A Framework for WE from Linearly Verifiable SNARKs and Applications

Sanjam Garg, Mohammad Hajiabadi, Dimitris Kolonelos, Abhiram Kothapalli, and
Guru-Vamsi Policharla

Witness Encryption [GGSW13, GKPVZ13]

Witness Encryption [GGSW13, GKPVZ13]

Consider an NP relation: $R \subset \{0,1\}^* \times \{0,1\}^*$

Witness Encryption [GGSW13, GKPVZ13]

Consider an NP relation: $R \subset \{0,1\}^* \times \{0,1\}^*$

(Extractable) Witness Encryption: Encrypt to a statement x.

Can decrypt iff you know witness w such that $(x, w) \in R$.

Witness Encryption [GGSW13, GKPVZ13]

Consider an NP relation: $R \subset \{0,1\}^* \times \{0,1\}^*$

(Extractable) Witness Encryption: Encrypt to a statement x.
Can decrypt iff you know witness w such that $(x, w) \in R$.

Witness encryption for all of NP is very powerful — recent progress but no concretely efficient constructions. [CVW18,Tsa22,VWW22]

Witness Encryption [GGSW13, GKPVZ13]

Consider an NP relation: $R \subset \{0,1\}^* \times \{0,1\}^*$

(Extractable) Witness Encryption: Encrypt to a statement x.
Can decrypt iff you know witness w such that $(x, w) \in R$.

Witness encryption for all of NP is very powerful — recent progress but no concretely efficient constructions. [CVW18,Tsa22,VWW22]

Today: Focus on efficient WE for special relations and applications.

Not going to build WE for NP

Many Examples!

Over the last 25 years:

**not an exhaustive list*

Many Examples!

Over the last 25 years:

Identity-Based Encryption
[BF01]

Hash-Proof Systems
[CS02], [BC16]

**not an exhaustive list*

Many Examples!

Over the last 25 years:

Hash Encryption
[CDG+17], [DG17]

Identity-Based Encryption
[BF01]

Registration Based Encryption
[GHMR18], [GKMR23], [FKdP23]

(Distributed) Broadcast Encryption
[BGW05], [WQZD10], [KMW23]

Committed Value WE
[GS17], [BL20], [CFK24]

Hash-Proof Systems
[CS02], [BC16]

Registered Functional Encryption
[FFM+23]

Laconic PSI/OT
[ALOS22], [DKL+23], [FHAS24], [BGJP25]

Registered Attribute Based Encryption
[HLWW23], [FWW23], [ZZGQ23], [AT24], [GLWW24]

Silent Threshold Encryption
[GKPW24]

Batched Threshold Encryption
[CGPP24], [CGPW24], [AFP24]

**not an exhaustive list*

Many Examples!

Over the last 25 years:

Registration Based Encryption
[GHMR18], [GKMR23], [FKdP23]

Hash Encryption

Identity-Based Encryption

At first glance, constructions seem "arbitrary" and unrelated 😕

Can we <u>systematically</u> study special purpose WE?

Laconic PSI/OT
[ALOS22], [DKL+23], [FHAS24], [BGJP25]

Registered Attribute Based Encryption
[HLWW23], [FWW23], [ZZGQ23], [AT24], [GLWW24]

Silent Threshold Encryption
[GKPW24]

Batched Threshold Encryption
[CGPP24], [CGPW24], [AFP24]

**not an exhaustive list*

Taxonomy of WE

Taxonomy of WE schemes

$(x, w) \in R$

Gen 1

$\text{Enc}(x, m) \rightarrow \text{ct}$

$\text{Dec}(w, \text{ct}) \rightarrow m$

Taxonomy of WE schemes

$(x, w) \in R$

Gen 1

$\mathsf{Enc}(x, m) \rightarrow \mathsf{ct}$

$\mathsf{Dec}(w, \mathsf{ct}) \rightarrow m$

$T_E = T_D = O(|R|)$

$|\mathsf{ct}| = O(|R|)$

Taxonomy of WE schemes

$(x, w) \in R$

Gen 1

Gen 2

$\text{Enc}(x, m) \to \text{ct}$

$\text{Dec}(w, \text{ct}) \to m$

$T_E = T_D = O(|R|)$

$|\text{ct}| = O(|R|)$

Taxonomy of WE schemes

$(x, w) \in R$

Gen 1	Gen 2
$\text{Enc}(x, m) \to \text{ct}$	$h \leftarrow \text{Hash}(x)$
$\text{Dec}(w, \text{ct}) \to m$	$\lvert h \rvert \ll \lvert x \rvert$

$T_E = T_D = O(\lvert R \rvert)$
$\lvert \text{ct} \rvert = O(\lvert R \rvert)$

Taxonomy of WE schemes

$(x, w) \in R$

Gen 1	Gen 2
$\text{Enc}(x, m) \to \text{ct}$	$h \leftarrow \text{Hash}(x)$
$\text{Dec}(w, \text{ct}) \to m$	$\lvert h \rvert \ll \lvert x \rvert$
	$\text{Enc}(h, m) \to \text{ct}$

$T_E = T_D = O(\lvert R \rvert)$

$\lvert \text{ct} \rvert = O(\lvert R \rvert)$

Taxonomy of WE schemes

$(x, w) \in R$

Gen 1	Gen 2
$\text{Enc}(x, m) \to \text{ct}$	$h \leftarrow \text{Hash}(x)$
$\text{Dec}(w, \text{ct}) \to m$	$\lvert h \rvert \ll \lvert x \rvert$
	$\text{Enc}(h, m) \to \text{ct}$
	$\text{Dec}(w, \text{ct}) \to m$

$T_E = T_D = O(\lvert R \rvert)$

$\lvert \text{ct} \rvert = O(\lvert R \rvert)$

Taxonomy of WE schemes

$(x, w) \in R$

Gen 1	Gen 2
$\text{Enc}(x, m) \to \text{ct}$	$h \leftarrow \text{Hash}(x)$
$\text{Dec}(w, \text{ct}) \to m$	$\lvert h \rvert \ll \lvert x \rvert$
	$\text{Enc}(h, m) \to \text{ct}$
	$\text{Dec}(w, \text{ct}) \to m$

Can build "Laconic" primitives!

- **Laconic OT**: Hash the receiver's choice bits
- **Laconic PSI**: Hash the receiver's database

$$T_E = T_D = O(\lvert R \rvert)$$
$$\lvert \text{ct} \rvert = O(\lvert R \rvert)$$

Taxonomy of WE schemes

$(x, w) \in R$

Gen 1	Gen 2	Gen 3
$\mathsf{Enc}(x, m) \rightarrow \mathsf{ct}$	$h \leftarrow \mathsf{Hash}(x)$	
$\mathsf{Dec}(w, \mathsf{ct}) \rightarrow m$	$\lvert h \rvert \ll \lvert x \rvert$	
	$\mathsf{Enc}(h, m) \rightarrow \mathsf{ct}$	
	$\mathsf{Dec}(w, \mathsf{ct}) \rightarrow m$	

$$T_E = T_D = O(|R|)$$
$$|\mathsf{ct}| = O(|R|)$$

Taxonomy of WE schemes

$(x, w) \in R$

Gen 1	Gen 2	Gen 3
$\mathsf{Enc}(x, m) \rightarrow \mathsf{ct}$	$h \leftarrow \mathsf{Hash}(x)$	$(h, \pi) \in R'$
$\mathsf{Dec}(w, \mathsf{ct}) \rightarrow m$	$\lvert h \rvert \ll \lvert x \rvert$	$\lvert R' \rvert \ll \lvert R \rvert$
	$\mathsf{Enc}(h, m) \rightarrow \mathsf{ct}$	
	$\mathsf{Dec}(w, \mathsf{ct}) \rightarrow m$	

$T_E = T_D = O(\lvert R \rvert)$

$\lvert \mathsf{ct} \rvert = O(\lvert R \rvert)$

Taxonomy of WE schemes

$(x, w) \in R$

Computational Reduction
(SNARK the relation!)

Gen 1	Gen 2	Gen 3
$\mathsf{Enc}(x, m) \to \mathsf{ct}$	$h \leftarrow \mathsf{Hash}(x)$	$(h, \pi) \in R'$
$\mathsf{Dec}(w, \mathsf{ct}) \to m$	$\lvert h \rvert \ll \lvert x \rvert$	$\lvert R' \rvert \ll \lvert R \rvert$
	$\mathsf{Enc}(h, m) \to \mathsf{ct}$	
	$\mathsf{Dec}(w, \mathsf{ct}) \to m$	

$T_E = T_D = O(\lvert R \rvert)$
$\lvert \mathsf{ct} \rvert = O(\lvert R \rvert)$

Taxonomy of WE schemes

$(x, w) \in R$

Computational Reduction
(SNARK the relation!)

Gen 1	Gen 2	Gen 3
$\mathsf{Enc}(x, m) \to \mathsf{ct}$	$h \leftarrow \mathsf{Hash}(x)$	$(h, \pi) \in R'$
$\mathsf{Dec}(w, \mathsf{ct}) \to m$	$\lvert h \rvert \ll \lvert x \rvert$	$\lvert R' \rvert \ll \lvert R \rvert$
	$\mathsf{Enc}(h, m) \to \mathsf{ct}$	$\mathsf{Enc}(h, m) \to \mathsf{ct}$
	$\mathsf{Dec}(w, \mathsf{ct}) \to m$	

$$T_E = T_D = O(\lvert R \rvert)$$
$$\lvert \mathsf{ct} \rvert = O(\lvert R \rvert)$$

Taxonomy of WE schemes

$(x, w) \in R$

Computational Reduction
(SNARK the relation!)

Gen 1	Gen 2	Gen 3
$\text{Enc}(x, m) \to \text{ct}$	$h \leftarrow \text{Hash}(x)$	$(h, \pi) \in R'$
$\text{Dec}(w, \text{ct}) \to m$	$\lvert h \rvert \ll \lvert x \rvert$	$\lvert R' \rvert \ll \lvert R \rvert$
	$\text{Enc}(h, m) \to \text{ct}$	$\text{Enc}(h, m) \to \text{ct}$
	$\text{Dec}(w, \text{ct}) \to m$	$\text{Dec}(\pi, \text{ct}) \to m$

$T_E = T_D = O(\lvert R \rvert)$

$\lvert \text{ct} \rvert = O(\lvert R \rvert)$

Taxonomy of WE schemes

$(x, w) \in R$

Computational Reduction
(SNARK the relation!)

Gen 1	Gen 2	Gen 3
$\mathsf{Enc}(x, m) \to \mathsf{ct}$	$h \leftarrow \mathsf{Hash}(x)$	$(h, \pi) \in R'$
$\mathsf{Dec}(w, \mathsf{ct}) \to m$	$\lvert h \rvert \ll \lvert x \rvert$	$\lvert R' \rvert \ll \lvert R \rvert$
	$\mathsf{Enc}(h, m) \to \mathsf{ct}$	$\mathsf{Enc}(h, m) \to \mathsf{ct}$
	$\mathsf{Dec}(w, \mathsf{ct}) \to m$	$\mathsf{Dec}(\pi, \mathsf{ct}) \to m$

Gen 1:
$T_E = T_D = O(\lvert R \rvert)$
$\lvert \mathsf{ct} \rvert = O(\lvert R \rvert)$

Gen 3:
$T_E = T_D = O(\lvert R' \rvert)$
$\lvert \mathsf{ct} \rvert = O(\lvert R' \rvert)$

Classification

Gen 1

Identity-Based Encryption
[BF01]

Hash-Proof Systems
[CS02], [BC16]

Committed Value WE
[GS17], [BL20]

Gen 2

Hash Encryption
[CDG+17], [DG17]

Gen 3

(Distributed) Broadcast Encryption
[BGW05], [WQZD10], [KMW23]

Registration Based Encryption
[GHMR18], [GKMR23], [FKdP23]

Registered Attribute Based Encryption
[HLWW23], [FWW23], [ZZGQ23], [AT24], [GLWW24]

Batched Threshold Encryption
[CGPP24], [CGPW24], [AFP24]

Committed Value WE
[CFK24]

Laconic PSI/OT
[ALOS22], [DKL+23], [FHAS24], [BGJP25]

Registered Functional Encryption
[FFM+23]

Silent Threshold Encryption
[GKPW24]

Classification

Gen 1

Identity-Based Encryption
[BF01]

Hash-Proof Systems
[CS02], [BC16]

Committed Value WE
[GS17], [BL20]

Gen 2

Today: A framework to build Gen 3 WE and applications

Hash Encryption
[CDG+17], [DG17]

Gen 3

(Distributed) Broadcast Encryption
[BGW05], [WQZD10], [KMW23]

Registration Based Encryption
[GHMR18], [GKMR23], [FKdP23]

Registered Attribute Based Encryption
[HLWW23], [FWW23], [ZZGQ23], [AT24], [GLWW24]

Batched Threshold Encryption
[CGPP24], [CGPW24], [AFP24]

Committed Value WE
[CFK24]

Laconic PSI/OT
[ALOS22], [DKL+23], [FHAS24], [BGJP25]

Registered Functional Encryption
[FFM+23]

Silent Threshold Encryption
[GKPW24]

Our Results

Our Results

Gadget-based framework for WE

Our Results

Gadget-based framework for WE

- Similar to ZK libraries: "glue" together gadgets written by experts

Our Results

Gadget-based framework for WE

- Similar to ZK libraries: “glue” together gadgets written by experts
- No prior knowledge of SNARKs needed!

Our Results

Gadget-based framework for WE

- Similar to ZK libraries: “glue” together gadgets written by experts
- No prior knowledge of SNARKs needed!
- Security in the GGM

Our Results

Gadget-based framework for WE

- Similar to ZK libraries: “glue” together gadgets written by experts
- No prior knowledge of SNARKs needed!
- Security in the GGM

Recover previous results

- Registration Based Encryption
- Distributed Broadcast Encryption
- Silent/Batched Threshold Encryption
- … and more!

Our Results

Gadget-based framework for WE

- Similar to ZK libraries: "glue" together gadgets written by experts
- No prior knowledge of SNARKs needed!
- Security in the GGM

Recover previous results

- Registration Based Encryption
- Distributed Broadcast Encryption
- Silent/Batched Threshold Encryption
- … and more!

Improve best known result [GLWW24]

Registered ABE with a Linear CRS

Our Results

Gadget-based framework for WE

- Similar to ZK libraries: “glue” together gadgets written by experts
- No prior knowledge of SNARKs needed!
- Security in the GGM

Recover previous results

- Registration Based Encryption
- Distributed Broadcast Encryption
- Silent/Batched Threshold Encryption
- … and more!

Improve best known result [GLWW24]

Registered ABE with a Linear CRS

New feasibility results

Registered Threshold Encryption

What class of relations support efficient WE?

Relations with “Linear” verifiers

Relations with “Linear” verifiers

Express the verification circuit for $R_L(x, w) = 1$ as a set of PPEs.

Relations with "Linear" verifiers

Express the verification circuit for $R_L(x, w) = 1$ as a set of PPEs.

$$\prod e(x_i, x_j) \cdot \prod e(x_i, w_j) \cdot \prod e(w_i, x_j) \cdot \prod e(w_i, w_j) = c_T$$

Relations with "Linear" verifiers

Express the verification circuit for $R_L(x, w) = 1$ as a set of PPEs.

$$\prod e(x_i, x_j) \cdot \prod e(x_i, w_j) \cdot \prod e(w_i, x_j) \cdot \prod e(w_i, w_j) = c_T$$

Relations with “Linear” verifiers

Express the verification circuit for $R_L(x, w) = 1$ as a set of PPEs.

$$\prod e(x_i, x_j) \cdot \prod e(x_i, w_j) \cdot \prod e(w_i, x_j) \cdot \prod e(w_i, w_j) = c_T$$

Compiler [BC16, BL20, GKPW24]: Linear PPE → WE

The Missing Piece

Linear Relation

PPE Constraint System: $\prod e(x_i, x_j) \cdot \prod e(x_i, w_j) \cdot \prod e(w_i, x_j) = c_T$

The Missing Piece

Natural Relation	$\mathcal{R} = \{(x, w) \mid f(x, w) = 1\}$
Linear Relation	PPE Constraint System: $\prod e(x_i, x_j) \cdot \prod e(x_i, w_j) \cdot \prod e(w_i, x_j) = c_T$

The Missing Piece

Natural Relation

$$\mathscr{R} = \{(x, w) \mid f(x, w) = 1\}$$

???

How do we *linearize* natural relations?
How do we leverage SNARK machinery for *succinctness*?

Linear Relation

PPE Constraint System: $\prod e(x_i, x_j) \cdot \prod e(x_i, w_j) \cdot \prod e(w_i, x_j) = c_T$

Closing the gap:
Our Framework to build WE

Our Framework

Our Framework

Goal: Simplify the process of translating:

Our Framework

Goal: Simplify the process of translating:

Natural Relations → Linear Relations

Our Framework

Goal: Simplify the process of translating:

Natural Relations → Linear Relations

- We provide "Gadgets" — Witness Encryption for useful relations:

Our Framework

Goal: Simplify the process of translating:

Natural Relations → Linear Relations

- We provide "Gadgets" — Witness Encryption for useful relations:

1. Signatures
2. Algebraic PRF
3. Inner Product
4. Zero Check
5. Degree Check

... and more!

Our Framework

Goal: Simplify the process of translating:

Natural Relations → Linear Relations

- We provide "Gadgets" — Witness Encryption for useful relations:

1. Signatures	4. Zero Check
2. Algebraic PRF	5. Degree Check
3. Inner Product	… and more!

- Gadgets can be composed to build WE for larger relations!

Our Framework

Goal: Simplify the process of translating:

Natural Relations → Linear Relations

- We provide "Gadgets" — Witness Encryption for useful relations:

1. Signatures	4. Zero Check
2. Algebraic PRF	5. Degree Check
3. Inner Product	… and more!

- Gadgets can be composed to build WE for larger relations!
- Gadgets fully capture succinctness!

Our Framework

Goal: Simplify the process of translating:

Natural Relations → Linear Relations

- We provide "Gadgets" — Witness Encryption for useful relations:

1. Signatures	4. Zero Check
2. Algebraic PRF	5. Degree Check
3. Inner Product	… and more!

- Gadgets can be composed to build WE for larger relations!
- Gadgets fully capture succinctness!
- Easy to use and extend with new gadgets!

Our Framework

Natural Relation	$\mathscr{R} = \{(x, w) \mid f(x, w) = 1\}$

Our Framework

Natural Relation

$$\mathscr{R} = \{(x, w) \mid f(x, w) = 1\}$$

This Work

Inner Product × Signature Check × Degree Check × ...

Our Framework

Natural Relation

$$\mathscr{R} = \{(x, w) \mid f(x, w) = 1\}$$

This Work ↓

Inner Product × Signature Check × Degree Check × ...

Linear Relation

PPE Constraint System: $\prod e(x_i, x_j) \cdot \prod e(x_i, w_j) \cdot \prod e(w_i, x_j) = c_T$

Remainder of the Talk

Gadget-based framework for WE

- Similar to ZK libraries: "glue" together gadgets written by experts
- No prior knowledge of SNARKs needed!
- Security in the GGM ✅

Recover previous results

- Registration Based Encryption
- Distributed Broadcast Encryption
- Silent/Batched Threshold Encryption
- … and more!

Improve best known result [GLWW24]

Registered ABE with a Linear CRS

New feasibility results

Registered Threshold Encryption

Remainder of the Talk

Gadget-based framework for WE

- Similar to ZK libraries: “glue” together gadgets written by experts
- No prior knowledge of SNARKs needed!
- Security in the GGM ✅

Recover previous results

- Registration Based Encryption
- Distributed Broadcast Encryption
- Silent/Batched Threshold Encryption
- … and more!

Improve best known result [GLWW24]

Registered ABE with a Linear CRS

New feasibility results

Registered Threshold Encryption

Remainder of the Talk

Gadget-based framework for WE

- Similar to ZK libraries: "glue" together gadgets written by experts
- No prior knowledge of SNARKs needed!
- Security in the GGM ✅

Recover previous results

- Registration Based Encryption
- Distributed Broadcast Encryption
- Silent/Batched Threshold Encryption
- … and more!

Improve best known result [GLWW24]

Registered ABE with a Linear CRS

New feasibility results

Registered Threshold Encryption

Distributed Broadcast Encryption

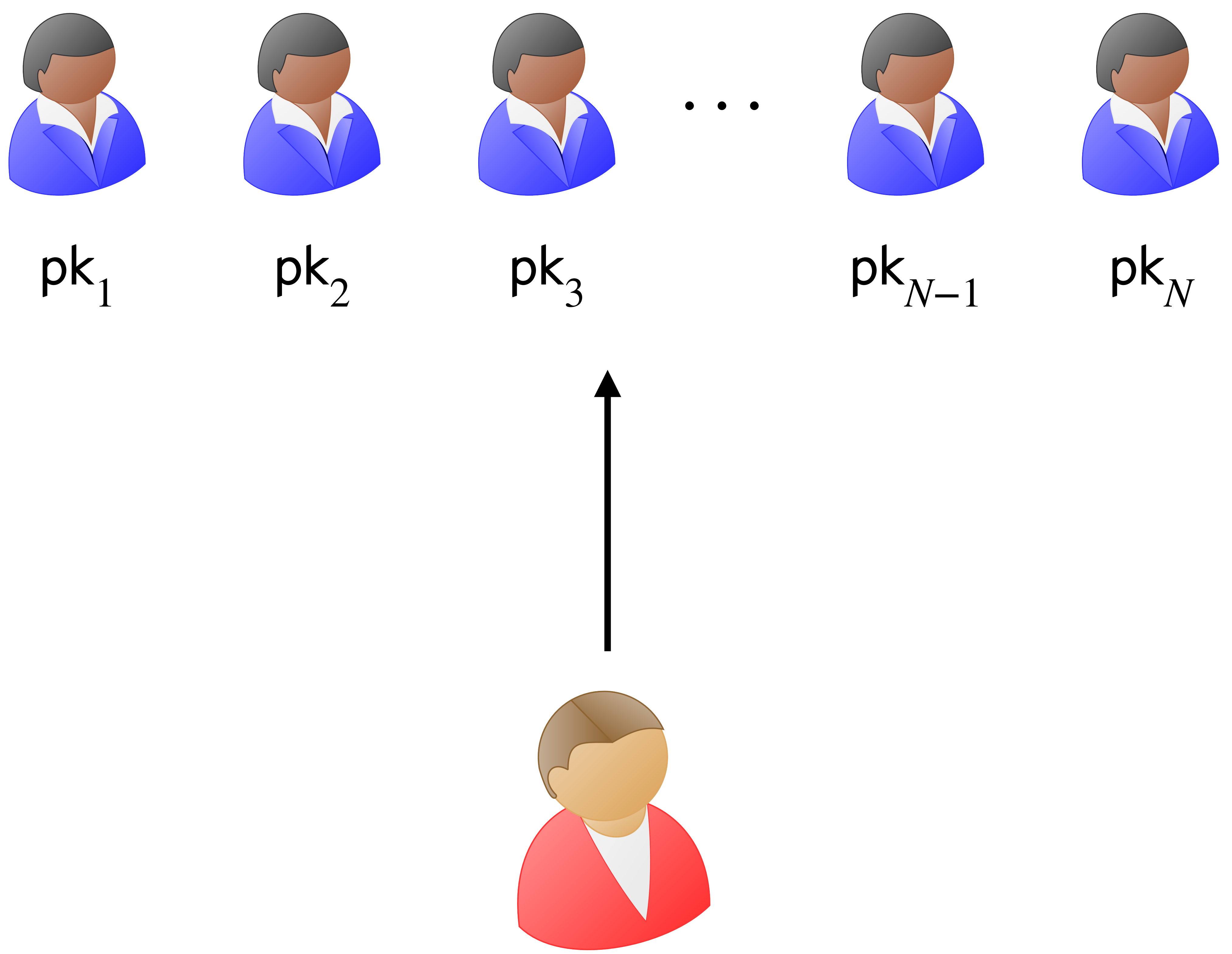
Distributed Broadcast Encryption [WQZD10,BZ14]

Goal: Send a message to n parties with $|\mathsf{ct}| = O(1)$

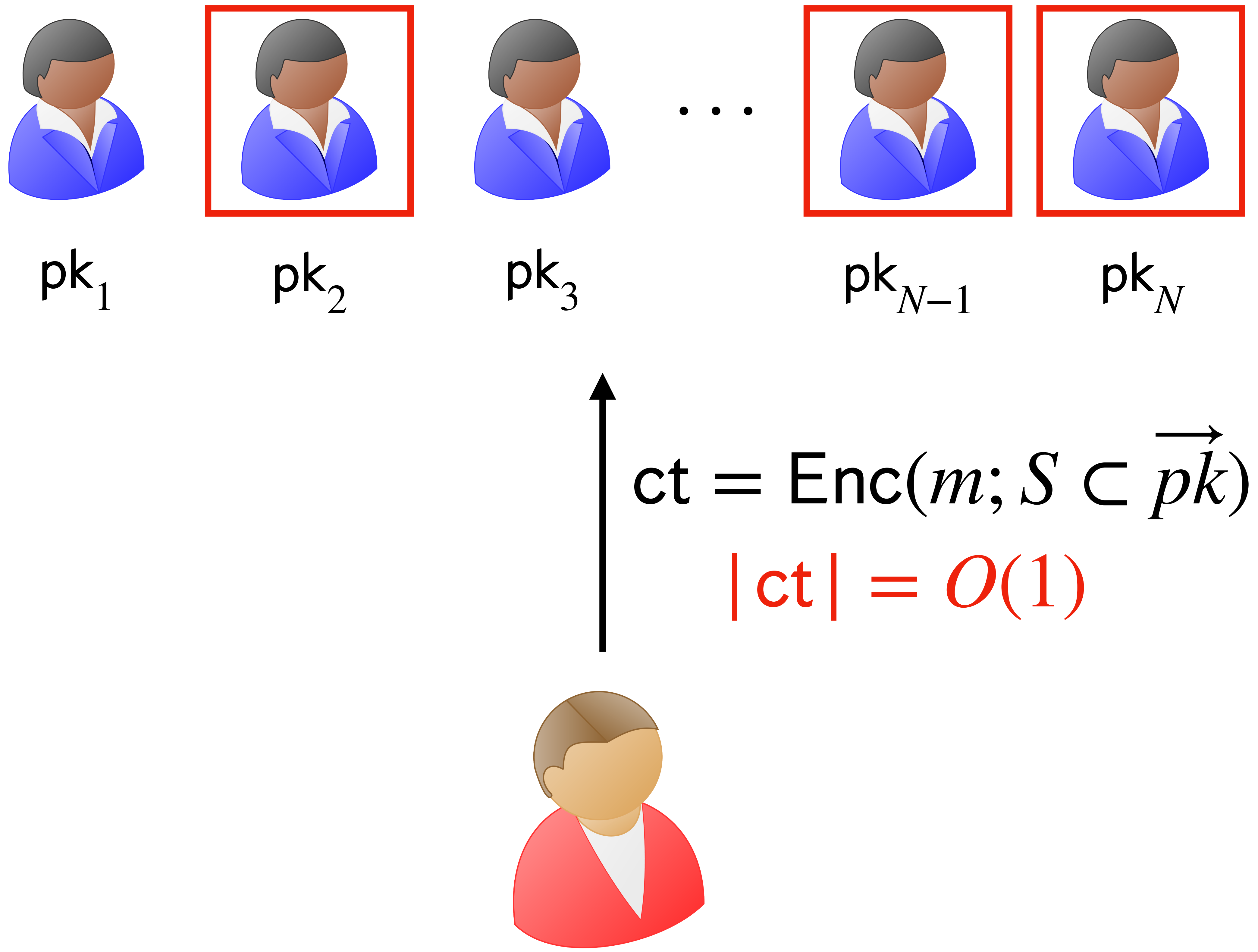
No interaction during setup except for a PKI.

(Note: Can achieve $|\mathsf{ct}| = O(n)$ using public key encryption)

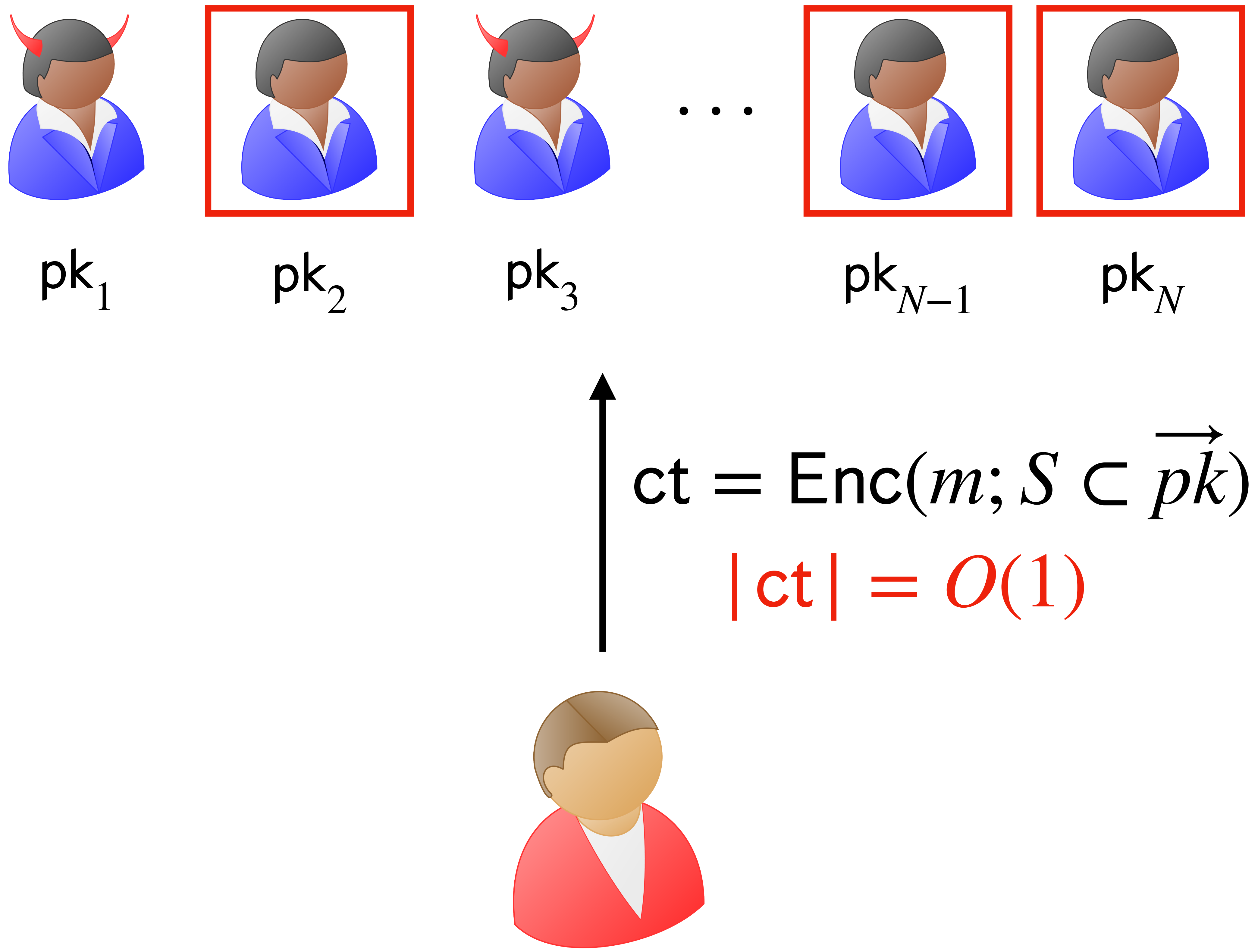
Distributed Broadcast Encryption [WQZD10,BZ14]



Distributed Broadcast Encryption [WQZD10,BZ14]



Distributed Broadcast Encryption [WQZD10,BZ14]



Semantic security against $S' = \overrightarrow{pk} \backslash S$

Step #1: Identify a Relation

Goal: Send a message to n parties with $|\mathsf{ct}| = O(1)$

Step #1: Identify a Relation

Goal: Send a message to n parties with $|\mathsf{ct}| = O(1)$

Let each party have a public key pair:

$$\{\mathsf{pk}_i = g^{\mathsf{sk}_i}\}_{i \in [n]} \in \mathbb{G}_1$$

Step #1: Identify a Relation

Goal: Send a message to n parties with $|\mathsf{ct}| = O(1)$

Let each party have a public key pair:

$$\{\mathsf{pk}_i = g^{\mathsf{sk}_i}\}_{i \in [n]} \in \mathbb{G}_1$$

Suppose we had a *succinct* ($|\mathsf{ct}| = O(1)$) WE for the following relation:

Step #1: Identify a Relation

Goal: Send a message to n parties with $|\mathsf{ct}| = O(1)$

Let each party have a public key pair:

$$\{\mathsf{pk}_i = g^{\mathsf{sk}_i}\}_{i \in [n]} \in \mathbb{G}_1$$

Suppose we had a *succinct* ($|\mathsf{ct}| = O(1)$) WE for the following relation:

You can decrypt my ciphertext iff you know a secret key

$$\{\mathsf{sk}_i : (\mathsf{pk}_i = g^{\mathsf{sk}_i}) \wedge (\mathsf{pk}_i \in \overrightarrow{\mathsf{pk}})\}$$

How do we build the WE?

Inner-Product Gadget

Inner-Product Gadget

Statement: $(u_1, u_2, \ldots, u_n) \in \mathbb{G}^n, \; v \in \mathbb{G}$

Inner-Product Gadget

Statement: $(u_1, u_2, \ldots, u_n) \in \mathbb{G}^n,\ v \in \mathbb{G}$

Witness: $(w_1, w_2, \ldots, w_n) \in \mathbb{F}^n$ such that $\prod_i u_i^{w_i} = v$.

Inner-Product Gadget

Statement: $(u_1, u_2, \ldots, u_n) \in \mathbb{G}^n, \; v \in \mathbb{G}$

Witness: $(w_1, w_2, \ldots, w_n) \in \mathbb{F}^n$ such that $\prod_i u_i^{w_i} = v$.

> You can decrypt my ciphertext iff you know $\vec{w}$ such that:
> $$\prod_i u_i^{w_i} = v$$

Inner-Product Gadget

Statement: $(u_1, u_2, \ldots, u_n) \in \mathbb{G}^n,\ v \in \mathbb{G}$

Witness: $(w_1, w_2, \ldots, w_n) \in \mathbb{F}^n$ such that $\prod_i u_i^{w_i} = v$.

> You can decrypt my ciphertext iff you know $\overrightarrow{w}$ such that:
>
> $$\prod_i u_i^{w_i} = v$$

1. The above WE has $|\mathrm{ct}| = O(1)$!

Inner-Product Gadget

Statement: $(u_1, u_2, \ldots, u_n) \in \mathbb{G}^n, \; v \in \mathbb{G}$

Witness: $(w_1, w_2, \ldots, w_n) \in \mathbb{F}^n$ such that $\prod_i u_i^{w_i} = v$.

You can decrypt my ciphertext iff you know $\vec{w}$ such that:

$$\prod_i u_i^{w_i} = v$$

1. The above WE has $|\mathsf{ct}| = O(1)$!
2. Enc takes as input a <u>succinct</u> commitment to $\vec{u}$, and runs in $O(1)$ time

Step #2: Reduce DBE $\rightarrow$ Inner-Product

Step #2: Reduce DBE → Inner-Product

- Each user has a public key pair:

$$\{\mathsf{pk}_i = g^{\mathsf{sk}_i}\}_{i\in[n]} \in \mathbb{G}$$

Step #2: Reduce DBE → Inner-Product

- Each user has a public key pair:

$$\{\mathsf{pk}_i = g^{\mathsf{sk}_i}\}_{i \in [n]} \in \mathbb{G}$$

- $\mathsf{Enc}(m, \overrightarrow{\mathsf{pk}})$:
  - Encrypt using the Inner-Product gadget with $\vec{u} = \overrightarrow{\mathsf{pk}}$ and $v = g$

Step #2: Reduce DBE → Inner-Product

- Each user has a public key pair:

$$\{\mathsf{pk}_i = g^{\mathsf{sk}_i}\}_{i \in [n]} \in \mathbb{G}$$

- Enc$(m, \overrightarrow{\mathsf{pk}})$:
  - Encrypt using the Inner-Product gadget with $\vec{u} = \overrightarrow{\mathsf{pk}}$ and $v = g$

You can decrypt my ciphertext iff you know $\vec{w}$ such that:

$$\prod_i \mathsf{pk}_i^{w_i} = g$$

Step #2: Reduce DBE → Inner-Product

- Each user has a public key pair:

$$\{\mathsf{pk}_i = g^{\mathsf{sk}_i}\}_{i \in [n]} \in \mathbb{G}$$

- Enc$(m, \overrightarrow{\mathsf{pk}})$:
  - Encrypt using the Inner-Product gadget with $\vec{u} = \overrightarrow{\mathsf{pk}}$ and $v = g$

You can decrypt my ciphertext iff you know $\vec{w}$ such that:

$$\prod_i \mathsf{pk}_i^{w_i} = g$$

Honest users decrypt using $\vec{w} = (0, \ldots, \mathsf{sk}_i^{-1}, \ldots, 0)$

Step #2: Reduce DBE → Inner-Product

- Each user has a public key pair:

$$\{\mathsf{pk}_i = g^{\mathsf{sk}_i}\}_{i \in [n]} \in \mathbb{G}$$

- Enc$(m, \overrightarrow{\mathsf{pk}})$:
  - Encrypt using the Inner-Product gadget with $\vec{u} = \overrightarrow{\mathsf{pk}}$ and $v = g$

You can decrypt my ciphertext iff you know $\vec{w}$ such that:

$$\prod_i \mathsf{pk}_i^{w_i} = g$$

Adversary can be reduced to solving DLOG

Registered Attribute Based Encryption

Registered Attribute Based Encryption [HLWW23]

Goal: Attribute Based Encryption without a Trusted Party

No interaction during setup except for a PKI

+ some notions of efficiency

Registered Attribute Based Encryption [HLWW23]

M users

Registered Attribute Based Encryption [HLWW23]

M users

Bulletin Board					
Public Key:	pk_1	pk_2	pk_3	pk_4	pk_5
Region:	EU	EU	USA	EU	USA
Area:	Crypto	Crypto	ML	ML	Crypto

Registered Attribute Based Encryption [HLWW23]

M users

Bulletin Board					
Public Key:	pk_1	pk_2	pk_3	pk_4	pk_5
Region:	EU	EU	USA	EU	USA
Area:	Crypto	Crypto	ML	ML	Crypto

Want to encrypt a message to “All cryptographers in EU region” … but:

Registered Attribute Based Encryption [HLWW23]

M users

Bulletin Board					
Public Key:	pk_1	pk_2	pk_3	pk_4	pk_5
Region:	EU	EU	USA	EU	USA
Area:	Crypto	Crypto	ML	ML	Crypto

Want to encrypt a message to "All cryptographers in EU region" … but:

1. Don't want to read the entire bulletin board

Registered Attribute Based Encryption [HLWW23]

M users

Bulletin Board					
Public Key:	pk_1	pk_2	pk_3	pk_4	pk_5
Region:	EU	EU	USA	EU	USA
Area:	Crypto	Crypto	ML	ML	Crypto

Want to encrypt a message to "All cryptographers in EU region" … but:

1. Don't want to read the entire bulletin board
2. Enc, Dec, and $|ct|$ should be succinct — polylog(M) (# of users)

Registered Attribute Based Encryption [HLWW23]

M users

Bulletin Board					
Public Key:	pk_1	pk_2	pk_3	pk_4	pk_5
Region:	EU	EU	USA	EU	USA
Area:	Crypto	Crypto	ML	ML	Crypto

Want to encrypt a message to "All cryptographers in EU region" … but:

1. Don't want to read the entire bulletin board → Will compress using a "helper". Only trusted for integrity.
2. Enc, Dec, and $|ct|$ should be succinct — $\mathrm{polylog}(M)$ (# of users)

Registered Attribute Based Encryption [HLWW23]

M users

Bulletin Board					
Public Key:	pk_1	pk_2	pk_3	pk_4	pk_5
Region:	EU	EU	USA	EU	USA
Area:	Crypto	Crypto	ML	ML	Crypto

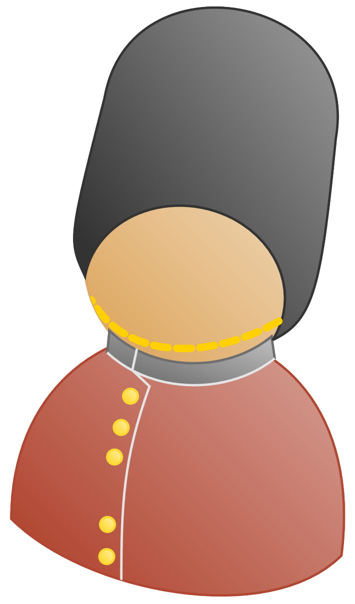
Want to encrypt a message to "All cryptographers in EU region" … but:

1. Don't want to read the entire bulletin board → Will compress using a "helper". Only trusted for integrity.
2. Enc, Dec, and $|ct|$ should be succinct — $\text{polylog}(M)$ (# of users) → Gen 3 WE

Registered Attribute Based Encryption [HLWW23]

M users

Bulletin Board					
Public Key:	pk_1	pk_2	pk_3	pk_4	pk_5
Region:	EU	EU	USA	EU	USA
Area:	Crypto	Crypto	ML	ML	Crypto



Key Curator

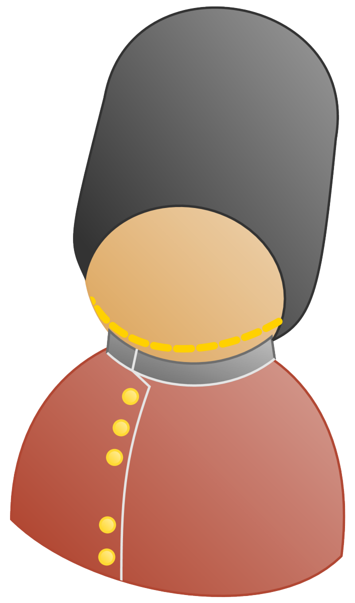
↓

Short aPK

Registered Attribute Based Encryption [HLWW23]

M users

Bulletin Board					
Public Key:	pk_1	pk_2	pk_3	pk_4	pk_5
Region:	EU	EU	USA	EU	USA
Area:	Crypto	Crypto	ML	ML	Crypto



Key Curator

↓

Short aPK

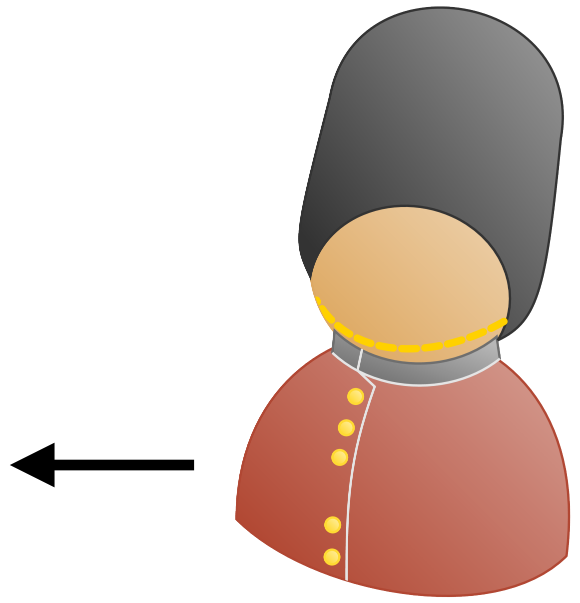
$$\text{Enc}(\text{aPK}, m, \text{"Crypto"} \wedge \text{"EU"}) \to |\text{ct}|$$

Registered Attribute Based Encryption [HLWW23]

M users

Bulletin Board					
Public Key:	pk_1	pk_2	pk_3	pk_4	pk_5
Region:	EU	EU	USA	EU	USA
Area:	Crypto	Crypto	ML	ML	Crypto

Helper Key:	hk_1	hk_2	hk_3	hk_4	hk_5



Key Curator

↓

Short aPK

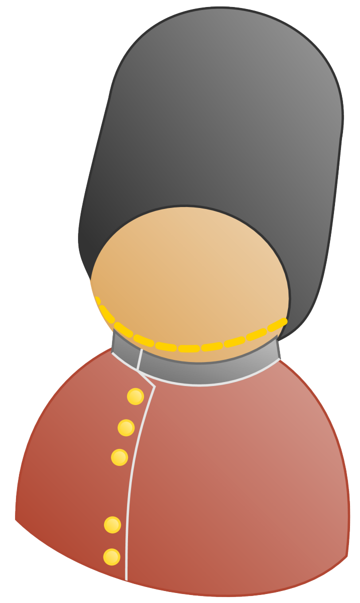
$$\text{Enc}(\text{aPK}, m, \text{"Crypto"} \land \text{"EU"}) \rightarrow |\text{ct}|$$

Registered Attribute Based Encryption [HLWW23]

M users

Bulletin Board					
Public Key:	pk_1	pk_2	pk_3	pk_4	pk_5
Region:	EU	EU	USA	EU	USA
Area:	Crypto	Crypto	ML	ML	Crypto

Helper Key:	hk_1	hk_2	hk_3	hk_4	hk_5



Key Curator

Short aPK

$$\mathsf{Enc}(\mathsf{aPK}, m, \text{"Crypto"} \wedge \text{"EU"}) \rightarrow |\mathsf{ct}|$$

$$\mathsf{Dec}(\mathsf{aPK}, \mathsf{hk}_2, \mathsf{sk}_2, \mathsf{ct}) \rightarrow m$$

Step #1: Identify a Relation

M users

Bulletin Board					
USA:	-	-	pk_3	-	pk_5
EU:	pk_1	pk_2	-	pk_4	-
Crypto:	pk_1	pk_2	-	-	pk_5
ML:	-	-	pk_3	pk_4	-

Step #1: Identify a Relation

M users

Bulletin Board					
USA:	-	-	pk_3	-	pk_5
EU:	pk_1	pk_2	-	pk_4	-
Crypto:	pk_1	pk_2	-	-	pk_5
ML:	-	-	pk_3	pk_4	-

You can decrypt my ciphertext iff you know a secret key

$$\{sk : (pk = g^{sk}) \wedge (pk \in \text{Crypto}) \wedge (pk \in \text{EU})\}$$

How do we build it?

Bulletin Board					
EU:	pk_1	pk_2	0	pk_4	0
Crypto:	pk_1	pk_2	0	0	pk_5

How do we build it?

Bulletin Board					
EU:	pk_1	pk_2	0	pk_4	0
Crypto:	pk_1	pk_2	0	0	pk_5

You can decrypt my ciphertext iff you know a secret key

$$\{sk : (pk = g^{sk}) \wedge (pk \in Crypto) \wedge (pk \in EU)\}$$

How do we build it?

Bulletin Board					
EU:	pk_1	pk_2	0	pk_4	0
Crypto:	pk_1	pk_2	0	0	pk_5

You can decrypt my ciphertext iff you know a secret key

$$\{\mathsf{sk} : (\mathsf{pk} = g^{\mathsf{sk}}) \wedge (\mathsf{pk} \in \mathsf{Crypto}) \wedge (\mathsf{pk} \in \mathsf{EU})\}$$

$$\{\mathsf{sk}, \vec{w} : (\mathsf{pk} = g^{\mathsf{sk}}) \wedge (\mathsf{pk} = \prod \mathsf{Crypto}_i^{w_i}) \wedge (\mathsf{pk} = \prod \mathsf{EU}_i^{w_i})\}$$

How do we build it?

Bulletin Board					
EU:	pk_1	pk_2	0	pk_4	0
Crypto:	pk_1	pk_2	0	0	pk_5

You can decrypt my ciphertext iff you know a secret key

$$\{\mathsf{sk} : (\mathsf{pk} = g^{\mathsf{sk}}) \wedge (\mathsf{pk} \in \mathsf{Crypto}) \wedge (\mathsf{pk} \in \mathsf{EU})\}$$

$$\{\mathsf{sk}, \vec{w} : (\mathsf{pk} = g^{\mathsf{sk}}) \wedge (\mathsf{pk} = \prod \mathsf{Crypto}_i^{w_i}) \wedge (\mathsf{pk} = \prod \mathsf{EU}_i^{w_i})\}$$

Almost works… but adversary can use "empty" slots

Zero-Check Gadget

Zero-Check Gadget

Statement: $S \subset [n]$

Zero-Check Gadget

Statement: $S \subset [n]$

Witness: $(w_1, w_2, \ldots, w_n) \in \mathbb{F}^n$ such that $\{w_i = 0\}_{i \in S}$

Zero-Check Gadget

Statement: $S \subset [n]$

Witness: $(w_1, w_2, \ldots, w_n) \in \mathbb{F}^n$ such that $\{w_i = 0\}_{i \in S}$

> You can decrypt my ciphertext iff you know $\vec{w}$ such that:
>
> $$\{w_i = 0\}_{i \in S}$$

Zero-Check Gadget

Statement: $S \subset [n]$

Witness: $(w_1, w_2, \ldots, w_n) \in \mathbb{F}^n$ such that $\{w_i = 0\}_{i \in S}$

You can decrypt my ciphertext iff you know $\vec{w}$ such that:

$$\{w_i = 0\}_{i \in S}$$

1. The above WE has $|\mathsf{ct}| = O(1)$!

Zero-Check Gadget

Statement: $S \subset [n]$

Witness: $(w_1, w_2, \ldots, w_n) \in \mathbb{F}^n$ such that $\{w_i = 0\}_{i \in S}$

You can decrypt my ciphertext iff you know $\vec{w}$ such that:

$$\{w_i = 0\}_{i \in S}$$

1. The above WE has $|\text{ct}| = O(1)$!
2. Enc takes as input a succinct commitment to S, and runs in $O(1)$ time

Inner-Product + Zero-Check → rABE

Bulletin Board					
EU:	pk_1	pk_2	0	pk_4	0
Crypto:	pk_1	pk_2	0	0	pk_5

Inner-Product + Zero-Check → rABE

Bulletin Board					
EU:	pk_1	pk_2	0	pk_4	0
Crypto:	pk_1	pk_2	0	0	pk_5

You can decrypt my ciphertext iff you know a secret key

$$\{\mathsf{sk} : (\mathsf{pk} = g^{\mathsf{sk}}) \wedge (\mathsf{pk} \in \mathsf{Crypto}) \wedge (\mathsf{pk} \in \mathsf{EU})\}$$

Inner-Product + Zero-Check → rABE

Bulletin Board					
EU:	pk_1	pk_2	0	pk_4	0
Crypto:	pk_1	pk_2	0	0	pk_5

You can decrypt my ciphertext iff you know a secret key

$$\{\mathsf{sk} : (\mathsf{pk} = g^{\mathsf{sk}}) \wedge (\mathsf{pk} \in \mathsf{Crypto}) \wedge (\mathsf{pk} \in \mathsf{EU})\}$$

$$\{\mathsf{sk}, \vec{w} : (\mathsf{pk} = g^{\mathsf{sk}}) \wedge (\mathsf{pk} = \prod \mathsf{Crypto}_i^{w_i}) \wedge (\mathsf{pk} = \prod \mathsf{EU}_i^{w_i}) \wedge \{w_i = 0\}_{i \in Z_{\mathsf{Crypto}}} \wedge \{w_i = 0\}_{i \in Z_{\mathsf{EU}}} \wedge \vec{w} \neq 0\}$$

Efficiency Analysis

$\overrightarrow{\text{Crypto}} = (\text{pk}_1, \text{pk}_2, 0, 0, \text{pk}_5)$ $\overrightarrow{\text{EU}} = (\text{pk}_1, \text{pk}_2, 0, \text{pk}_4, 0)$

$$\{\text{sk}, \vec{w} : (\text{pk} = g^{\text{sk}}) \wedge (\text{pk} = \prod \text{Crypto}_i^{w_i}) \wedge (\text{pk} = \prod \text{EU}_i^{w_i})$$
$$\wedge \{w_i = 0\}_{i \in Z_{\text{Crypto}}} \wedge \{w_i = 0\}_{i \in Z_{\text{EU}}} \wedge \vec{w} \neq 0\}$$

Efficiency Analysis

$$\overrightarrow{\mathsf{Crypto}} = (\mathsf{pk}_1, \mathsf{pk}_2, 0, 0, \mathsf{pk}_5) \qquad \overrightarrow{\mathsf{EU}} = (\mathsf{pk}_1, \mathsf{pk}_2, 0, \mathsf{pk}_4, 0)$$

$$\{\mathsf{sk}, \vec{w} : (\mathsf{pk} = g^{\mathsf{sk}}) \wedge (\mathsf{pk} = \prod \mathsf{Crypto}_i^{w_i}) \wedge (\mathsf{pk} = \prod \mathsf{EU}_i^{w_i}) \wedge \{w_i = 0\}_{i \in Z_{\mathsf{Crypto}}} \wedge \{w_i = 0\}_{i \in Z_{\mathsf{EU}}} \wedge \vec{w} \neq 0\}$$

Key Curator computes:

Efficiency Analysis

$\overrightarrow{\text{Crypto}} = (\text{pk}_1, \text{pk}_2, 0, 0, \text{pk}_5)$ $\overrightarrow{\text{EU}} = (\text{pk}_1, \text{pk}_2, 0, \text{pk}_4, 0)$

$$\{\text{sk}, \vec{w} : (\text{pk} = g^{\text{sk}}) \wedge (\text{pk} = \prod \text{Crypto}_i^{w_i}) \wedge (\text{pk} = \prod \text{EU}_i^{w_i})$$
$$\wedge \{w_i = 0\}_{i \in Z_{\text{Crypto}}} \wedge \{w_i = 0\}_{i \in Z_{\text{EU}}} \wedge \vec{w} \neq 0\}$$

Key Curator computes:

1. *Succinct* commitments to $\overrightarrow{\text{Crypto}}$, $\overrightarrow{\text{EU}}$, and Z_{Crypto}, Z_{EU}

Efficiency Analysis

$$\overrightarrow{\text{Crypto}} = (\text{pk}_1, \text{pk}_2, 0, 0, \text{pk}_5) \qquad \overrightarrow{\text{EU}} = (\text{pk}_1, \text{pk}_2, 0, \text{pk}_4, 0)$$

$$\{\text{sk}, \vec{w} : (\text{pk} = g^{\text{sk}}) \wedge (\text{pk} = \prod \text{Crypto}_i^{w_i}) \wedge (\text{pk} = \prod \text{EU}_i^{w_i}) \wedge \{w_i = 0\}_{i \in Z_{\text{Crypto}}} \wedge \{w_i = 0\}_{i \in Z_{\text{EU}}} \wedge \vec{w} \neq 0\}$$

Key Curator computes:

1. *Succinct* commitments to $\overrightarrow{\text{Crypto}}$, $\overrightarrow{\text{EU}}$, and Z_{Crypto}, Z_{EU}
2. **Helper key:** Witness for Inner Product and Zero Check

Efficiency Analysis

$\overrightarrow{\text{Crypto}} = (\text{pk}_1, \text{pk}_2, 0, 0, \text{pk}_5)$ $\overrightarrow{\text{EU}} = (\text{pk}_1, \text{pk}_2, 0, \text{pk}_4, 0)$

$$\{\text{sk}, \overrightarrow{w} : (\text{pk} = g^{\text{sk}}) \wedge (\text{pk} = \prod \text{Crypto}_i^{w_i}) \wedge (\text{pk} = \prod \text{EU}_i^{w_i})$$
$$\wedge \{w_i = 0\}_{i \in Z_{\text{Crypto}}} \wedge \{w_i = 0\}_{i \in Z_{\text{EU}}} \wedge \overrightarrow{w} \neq 0\}$$

Key Curator computes:

1. *Succinct* commitments to $\overrightarrow{\text{Crypto}}$, $\overrightarrow{\text{EU}}$, and Z_{Crypto}, Z_{EU}
2. **Helper key:** Witness for Inner Product and Zero Check

Avoid reading entire bulletin board AND $(\text{Enc}, |\text{ct}|, \text{Dec})$ **are succinct**

Line of Work on Improving CRS

	$\lvert \mathsf{crs} \rvert$	Policy	Setting
[HLWW23, §5]	1	Circuit	iO
[FWW23]	1	Circuit	WE
[HLWW23, §7]	$\lvert\mathbb{U}\rvert M^2$	MSP	Composite, static
[ZZGQ23]	$\lvert\mathbb{U}\rvert M^2$	ABP	Prime, static
[AT24]	M^2	SP	Prime, static
[GLWW24, §4]	$M^{1+o(1)}$	MSP	Prime, q-type
[GLWW24, §5]	$\lvert\mathbb{U}\rvert M^{1+o(1)}$	MSP	Composite, static
Our Scheme	M	DNF	Prime, GGM

M users and $\lvert \mathbb{U} \rvert$ attributes

Matches CRS size of “weaker” primitives like RBE

Thank you!