1}{(X+j)^2} \quad \forall j \in [\ell].$$

Poles	$-i$	$-j$	$-k$
$-i$	$\frac{1}{(X+i)^2}$	$\frac{1}{j-i}\left(\frac{1}{X+i}-\frac{1}{X+j}\right)$	$\frac{1}{k-i}\left(\frac{1}{X+i}-\frac{1}{X+k}\right)$
$-j$	$\frac{1}{j-i}\left(\frac{1}{X+i}-\frac{1}{X+j}\right)$	$\frac{1}{(X+j)^2}$	$\frac{1}{k-j}\left(\frac{1}{X+k}-\frac{1}{X+j}\right)$
$-k$	$\frac{1}{k-i}\left(\frac{1}{X+i}-\frac{1}{X+k}\right)$	$\frac{1}{k-j}\left(\frac{1}{X+k}-\frac{1}{X+j}\right)$	$\frac{1}{(X+k)^2}$

Ciphertext for index $i \in [\ell]$

One-time pad* message using *quadratic fraction *on the diagonal multiplied by a random value:*

$$m + \frac{r_i}{(X+i)^2}$$

Pre-decrypting batch

The pre-decryptor computes the pre-decryption key:

$$\mathsf{sbk} = \sum_{j \in [\ell]} \frac{r_j}{X+j}$$

Decryption for index $i \in [\ell]$

Compute $\mathsf{sbk} \cdot \left(\frac{1}{X+i}\right)$

$$= \frac{r_i}{(X+i)^2} + \sum_{j \neq i} \frac{r_i}{(X+j)} \cdot \frac{1}{(X+i)}$$

off-diagonal products linearize and can be subtracted out

Intuition

All parties have *linear* fractions in $\mathbb{G}_2$:

$$\frac{1}{X+j} \quad \forall j \in [\ell];$$

and *quadratic* fractions in $\mathbb{G}_T$:

$$\frac{1}{(X+j)^2} \quad \forall j \in [\ell].$$

Poles	$-i$	$-j$	$-k$
$-i$	$\frac{1}{(X+i)^2}$	$\frac{1}{j-i}\left(\frac{1}{X+i}-\frac{1}{X+j}\right)$	$\frac{1}{k-i}\left(\frac{1}{X+i}-\frac{1}{X+k}\right)$
$-j$	$\frac{1}{j-i}\left(\frac{1}{X+i}-\frac{1}{X+j}\right)$	$\frac{1}{(X+j)^2}$	$\frac{1}{k-j}\left(\frac{1}{X+k}-\frac{1}{X+j}\right)$
$-k$	$\frac{1}{k-i}\left(\frac{1}{X+i}-\frac{1}{X+k}\right)$	$\frac{1}{k-j}\left(\frac{1}{X+k}-\frac{1}{X+j}\right)$	$\frac{1}{(X+k)^2}$

Ciphertext for index $i \in [\ell]$

One-time pad* message using *quadratic fraction *on the diagonal multiplied by a random value:*

$$m + \frac{r_i}{(X+i)^2}$$

Pre-decrypting batch

The pre-decryptor computes the pre-decryption key:

$$\mathsf{sbk} = \sum_{j \in [\ell]} \frac{r_j}{X+j}$$

Decryption for index $i \in [\ell]$

Compute $\mathsf{sbk} \cdot \left(\frac{1}{X+i}\right)$

$$= \frac{r_i}{(X+i)^2} + \sum_{j \neq i} \frac{r_j}{(X+j)} \cdot \frac{1}{(X+i)}$$

Intuition

All parties have *linear* fractions in $\mathbb{G}_2$:

$$\frac{1}{X+j} \quad \forall j \in [\ell];$$

and *quadratic* fractions in $\mathbb{G}_T$:

$$\frac{1}{(X+j)^2} \quad \forall j \in [\ell].$$

Poles	$-i$	$-j$	$-k$
$-i$	$\frac{1}{(X+i)^2}$	$\frac{1}{j-i}\left(\frac{1}{X+i}-\frac{1}{X+j}\right)$	$\frac{1}{k-i}\left(\frac{1}{X+i}-\frac{1}{X+k}\right)$
$-j$	$\frac{1}{j-i}\left(\frac{1}{X+i}-\frac{1}{X+j}\right)$	$\frac{1}{(X+j)^2}$	$\frac{1}{k-j}\left(\frac{1}{X+k}-\frac{1}{X+j}\right)$
$-k$	$\frac{1}{k-i}\left(\frac{1}{X+i}-\frac{1}{X+k}\right)$	$\frac{1}{k-j}\left(\frac{1}{X+k}-\frac{1}{X+j}\right)$	$\frac{1}{(X+k)^2}$

Ciphertext for index $i \in [\ell]$

One-time pad* message using *quadratic fraction *on the diagonal multiplied by a random value:*

$$m + \frac{r_i}{(X+i)^2}$$

Pre-decrypting batch

The pre-decryptor computes the pre-decryption key:

$$\mathsf{sbk} = \sum_{j \in [\ell]} \frac{r_j}{X+j}$$

Decryption for index $i \in [\ell]$

Compute $\mathsf{sbk} \cdot \left(\frac{1}{X+i}\right)$

$$= \frac{r_i}{(X+i)^2} + \sum_{j \neq i} \frac{r_j}{(X+j)} \cdot \frac{1}{(X+i)}$$

and then you're just left with the one-time pad!

The Index-Dependent Construction

[Boneh-N-Roy-Tas26]

ℓ: **max block size**

Recall (asymmetric) bilinear group: $\mathcal{G} = \left(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e\right)$;

additive notation: $[x]_1 = x \cdot g_1, [y]_2 = y \cdot g_2,$ and $[z]_T = z \cdot e(g_1, g_2)$

The Index-Dependent Construction

[Boneh-N-Roy-Tas26]

ℓ: **max block size**

Recall (asymmetric) bilinear group: $\mathcal{G} = \left(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e\right)$;

additive notation: $[x]_1 = x \cdot g_1, [y]_2 = y \cdot g_2,$ and $[z]_T = z \cdot e(g_1, g_2)$

Setup(1^λ, batch size 1^ℓ)

Sample $x \xleftarrow{\$} \mathbb{Z}_p$, and output

$$\mathsf{sk} := x,$$

$$\mathsf{dk} := \text{linear fractions: } \left[\frac{1}{x+1}\right]_2, \ldots, \left[\frac{1}{x+\ell}\right]_2,$$

$$\mathsf{ek} := \text{quadratic fractions } \left[\frac{1}{(x+1)^2}\right]_T, \ldots, \left[\frac{1}{(x+\ell)^2}\right]_T$$

The Index-Dependent Construction

[Boneh-N-Roy-Tas26]

ℓ: **max block size**

Recall (asymmetric) bilinear group: $\mathcal{G} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e)$;

additive notation: $[x]_1 = x \cdot g_1, [y]_2 = y \cdot g_2$, and $[z]_T = z \cdot e(g_1, g_2)$

Setup(1^λ, batch size 1^ℓ)

Sample $x \xleftarrow{\$} \mathbb{Z}_p$, and output

$$\mathsf{sk} := x,$$

$$\mathsf{dk} := \text{linear fractions: } \left[\frac{1}{x+1}\right]_2, \ldots, \left[\frac{1}{x+\ell}\right]_2,$$

$$\mathsf{ek} := \text{quadratic fractions } \left[\frac{1}{(x+1)^2}\right]_T, \ldots, \left[\frac{1}{(x+\ell)^2}\right]_T$$

Enc(ek, index $i \in [\ell]$, msg $m \in \mathbb{G}_T$)

Sample $r \xleftarrow{\$} \mathbb{Z}_p$, and output

ciphertext with two components

(1) $[r]_1$ (2) $r\left[\frac{1}{(x+i)^2}\right]_T + m$

The Index-Dependent Construction

[Boneh-N-Roy-Tas26]

ℓ: **max block size**

Recall (asymmetric) bilinear group: $\mathcal{G} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e)$;

additive notation: $[x]_1 = x \cdot g_1, [y]_2 = y \cdot g_2$, and $[z]_T = z \cdot e(g_1, g_2)$

Setup(1^λ, batch size 1^ℓ)

Sample $x \xleftarrow{\$} \mathbb{Z}_p$, and output

$$\mathsf{sk} := x,$$

$$\mathsf{dk} := \text{linear fractions: } \left[\frac{1}{x+1}\right]_2, \ldots, \left[\frac{1}{x+\ell}\right]_2,$$

$$\mathsf{ek} := \text{quadratic fractions } \left[\frac{1}{(x+1)^2}\right]_T, \ldots, \left[\frac{1}{(x+\ell)^2}\right]_T$$

Enc(ek, index $i \in [\ell]$, msg $m \in \mathbb{G}_T$)

Sample $r \xleftarrow{\$} \mathbb{Z}_p$, and output

ciphertext with two components

(1) $[r]_1$ (2) $r\left[\frac{1}{(x+i)^2}\right]_T + m$

(3) NIZK proof of randomness

$\Pi.\mathsf{Prove}(x = (\mathsf{ct}[1], \mathsf{ct}[2]), w = r)$

The Index-Dependent Construction

[Boneh-N-Roy-Tas26]

ℓ: **max block size**

Recall (asymmetric) bilinear group: $\mathcal{G} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e)$;

additive notation: $[x]_1 = x \cdot g_1, [y]_2 = y \cdot g_2$, and $[z]_T = z \cdot e(g_1, g_2)$

Setup(1^λ, batch size 1^ℓ)

Sample $x \xleftarrow{\$} \mathbb{Z}_p$, and output

$$\mathsf{sk} := x,$$

$$\mathsf{dk} := \text{linear fractions: } \left[\frac{1}{x+1}\right]_2, \ldots, \left[\frac{1}{x+\ell}\right]_2,$$

$$\mathsf{ek} := \text{quadratic fractions } \left[\frac{1}{(x+1)^2}\right]_T, \ldots, \left[\frac{1}{(x+\ell)^2}\right]_T$$

Enc(ek, index $i \in [\ell]$, msg $m \in \mathbb{G}_T$)

Sample $r \xleftarrow{\$} \mathbb{Z}_p$, and output

ciphertext with two components

(1) $[r]_1$ (2) $r \left[\frac{1}{(x+i)}\right]$

needed for CCA-2 security

(3) NIZK proof of randomness

Π.Prove$(x = (\mathsf{ct}[1], \mathsf{ct}[2]), w = r)$

The Index-Dependent Construction

[Boneh-N-Roy-Tas26]

ℓ: **max block size**

Recall (asymmetric) bilinear group: $\mathcal{G} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e)$;

additive notation: $[x]_1 = x \cdot g_1, [y]_2 = y \cdot g_2$, and $[z]_T = z \cdot e(g_1, g_2)$

$$\mathsf{PreDec}(\mathsf{sk}, \{\mathsf{ct}_i\}_i)$$

Check NIZK proofs, and if valid, **output pre-decryption key as sum of partial fractions:**

$$\mathsf{sbk} := \sum_{i \in [\ell]} \frac{1}{x+i} \cdot \mathsf{ct}_i[1] = \sum_{i \in [\ell]} \left[\frac{r_i}{x+i} \right]_1$$

The Index-Dependent Construction

[Boneh-N-Roy-Tas26]

ℓ: **max block size**

Recall (asymmetric) bilinear group: $\mathcal{G} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e)$;

additive notation: $[x]_1 = x \cdot g_1, [y]_2 = y \cdot g_2,$ and $[z]_T = z \cdot e(g_1, g_2)$

$\mathsf{PreDec}(\mathsf{sk}, \{\mathsf{ct}_i\}_i)$

Check NIZK proofs, and if valid, **output pre-decryption key as sum of partial fractions:**

$$\mathsf{sbk} := \sum_{i \in [\ell]} \frac{1}{x+i} \cdot \mathsf{ct}_i[1] = \sum_{i \in [\ell]} \left[\frac{r_i}{x+i}\right]_1$$

$\mathsf{Dec}(\mathsf{dk}, \mathsf{sbk}, \{\mathsf{ct}_i\}_i)$

To recover m_j, compute:

$$e\left(\mathsf{sbk}, \left[\frac{1}{x+j}\right]_2\right) = e\left(\sum_{i \in [\ell]} \left[\frac{r_i}{x+i}\right]_1, \left[\frac{1}{x+j}\right]_2\right)$$

The Index-Dependent Construction

[Boneh-N-Roy-Tas26]

ℓ: **max block size**

Recall (asymmetric) bilinear group: $\mathcal{G} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e)$;

additive notation: $[x]_1 = x \cdot g_1, [y]_2 = y \cdot g_2$, and $[z]_T = z \cdot e(g_1, g_2)$

$\mathsf{PreDec}(\mathsf{sk}, \{\mathsf{ct}_i\}_i)$

Check NIZK proofs, and if valid, **output pre-decryption key as sum of partial fractions:**

$$\mathsf{sbk} := \sum_{i \in [\ell]} \frac{1}{x+i} \cdot \mathsf{ct}_i[1] = \sum_{i \in [\ell]} \left[\frac{r_i}{x+i}\right]_1$$

$\mathsf{Dec}(\mathsf{dk}, \mathsf{sbk}, \{\mathsf{ct}_i\}_i)$

To recover m_j, compute:

$$e\left(\mathsf{sbk}, \left[\frac{1}{x+j}\right]_2\right) = e\left(\sum_{i \in [\ell]} \left[\frac{r_i}{x+i}\right]_1, \left[\frac{1}{x+j}\right]_2\right)$$

$$= r_j \left[\frac{1}{(x+j)^2}\right]_T + r_j \sum_{i \neq j} \left[\frac{1}{(x+i)} \cdot \frac{1}{(x+j)}\right]_T$$

The Index-Dependent Construction

[Boneh-N-Roy-Tas26]

ℓ: **max block size**

Recall (asymmetric) bilinear group: $\mathcal{G} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e)$;

additive notation: $[x]_1 = x \cdot g_1, [y]_2 = y \cdot g_2$, and $[z]_T = z \cdot e(g_1, g_2)$

$\mathsf{PreDec}(\mathsf{sk}, \{\mathsf{ct}_i\}_i)$

Check NIZK proofs, and if valid, **output pre-decryption key as sum of partial fractions:**

$$\mathsf{sbk} := \sum_{i \in [\ell]} \frac{1}{x+i} \cdot \mathsf{ct}_i[1] = \sum_{i \in [\ell]} \left[\frac{r_i}{x+i}\right]_1$$

$\mathsf{Dec}(\mathsf{dk}, \mathsf{sbk}, \{\mathsf{ct}_i\}_i)$

To recover m_j, compute:

$$e\left(\mathsf{sbk}, \left[\frac{1}{x+j}\right]_2\right) = e\left(\sum_{i \in [\ell]} \left[\frac{r_i}{x+i}\right]_1, \left[\frac{1}{x+j}\right]_2\right)$$

$$= r_j \left[\frac{1}{(x+j)^2}\right]_T + r_j \sum_{i \neq j} \left[\frac{1}{(x+i)} \cdot \frac{1}{(x+j)}\right]_T$$

$$= r_j \left[\frac{1}{(x+j)^2}\right]_T + \sum_{i \neq j} \frac{1}{j-i} \cdot e\left([r_j]_1, \left[\frac{1}{(x+i)}\right]_2 - \left[\frac{1}{(x+j)}\right]_2\right)$$

The Index-Dependent Construction

[Boneh-N-Roy-Tas26]

ℓ: **max block size**

Recall (asymmetric) bilinear group: $\mathcal{G} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e)$;

additive notation: $[x]_1 = x \cdot g_1, [y]_2 = y \cdot g_2$, and $[z]_T = z \cdot e(g_1, g_2)$

$\mathsf{PreDec}(\mathsf{sk}, \{\mathsf{ct}_i\}_i)$

Check NIZK proofs, and if valid, **output pre-decryption key as sum of partial fractions:**

$$\mathsf{sbk} := \sum_{i \in [\ell]} \frac{1}{x+i} \cdot \mathsf{ct}_i[1] = \sum_{i \in [\ell]} \left[\frac{r_i}{x+i}\right]_1$$

$\mathsf{Dec}(\mathsf{dk}, \mathsf{sbk}, \{\mathsf{ct}_i\}_i)$

To recover m_j, compute:

$$e\left(\mathsf{sbk}, \left[\frac{1}{x+j}\right]_2\right) = e\left(\sum_{i \in [\ell]} \left[\frac{r_i}{x+i}\right]_1, \left[\frac{1}{x+j}\right]_2\right)$$

$$= r_j \left[\frac{1}{(x+j)^2}\right]_T + r_j \sum_{i \neq j} \left[\frac{1}{(x+i)} \cdot \frac{1}{(x+j)}\right]_T$$

$$= r_j \left[\frac{1}{(x+j)^2}\right]_T + \sum_{i \neq j} \frac{1}{j-i} \cdot e\left([r_j]_1, \left[\frac{1}{(x+i)}\right]_2 - \left[\frac{1}{(x+j)}\right]_2\right)$$

subtract out linear terms

The Index-Dependent Construction

[Boneh-N-Roy-Tas26]

ℓ: **max block size**

Recall (asymmetric) bilinear group: $\mathcal{G} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e)$;

additive notation: $[x]_1 = x \cdot g_1, [y]_2 = y \cdot g_2$, and $[z]_T = z \cdot e(g_1, g_2)$

$\mathsf{PreDec}(\mathsf{sk}, \{\mathsf{ct}_i\}_i)$

Check NIZK proofs, and if valid, **output pre-decryption key as sum of partial fractions:**

$$\mathsf{sbk} := \sum_{i \in [\ell]} \frac{1}{x+i} \cdot \mathsf{ct}_i[1] = \sum_{i \in [\ell]} \left[\frac{r_i}{x+i}\right]_1$$

$\mathsf{Dec}(\mathsf{dk}, \mathsf{sbk}, \{\mathsf{ct}_i\}_i)$

To recover m_j, compute:

$$e\left(\mathsf{sbk}, \left[\frac{1}{x+j}\right]_2\right) = e\left(\sum_{i \in [\ell]} \left[\frac{r_i}{x+i}\right]_1, \left[\frac{1}{x+j}\right]_2\right)$$

$$= r_j \left[\frac{1}{(x+j)^2}\right]_T + r_j \sum_{i \neq j} \left[\frac{1}{(x+i)} \cdot \frac{1}{(x+j)}\right]_T$$

$$= r_j \left[\frac{1}{(x+j)^2}\right]_T + \sum_{i \neq j} \frac{1}{j-i} \cdot e\left([r_j]_1, \left[\frac{1}{(x+i)}\right]_2 - \left[\frac{1}{(x+j)}\right]_2\right)$$

subtract out linear terms

That was a lot of algebra for a presentation…

That was a lot of algebra for a presentation…

This is a good breakpoint!

That was a lot of algebra for a presentation…

This is a good breakpoint!

If you stop listening here, just remember that **we can upgrade the construction and remove the index-dependent relaxation** using techniques used to construct Broadcast/Threshold Encryption in our prior work [JNR26].

That was a lot of algebra for a presentation…

This is a good breakpoint!

If you stop listening here, just remember that **we can upgrade the construction and remove the index-dependent relaxation** using techniques used to construct Broadcast/Threshold Encryption in our prior work [JNR26].

But, if you have appetite for more, I’m going to go on and tell you about the other really cool, less obvious property.

Upgrading The Construction

Upgrading The Construction

The index-dependent construction uses the quadratic fraction explicitly to one-time pad the message

Upgrading The Construction

The index-dependent construction uses the quadratic fraction explicitly to one-time pad the message

You need to know to an index in advance to do so

Upgrading The Construction

The index-dependent construction uses the quadratic fraction explicitly to one-time pad the message

You need to know to an index in advance to do so

$$r\left[\frac{1}{(x+i)^2}\right]_T + m$$

Upgrading The Construction

The index-dependent construction uses the quadratic fraction explicitly to one-time pad the message

You need to know to an index in advance to do so

$$r\left[\frac{1}{(x+i)^2}\right]_T + m$$

Can we use a single fraction *independent* of the index?

Upgrading The Construction

The index-dependent construction uses the quadratic fraction explicitly to one-time pad the message

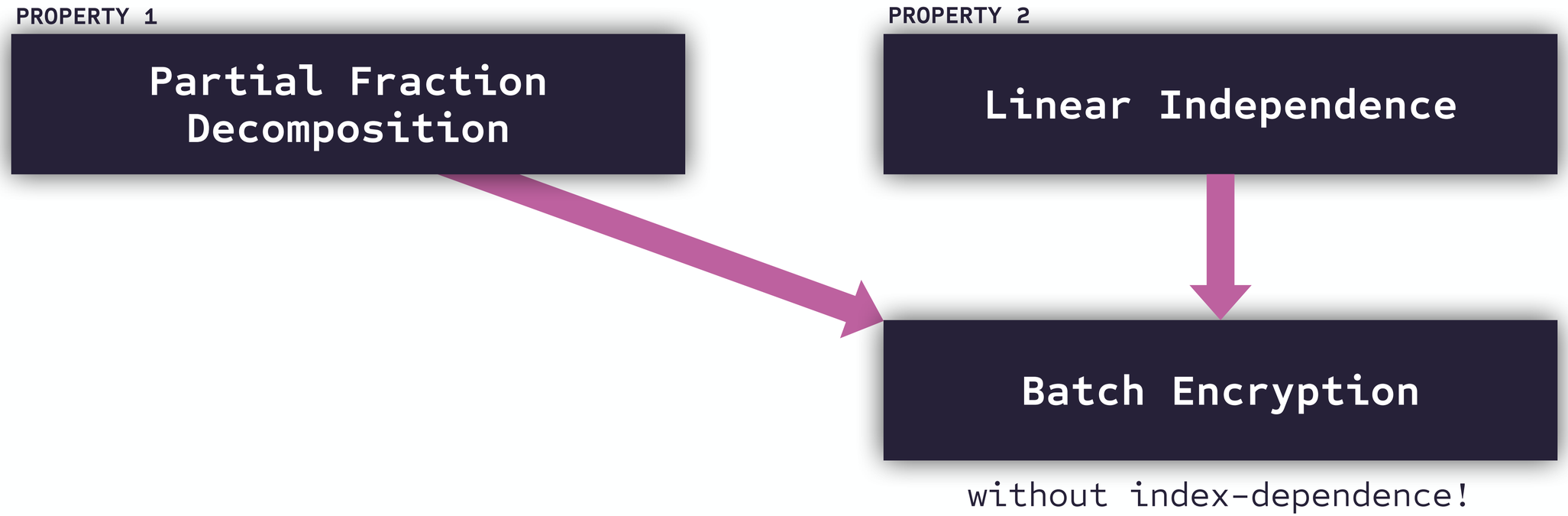
You need to know to an index in advance to do so

$$r\left[\frac{1}{(x+i)^2}\right]_T + m$$

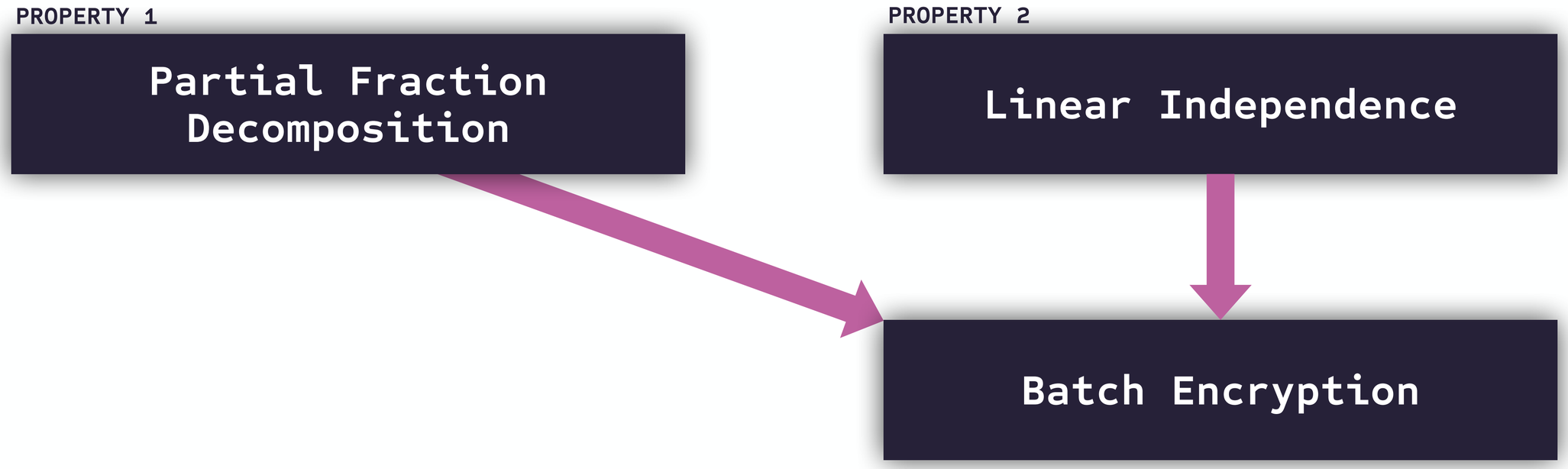
Can we use a single fraction *independent* of the index?

> **Yes, we can!**
> **But we're going to need another property of these fractions**

Property 2: Linear Independence



Property 2: Linear Independence



without index-dependence!

the second property has finally arrived...
I hope you're not half-asleep

Property 2: Linear Independence

Property 2: Linear Independence

Earlier we saw, partial fraction products can be decomposed

$$\frac{1}{X+i} \cdot \frac{1}{X+j} = \frac{1}{j-i}\left(\frac{1}{X+i} - \frac{1}{X+j}\right)$$

Property 2: Linear Independence

Earlier we saw, partial fraction products can be decomposed

$$\frac{1}{X+i} \cdot \frac{1}{X+j} = \frac{1}{j-i}\left(\frac{1}{X+i} - \frac{1}{X+j}\right)$$

The decomposition theorem can be used to derive *linearly-independent* relations.

Property 2: Linear Independence

Earlier we saw, partial fraction products can be decomposed

$$\frac{1}{X+i} \cdot \frac{1}{X+j} = \frac{1}{j-i}\left(\frac{1}{X+i} - \frac{1}{X+j}\right)$$

The decomposition theorem can be used to derive *linearly-independent* relations.

Suppose you start with a simple relation: for $i \neq j$,

Property 2: Linear Independence

Earlier we saw, partial fraction products can be decomposed

$$\frac{1}{X+i} \cdot \frac{1}{X+j} = \frac{1}{j-i}\left(\frac{1}{X+i} - \frac{1}{X+j}\right)$$

The decomposition theorem can be used to derive *linearly-independent* relations.

Suppose you start with a simple relation: for $i \neq j$,

$$\left(\frac{1}{X+i} + \frac{1}{X+j}\right)$$

Property 2: Linear Independence

Earlier we saw, partial fraction products can be decomposed

$$\frac{1}{X+i} \cdot \frac{1}{X+j} = \frac{1}{j-i}\left(\frac{1}{X+i} - \frac{1}{X+j}\right)$$

The decomposition theorem can be used to derive *linearly-independent* relations.

Suppose you start with a simple relation: for $i \neq j$, $(k \neq i, j)$

$$\left(\frac{1}{X+i} + \frac{1}{X+j}\right)\frac{1}{X+k}$$

and then multiply it with another fraction...

Property 2: Linear Independence

Earlier we saw, partial fraction products can be decomposed

$$\frac{1}{X+i}\cdot\frac{1}{X+j}=\frac{1}{j-i}\left(\frac{1}{X+i}-\frac{1}{X+j}\right)$$

The decomposition theorem can be used to derive *linearly-independent* relations.

Suppose you start with a simple relation: for $i \neq j$, $(k \neq i, j)$

$$\left(\frac{1}{X+i}+\frac{1}{X+j}\right)\frac{1}{X+k}=\frac{1}{k-i}\left(\frac{1}{X+i}-\frac{1}{X+k}\right)$$

and then multiply it
with another
fraction...

Property 2: Linear Independence

Earlier we saw, partial fraction products can be decomposed

$$\frac{1}{X+i}\cdot\frac{1}{X+j}=\frac{1}{j-i}\left(\frac{1}{X+i}-\frac{1}{X+j}\right)$$

The decomposition theorem can be used to derive *linearly-independent* relations.

Suppose you start with a simple relation: for $i \neq j$, $(k \neq i, j)$

$$\left(\frac{1}{X+i}+\frac{1}{X+j}\right)\frac{1}{X+k}=\frac{1}{k-i}\left(\frac{1}{X+i}-\frac{1}{X+k}\right)+\frac{1}{k-j}\left(\frac{1}{X+j}-\frac{1}{X+k}\right)$$

and then multiply it with another fraction...

Property 2: Linear Independence

Earlier we saw, partial fraction products can be decomposed

$$\frac{1}{X+i}\cdot\frac{1}{X+j}=\frac{1}{j-i}\left(\frac{1}{X+i}-\frac{1}{X+j}\right)$$

The decomposition theorem can be used to derive *linearly-independent* relations.

Suppose you start with a simple relation: for $i \neq j$, $(k \neq i, j)$

$$\left(\frac{1}{X+i}+\frac{1}{X+j}\right)\frac{1}{X+k}=\frac{1}{k-i}\left(\frac{1}{X+i}-\frac{1}{X+k}\right)+\frac{1}{k-j}\left(\frac{1}{X+j}-\frac{1}{X+k}\right)$$

and then multiply it with another fraction...

$$=\frac{1}{k-i}\cdot\frac{1}{X+i}+\frac{1}{k-j}\cdot\frac{1}{X+j}-\left(\frac{1}{k-i}+\frac{1}{k-j}\right)\frac{1}{X+k}$$

Property 2: Linear Independence

$$\left(\frac{1}{X+i}+\frac{1}{X+j}\right)\frac{1}{X+k}=\frac{1}{k-i}\cdot\frac{1}{X+i}+\frac{1}{k-j}\cdot\frac{1}{X+j}-\left(\frac{1}{k-i}+\frac{1}{k-j}\right)\frac{1}{X+k}$$

Property 2: Linear Independence

$$\left(\frac{1}{X+i}+\frac{1}{X+j}\right)\frac{1}{X+k}=\frac{1}{k-i}\cdot\frac{1}{X+i}+\frac{1}{k-j}\cdot\frac{1}{X+j}-\left(\frac{1}{k-i}+\frac{1}{k-j}\right)\frac{1}{X+k}$$



Property 2: Linear Independence

$$\left(\frac{1}{X+i}+\frac{1}{X+j}\right)\frac{1}{X+k}=\frac{1}{k-i}\cdot\frac{1}{X+i}+\frac{1}{k-j}\cdot\frac{1}{X+j}-\left(\frac{1}{k-i}+\frac{1}{k-j}\right)\frac{1}{X+k}$$

$$\left(\frac{1}{X+i}+\frac{1}{X+j}\right)\frac{1}{X+k}+\left(\frac{1}{k-i}+\frac{1}{k-j}\right)\frac{1}{X+k}=\frac{1}{k-i}\cdot\frac{1}{X+i}+\frac{1}{k-j}\cdot\frac{1}{X+j}$$

Property 2: Linear Independence

$$\left(\frac{1}{X+i}+\frac{1}{X+j}\right)\frac{1}{X+k}=\frac{1}{k-i}\cdot\frac{1}{X+i}+\frac{1}{k-j}\cdot\frac{1}{X+j}-\left(\frac{1}{k-i}+\frac{1}{k-j}\right)\frac{1}{X+k}$$

$$\left(\frac{1}{X+i}+\frac{1}{X+j}\right)\frac{1}{X+k}+\left(\frac{1}{k-i}+\frac{1}{k-j}\right)\frac{1}{X+k}=\frac{1}{k-i}\cdot\frac{1}{X+i}+\frac{1}{k-j}\cdot\frac{1}{X+j}$$

Property 2: Linear Independence

$$\left(\frac{1}{X+i}+\frac{1}{X+j}\right)\frac{1}{X+k}=\frac{1}{k-i}\cdot\frac{1}{X+i}+\frac{1}{k-j}\cdot\frac{1}{X+j}-\left(\frac{1}{k-i}+\frac{1}{k-j}\right)\frac{1}{X+k}$$

$$\left(\frac{1}{X+i}+\frac{1}{X+j}\right)\frac{1}{X+k}+\left(\frac{1}{k-i}+\frac{1}{k-j}\right)\frac{1}{X+k}=\frac{1}{k-i}\cdot\frac{1}{X+i}+\frac{1}{k-j}\cdot\frac{1}{X+j}$$

Property 2: Linear Independence

$$\left(\frac{1}{X+i}+\frac{1}{X+j}\right)\frac{1}{X+k}=\frac{1}{k-i}\cdot\frac{1}{X+i}+\frac{1}{k-j}\cdot\frac{1}{X+j}-\left(\frac{1}{k-i}+\frac{1}{k-j}\right)\frac{1}{X+k}$$

$$\left(\frac{1}{X+i}+\frac{1}{X+j}\right)\frac{1}{X+k}+\left(\frac{1}{k-i}+\frac{1}{k-j}\right)\frac{1}{X+k}=\frac{1}{k-i}\cdot\frac{1}{X+i}+\frac{1}{k-j}\cdot\frac{1}{X+j}$$

(1) $\dfrac{1}{X+i}+\dfrac{1}{X+j}$

(2) $\dfrac{1}{k-i}\cdot\dfrac{1}{X+i}+\dfrac{1}{k-j}\cdot\dfrac{1}{X+j}$

Property 2: Linear Independence

$$\left(\frac{1}{X+i}+\frac{1}{X+j}\right)\frac{1}{X+k}=\frac{1}{k-i}\cdot\frac{1}{X+i}+\frac{1}{k-j}\cdot\frac{1}{X+j}-\left(\frac{1}{k-i}+\frac{1}{k-j}\right)\frac{1}{X+k}$$

$$\left(\frac{1}{X+i}+\frac{1}{X+j}\right)\frac{1}{X+k}+\left(\frac{1}{k-i}+\frac{1}{k-j}\right)\frac{1}{X+k}=\frac{1}{k-i}\cdot\frac{1}{X+i}+\frac{1}{k-j}\cdot\frac{1}{X+j}$$

(1) $\dfrac{1}{X+i}+\dfrac{1}{X+j}$

(2) $\dfrac{1}{k-i}\cdot\dfrac{1}{X+i}+\dfrac{1}{k-j}\cdot\dfrac{1}{X+j}$

(1) and (2) are $\mathbb{F}$-linearly independent

Property 2: Linear Independence

$$\left(\frac{1}{X+i}+\frac{1}{X+j}\right)\frac{1}{X+k}=\frac{1}{k-i}\cdot\frac{1}{X+i}+\frac{1}{k-j}\cdot\frac{1}{X+j}-\left(\frac{1}{k-i}+\frac{1}{k-j}\right)\frac{1}{X+k}$$

$$\left(\frac{1}{X+i}+\frac{1}{X+j}\right)\frac{1}{X+k}+\left(\frac{1}{k-i}+\frac{1}{k-j}\right)\frac{1}{X+k}=\frac{1}{k-i}\cdot\frac{1}{X+i}+\frac{1}{k-j}\cdot\frac{1}{X+j}$$

(1) $\dfrac{1}{X+i}+\dfrac{1}{X+j}$

(2) $\dfrac{1}{k-i}\cdot\dfrac{1}{X+i}+\dfrac{1}{k-j}\cdot\dfrac{1}{X+j}$

(1) and (2) are $\mathbb{F}$-linearly independent

follows from the invertibility of *Cauchy Matrices*

Property 2: Linear Independence

Property 2: Linear Independence

Takeaway

Taking a product of a relation with distinct fractions yields a linearly independent relation.

Property 2: Linear Independence

> **Takeaway**
> **Taking a product of a relation with distinct fractions yields a linearly independent relation.**

Embed a linear relation over all batch positions and the one-time pad in the ciphertext.

Property 2: Linear Independence

> **Takeaway**
> **Taking a product of a relation with distinct fractions yields a linearly independent relation.**

Embed a linear relation over all batch positions and the one-time pad in the ciphertext.

Pre-decryption and Decryption **involves deriving another relation** and **solving the system to recover the one-time pad**

The Final Construction

[Boneh-**N**-Roy-Tas26]

ℓ: **max block size**

Recall (asymmetric) bilinear group: $\mathcal{G} = \left(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e\right)$;

additive notation: $[x]_1 = x \cdot g_1, [y]_2 = y \cdot g_2$, and $[z]_T = z \cdot e(g_1, g_2)$

The Final Construction

[Boneh-N-Roy-Tas26]

ℓ: **max block size**

Recall (asymmetric) bilinear group: $\mathcal{G} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e)$;

additive notation: $[x]_1 = x \cdot g_1, [y]_2 = y \cdot g_2$, and $[z]_T = z \cdot e(g_1, g_2)$

Setup(1^λ, batch size 1^ℓ)

The Final Construction

[Boneh-N-Roy-Tas26]

ℓ: **max block size**

Recall (asymmetric) bilinear group: $\mathcal{G} = \left(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e\right)$;

additive notation: $[x]_1 = x \cdot g_1, [y]_2 = y \cdot g_2,$ and $[z]_T = z \cdot e(g_1, g_2)$

$\mathsf{Setup}(1^\lambda,$ batch size $1^\ell)$

Sample $x \xleftarrow{\$} \mathbb{Z}_p,$ and output

$\mathsf{sk} := x,$

The Final Construction

[Boneh-N-Roy-Tas26]

ℓ: **max block size**

Recall (asymmetric) bilinear group: $\mathcal{G} = \left(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e\right)$;

additive notation: $[x]_1 = x \cdot g_1, [y]_2 = y \cdot g_2,$ and $[z]_T = z \cdot e(g_1, g_2)$

$$\mathsf{Setup}(1^\lambda,\ \text{batch size } 1^\ell)$$

Sample $x \xleftarrow{\$} \mathbb{Z}_p$, and output

$$\mathsf{sk} := x,$$

$$\mathsf{dk} := \text{linear fractions: } \left[\frac{1}{x+1}\right]_2, \ldots, \left[\frac{1}{x+\ell}\right]_2,$$

The Final Construction

[Boneh-N-Roy-Tas26]

ℓ: **max block size**

Recall (asymmetric) bilinear group: $\mathcal{G} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e)$;

additive notation: $[x]_1 = x \cdot g_1, [y]_2 = y \cdot g_2$, and $[z]_T = z \cdot e(g_1, g_2)$

$$\mathsf{Setup}(1^\lambda, \text{ batch size } 1^\ell)$$

Sample $x \xleftarrow{\$} \mathbb{Z}_p$, and output

$$\mathsf{sk} := x,$$

$$\mathsf{dk} := \text{ linear fractions: } \left[\frac{1}{x+1}\right]_2, \ldots, \left[\frac{1}{x+\ell}\right]_2,$$

$$\mathsf{ek} := \text{linear fractions } \left[\frac{1}{x-1}\right]_1, \left[\frac{1}{x}\right]_1 \left[\frac{1}{x+1}\right]_1, \ldots, \left[\frac{1}{x+\ell}\right]_1$$

The Final Construction

[Boneh-N-Roy-Tas26]

ℓ: **max block size**

Recall (asymmetric) bilinear group: $\mathcal{G} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e)$;

additive notation: $[x]_1 = x \cdot g_1, [y]_2 = y \cdot g_2$, and $[z]_T = z \cdot e(g_1, g_2)$

$$\mathsf{Setup}(1^\lambda, \text{ batch size } 1^\ell)$$

Sample $x \xleftarrow{\$} \mathbb{Z}_p$, and output

$$\mathsf{sk} := x,$$

$$\mathsf{dk} := \text{ linear fractions: } \left[\frac{1}{x+1}\right]_2, \ldots, \left[\frac{1}{x+\ell}\right]_2,$$

$$\mathsf{ek} := \text{linear fractions } \left[\frac{1}{x-1}\right]_1, \left[\frac{1}{x}\right]_1 \left[\frac{1}{x+1}\right]_1, \ldots, \left[\frac{1}{x+\ell}\right]_1$$

no longer need quadratics

The Final Construction

[Boneh-N-Roy-Tas26]

ℓ: **max block size**

Recall (asymmetric) bilinear group: $\mathcal{G} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e)$;

additive notation: $[x]_1 = x \cdot g_1, [y]_2 = y \cdot g_2$, and $[z]_T = z \cdot e(g_1, g_2)$

$\mathsf{Setup}(1^\lambda,\ \text{batch size } 1^\ell)$

Sample $x \xleftarrow{\$} \mathbb{Z}_p$, and output

$\mathsf{sk} := x,$

$\mathsf{dk} :=$ linear fractions: $\left[\frac{1}{x+1}\right]_2, \ldots, \left[\frac{1}{x+\ell}\right]_2,$

$\mathsf{ek} :=$ linear fractions $\left[\frac{1}{x-1}\right]_1, \left[\frac{1}{x}\right]_1 \left[\frac{1}{x+1}\right]_1, \ldots, \left[\frac{1}{x+\ell}\right]_1$

new partial fractions

no longer need quadratics

The Final Construction

[Boneh-**N**-Roy-Tas26]

ℓ: **max block size**

Recall (asymmetric) bilinear group: $\mathcal{G} = \left(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e\right)$;

additive notation: $[x]_1 = x \cdot g_1, [y]_2 = y \cdot g_2$, and $[z]_T = z \cdot e(g_1, g_2)$

The Final Construction

[Boneh-N-Roy-Tas26]

ℓ: **max block size**

Recall (asymmetric) bilinear group: $\mathcal{G} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e)$;

additive notation: $[x]_1 = x \cdot g_1, [y]_2 = y \cdot g_2$, and $[z]_T = z \cdot e(g_1, g_2)$

Enc(ek, msg $m \in \mathbb{G}_T$) $\rightarrow$ ct

The Final Construction

[Boneh-N-Roy-Tas26]

ℓ: **max block size**

Recall (asymmetric) bilinear group: $\mathcal{G} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e)$;

additive notation: $[x]_1 = x \cdot g_1, [y]_2 = y \cdot g_2$, and $[z]_T = z \cdot e(g_1, g_2)$

Enc(ek, msg $m \in \mathbb{G}_T$) $\rightarrow$ ct

Sample $r \xleftarrow{\$} \mathbb{Z}_p$, and output

ciphertext with **three** components

The Final Construction

[Boneh-N-Roy-Tas26]

ℓ: **max block size**

Recall (asymmetric) bilinear group: $\mathcal{G} = \left(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e\right)$;

additive notation: $[x]_1 = x \cdot g_1, [y]_2 = y \cdot g_2,$ and $[z]_T = z \cdot e(g_1, g_2)$

$$\mathsf{Enc}(\mathsf{ek},\ \mathtt{msg}\ m \in \mathbb{G}_T) \to \mathsf{ct}$$

Sample $r \xleftarrow{\$} \mathbb{Z}_p$, and output

ciphertext with **three** components

(1) $[r]_1$

The Final Construction

[Boneh-N-Roy-Tas26]

ℓ: **max block size**

Recall (asymmetric) bilinear group: $\mathcal{G} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e)$;

additive notation: $[x]_1 = x \cdot g_1, [y]_2 = y \cdot g_2$, and $[z]_T = z \cdot e(g_1, g_2)$

$$\mathsf{Enc}(\mathsf{ek},\ \mathtt{msg}\ m \in \mathbb{G}_T) \to \mathsf{ct}$$

Sample $r \xleftarrow{\$} \mathbb{Z}_p$, and output

ciphertext with **three** components

(1) $[r]_1$

(2) $r \left[\frac{1}{x}\right]_T + m$

The Final Construction

[Boneh-N-Roy-Tas26]

ℓ: **max block size**

Recall (asymmetric) bilinear group: $\mathcal{G} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e)$;

additive notation: $[x]_1 = x \cdot g_1, [y]_2 = y \cdot g_2$, and $[z]_T = z \cdot e(g_1, g_2)$

$$\text{Enc(ek, msg } m \in \mathbb{G}_T) \to \text{ct}$$

Sample $r \xleftarrow{\$} \mathbb{Z}_p$, and output

ciphertext with **three** components

(1) $[r]_1$

(2) $r \left[\frac{1}{x}\right]_T + m$

replace quadratic with
linear fraction one-time pad

The Final Construction

[Boneh-**N**-Roy-Tas26]

ℓ: **max block size**

Recall (asymmetric) bilinear group: $\mathcal{G} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e)$;

additive notation: $[x]_1 = x \cdot g_1, [y]_2 = y \cdot g_2$, and $[z]_T = z \cdot e(g_1, g_2)$

$$\mathsf{Enc}(\mathsf{ek},\ \mathtt{msg}\ m \in \mathbb{G}_T) \to \mathsf{ct}$$

Sample $r \xleftarrow{\$} \mathbb{Z}_p$, and output

ciphertext with **three** components

(1) $[r]_1$ (2) $r\left[\frac{1}{x}\right]_T + m$ (3) $r\left(\sum_{i=-1}^{\ell}\left[\frac{1}{x+i}\right]_1\right)$

replace quadratic with
linear fraction one-time pad

The Final Construction

[Boneh-N-Roy-Tas26]

ℓ: **max block size**

Recall (asymmetric) bilinear group: $\mathcal{G} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e)$;

additive notation: $[x]_1 = x \cdot g_1, [y]_2 = y \cdot g_2$, and $[z]_T = z \cdot e(g_1, g_2)$

$$\text{Enc(ek, msg } m \in \mathbb{G}_T) \to \text{ct}$$

Sample $r \xleftarrow{\$} \mathbb{Z}_p$, and output

ciphertext with **three** components

(1) $[r]_1$

(2) $r\left[\frac{1}{x}\right]_T + m$

(3) $r\left(\sum_{i=-1}^{\ell}\left[\frac{1}{x+i}\right]_1\right)$

replace quadratic with **linear fraction one-time pad**

publish linear relation over fractions

The Final Construction

[Boneh-N-Roy-Tas26]

ℓ: **max block size**

Recall (asymmetric) bilinear group: $\mathcal{G} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e)$;

additive notation: $[x]_1 = x \cdot g_1, [y]_2 = y \cdot g_2$, and $[z]_T = z \cdot e(g_1, g_2)$

$$\text{Enc}(\text{ek}, \ \text{msg} \ m \in \mathbb{G}_T) \to \text{ct}$$

Sample $r \xleftarrow{\$} \mathbb{Z}_p$, and output

ciphertext with **three** components

(1) $[r]_1$

(2) $r\left[\frac{1}{x}\right]_T + m$

(3) $r\left(\sum_{i=-1}^{\ell}\left[\frac{1}{x+i}\right]_1\right)$

replace quadratic with **linear fraction one-time pad**

publish linear relation over fractions

Can be upgraded to CCA-2 using NIZK in the same way as before

Decryption Using Linear Independence

$$\mathsf{ct} = \quad (1)\ [r]_1 \qquad (2)\ r\left[\frac{1}{x}\right]_T + m \qquad (3)\ r\left(\sum_{i=-1}^{\ell}\left[\frac{1}{x+i}\right]_1\right)$$

Decryption Using Linear Independence

$$\mathsf{ct} = \quad (1)\ [r]_1 \qquad (2)\ r\left[\frac{1}{x}\right]_T + m \qquad (3)\ r\left(\sum_{i=-1}^{\ell}\left[\frac{1}{x+i}\right]_1\right)$$

Let’s try to derive another relation from the ciphertext:

Decryption Using Linear Independence

$$\mathsf{ct} = \quad (1)\ [r]_1 \qquad (2)\ r\left[\frac{1}{x}\right]_T + m \qquad (3)\ r\left(\sum_{i=-1}^{\ell}\left[\frac{1}{x+i}\right]_1\right)$$

Let’s try to derive another relation from the ciphertext:

$$\mathsf{dk} := \text{ linear fractions: } \left[\frac{1}{x+1}\right]_2, \ldots, \left[\frac{1}{x+\ell}\right]_2$$

Decryption Using Linear Independence

$$\mathsf{ct} = \quad (1)\ [r]_1 \qquad (2)\ r\left[\frac{1}{x}\right]_T + m \qquad (3)\ r\left(\sum_{i=-1}^{\ell}\left[\frac{1}{x+i}\right]_1\right)$$

Let’s try to derive another relation from the ciphertext:

$$\mathsf{dk} := \text{ linear fractions: } \left[\frac{1}{x+1}\right]_2, \ldots, \left[\frac{1}{x+\ell}\right]_2$$

Multiply fraction in dk with the linear relation in ct

Decryption Using Linear Independence

$$\mathsf{ct} = \quad (1)\ [r]_1 \qquad (2)\ r\left[\frac{1}{x}\right]_T + m$$

Let’s try to derive another relation from the ciphertext:

$$\mathsf{dk} := \text{ linear fractions: } \left[\frac{1}{x+1}\right]_2, \ldots, \left[\frac{1}{x+\ell}\right]_2$$

Multiply fraction in dk with the linear relation in ct

$$r\left(\sum_{i=-1}^{\ell}\left[\frac{1}{x+i}\right]_1\right)$$

Decryption Using Linear Independence

$$\mathsf{ct} = \quad (1)\ [r]_1 \qquad\qquad (2)\ r\left[\frac{1}{x}\right]_T + m$$

Let’s try to derive another relation from the ciphertext:

$$\mathsf{dk} := \ \texttt{linear fractions:}\ \left[\frac{1}{x+1}\right]_2, \ldots, \left[\frac{1}{x+\ell}\right]_2$$

Multiply fraction in dk with the linear relation in ct

$$e\left(r\left(\sum_{i=-1}^{\ell}\left[\frac{1}{x+i}\right]_1\right),\ \left[\frac{1}{x+j}\right]_2\right)$$

Decryption Using Linear Independence

$$\mathsf{ct} = \quad (1)\ [r]_1 \qquad (2)\ r\left[\frac{1}{x}\right]_T + m$$

Let’s try to derive another relation from the ciphertext:

$$\mathsf{dk} := \text{ linear fractions: } \left[\frac{1}{x+1}\right]_2, \ldots, \left[\frac{1}{x+\ell}\right]_2$$

Multiply fraction in dk with the linear relation in ct

$$e\left(r\left(\sum_{i=-1}^{\ell}\left[\frac{1}{x+i}\right]_1\right), \left[\frac{1}{x+j}\right]_2\right)$$

some algebra later...

Decryption Using Linear Independence

$$\mathsf{ct} = \quad (1)\ [r]_1 \qquad (2)\ r\left[\frac{1}{x}\right]_T + m$$

Let’s try to derive another relation from the ciphertext:

$$\mathsf{dk} := \text{ linear fractions: } \left[\frac{1}{x+1}\right]_2, \ldots, \left[\frac{1}{x+\ell}\right]_2$$

Multiply fraction in dk with the linear relation in ct

$$e\left(r\left(\sum_{i=-1}^{\ell}\left[\frac{1}{x+i}\right]_1\right), \left[\frac{1}{x+j}\right]_2\right) \xrightarrow{\text{some algebra later...}} r\left(\sum_{\substack{i=-1 \\ i\neq j}}^{\ell}\frac{1}{j-i}\left(\left[\frac{1}{x+i}\right]_T - \left[\frac{1}{x+j}\right]_T\right)\right)$$

Decryption Using Linear Independence

$$\mathsf{ct} = \quad (1)\ [r]_1 \qquad (2)\ r\left[\frac{1}{x}\right]_T + m$$

Let’s try to derive another relation from the ciphertext:

$$\mathsf{dk} := \text{ linear fractions: } \left[\frac{1}{x+1}\right]_2, \ldots, \left[\frac{1}{x+\ell}\right]_2$$

Multiply fraction in dk with the linear relation in ct

$$e\left(r\left(\sum_{i=-1}^{\ell}\left[\frac{1}{x+i}\right]_1\right), \left[\frac{1}{x+j}\right]_2\right) \xrightarrow{\text{some algebra later...}} r\left(\sum_{\substack{i=-1 \\ i\neq j}}^{\ell}\frac{1}{j-i}\left(\left[\frac{1}{x+i}\right]_T - \left[\frac{1}{x+j}\right]_T\right)\right)$$

you get a linear relation!

Decryption Using Linear Independence

$$\mathsf{ct} = \quad (1)\ [r]_1 \qquad (2)\ r\left[\frac{1}{x}\right]_T + m$$

Let’s try to derive another relation from the ciphertext:

$$\mathsf{dk} := \text{ linear fractions: } \left[\frac{1}{x+1}\right]_2, \ldots, \left[\frac{1}{x+\ell}\right]_2$$

Multiply fraction in dk with the linear relation in ct

$$e\left(r\left(\sum_{i=-1}^{\ell}\left[\frac{1}{x+i}\right]_1\right), \left[\frac{1}{x+j}\right]_2\right)$$

some algebra later...

$$r\left(\sum_{\substack{i=-1 \\ i\neq j}}^{\ell}\frac{1}{j-i}\left(\left[\frac{1}{x+i}\right]_T - \left[\frac{1}{x+j}\right]_T\right)\right) + r\left[\frac{1}{(x+j)^2}\right]_T$$

you get a linear relation!

but there’s an annoying quadratic term…

Pre-Decryptor Helps With Quadratic

$$r\left(\sum_{\substack{i=-1 \\ i \neq j}}^{\ell} \frac{1}{j-i}\left(\left[\frac{1}{x+i}\right]_T - \left[\frac{1}{x+j}\right]_T\right)\right) + r\left[\frac{1}{(x+j)^2}\right]_T$$

Pre-Decryptor Helps With Quadratic

$$r\left(\sum_{\substack{i=-1\\ i\neq j}}^{\ell}\frac{1}{j-i}\left(\left[\frac{1}{x+i}\right]_T-\left[\frac{1}{x+j}\right]_T\right)\right)+r\left[\frac{1}{(x+j)^2}\right]_T$$

$$\mathsf{PreDec}(\mathsf{sk},\{\mathsf{ct}_i\}_i)$$

Order the ciphertexts, and

output pre-decryption key as sum of partial fractions:

$$\mathsf{sbk}:=\sum_{i\in[\ell]}\frac{1}{x+i}\cdot\mathsf{ct}_i[1]=\sum_{i\in[\ell]}\left[\frac{r_i}{x+i}\right]_1$$

Pre-Decryptor Helps With Quadratic

$$r\left(\sum_{\substack{i=-1\\ i\neq j}}^{\ell}\frac{1}{j-i}\left(\left[\frac{1}{x+i}\right]_T-\left[\frac{1}{x+j}\right]_T\right)\right)+r\left[\frac{1}{(x+j)^2}\right]_T$$

$\mathsf{PreDec}(\mathsf{sk},\{\mathsf{ct}_i\}_i)$

Order the ciphertexts, and

output pre-decryption key as sum of partial fractions:

$$\mathsf{sbk}:=\sum_{i\in[\ell]}\frac{1}{x+i}\cdot\mathsf{ct}_i[1]=\sum_{i\in[\ell]}\left[\frac{r_i}{x+i}\right]_1$$

Multiplying sbk with fraction in dk gives the quadratic fraction

Pre-Decryptor Helps With Quadratic

$$r\left(\sum_{\substack{i=-1\\ i\neq j}}^{\ell}\frac{1}{j-i}\left(\left[\frac{1}{x+i}\right]_T-\left[\frac{1}{x+j}\right]_T\right)\right)+r\left[\frac{1}{(x+j)^2}\right]_T$$

$\mathsf{PreDec}(\mathsf{sk},\{\mathsf{ct}_i\}_i)$

Order the ciphertexts, and

output pre-decryption key as sum of partial fractions:

$$\mathsf{sbk}:=\sum_{i\in[\ell]}\frac{1}{x+i}\cdot\mathsf{ct}_i[1]=\sum_{i\in[\ell]}\left[\frac{r_i}{x+i}\right]_1$$

Multiplying sbk with fraction in dk gives the quadratic fraction $e\left(\mathsf{sbk},\left[\frac{1}{x+j}\right]_2\right)$

Pre-Decryptor Helps With Quadratic

$$r\left(\sum_{\substack{i=-1\\ i\neq j}}^{\ell}\frac{1}{j-i}\left(\left[\frac{1}{x+i}\right]_T-\left[\frac{1}{x+j}\right]_T\right)\right)+r\left[\frac{1}{(x+j)^2}\right]_T$$

$\mathsf{PreDec}(\mathsf{sk},\{\mathsf{ct}_i\}_i)$

Order the ciphertexts, and

output pre-decryption key as sum of partial fractions:

$$\mathsf{sbk}:=\sum_{i\in[\ell]}\frac{1}{x+i}\cdot\mathsf{ct}_i[1]=\sum_{i\in[\ell]}\left[\frac{r_i}{x+i}\right]_1$$

Multiplying sbk with fraction in dk gives the quadratic fraction $e\left(\mathsf{sbk},\left[\frac{1}{x+j}\right]_2\right)$



some algebra later...

Pre-Decryptor Helps With Quadratic

$$r\left(\sum_{\substack{i=-1\\ i\neq j}}^{\ell}\frac{1}{j-i}\left(\left[\frac{1}{x+i}\right]_T-\left[\frac{1}{x+j}\right]_T\right)\right)+r\left[\frac{1}{(x+j)^2}\right]_T$$

$\mathsf{PreDec}(\mathsf{sk},\{\mathsf{ct}_i\}_i)$

Order the ciphertexts, and

output pre-decryption key as sum of partial fractions:

$$\mathsf{sbk}:=\sum_{i\in[\ell]}\frac{1}{x+i}\cdot\mathsf{ct}_i[1]=\sum_{i\in[\ell]}\left[\frac{r_i}{x+i}\right]_1$$

Multiplying sbk with fraction in dk gives the quadratic fraction $e\left(\mathsf{sbk},\left[\frac{1}{x+j}\right]_2\right)$



some algebra later...

$$r\left[\frac{1}{(x+j)^2}\right]_T$$

Decryptor solves a linear system

$$r\left(\sum_{\substack{i=-1 \\ i \neq j}}^{\ell} \frac{1}{j-i}\left(\left[\frac{1}{x+i}\right]_T - \left[\frac{1}{x+j}\right]_T\right)\right) + r\left[\frac{1}{(x+j)^2}\right]_T$$

Decryptor solves a linear system

$$r\left(\sum_{\substack{i=-1 \\ i \neq j}}^{\ell} \frac{1}{j-i}\left(\left[\frac{1}{x+i}\right]_T - \left[\frac{1}{x+j}\right]_T\right)\right) + r\left[\frac{1}{(x+j)^2}\right]_T$$

With the quadratic term gone, the decryptor can solve the linear system

And recovers $r\left[\frac{1}{x}\right]_T$

Decryptor solves a linear system

$$r\left(\sum_{\substack{i=-1\\ i\neq j}}^{\ell}\frac{1}{j-i}\left(\left[\frac{1}{x+i}\right]_T-\left[\frac{1}{x+j}\right]_T\right)\right)+r\left[\frac{1}{(x+j)^2}\right]_T$$

With the quadratic term gone, the decryptor can solve the linear system

And recovers $r\left[\frac{1}{x}\right]_T$

$$\mathsf{Enc}(\mathsf{ek},\ \texttt{msg}\ m\in\mathbb{G}_T)\to\mathsf{ct}$$

Sample $r\xleftarrow{\$}\mathbb{Z}_p$, and output

ciphertext with **three** components

(1) $[r]_1$ (2) $r\left[\frac{1}{x}\right]_T+m$ (3) $r\left(\sum_{i=-1}^{\ell}\left[\frac{1}{x+i}\right]_1\right)$

Decryptor solves a linear system

$$r\left(\sum_{\substack{i=-1 \\ i\neq j}}^{\ell}\frac{1}{j-i}\left(\left[\frac{1}{x+i}\right]_T-\left[\frac{1}{x+j}\right]_T\right)\right)+r\left[\frac{1}{(x+j)^2}\right]_T$$

With the quadratic term gone, the decryptor can solve the linear system

And recovers $r\left[\frac{1}{x}\right]_T$

$$\mathsf{Enc}(\mathsf{ek},\ \texttt{msg}\ m\in\mathbb{G}_T)\to\mathsf{ct}$$

Sample $r\xleftarrow{\$}\mathbb{Z}_p$, and output

ciphertext with **three** components

(1) $[r]_1$ (2) $r\left[\frac{1}{x}\right]_T+m$ (3) $r\left(\sum_{i=-1}^{\ell}\left[\frac{1}{x+i}\right]_1\right)$

which is exactly the one-time pad!

Decryptor solves a linear system

$$r\left(\sum_{\substack{i=-1\\ i\neq j}}^{\ell}\frac{1}{j-i}\left(\left[\frac{1}{x+i}\right]_T-\left[\frac{1}{x+j}\right]_T\right)\right)+r\left[\frac{1}{(x+j)^2}\right]_T$$

With the quadratic term gone, the decryptor can solve the linear system

And recovers $r\left[\frac{1}{x}\right]_T$

Enc(ek, msg $m \in \mathbb{G}_T$) $\rightarrow$ ct

Sample $r \xleftarrow{\$} \mathbb{Z}_p$, and output

ciphertext with **three** components

(1) $[r]_1$

(2) $r\left[\frac{1}{x}\right]_T + m$

(3) $r\left(\sum_{i=-1}^{\ell}\left[\frac{1}{x+i}\right]_1\right)$

which is exactly the one-time pad!

see paper for more details

Encryption and Decryption Can Be More Efficient

Encryption and Decryption Can Be More Efficient

As presented,

Encryption and Decryption Can Be More Efficient

As presented,

ciphertext contains

3

group elements

Encryption and Decryption Can Be More Efficient

As presented,

ciphertext contains

3

group elements

decryption requires

$O(\ell^2)$

pairings and group operations

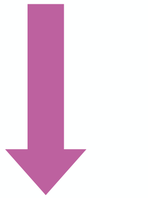
Encryption and Decryption Can Be More Efficient

As presented,

ciphertext contains

3

group elements



2

group elements

decryption requires

$O(\ell^2)$

pairings and group operations

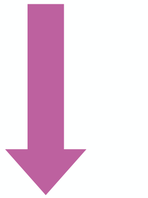
Encryption and Decryption Can Be More Efficient

As presented,

ciphertext contains

3

group elements



2

group elements

decryption requires

$O(\ell^2)$

pairings and group operations



$O(\ell)$

pairings

$O(\ell \log \ell)$

group operations

Encryption and Decryption Can Be More Efficient

As presented,

ciphertext contains

3

group elements



2

group elements

decryption requires

$O(\ell^2)$

pairings and group operations



$O(\ell)$

pairings

$O(\ell \log \ell)$

group operations

see paper for details

Prior Work Limitations

Prior Work Limitations

Ciphertext is tied to an index

Users must **encrypt to an index/position** in the block/batch. Thus, they **need to coordinate** and know their **position ahead of time**.

The ciphertext is *only* for the fixed index.

Index-dependent Batch Encryption

Prior Work Limitations

Ciphertext is tied to an index

Users must **encrypt to an index/position** in the block/batch. Thus, they **need to coordinate** and know their **position ahead of time**.

The ciphertext is *only* for the fixed index.

Index-dependent Batch Encryption

Ciphertext is tied to an epoch

Ciphertext is no longer tied to a position but is **only valid for the current batch** identified by an epoch.

If it’s not included, it **needs to be re-encrypted**.

“Epoch-based”

Prior Work Limitations

Ciphertext is tied to an index

Users must **encrypt to an index/position** in the block/batch. Thus, they **need to coordinate** and know their **position ahead of time**.

The ciphertext is *only* for the fixed index.

Index-dependent Batch Encryption

Ciphertext is tied to an epoch

Ciphertext is no longer tied to a position but is **only valid for the current batch** identified by an epoch.

If it's not included, it **needs to be re-encrypted**.

"Epoch-based"

Large CRS

Index-independent and *epoch-less* schemes suffer from a large exponential-size CRS

Our Work

[Boneh-**N**-Roy-Tas26]

Our Work

[Boneh-**N**-Roy-Tas26]

Best of all worlds!

We achieve batch encryption that is

1. **Independent of index:** no coordination required
2. **Epochless:** pending messages don’t need to be re-encrypted
3. **Much shorter practical CRS:** linear in batch size
4. **Shortest ciphertext:** only 2 group elements!

Our Work

[Boneh-**N**-Roy-Tas26]

(Almost) Best of all worlds!

We achieve batch encryption that is

1. **Independent of index:** no coordination required
2. **Epochless:** pending messages don’t need to be re-encrypted
3. **Much shorter practical CRS:** linear in batch size
4. **Shortest ciphertext:** only 2 group elements!

Secret key size is $\mathcal{O}(\ell)$:

linear in the maximum size of the batch

Our Work

[Boneh-**N**-Roy-Tas26]

(Almost) Best of all worlds!

We achieve batch encryption that is

1. **Independent of index:** no coordination required
2. **Epochless:** pending messages don't need to be re-encrypted
3. **Much shorter practical CRS:** linear in batch size
4. **Shortest ciphertext:** only 2 group elements!

Secret key size is $\mathcal{O}(\ell)$:

linear in the maximum size of the batch

Shortly after our eprint was out, two subsequent works

[Pol26, ADG+26]

achieved the same using a different technique

Proving Security

Proving Security

another q-type assumption

Proving Security

another q-type assumption

informally, captures the hardness of solving the linear system for the one-time pad fraction $\left[\frac{1}{x}\right]_T$

Proving Security

another q-type assumption

informally, captures the hardness of solving the linear system for the one-time pad fraction $\left[\frac{1}{x}\right]_T$



Assumption holds in Generic (Bilinear) Group Model

Implementation

Implementation

Prototypical implementation (not very optimized) using arkworks:
BLS12-381, single core Apple M4 Pro, 24 GB memory

Implementation

Prototypical implementation (not very optimized) using arkworks:
BLS12-381, single core Apple M4 Pro, 24 GB memory

BE: Index-independent batch encryption with **3 group elements**

Implementation

Prototypical implementation (not very optimized) using arkworks:
BLS12-381, single core Apple M4 Pro, 24 GB memory

BE: Index-independent batch encryption with **3 group elements**

$\mathsf{BE}_{\mathrm{short}}$: Index-independent batch encryption with **2 group elements**

Implementation

Prototypical implementation (not very optimized) using arkworks:
BLS12-381, single core Apple M4 Pro, 24 GB memory

BE: Index-independent batch encryption with **3 group elements**

$\mathsf{BE}_{\mathrm{short}}$: Index-independent batch encryption with **2 group elements**

Batch size ℓ	Enc		PreDec		Dec	
	BE	$\mathsf{BE}_{\mathrm{short}}$	BE	$\mathsf{BE}_{\mathrm{short}}$	BE	$\mathsf{BE}_{\mathrm{short}}$
4	599 µs	437 µs	1.53 ms	0.82 ms	19.2 ms	15.3 ms
16	599 µs	437 µs	2.83 ms	1.93 ms	101 ms	76.3 ms
64	599 µs	437 µs	8.12 ms	6.93 ms	501 ms	364 ms
128	599 µs	437 µs	14.5 ms	13.2 ms	1.09 s	787 ms
256	599 µs	437 µs	27.9 ms	25.6 ms	2.38 s	1.69 s
512	599 µs	437 µs	54.3 ms	50.1 ms	5.12 s	3.61 s

Future Work

Future Work

Applying this technique to other constructions?

Future Work

Applying this technique to other constructions?

in the appendix, we sketched a Hadamard product proof construction using partial fractions

Future Work

Applying this technique to other constructions?



in the appendix, we sketched a Hadamard product proof construction using partial fractions

Recently, [Jutla-Roy26] used that to do Groth16 proofs with only 4 FFTs (instead of 6)

Future Work

Applying this technique to other constructions?



in the appendix, we sketched a Hadamard product proof construction using partial fractions

Recently, [Jutla-Roy26] used that to do Groth16 proofs with only 4 FFTs (instead of 6)

Generalizing to higher-order fractions?

Could perhaps lead to more expressivity or packing more information per fraction

Future Work

Applying this technique to other constructions?

in the appendix, we sketched a Hadamard product proof construction using partial fractions

Recently, [Jutla-Roy26] used that to do Groth16 proofs with only 4 FFTs (instead of 6)

Generalizing to higher-order fractions?

Could perhaps lead to more expressivity or packing more information per fraction

we're actively working on this, so if you have any ideas, we'd love to collaborate :)

Takeaways

Partial Fraction Decomposition

Allows you to **linearize cross-terms:** $\frac{1}{X+i} \cdot \frac{1}{X+j} = \frac{1}{j-i}\left(\frac{1}{X+i} - \frac{1}{X+j}\right),$

and **mask values with higher-order fractions:** $\frac{1}{(X+k)^2}$

Linear Independence

Product with partial fractions allows you to derive linearly independent relations:

$$\left(\frac{1}{X+i} + \frac{1}{X+j}\right) \rightsquigarrow \frac{1}{k-i} \cdot \frac{1}{X+i} + \frac{1}{k-j} \cdot \frac{1}{X+j}$$