$$\frac{1}{X+a} \qquad \sum \frac{1}{X+a_i} \qquad \frac{1}{X+b}$$

Partial Fraction Techniques For Cryptography

$$\frac{1}{X+a} \cdot \frac{1}{X+b} = \frac{1}{b-a}\left(\frac{1}{X+a} - \frac{1}{X+b}\right)$$

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

Rohit Nema (Stanford) | Bay Area Crypto Day | Nov. 14, 2025
joint work with Arnab Roy (Mysten Labs) and Charanjit Jutla (IBM)

Summary

- We use Partial Fraction Decomposition (of rational functions with linear denominator) to do cryptography

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

- Not the first to use rational functions (e.g. [BB04b,DY05,JL09,Wee16,Hab22,GV22])
- Two interesting properties: (1) *membership testing* and (2) *linear independence*

$$(1) \Longrightarrow \textbf{Key-Value Commitments} \qquad (2) \Longrightarrow \textbf{Dynamic Threshold Encryption}$$

- Prove security using new (*adaptive)* falsifiable q-type assumptions (secure in the (bilinear) generic group model)

Partial Fraction Decomposition

the mathematical foundation of our work

- rewrite product of rational functions as a sum (with easily computable closed-form coefficients)

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

well-known

Theorem. Suppose $n \geq 2$. Let $a_1, \ldots, a_n \in \mathbb{F}$ be distinct. Then, there exists unique coefficients $c_1, \ldots, c_n \in \mathbb{F}$ such that,

$$\prod_{i \in [n]} \frac{1}{X + a_i} = \sum_{i \in [n]} \frac{c_i}{X + a_i}$$

where $c_i = \prod_{j \neq i} \frac{1}{a_j - a_i}$ for each $i \in [n]$.

Partial Fraction Decomposition

(Non-)Membership Testing (1)

- Represent set $S = \{a_1, \ldots, a_n\}$ as a sum of fractions, $\displaystyle\sum_{i \in [n]} \frac{1}{X + a_i}$

- For $b \in \mathbb{F}$, examine the product, $P = \left(\displaystyle\sum_{i \in [n]} \frac{1}{X + a_i} \right) \left(\frac{1}{X + b} \right)$
  - If $b \notin S$, then there exists h_i such that, $P = \displaystyle\sum_{i \in [n]} \frac{z_i}{X + a_i} + \frac{z_{n+1}}{X + b}$
  - If $b \in S$, then no such decomposition exists

Partial Fraction Decomposition

Linear Independence (2)

- Consider again,

$$\left(\sum_{i\in[n]}\frac{1}{X+a_i}\right)\left(\frac{1}{X+b}\right)=\sum_{i\in[n]}\frac{z_i}{X+a_i}+\frac{z_{n+1}}{X+b}$$

- Using **Theorem 1**, it is easy to show that,

$$z_i=\frac{1}{b-a_i}\text{ and }z_{n+1}=-\sum_{i\in[n]}\frac{1}{b-a_i}$$

- For $m\leq n$ distinct $b_1,\ldots,b_m\notin S$, the following are $\mathbb{F}$-linearly independent,

$$\left\{\left(\sum_{i\in[n]}\frac{1}{X+a_i}+\sum_{i\in[n]}\frac{1}{b_j-a_i}\right)\left(\frac{1}{X+b_j}\right)\right\}_{j\in[m]}=\left\{\sum_{i\in[n]}\frac{1}{b_j-a_i}\cdot\frac{1}{X+a_i}\right\}_{j\in[m]}$$

Partial Fraction Decomposition

Linear Independence (2)

- For $m \leq n$ distinct $b_1, \ldots, b_m \notin S$, the following are $\mathbb{F}$-linearly independent,

$$\left\{ \left(\sum_{i \in [n]} \frac{1}{X + a_i} \right) \left(\frac{1}{X + b_j} \right) + \sum_{i \in [n]} \frac{1}{b_j - a_i} \right\}_{j \in [m]} = \left\{ \sum_{i \in [n]} \frac{1}{b_j - a_i} \cdot \frac{1}{X + a_i} \right\}_{j \in [m]}$$

- **Why?**
  - the coefficients $\left(\frac{1}{b_j - a_i} \right)$ form a full-rank C*auchy* matrix*!*

Cauchy Matrices

- For $\vec{a} \in \mathbb{F}^n, \vec{b} \in \mathbb{F}^m$, we define the Cauchy matrix $\mathscr{C}(\vec{a}, \vec{b}) = \left(c_{ij}\right)_{i \in [n], j \in [m]}$ as,

$$c_{ij} = \frac{1}{b_j - a_i}$$

- So,

$$\mathscr{C}(\vec{a}, \vec{b}) = \begin{bmatrix} \frac{1}{b_1 - a_1} & \frac{1}{b_2 - a_1} & \cdots & \frac{1}{b_m - a_1} \\ \frac{1}{b_1 - a_2} & \frac{1}{b_2 - a_2} & \cdots & \frac{1}{b_m - a_2} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{1}{b_1 - a_n} & \frac{1}{b_2 - a_n} & \cdots & \frac{1}{b_m - a_n} \end{bmatrix}$$

one can show this is full-rank

if $(a_1, \ldots, a_n, b_1, \ldots, b_m)$ are all distinct

Product of Partial Fractions as Cauchy Matrix

$$\left\{\left(\sum_{i\in[n]}\frac{1}{X+a_i}\right)\left(\frac{1}{X+b_j}\right)+\sum_{i\in[n]}\frac{1}{b_j-a_i}\right\}_{j\in[m]}=\left\{\sum_{i\in[n]}\frac{1}{b_j-a_i}\cdot\frac{1}{X+a_i}\right\}_{j\in[m]}$$

$$\mathscr{C}(\vec{a},\vec{b})^T\begin{bmatrix}\frac{1}{X+a_1}\\ \vdots\\ \frac{1}{X+a_n}\end{bmatrix}=\begin{bmatrix}\frac{1}{b_1-a_1} & \frac{1}{b_1-a_2} & \cdots & \frac{1}{b_1-a_n}\\ \frac{1}{b_2-a_1} & \frac{1}{b_2-a_2} & \cdots & \frac{1}{b_2-a_n}\\ \vdots & \vdots & \ddots & \vdots\\ \frac{1}{b_m-a_1} & \frac{1}{b_m-a_2} & \cdots & \frac{1}{b_m-a_n}\end{bmatrix}\begin{bmatrix}\frac{1}{X+a_1}\\ \vdots\\ \frac{1}{X+a_n}\end{bmatrix}$$

Product of Partial Fractions as Cauchy Matrix

A 2 x 2 Example

Let $n = 2,\ m = 2.$

Suppose $a_1 = 1, a_2 = 4, b_1 = 3,$ and $b_2 = 8.$

$$\left(\frac{1}{X+1}+\frac{1}{X+4}\right)\left(\frac{1}{X+3}\right)+\left(\frac{1}{3-1}+\frac{1}{3-4}\right)\frac{1}{X+3}=\frac{1}{3-1}\cdot\frac{1}{X+1}+\frac{1}{3-4}\cdot\frac{1}{X+4}$$

$$\left(\frac{1}{X+1}+\frac{1}{X+4}\right)\left(\frac{1}{X+8}\right)+\left(\frac{1}{8-1}+\frac{1}{8-4}\right)\frac{1}{X+8}=\frac{1}{8-1}\cdot\frac{1}{X+1}+\frac{1}{8-4}\cdot\frac{1}{X+4}$$

$$\begin{bmatrix} \frac{1}{3-1} & \frac{1}{3-4} \\ \frac{1}{8-1} & \frac{1}{8-4} \end{bmatrix} \begin{bmatrix} \frac{1}{X+1} \\ \frac{1}{X+4} \end{bmatrix}$$

Partial Fraction Decomposition

Recap: Linear Independence (2)

- Consider again,

$$\left(\sum_{i\in[n]} \frac{1}{X+a_i}\right)\left(\frac{1}{X+b}\right) = \sum_{i\in[n]} \frac{z_i}{X+a_i} + \frac{z_{n+1}}{X+b}$$

- Using **Theorem 1**, it is easy to show that,

$$z_i = \frac{1}{b-a_i} \text{ and } z_{n+1} = -\sum_{i\in[n]} \frac{1}{b-a_i}$$

- For $m \leq n$ distinct $b_1, \ldots, b_m \notin S$, the following are $\mathbb{F}$-linearly independent,

$$\left\{\left(\sum_{i\in[n]} \frac{1}{X+a_i} + \sum_{i\in[n]} \frac{1}{b_j-a_i}\right)\left(\frac{1}{X+b_j}\right)\right\}_{j\in[m]} = \left\{\sum_{i\in[n]} \frac{1}{b_j-a_i}\cdot\frac{1}{X+a_i}\right\}_{j\in[m]}$$

First Construction

Key-Value Commitments

Key-Value Commitments (KVCs) [AR20]

using (Non-)Membership Testing (1)

- We have a dictionary, a set of key-value pairs,

$$D = \left\{ \left(k_i, v_i \right) \right\}_{i \in [n]}$$

- *Commit* using "linear" partial fractions,

$$C_D(X) = \sum_{i \in [n]} \frac{v_i}{X + k_i}$$

- We can test,

$$\frac{(k, v) \in D}{\begin{array}{c}\exists z_i \text{ such that,} \\ \left(C_D(X) - \frac{v}{X + k} \right) \left(\frac{1}{X + k} \right) = \sum_{i \in [n]} \frac{z_i}{X + k_i}\end{array}} \qquad \frac{(k, v) \notin D}{\begin{array}{c}\textit{no such decomposition exists} \\ \text{because of quadratic fraction,} \\ \frac{1}{(X + k)^2}\end{array}}$$

Key-Value Commitments (KVCs)

Instantiating Cryptographically

- Asymmetric bilinear group, $\mathscr{G} = \left(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e\right)$,
  - Additive notation: $[x]_i = g_i^x \in \mathbb{G}_i$, and $[y]_T = e([1]_1, [1]_2)^y \in \mathbb{G}_T$
- Trusted setup/DKG: sample $x \xleftarrow{\$} \mathbb{F}$,
- Let $K_i(X) = \dfrac{1}{X + k_i}$. Then, commit to dictionary D using,

additively homomorphic!
efficient to update

$$[C_D(x)]_1 = \left[\sum_{i \in [n]} \frac{v_i}{x + k_i}\right]_1 = \sum_{i \in [n]} v_i \left[K_i(x)\right]_1$$

- Membership test is a pairing equation,

where does verifier get these?

$$e\left([C_D(x)]_1 - v\left[\frac{1}{x+k}\right]_1, \left[\frac{1}{x+k}\right]_2\right) \stackrel{?}{=} e\left(\sum_{i \in [n]} z_i \left[K_i(x)\right]_1, [1]_2\right)$$

linear in $|D|$ to compute :(

Credential-based KVCs

- New *semi-trusted* party, the *Registrar*: knows x and D and runs in $O(|D|)$ time.
- Takes as input k_{n+1} and outputs a *credential*,

$$h_{n+1}^{(1)} = \left[\frac{1}{x + k_{n+1}}\right]_1, h_{n+1}^{(2)} = \left[\frac{1}{x + k_{n+1}}\right]_2, \Lambda = \sum_{i \in [n+1]} \left[\frac{z_i}{x + k_i}\right]_1$$

- *Semi-stateless* setting: stateless and $O(1)$ to update commitment after one-time linear computation of credential
  - after each update, user publishes *update information* and every user updates their proofs

Updating commitment

$$\left[C_D(x)\right]_1 + v_{n+1} h_{n+1}^{(1)}$$

Update information

$$\left(k_{n+1}, v_{n+1}, h_{n+1}^{(1)}\right)$$

Updating proof for user i

$$\Lambda + \frac{v_{n+1}}{k_{n+1} - k_i}\left(h_i^{(1)} - h_{n+1}^{(1)}\right)$$

$$+$$

verify $h_{n+1}^{(1)}$ is well-formed

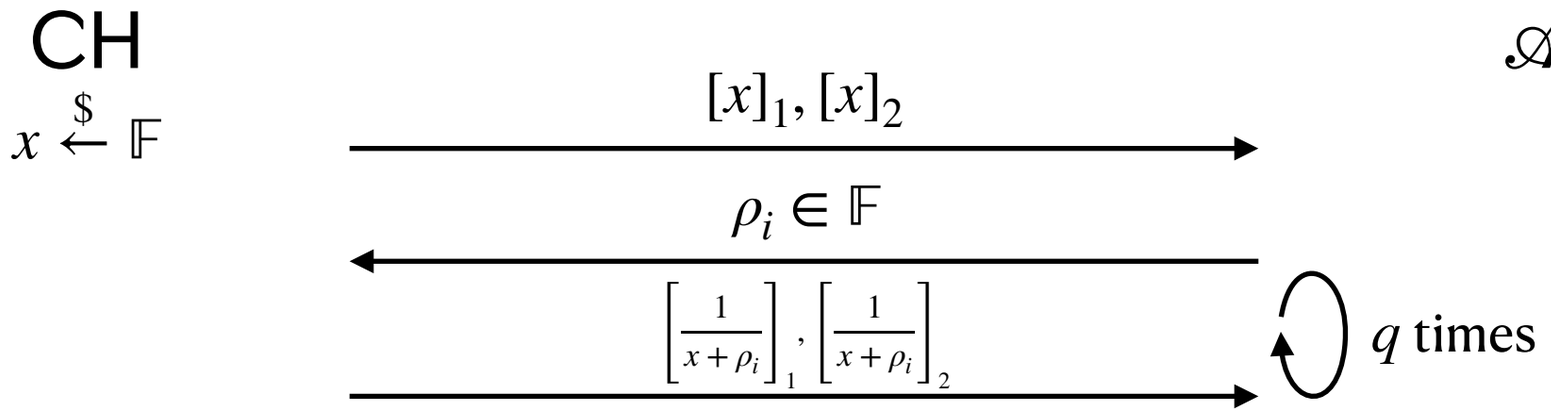
$$e\left(h_{n+1}^{(1)}, [x]_2 + [k_{n+1}]_2\right) \stackrel{?}{=} e([1]_1, [1]_2)$$

Credential-based KVCs

Defining and Proving Security

- Usual notion of *key-binding* (slightly modified to account for *unregistered* keys)
- Two new assumptions, (1) q-DistPole: reminiscent of BB signature *unforgeability,* and (2) q-SqPole: cannot compute a quadratic fraction from linear fractions

q-DistPole



$\mathscr{A}$ wins if,

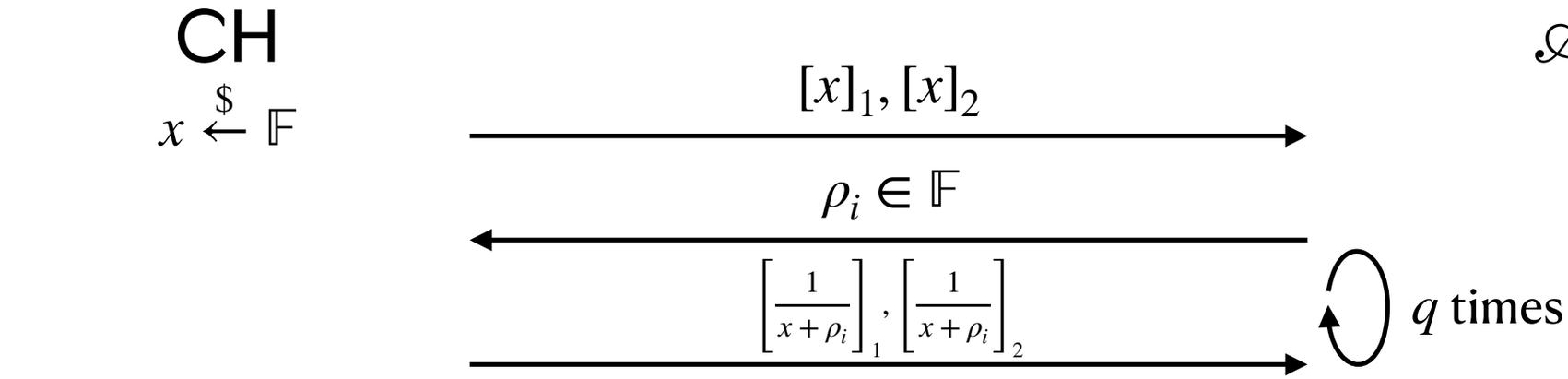
$$\rho \notin \{\rho_i\}_{i\in[q]} \text{ and } u = \left[\frac{1}{x+\rho}\right]_2$$

Credential-based KVCs

Defining and Proving Security

- Usual notion of *key-binding* (slightly modified to account for *unregistered* keys)
- Two new assumptions, (1) q-DistPole: reminiscent of BB signature *unforgeability,* and (2) q-SqPole: cannot compute a quadratic fraction from linear fractions

q-SqPole



$\mathscr{A}$ wins if,

$$u = \left[\frac{1}{(x+\rho_j)^2}\right]_2$$

Credential-based KVCs

Efficiency & Comparison with Prior Work

- In fact, loosely similar to RSA KVaC [AR20] which is instantiated in composite-order groups
  - vs. prime-order groups in our case. Much smaller in practice $\Longrightarrow$ less communication
  - computationally more efficient. No expensive Bezout coefficients calculation
- Cuckoo commitments [FKdP23] constructs KVCs from Vector Commitments generically using cuckoo hashing
  - our construction is *purely* algebraic avoiding non-algebraic operations such as cuckoo hashing
  - we support stateless updates while Cuckoo commitments does not

Second Construction
Dynamic Threshold Encryption

Dynamic Threshold Encryption (DTE) [DP08]

overview

- Public-key encryption with threshold decryption [DF89]
  - Given ciphertext, users can generate *decryption shares,* which can be combined to recover plaintext
- Dynamic selection of threshold and authorized set at encryption time
- In our setting, central authority issues keys using secret (DKG relatively simple and efficient in our case so can be multiple parties in MPC)
- For simplicity, assume users are in $[n]$ but can extend to arbitrary identities using a hash function...
- Define $p_i(X) = \dfrac{1}{X + i}$

$$\mathsf{pk}_i = \left[p_i(x)\right]_1 \in \mathbb{G}_1 \quad \text{and} \quad \mathsf{sk}_i = \left[p_i(x)\right]_2 \in \mathbb{G}_2$$

Dynamic Threshold Encryption (DTE)

preprocessing the encryption key

- Public preprocess encryption key as sum of public keys *not in* the authorized set or in other words, the *unauthorized set*
- For authorized set $S \subseteq [n]$ of size s,

$$\mathsf{ek}_S = \sum_{i \notin S} \mathsf{pk}_i = \left[\sum_{i \notin S} p_i(x) \right]_1$$

- Think of ek_S as a linear relation on $n - s$ *variables*, $p_i(x)$

Dynamic Threshold Encryption (DTE)

Using Linear Independence (2)

$$\mathsf{ek}_S = \sum_{i \notin S} \mathsf{pk}_i = \left[\sum_{i \notin S} p_i(x) \right]_1$$

- Multiply (pairing) with secret key of authorized user $j \in S$,

$$e\left(\mathsf{ek}_S, \mathsf{sk}_j\right) = \left[\sum_{i \notin S} p_i(x) \cdot p_j(x) \right]_T = \left[\sum_{i \notin S} \frac{1}{j-i} \left(p_i(x) - p_j(x) \right) \right]_T$$

$$\text{So, } e\left(\mathsf{ek}_S, \mathsf{sk}_j\right) + \left[\sum_{i \notin S} \frac{1}{j-i} p_j(x) \right]_T = \left[\sum_{i \notin S} \frac{1}{j-i} p_i(x) \right]_T$$

- Which are linearly independent for distinct $j, j' \in S$ by (2)

Dynamic Threshold Encryption (DTE)

Using Linear Independence (2)

$$\mathsf{ek}_S = \sum_{i \notin S} \mathsf{pk}_i = \left[\sum_{i \notin S} p_i(x) \right]_1 \qquad e\left(\mathsf{ek}_S, \mathsf{sk}_j\right) + \left[\sum_{i \notin S} \frac{1}{j-i} p_j(x) \right]_T = \left[\sum_{i \notin S} \frac{1}{j-i} p_i(x) \right]_T$$

- Simplifying,

$$e\left(\mathsf{ek}_S, \mathsf{sk}_j\right) + \sum_{i \notin S} \frac{1}{j-i} \left[p_j(x)\right]_T = \sum_{i \notin S} \frac{1}{j-i} \left[p_i(x)\right]_T$$

$$\implies e\left(\mathsf{ek}_S + \sum_{i \notin S} \frac{1}{j-i} \left[1\right]_1, \mathsf{sk}_j\right) = \sum_{i \notin S} \frac{1}{j-i} \left[p_i(x)\right]_T$$

- So, we obtain linear relations over variables $\left[p_i(x)\right]_T$

- *Intuition*. Every authorized user can contribute one linear relation. Given enough linear relations, we obtain a full-rank system and can solve for some mask that hides the plaintext

Dynamic Threshold Encryption (DTE)

Adding Mask + Generating the Ciphertext

$$\mathsf{ek}_S = \sum_{i \notin S} \mathsf{pk}_i = \left[\sum_{i \notin S} p_i(x) \right]_1 \qquad e\left(\mathsf{ek}_S + \sum_{i \notin S} \frac{1}{j-i} \left[1\right]_1, \mathsf{sk}_j \right) = \sum_{i \notin S} \frac{1}{j-i} \left[p_i(x)\right]_T$$

- Sample a "target" masking key, $\mathsf{tk} = \left[\frac{1}{x-1} = p_{-1}(x) \right]_1$
- We add it to ek_S so the linear relations are also over $\left[p_{-1}(x)\right]_T$
- Randomize the entire linear system so keys are reusable... Sample $r \xleftarrow{\$} \mathbb{F}$
- For message $m \in \mathbb{G}_T$,

$$\mathsf{ct} = \left([r]_1,\ r \cdot (\mathsf{ek}_S + \mathsf{tk}),\ r \cdot \left[p_{-1}(x)\right]_T + m \right)$$

random mask in $\mathbb{G}_1$

to extract linear relation of randomized system

mask recoverable after solving linear system

Dynamic Threshold Encryption (DTE)

Modifying Partial Decryption

$$\mathsf{ek}_S = \sum_{i \notin S} \mathsf{pk}_i = \left[\sum_{i \notin S} p_i(x) \right]_1 \qquad e\left(\mathsf{ek}_S + \sum_{i \notin S} \frac{1}{j-i} \left[1\right]_1, \mathsf{sk}_j \right) = \sum_{i \notin S} \frac{1}{j-i} \left[p_i(x)\right]_T \qquad \mathsf{ct} = \left([r]_1,\ r \cdot (\mathsf{ek}_S + \mathsf{tk}),\ r \cdot \left[p_{-1}(x)\right]_T + m \right)$$

- Use ct_1 and ct_2 instead,

$$e\left(\mathsf{ct}_2 + \left(\sum_{i \notin S} \frac{1}{j-i} + \frac{1}{j+1} \right) \mathsf{ct}_1, \mathsf{sk}_j \right)$$

$$= \left[r \left(\sum_{i \notin S} p_i(x) + p_{-1}(x) + \sum_{i \notin S} \frac{1}{j-i} + \frac{1}{j+1} \right) p_j(x) \right]_T$$

$$= \sum_{i \notin S} \frac{r}{j-i} \left[p_i(x)\right]_T + \frac{r}{j+1} \left[p_{-1}(x)\right]_T$$

Dynamic Threshold Encryption (DTE)

Adding Dummies for Arbitrary Threshold

$$\mathsf{ek}_S = \sum_{i \notin S} \mathsf{pk}_i = \left[\sum_{i \notin S} p_i(x) \right]_1 \qquad e\left(\mathsf{ct}_2 + \left(\sum_{i \notin S} \frac{1}{j-i} + \frac{1}{j+1} \right) \mathsf{ct}_1, \mathsf{sk}_j \right) \qquad \mathsf{ct} = \left([r]_1,\ r \cdot (\mathsf{ek}_S + \mathsf{tk}),\ r \cdot \left[p_{-1}(x) \right]_T + m \right)$$

- Number of variables in system ≡ number of partial decryptions required to unmask message
- How do we go from $n - s + 1$ variables to something that depends on the threshold t?
- We add *dummy* partial fractions to the encryption key
  - Act like *variables* in the linear system
  - Add $s + t$ dummies to get a linear system over $n + t + 1$ variables
- How to decrypt with only t partial decryption shares?
  - Give n secret keys “for free” in the aggregation/combining key ck
- Dummies can be added dynamically at encryption time

Dynamic Threshold Encryption (DTE)

Consolidating Dummies

$$\mathsf{ek}_S = \sum_{i \notin S} \mathsf{pk}_i = \left[\sum_{i \notin S} p_i(x) \right]_1 \qquad e\left(\mathsf{ct}_2 + \left(\sum_{i \notin S} \frac{1}{j-i} + \frac{1}{j+1} \right) \mathsf{ct}_1, \mathsf{sk}_j \right) \qquad \mathsf{ct} = \left([r]_1,\ r \cdot (\mathsf{ek}_S + \mathsf{tk}),\ r \cdot \left[p_{-1}(x) \right]_T + m \right)$$

- As is, need $2n$ dummies to support any S and t
- The identity of the dummies don't matter—we just need an appropriate number of them
  - Borrow trick from recent work by Waters and Wu [WW25] to publish sums of dummies in increasing powers of 2
- Thus, only need $O(\log n)$ public parameters!
- (We ignore dummies henceforth for clarity in this presentation...)

Dynamic Threshold Encryption (DTE)

Share Verification + Reducing Communication

$$\mathsf{ek}_S = \sum_{i \notin S} \mathsf{pk}_i = \left[\sum_{i \notin S} p_i(x) \right]_1 \qquad e\left(\mathsf{ct}_2 + \left(\sum_{i \notin S} \frac{1}{j-i} + \frac{1}{j+1} \right) \mathsf{ct}_1, \mathsf{sk}_j \right) \qquad \mathsf{ct} = \left([r]_1,\ r \cdot (\mathsf{ek}_S + \mathsf{tk}),\ r \cdot \left[p_{-1}(x) \right]_T + m \right)$$

- Verify partial decryption share is valid with respect to user's verification key vk_i (which will end up being equal to pk_i in our case). We modify partial decryption to support this...
- Converts one $\mathbb{G}_T$ element into one $\mathbb{G}_1$ and one $\mathbb{G}_2$ element, reducing communication in practice
- Sample $r_i \xleftarrow{\$} \mathbb{F}$ and output, $\partial_i = \left(\frac{1}{r_i}\mathsf{ct}_1, \frac{1}{r_i}\mathsf{ct}_2, \left[r_i \right]_2, r_i \cdot \mathsf{sk}_i \right)$
- To verify, check ct and randomized sk_i are consistent with randomness r_i and pk_i,

$$e\left(pk_i, \left[r_i \right]_2 \right) \stackrel{?}{=} e\left([1]_1, r_i \cdot \mathsf{sk}_i \right) \textit{ and } e\left(\frac{1}{r_i}\mathsf{ct}_1, \left[r_i \right]_2 \right) \stackrel{?}{=} e\left(\mathsf{ct}_1, [1]_2 \right) \textit{ and } e\left(\frac{1}{r_i}\mathsf{ct}_2, \left[r_i \right]_2 \right) \stackrel{?}{=} e\left(\mathsf{ct}_2, [1]_2 \right)$$

- Can be batched into only 3 pairings total (vs. 6 above)

Dynamic Threshold Encryption (DTE)

Combining/Aggregation

$$\mathsf{ek}_S = \sum_{i \notin S} \mathsf{pk}_i = \left[\sum_{i \notin S} p_i(x) \right]_1 \qquad \partial_i = \left(\frac{1}{r_i}\mathsf{ct}_1, \frac{1}{r_i}\mathsf{ct}_2, \left[r_i\right]_2, r_i \cdot \mathsf{sk}_i \right) \qquad \mathsf{ct} = \left([r]_1,\ r \cdot (\mathsf{ek}_S + \mathsf{tk}),\ r \cdot \left[p_{-1}(x)\right]_T + m \right)$$

- Combiner has ck with *free* secret keys (distinct from real users)
- Derives n more linear relations (it turns out it doesn't need to do n additional pairings for this... just 1!)
- Solves for $r \cdot \left[p_{-1}(x)\right]_T$ using inverse of Cauchy matrix
  - Only need first row of the inverse. Closed-form solution efficiently computable in $\tilde{O}(n)$

Dynamic Threshold Encryption (DTE)

CPA Security with Adaptive Corruptions

- A new assumption *q-Decisional Bilinear Rank-Deficit DH* (*q*-RankDef)
  - Briefly, given $\left[\frac{1}{x+\rho_i}\right]_1$ and *not too many** $\left[\frac{1}{x+\rho_i}\right]_2$ for adversarially chosen ρ_i, and $\left[\frac{1}{x+\rho_t}\right]_1$ for an adversarially chosen *target* ρ_t, $\mathcal{A}$ cannot distinguish the following,

$$\left([r]_1, r \cdot \left[\sum_i \frac{1}{x+\rho_i} + \frac{1}{x+\rho_t}\right]_1, r \cdot \left[\frac{1}{x+\rho_t}\right]_T\right)$$

$$\approx_c$$

$$\left([r]_1, r \cdot \left[\sum_i \frac{1}{x+\rho_i} + \frac{1}{x+\rho_t}\right]_1, \cdot [r']_T\right)$$

(* It has a *deficit amount of fractions in* $\mathbb{G}_2$.)

Dynamic Threshold Encryption (DTE)

CPA Security with Adaptive Corruptions

- $\mathcal{A}$ can pick ρ_i adaptively
- Can prove a strong notion of CPA security wherein $\mathcal{A}$ can corrupt parties adaptively both before and after the encryption key is sampled, and even after the challenge, easily using this assumption
- Indeed, assumption is strong because it is adaptive in nature
- Assumption holds in Bilinear GGM
- Interesting if it's possible to reduce this to a static assumption (like BB signature unforgeability [BB04b])

Dynamic Threshold Encryption (DTE)

CCA-2 Security

- Ciphertext is ElGamal-like
- Can use generic transformations to get CCA-2 security with little overheard
- Ciphertext can also be made *publicly verifiable*—to verify if the ciphertext is *decryptable*
- Two approaches: Groth-Sahai NIZK [GS08] (short CRS + large proof) or QA-NIZK [JR13] (larger CRS + shorter proof)

Dynamic Threshold Encryption (DTE)

Comparison with prior work

	Delerablée and Pointcheval [DP08]	Garg et al. [GKPW24]	Waters and Wu [WW25]	This work
$\|\mathsf{ek}\|$	$O(n)$	$O(1)$	$O(\log n)$	$O(\log s)$
$\|\mathsf{pp}\|$	–	$O(n)$	$O(n \log n)$	$O(\log n)$
$\|\mathsf{pk}\|$	$O(1)$	$O(n)$	$O(n)$	$O(1)$
$\|\mathsf{ct}\|$	$2\|\mathbb{G}_1\| + \|\mathbb{G}_T\|$	$2\|\mathbb{G}_1\| + 7\|\mathbb{G}_2\| + \|G_T\|$	$3\|\mathbb{G}\| + \|\mathbb{G}_T\|$	$2\|\mathbb{G}_1\| + \|\mathbb{G}_T\|$ ($+3\|\mathbb{G}_1\| + \|\mathbb{G}_T\|$ for CCA)
$\|\partial\|$	$3\|\mathbb{G}\| + \|\mathbb{G}_T\|$	$1\|\mathbb{G}_2\|$	$2\|\mathbb{G}\|$	$2(\|G_1\| + \|\mathbb{G}_2\|)$
ciphertext validation?	✓	X	X	✓
$\|\mathsf{ck}\|$	$O(n)$	$O(s)$	$O(s)$	$O(n)$
# of pairings for decryption	$O(t)$	0	3	$O(t)$
# of pairings for verifying share	2	2	2	3
No RO?	X	X	✓	✓
Adaptive?	X	✓	X	✓
Assumption	(static) q-type	generic group	(static) q-type	(adaptive) q-type

Other Constructions

Observing Patterns in Other Primitives

- Hadamard Product Proofs
  - Given commitments to $\vec{u}, \vec{v}, \vec{w} \in \mathbb{F}^n$, we want to prove,

$$\vec{u} \circ \vec{v} = \vec{w}$$

  - Commitment using partial fractions, e.g. $\mathsf{com}_{\vec{u}} = \left[\sum_{i \in [n]} \frac{u_i}{x + i} \right]_1$

    enables proving the relation easily
- Registration-based Encryption
  - algebraically simple construction with concrete savings

ePrint 2025/2081

eprint.iacr.org/2025/2081

References

BB04b. Dan Boneh and Xavier Boyen. Short signatures without random oracles.

DY05. Yevgeniy Dodis and Aleksandr Yampolskiy. A verifiable random function with short proofs and keys.

Wee16. Hoeteck Wee. Déjà q: Encore! un petit ibe.

Hab22. Ulrich Haböck. Multivariate lookups based on logarithmic derivatives.

GV22. Rishab Goyal and Vinod Vaikuntanathan. Locally verifiable signature and key aggregation.

JL09. Stanislaw Jarecki and Xiaomin Liu. Efficient oblivious pseudorandom function with applications to adaptive OT and secure computation of set intersection.

AR20. Shashank Agrawal and Srinivasan Raghuraman. KVaC: Key-Value Commitments for blockchains and beyond.

FKdP23. Dario Fiore, Dimitris Kolonelos, and Paola de Perthuis. Cuckoo commitments: Registration-based encryption and key-value map commitments for large spaces.

DP08. Cécile Delerablée and David Pointcheval. Dynamic threshold public-key encryption.

DF89. Yvo Desmedt and Yair Frankel. Threshold Cryptosystems.

WW25. Brent Waters and David J. Wu. Silent threshold cryptography from pairings: Expressive policies in the plain model.

GS08. Jens Groth and Amit Sahai. Efficient non-interactive proof systems for bilinear groups.

JR13. Charanjit S. Jutla and Arnab Roy. Shorter quasi-adaptive NIZK proofs for linear subspaces.

GKPW24. Sanjam Garg, Dimitris Kolonelos, Guru-Vamsi Policharla, and Mingyuan Wang. Threshold encryption with silent setup.