A General Framework for Registered Functional Encryption via User-Specific Pre-Constraining

Tapas Pal[1] **Robert Schädlich**[2]

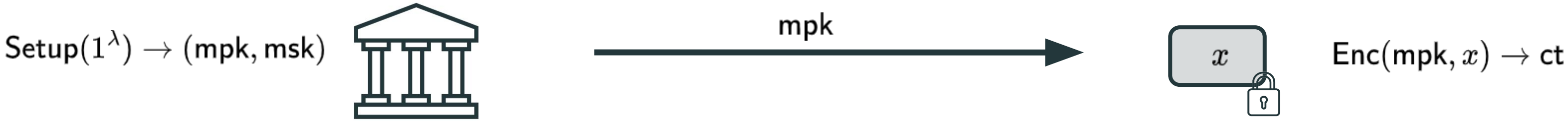
December 12, 2025

[1] Karlsruhe Institute of Technology, KASTEL Security Research Labs
[2] DIENS, École normale supérieure, PSL University, CNRS, Inria

Functional Encryption [TCC:BSW11]

$\mathsf{Setup}(1^\lambda) \to (\mathsf{mpk}, \mathsf{msk})$ 🏛

Functional Encryption [TCC:BSW11]



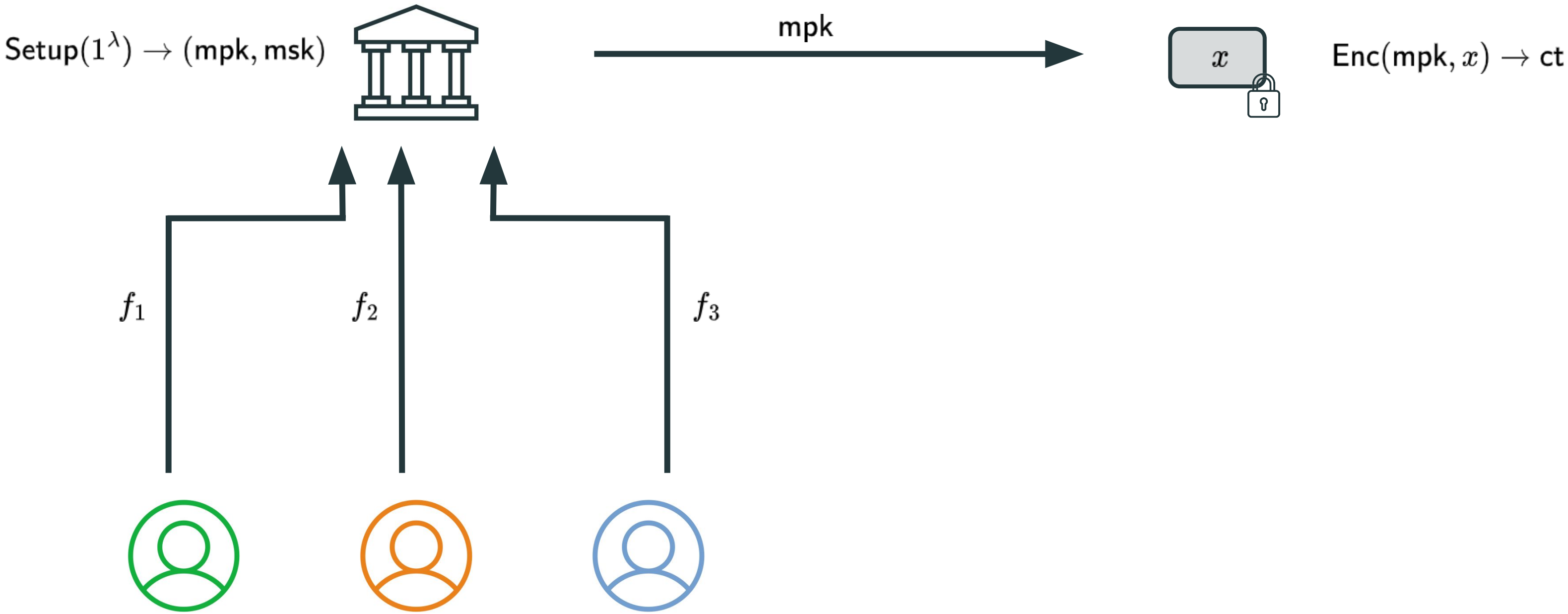
$\mathsf{Setup}(1^\lambda) \rightarrow (\mathsf{mpk}, \mathsf{msk})$

$\xrightarrow{\mathsf{mpk}}$

x

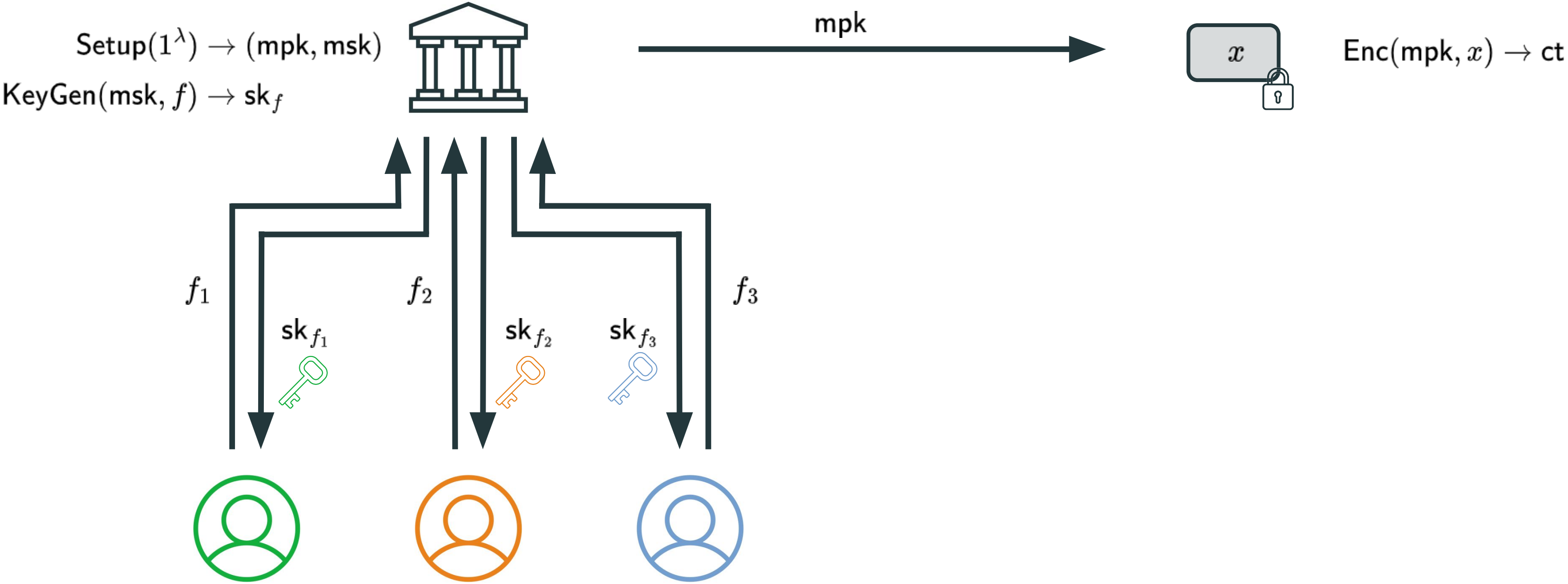
$\mathsf{Enc}(\mathsf{mpk}, x) \rightarrow \mathsf{ct}$

Functional Encryption [TCC:BSW11]



$\mathsf{Setup}(1^\lambda) \rightarrow (\mathsf{mpk}, \mathsf{msk})$
mpk
x
$\mathsf{Enc}(\mathsf{mpk}, x) \rightarrow \mathsf{ct}$
f_1
f_2
f_3

Functional Encryption [TCC:BSW11]

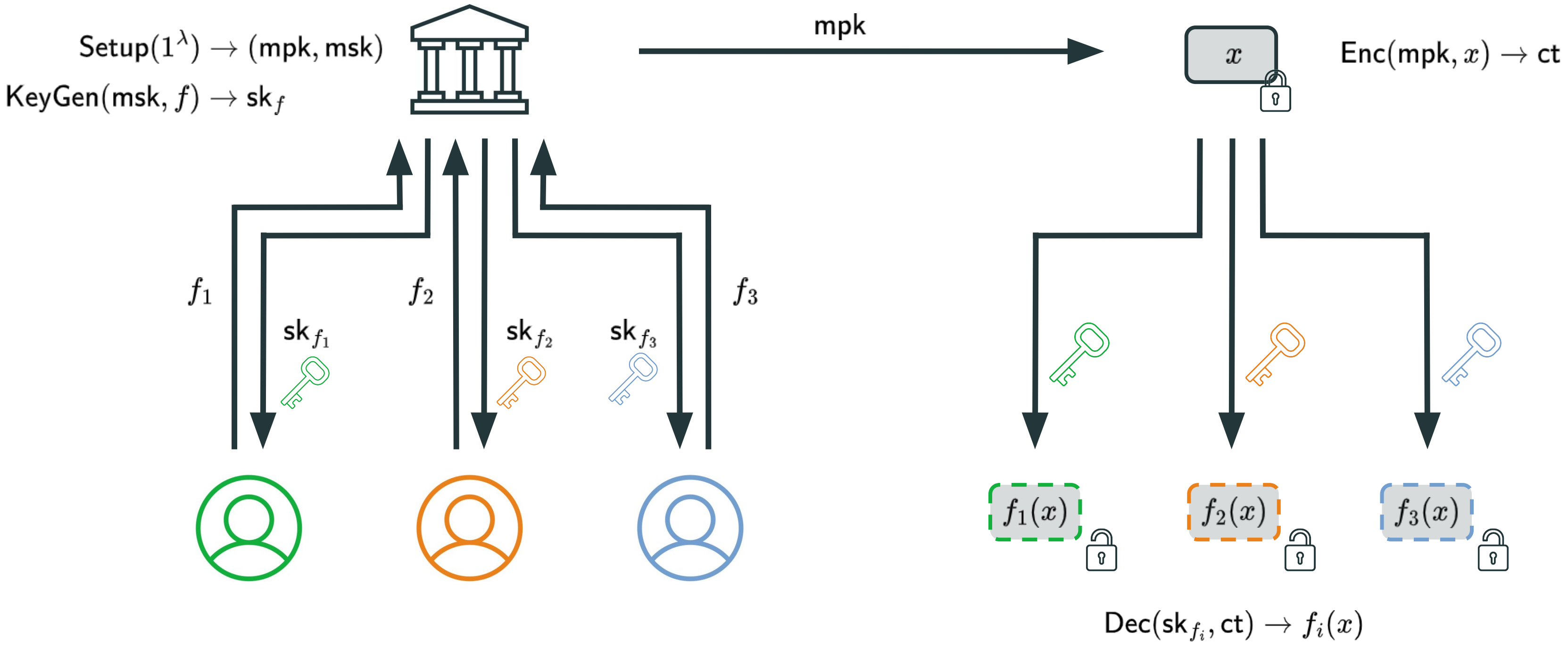


Setup$(1^\lambda) \rightarrow (\mathsf{mpk}, \mathsf{msk})$

KeyGen$(\mathsf{msk}, f) \rightarrow \mathsf{sk}_f$

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

Functional Encryption [TCC:BSW11]



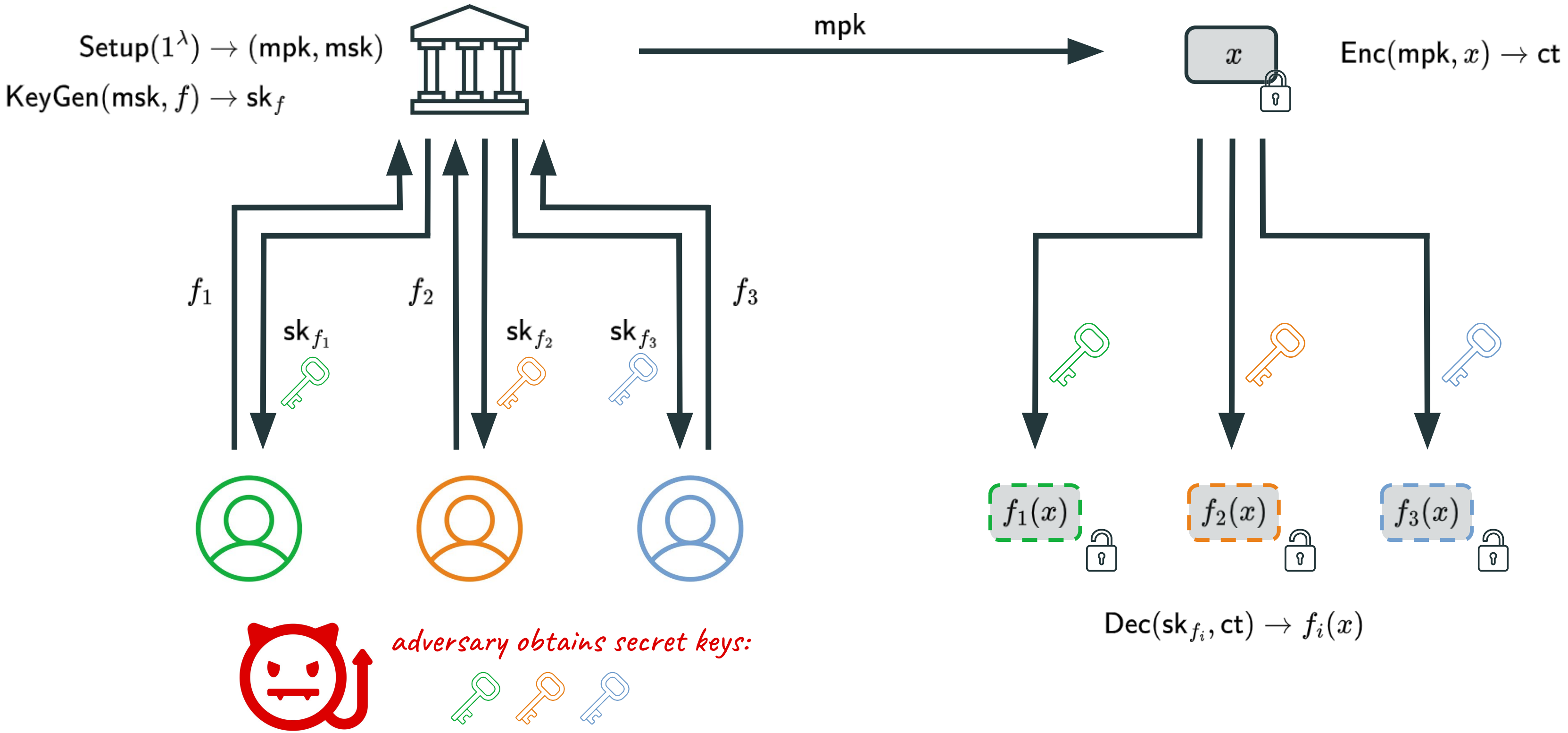
$\mathsf{Setup}(1^\lambda) \to (\mathsf{mpk}, \mathsf{msk})$

$\mathsf{KeyGen}(\mathsf{msk}, f) \to \mathsf{sk}_f$

$\mathsf{Enc}(\mathsf{mpk}, x) \to \mathsf{ct}$

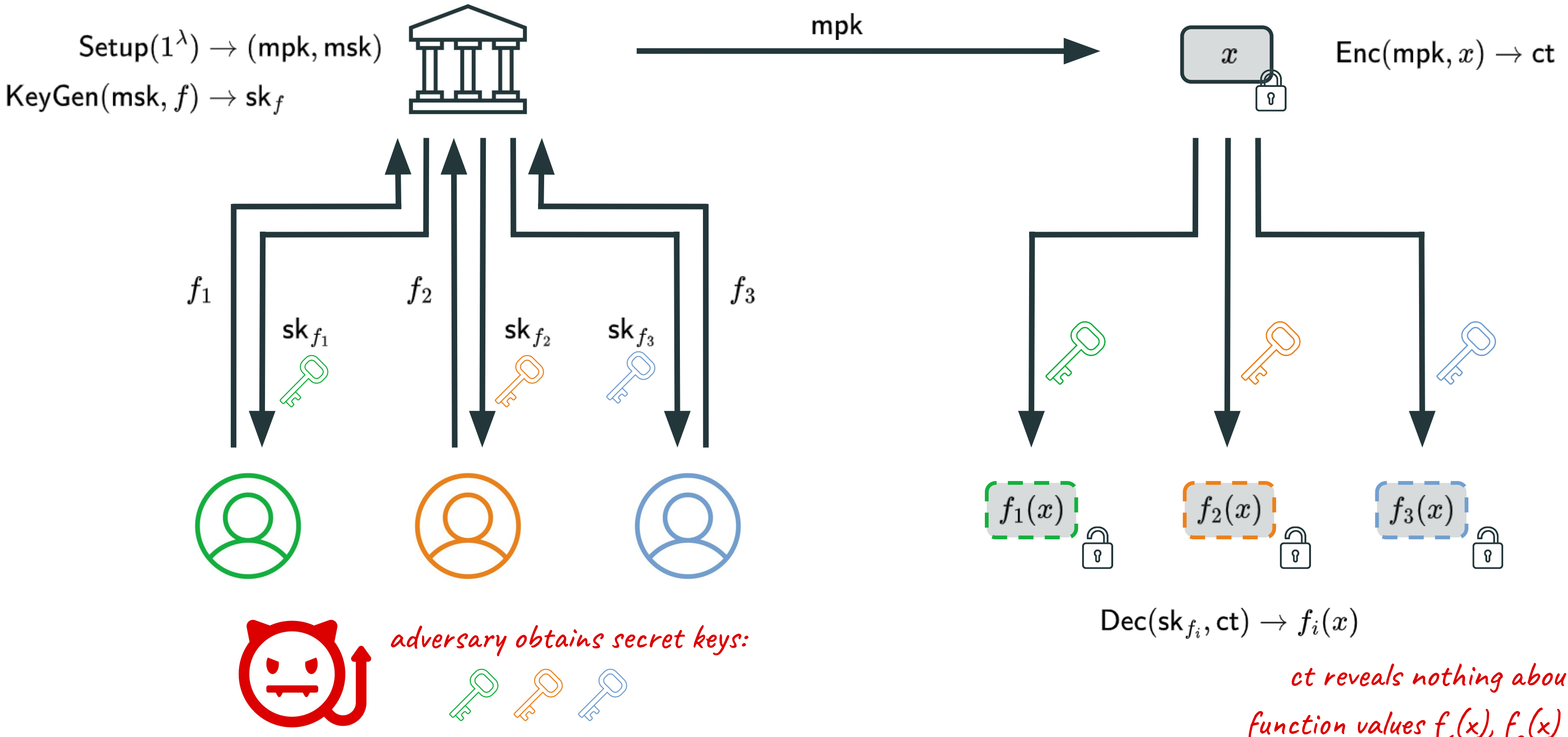
$\mathsf{Dec}(\mathsf{sk}_{f_i}, \mathsf{ct}) \to f_i(x)$

The Problem with FE Security



$\mathsf{Setup}(1^\lambda) \to (\mathsf{mpk}, \mathsf{msk})$
$\mathsf{KeyGen}(\mathsf{msk}, f) \to \mathsf{sk}_f$
mpk
x
$\mathsf{Enc}(\mathsf{mpk}, x) \to \mathsf{ct}$
f_1
sk_{f_1}
f_2
sk_{f_2}
sk_{f_3}
f_3
$f_1(x)$
$f_2(x)$
$f_3(x)$
$\mathsf{Dec}(\mathsf{sk}_{f_i}, \mathsf{ct}) \to f_i(x)$
adversary obtains secret keys:

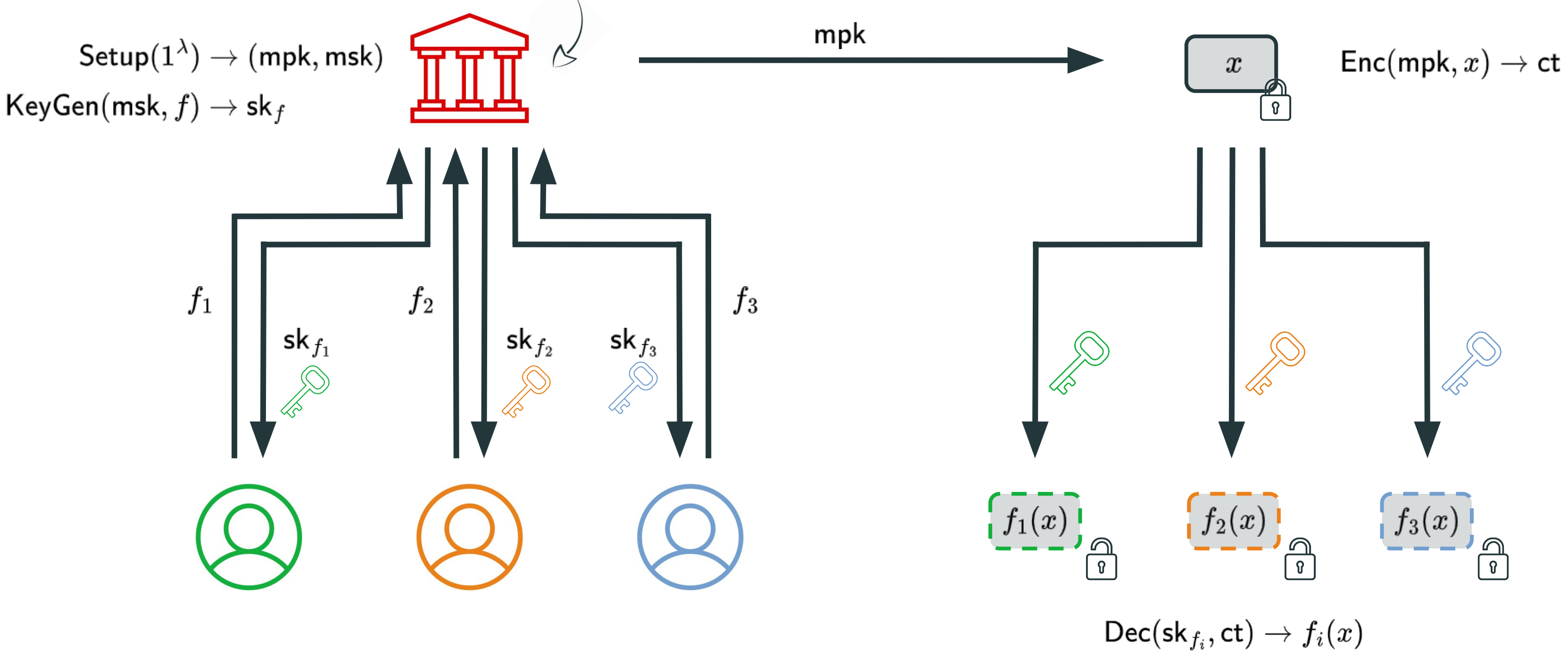
The Problem with FE Security


$\mathsf{Setup}(1^\lambda) \to (\mathsf{mpk}, \mathsf{msk})$
$\mathsf{KeyGen}(\mathsf{msk}, f) \to \mathsf{sk}_f$
mpk
x
$\mathsf{Enc}(\mathsf{mpk}, x) \to \mathsf{ct}$
f_1
sk_{f_1}
f_2
sk_{f_2}
sk_{f_3}
f_3
$f_1(x)$
$f_2(x)$
$f_3(x)$
$\mathsf{Dec}(\mathsf{sk}_{f_i}, \mathsf{ct}) \to f_i(x)$
adversary obtains secret keys:

ct reveals nothing about x except function values $f_1(x)$, $f_2(x)$ and $f_3(x)$

The Problem with FE Security

key-escrow problem: msk reveals f(x) for all f :(



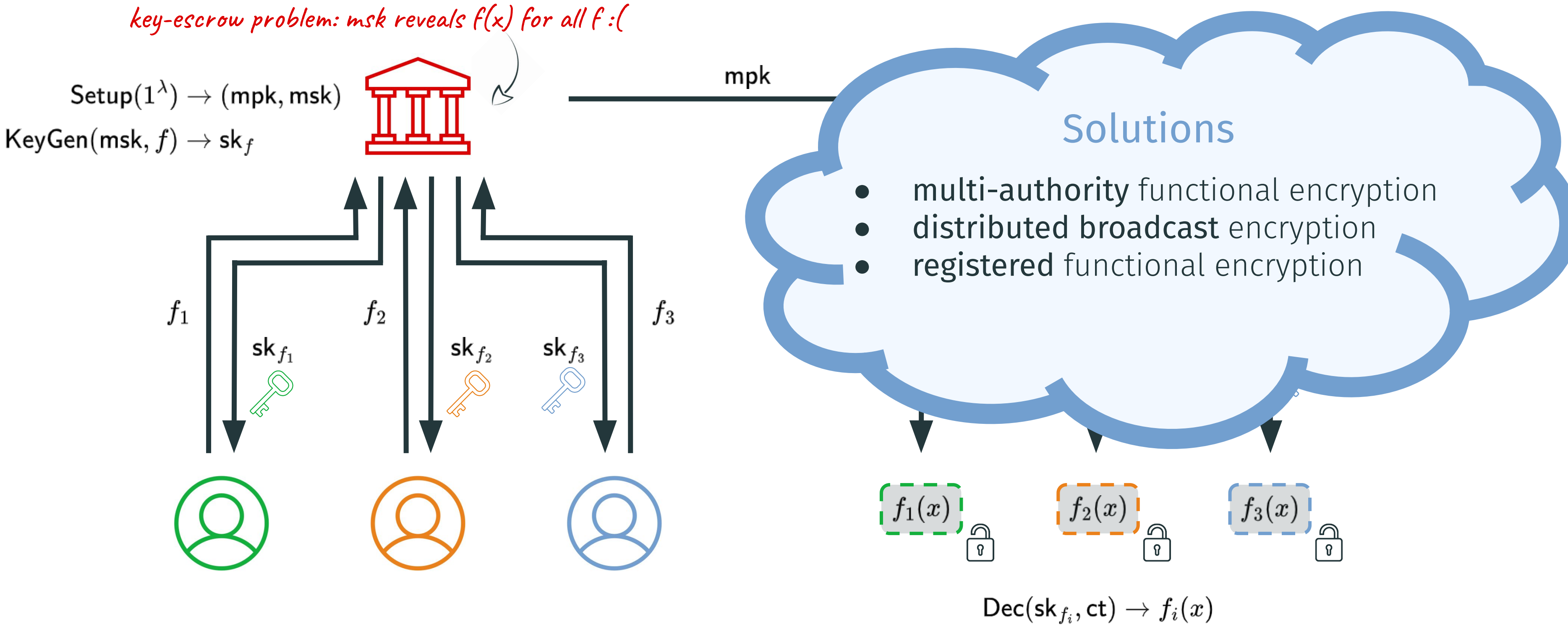
$\mathsf{Setup}(1^\lambda) \to (\mathsf{mpk}, \mathsf{msk})$

$\mathsf{KeyGen}(\mathsf{msk}, f) \to \mathsf{sk}_f$

$\mathsf{Enc}(\mathsf{mpk}, x) \to \mathsf{ct}$

$\mathsf{Dec}(\mathsf{sk}_{f_i}, \mathsf{ct}) \to f_i(x)$

The Problem with FE Security



key-escrow problem: msk reveals f(x) for all f :(

$\mathsf{Setup}(1^\lambda) \to (\mathsf{mpk}, \mathsf{msk})$

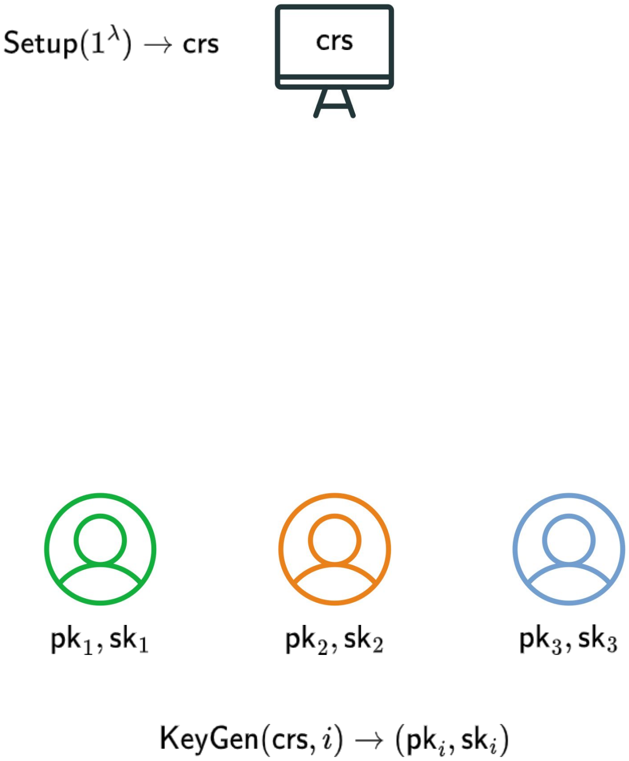
$\mathsf{KeyGen}(\mathsf{msk}, f) \to \mathsf{sk}_f$

Solutions

- **multi-authority** functional encryption
- **distributed broadcast** encryption
- **registered** functional encryption

$\mathsf{Dec}(\mathsf{sk}_{f_i}, \mathsf{ct}) \to f_i(x)$

Registered Functional Encryption* [AC:FFM+23]



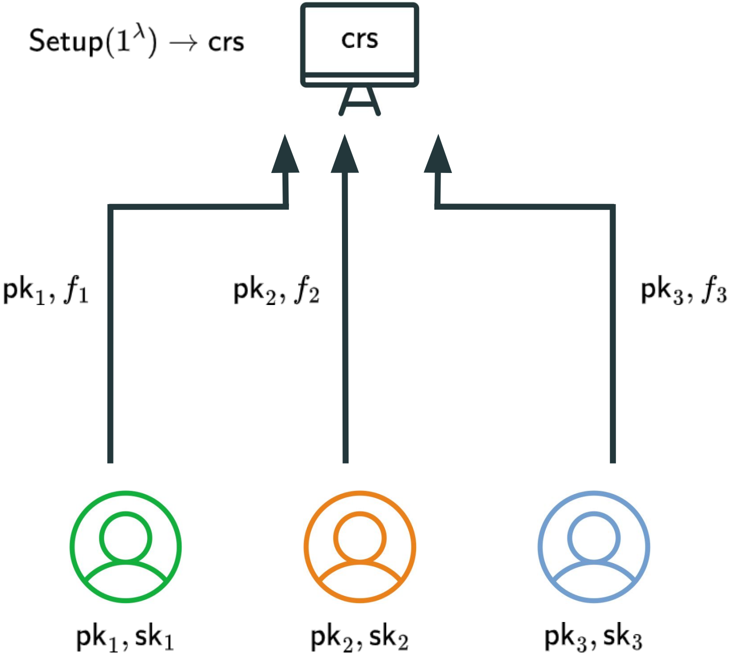
$\mathsf{Setup}(1^\lambda) \to \mathsf{crs}$

$\mathsf{pk}_1, \mathsf{sk}_1$
$\mathsf{pk}_2, \mathsf{sk}_2$
$\mathsf{pk}_3, \mathsf{sk}_3$

$\mathsf{KeyGen}(\mathsf{crs}, i) \to (\mathsf{pk}_i, \mathsf{sk}_i)$

* *not the full story, but good enough for now*

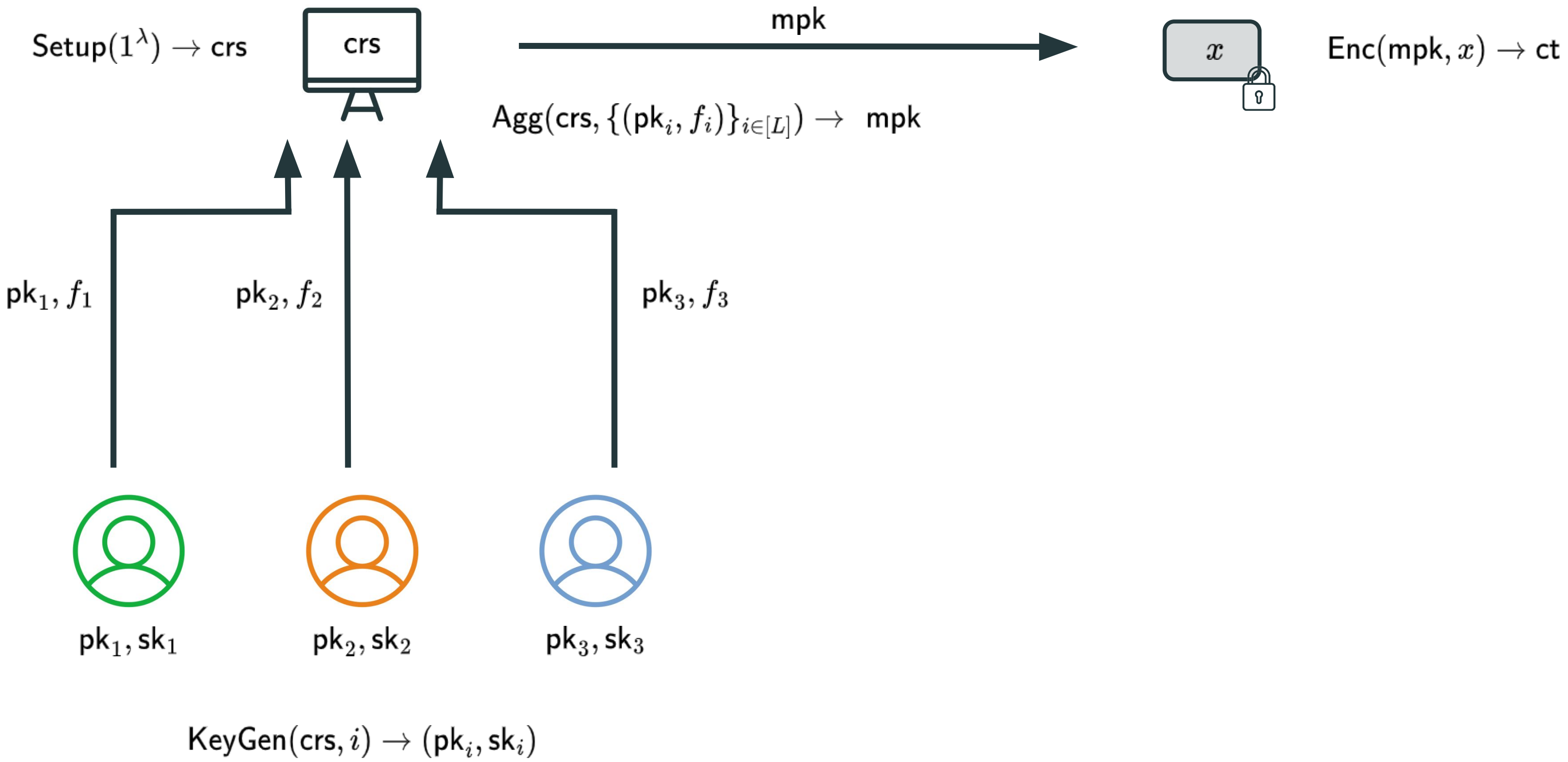
Registered Functional Encryption* [AC:FFM+23]


Setup(1^λ) → crs
crs
pk_1, f_1
pk_2, f_2
pk_3, f_3
$\mathsf{pk}_1, \mathsf{sk}_1$
$\mathsf{pk}_2, \mathsf{sk}_2$
$\mathsf{pk}_3, \mathsf{sk}_3$

$\mathsf{KeyGen}(\mathsf{crs}, i) \rightarrow (\mathsf{pk}_i, \mathsf{sk}_i)$

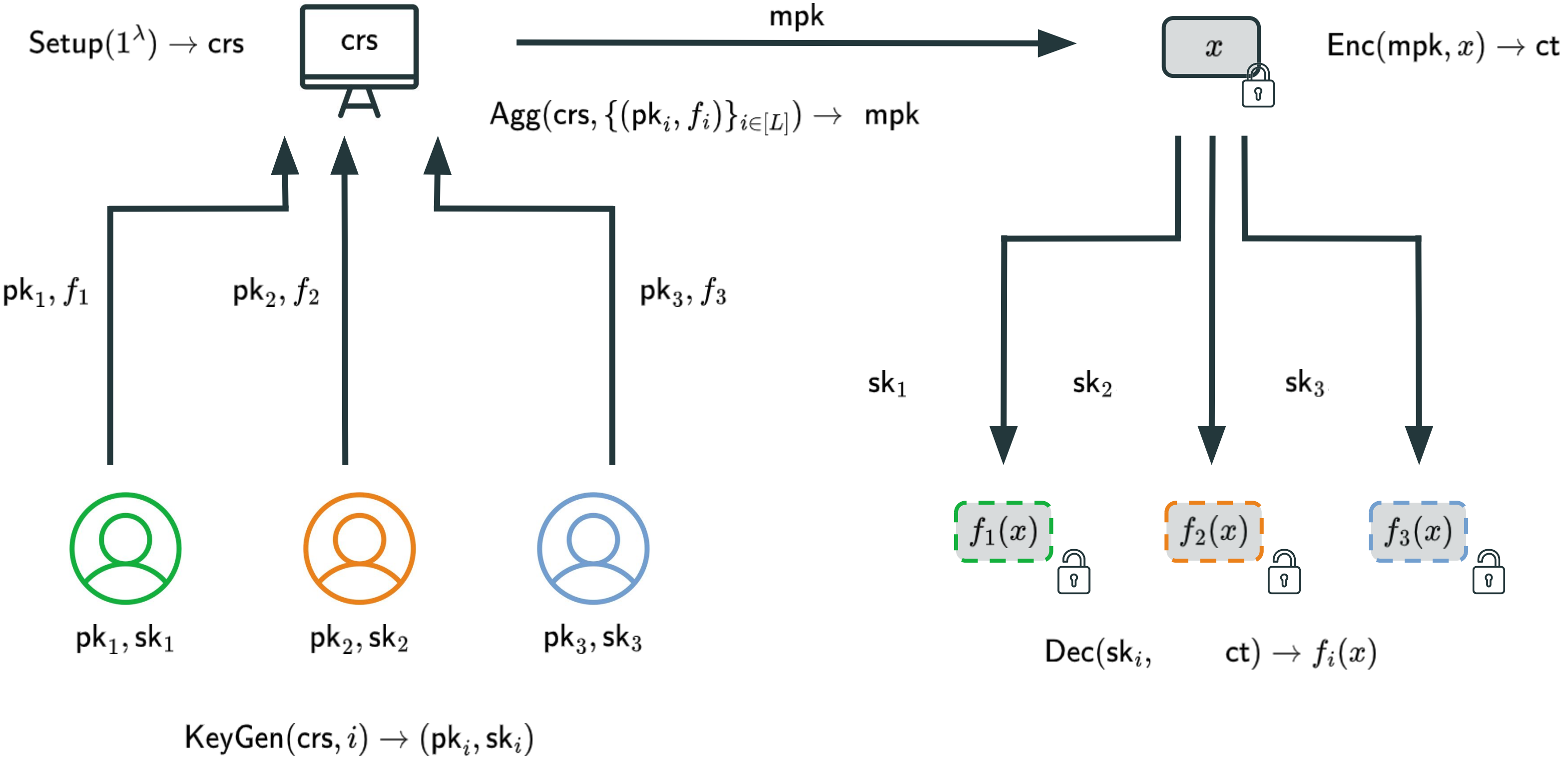
* *not the full story, but good enough for now*

Registered Functional Encryption* [AC:FFM+23]


$\mathsf{Setup}(1^\lambda) \to \mathsf{crs}$
crs
mpk
x
$\mathsf{Enc}(\mathsf{mpk}, x) \to \mathsf{ct}$
$\mathsf{Agg}(\mathsf{crs}, \{(\mathsf{pk}_i, f_i)\}_{i\in[L]}) \to \mathsf{mpk}$
pk_1, f_1
pk_2, f_2
pk_3, f_3
$\mathsf{pk}_1, \mathsf{sk}_1$
$\mathsf{pk}_2, \mathsf{sk}_2$
$\mathsf{pk}_3, \mathsf{sk}_3$
$\mathsf{KeyGen}(\mathsf{crs}, i) \to (\mathsf{pk}_i, \mathsf{sk}_i)$

* *not the full story, but good enough for now*

Registered Functional Encryption* [AC:FFM+23]



$\mathsf{Setup}(1^\lambda) \to \mathsf{crs}$

crs

mpk

$\mathsf{Enc}(\mathsf{mpk}, x) \to \mathsf{ct}$

x

$\mathsf{Agg}(\mathsf{crs}, \{(\mathsf{pk}_i, f_i)\}_{i \in [L]}) \to \mathsf{mpk}$

pk_1, f_1

pk_2, f_2

pk_3, f_3

sk_1

sk_2

sk_3

$f_1(x)$

$f_2(x)$

$f_3(x)$

$\mathsf{pk}_1, \mathsf{sk}_1$

$\mathsf{pk}_2, \mathsf{sk}_2$

$\mathsf{pk}_3, \mathsf{sk}_3$

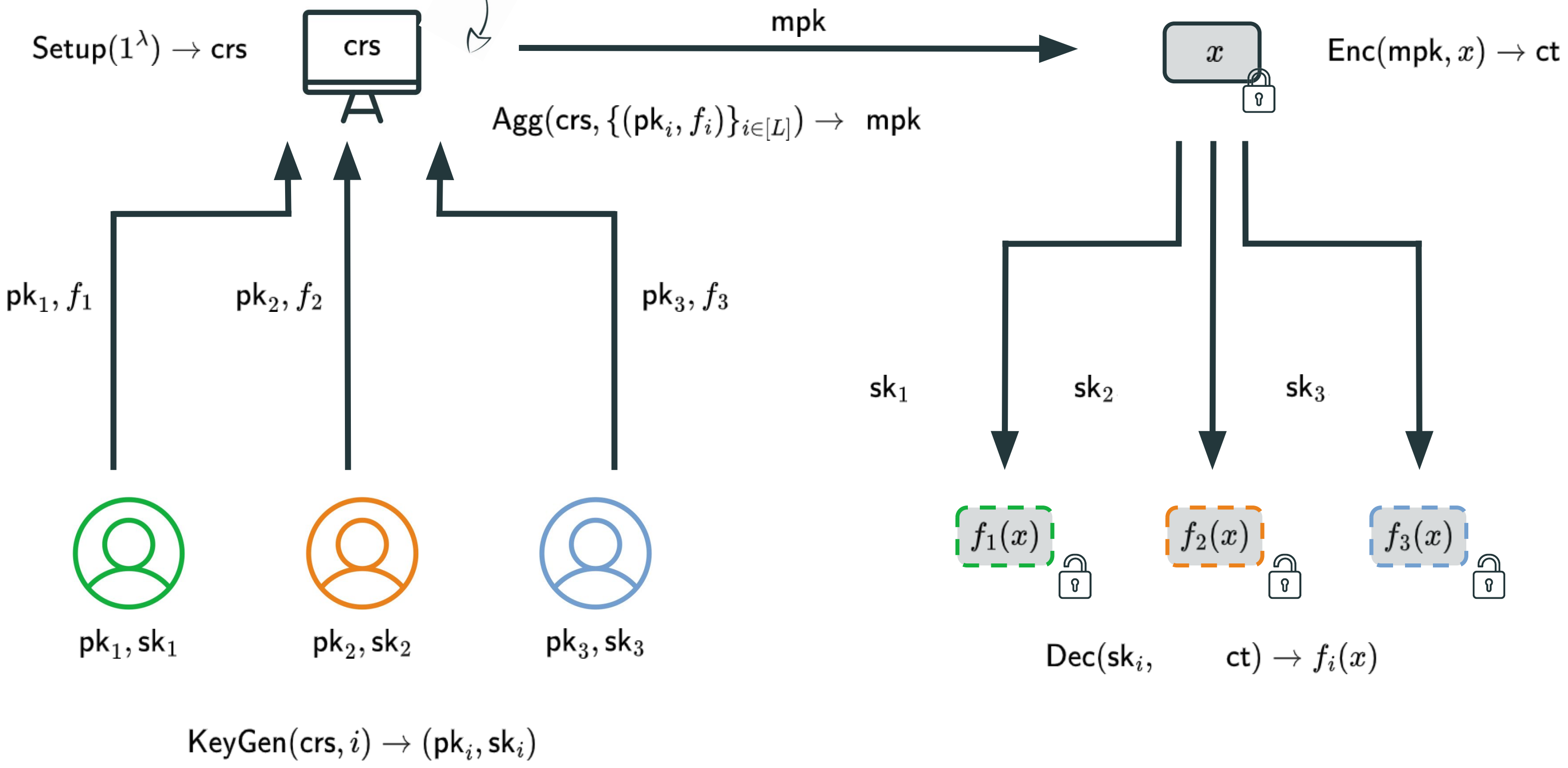
$\mathsf{Dec}(\mathsf{sk}_i, \quad \mathsf{ct}) \to f_i(x)$

$\mathsf{KeyGen}(\mathsf{crs}, i) \to (\mathsf{pk}_i, \mathsf{sk}_i)$

* *not the full story, but good enough for now*

Registered Functional Encryption* [AC:FFM+23]

key curator is deterministic & holds no secret => key-escrow problem resolved!



$\mathsf{Setup}(1^\lambda) \to \mathsf{crs}$

crs

mpk

$\mathsf{Agg}(\mathsf{crs}, \{(\mathsf{pk}_i, f_i)\}_{i\in[L]}) \to \ \mathsf{mpk}$

x

$\mathsf{Enc}(\mathsf{mpk}, x) \to \mathsf{ct}$

pk_1, f_1

pk_2, f_2

pk_3, f_3

sk_1

sk_2

sk_3

$f_1(x)$

$f_2(x)$

$f_3(x)$

$\mathsf{pk}_1, \mathsf{sk}_1$

$\mathsf{pk}_2, \mathsf{sk}_2$

$\mathsf{pk}_3, \mathsf{sk}_3$

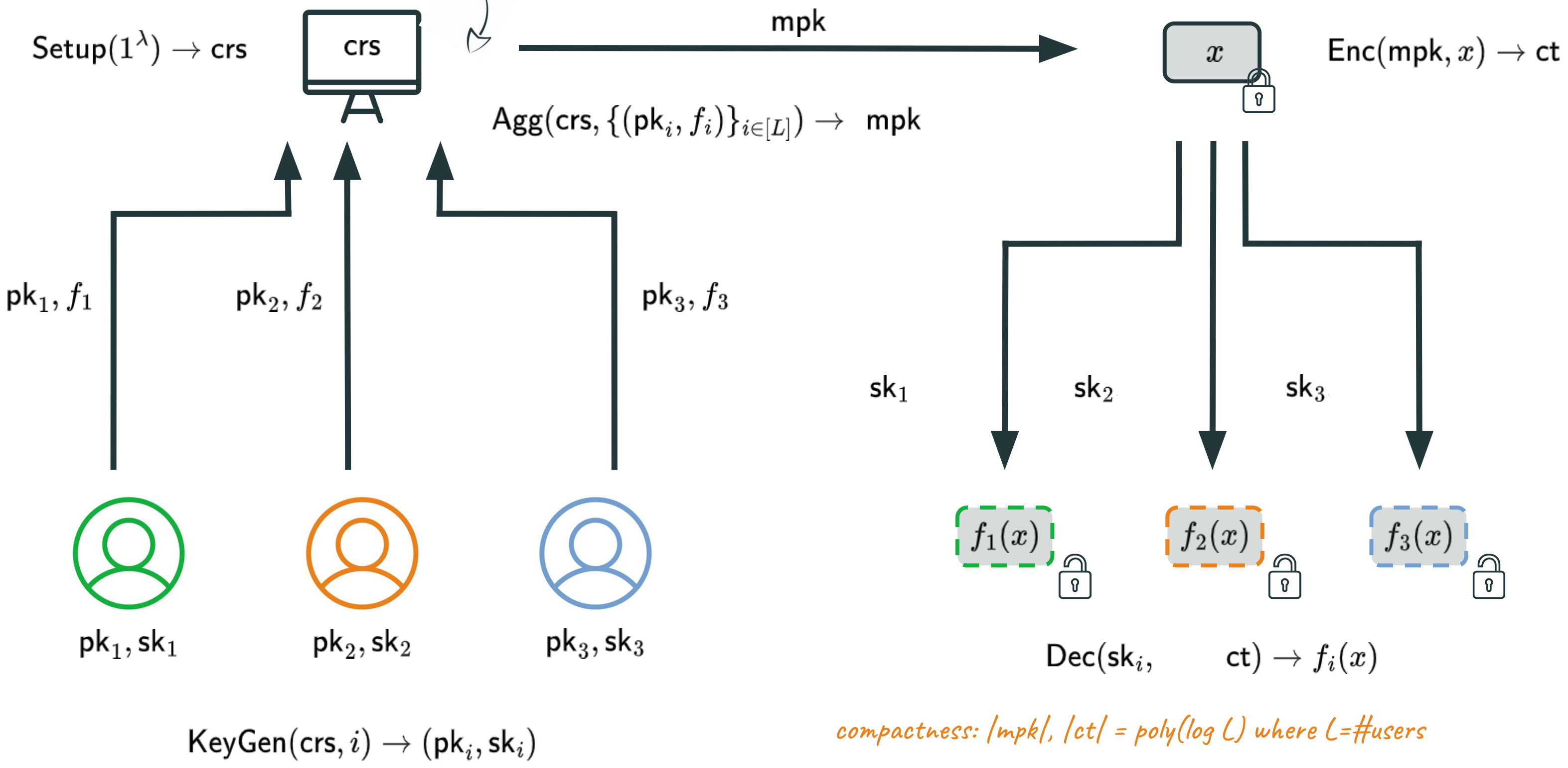
$\mathsf{Dec}(\mathsf{sk}_i, \quad \mathsf{ct}) \to f_i(x)$

$\mathsf{KeyGen}(\mathsf{crs}, i) \to (\mathsf{pk}_i, \mathsf{sk}_i)$

* *not the full story, but good enough for now*

Registered Functional Encryption* [AC:FFM+23]

key curator is deterministic & holds no secret => key-escrow problem resolved!



Setup$(1^\lambda) \rightarrow$ crs

crs

mpk

x

Enc(mpk, x) $\rightarrow$ ct

Agg(crs, $\{(\mathsf{pk}_i, f_i)\}_{i \in [L]}$) $\rightarrow$ mpk

pk_1, f_1

pk_2, f_2

pk_3, f_3

sk_1

sk_2

sk_3

$f_1(x)$

$f_2(x)$

$f_3(x)$

$\mathsf{pk}_1, \mathsf{sk}_1$

$\mathsf{pk}_2, \mathsf{sk}_2$

$\mathsf{pk}_3, \mathsf{sk}_3$

Dec(sk_i, ct) $\rightarrow f_i(x)$

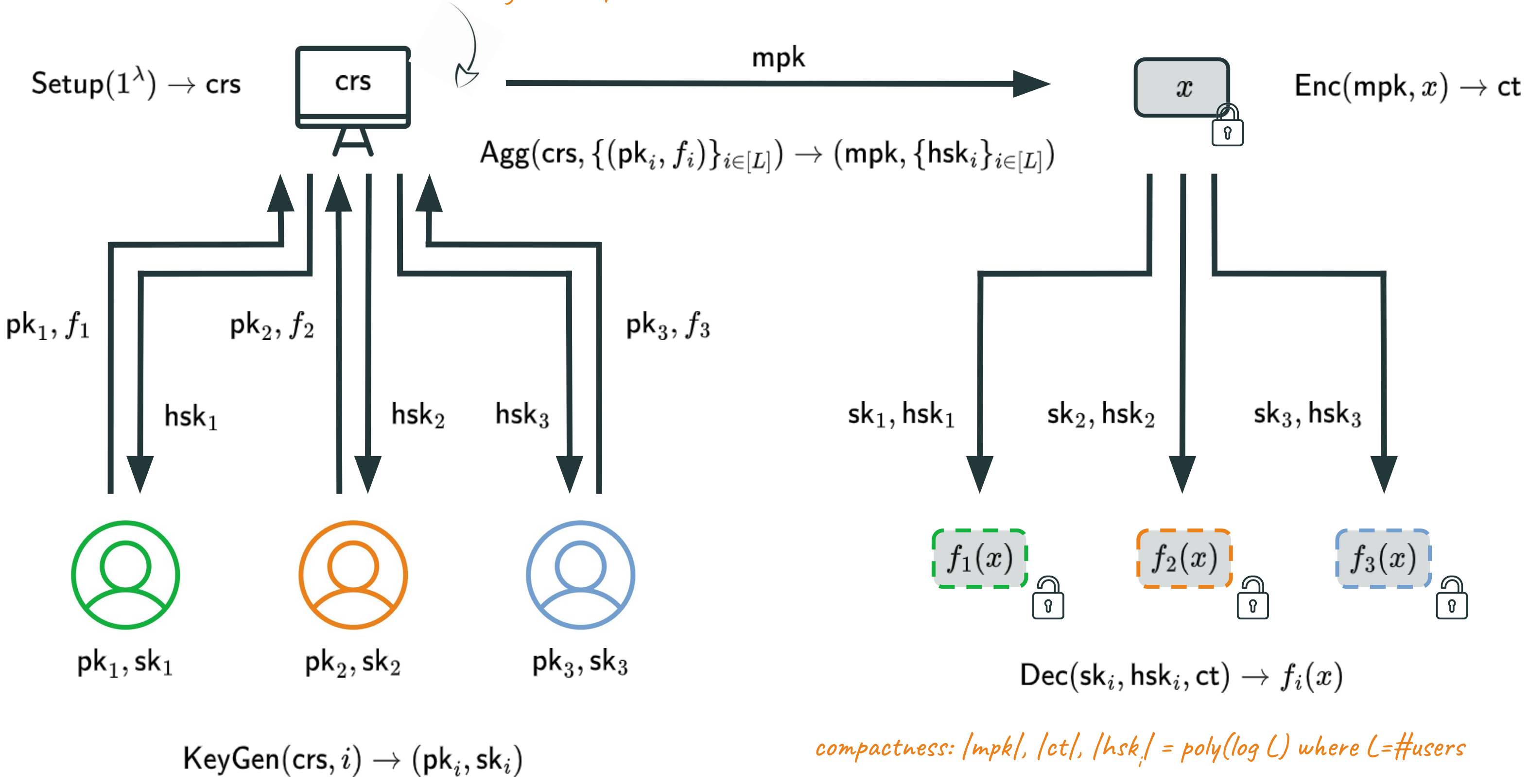
KeyGen(crs, i) $\rightarrow$ ($\mathsf{pk}_i, \mathsf{sk}_i$)

compactness: |mpk|, |ct| = poly(log L) where L=#users

** not the full story, but good enough for now*

Registered Functional Encryption* [AC:FFM+23]

key curator is deterministic & holds no secret => key-escrow problem resolved!

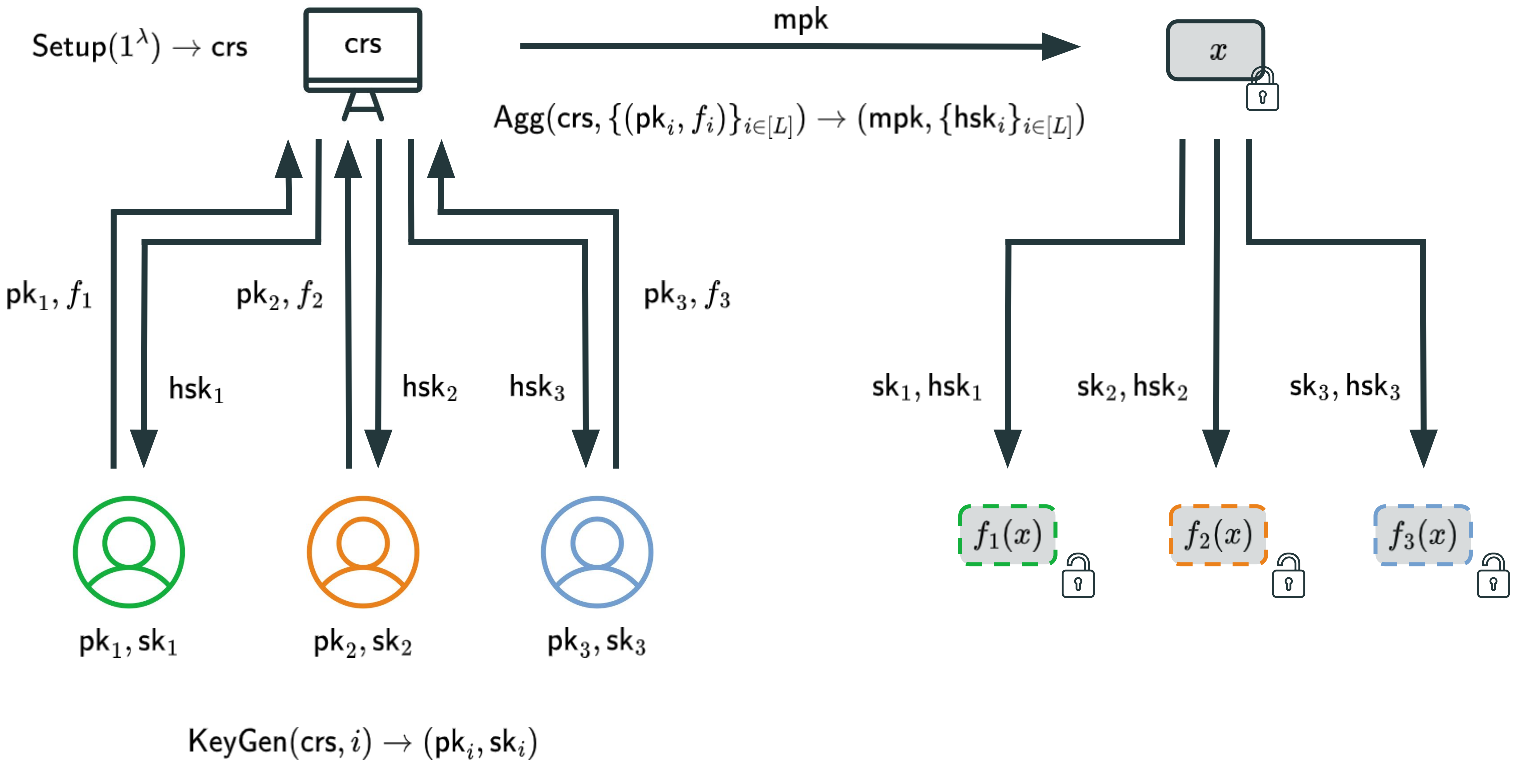


$\mathsf{Setup}(1^\lambda) \to \mathsf{crs}$
crs
mpk
x
$\mathsf{Enc}(\mathsf{mpk}, x) \to \mathsf{ct}$
$\mathsf{Agg}(\mathsf{crs}, \{(\mathsf{pk}_i, f_i)\}_{i\in[L]}) \to (\mathsf{mpk}, \{\mathsf{hsk}_i\}_{i\in[L]})$
pk_1, f_1
pk_2, f_2
pk_3, f_3
hsk_1
hsk_2
hsk_3
$\mathsf{sk}_1, \mathsf{hsk}_1$
$\mathsf{sk}_2, \mathsf{hsk}_2$
$\mathsf{sk}_3, \mathsf{hsk}_3$
$f_1(x)$
$f_2(x)$
$f_3(x)$
$\mathsf{pk}_1, \mathsf{sk}_1$
$\mathsf{pk}_2, \mathsf{sk}_2$
$\mathsf{pk}_3, \mathsf{sk}_3$
$\mathsf{Dec}(\mathsf{sk}_i, \mathsf{hsk}_i, \mathsf{ct}) \to f_i(x)$
$\mathsf{KeyGen}(\mathsf{crs}, i) \to (\mathsf{pk}_i, \mathsf{sk}_i)$

compactness: |mpk|, |ct|, |hsk$_i$| = poly(log L) where L=#users

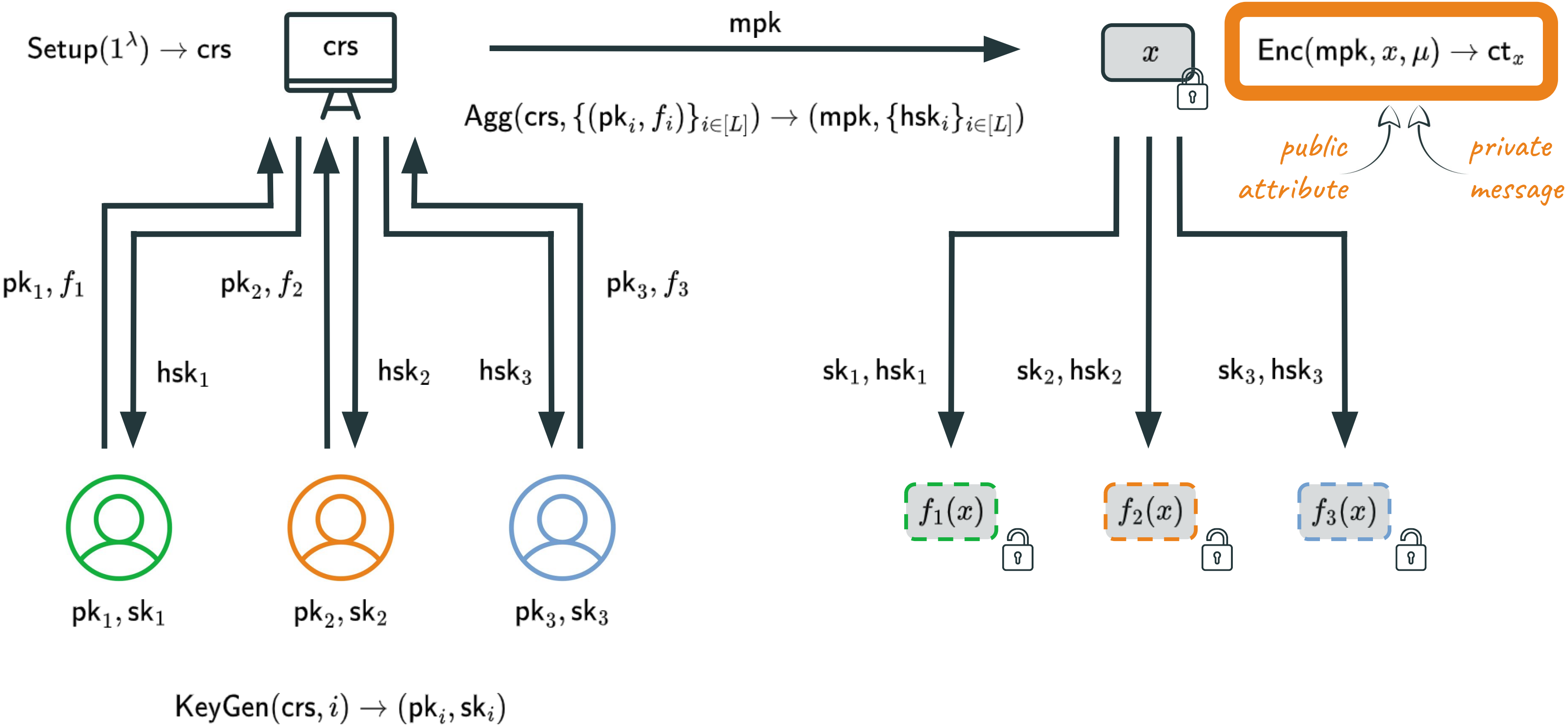
* *not the full story, but good enough for now*

Special Case: Registered Attribute-Based Encryption



$\mathsf{Setup}(1^\lambda) \to \mathsf{crs}$
crs
mpk
$\mathsf{Agg}(\mathsf{crs}, \{(\mathsf{pk}_i, f_i)\}_{i\in[L]}) \to (\mathsf{mpk}, \{\mathsf{hsk}_i\}_{i\in[L]})$
x
pk_1, f_1
pk_2, f_2
pk_3, f_3
hsk_1
hsk_2
hsk_3
$\mathsf{sk}_1, \mathsf{hsk}_1$
$\mathsf{sk}_2, \mathsf{hsk}_2$
$\mathsf{sk}_3, \mathsf{hsk}_3$
$f_1(x)$
$f_2(x)$
$f_3(x)$
$\mathsf{pk}_1, \mathsf{sk}_1$
$\mathsf{pk}_2, \mathsf{sk}_2$
$\mathsf{pk}_3, \mathsf{sk}_3$
$\mathsf{KeyGen}(\mathsf{crs}, i) \to (\mathsf{pk}_i, \mathsf{sk}_i)$

Special Case: Registered Attribute-Based Encryption



$\mathsf{Setup}(1^\lambda) \rightarrow \mathsf{crs}$

crs

mpk

$\mathsf{Agg}(\mathsf{crs}, \{(\mathsf{pk}_i, f_i)\}_{i\in[L]}) \rightarrow (\mathsf{mpk}, \{\mathsf{hsk}_i\}_{i\in[L]})$

x

$\mathsf{Enc}(\mathsf{mpk}, x, \mu) \rightarrow \mathsf{ct}_x$

public attribute

private message

pk_1, f_1

hsk_1

pk_2, f_2

hsk_2

hsk_3

pk_3, f_3

$\mathsf{sk}_1, \mathsf{hsk}_1$

$\mathsf{sk}_2, \mathsf{hsk}_2$

$\mathsf{sk}_3, \mathsf{hsk}_3$

$f_1(x)$

$f_2(x)$

$f_3(x)$

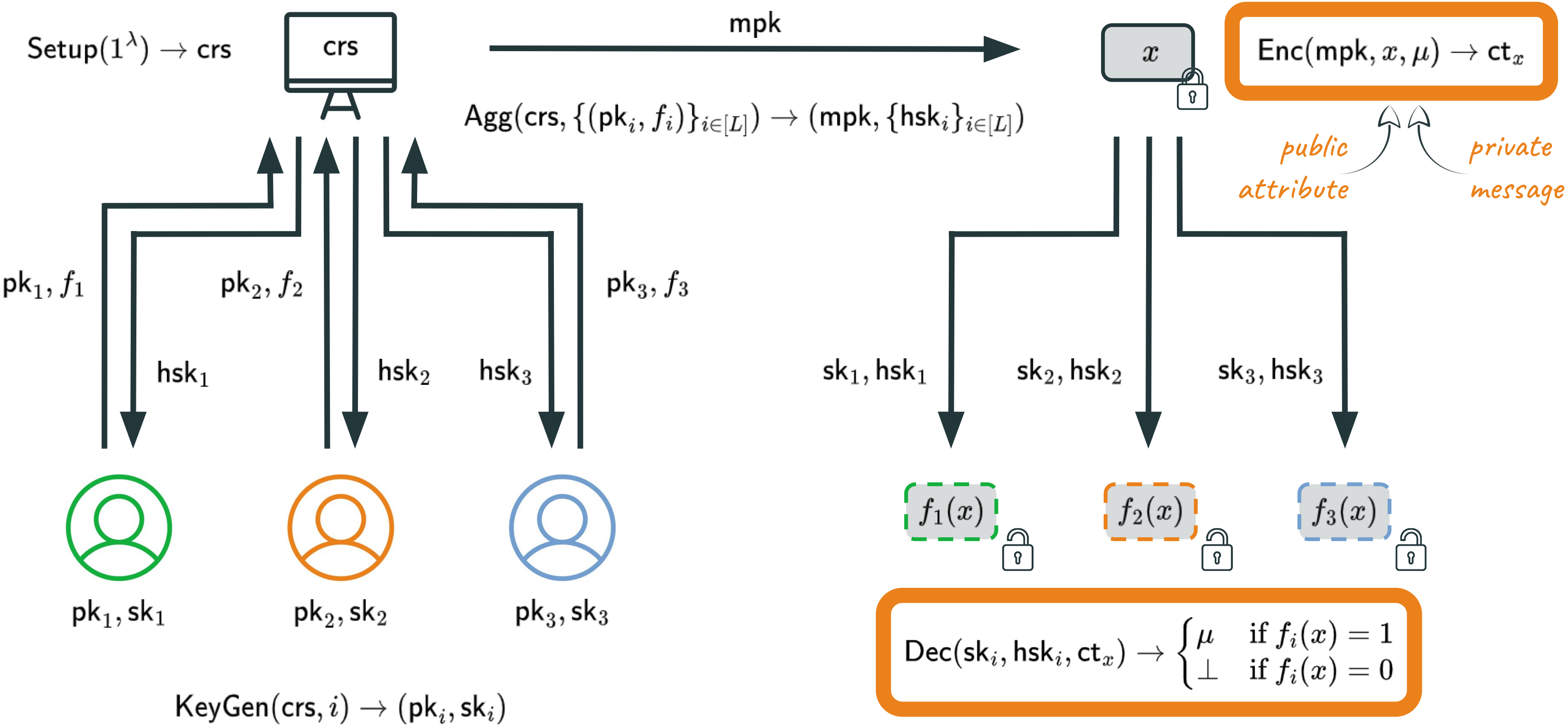
$\mathsf{pk}_1, \mathsf{sk}_1$

$\mathsf{pk}_2, \mathsf{sk}_2$

$\mathsf{pk}_3, \mathsf{sk}_3$

$\mathsf{KeyGen}(\mathsf{crs}, i) \rightarrow (\mathsf{pk}_i, \mathsf{sk}_i)$

Special Case: Registered Attribute-Based Encryption



$\mathsf{Setup}(1^\lambda) \to \mathsf{crs}$

crs

mpk

$\mathsf{Agg}(\mathsf{crs}, \{(\mathsf{pk}_i, f_i)\}_{i\in[L]}) \to (\mathsf{mpk}, \{\mathsf{hsk}_i\}_{i\in[L]})$

x

$\mathsf{Enc}(\mathsf{mpk}, x, \mu) \to \mathsf{ct}_x$

public attribute

private message

pk_1, f_1 pk_2, f_2 pk_3, f_3

hsk_1 hsk_2 hsk_3

$\mathsf{sk}_1, \mathsf{hsk}_1$ $\mathsf{sk}_2, \mathsf{hsk}_2$ $\mathsf{sk}_3, \mathsf{hsk}_3$

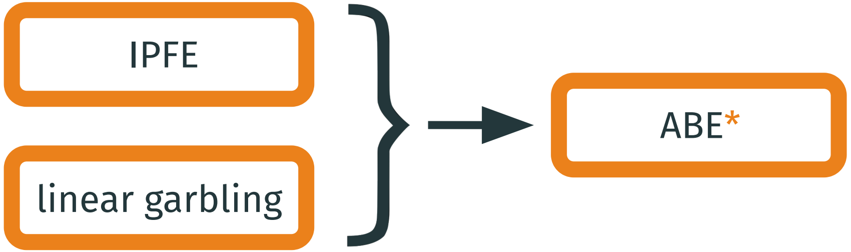
$f_1(x)$ $f_2(x)$ $f_3(x)$

$\mathsf{pk}_1, \mathsf{sk}_1$ $\mathsf{pk}_2, \mathsf{sk}_2$ $\mathsf{pk}_3, \mathsf{sk}_3$

$$\mathsf{Dec}(\mathsf{sk}_i, \mathsf{hsk}_i, \mathsf{ct}_x) \to \begin{cases} \mu & \text{if } f_i(x) = 1 \\ \perp & \text{if } f_i(x) = 0 \end{cases}$$

$\mathsf{KeyGen}(\mathsf{crs}, i) \to (\mathsf{pk}_i, \mathsf{sk}_i)$

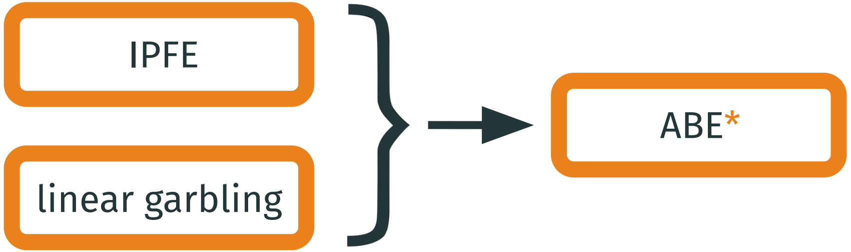
State of the Art. ABE ⇔ Registered ABE



* *natural generalization to FE*

- (Plain) ABE and FE.
  - ✔ **modular** – easy-to-verify building blocks
  - ✔ **powerful** – uniform models of computation, partially-hiding FE
  - ✔ **versatile** – flexible assumptions on different structures (pairings, lattices)

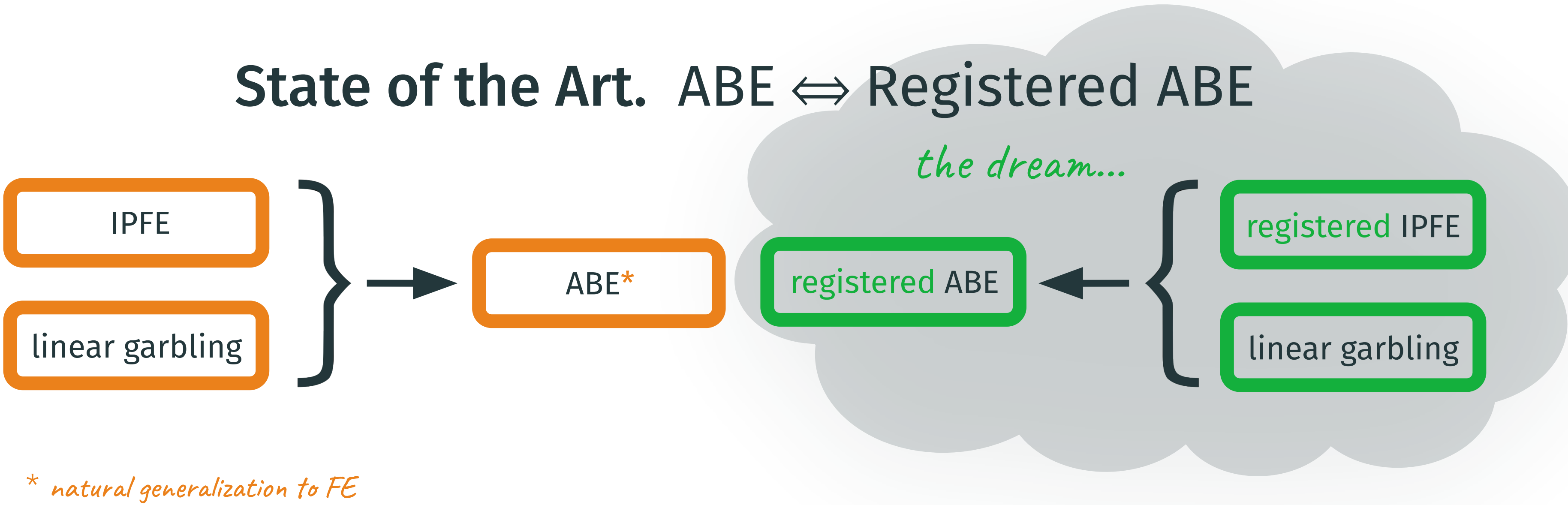
State of the Art. ABE ⟺ Registered ABE



* *natural generalization to FE*

- (Plain) ABE and FE.
  - ✓ **modular** – easy-to-verify building blocks
  - ✓ **powerful** – uniform models of computation, partially-hiding FE
  - ✓ **versatile** – flexible assumptions on different structures (pairings, lattices)
- Registered ABE and FE.
  - ✗ complex constructions, hard to verify
  - ✗ limited flexibility (few concrete functionalities and assumptions)

State of the Art. ABE ⇔ Registered ABE



* natural generalization to FE

- **(Plain) ABE and FE.**
  - ✔ **modular** – easy-to-verify building blocks
  - ✔ **powerful** – uniform models of computation, partially-hiding FE
  - ✔ **versatile** – flexible assumptions on different structures (pairings, lattices)

- **Registered ABE and FE.**
  - ✘ complex constructions, hard to verify
  - ✘ limited flexibility (few concrete functionalities and assumptions)

Linear Garbling [FOCS:AIK11, ICALP:IW14, EC:LL20] *(Generalization of LSSS)*

1. $\mathsf{Garble}(f, \sigma; \mathbf{r}) \to (L_1, \ldots, L_m)$
   - low-degree (affine) functions in *public* input $\mathbf{x}$ (“**label functions**”)
   - coefficient vectors $(\mathbf{L}_1, \ldots, \mathbf{L}_m)$ encode *secret* input σ and randomness $\mathbf{r}$

2. $\ell_1 = L_1(\mathbf{x}) = \boxed{\langle(1, \mathbf{x}), \mathbf{L}_1\rangle}, \ldots, \ell_m = L_m(\mathbf{x}) = \boxed{\langle(1, \mathbf{x}), \mathbf{L}_m\rangle}$
   - $\ell_1, \ldots, \ell_m$ (“**labels**”)

Linear Garbling [FOCS:AIK11, ICALP:IW14, EC:LL20] *(Generalization of LSSS)*

1. Garble$(f, \sigma; \mathbf{r}) \to (L_1, \ldots, L_m)$
   - low-degree (affine) functions in *public* input $\mathbf{x}$ ("**label functions**")
   - coefficient vectors $(\mathbf{L}_1, \ldots, \mathbf{L}_m)$ encode *secret* input $\boldsymbol{\sigma}$ and randomness $\mathbf{r}$

2. $\ell_1 = L_1(\mathbf{x}) = \boxed{\langle (1, \mathbf{x}), \mathbf{L}_1 \rangle}, \ldots, \ell_m = L_m(\mathbf{x}) = \boxed{\langle (1, \mathbf{x}), \mathbf{L}_m \rangle}$
   - $\ell_1, \ldots, \ell_m$ ("**labels**")

3. Eval$(f, \mathbf{x}, \ell_1, \ldots, \ell_m) \to \sigma \cdot f(\mathbf{x})$
   - high degree in $\mathbf{x}$
   - security: $\ell_1, \ldots, \ell_m$ reveal nothing about $\boldsymbol{\sigma}$ beyond $\boldsymbol{\sigma} \cdot f(\mathbf{x})$

Linear Garbling [FOCS:AIK11, ICALP:IW14, EC:LL20] *(Generalization of LSSS)*

1. Garble$(f, \sigma; \mathbf{r}) \to (L_1, \ldots, L_m)$
   - low-degree (affine) functions in *public* input $\mathbf{x}$ ("**label functions**")
   - coefficient vectors $(\mathbf{L}_1, \ldots, \mathbf{L}_m)$ encode *secret* input σ and randomness $\mathbf{r}$

Hide σ and r? ⇝ Hide these multiplications!

2. $\ell_1 = L_1(\mathbf{x}) = \boxed{\langle (1, \mathbf{x}), \mathbf{L}_1 \rangle}, \ldots, \ell_m = L_m(\mathbf{x}) = \boxed{\langle (1, \mathbf{x}), \mathbf{L}_m \rangle}$
   - $\ell_1, \ldots, \ell_m$ ("labels")

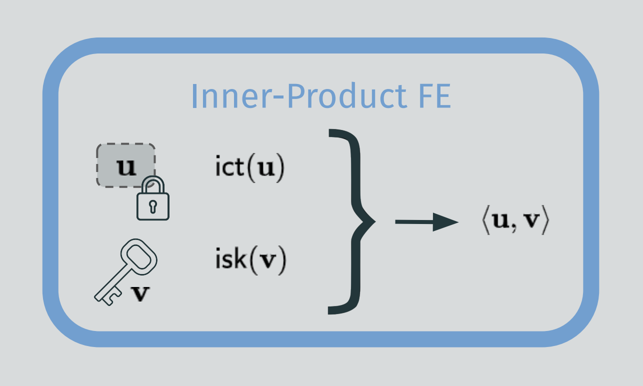
3. Eval$(f, \mathbf{x}, \ell_1, \ldots, \ell_m) \to \sigma \cdot f(\mathbf{x})$
   - high degree in $\mathbf{x}$
   - security: $\ell_1, \ldots, \ell_m$ reveal nothing about σ beyond $\sigma \cdot f(\mathbf{x})$

Linear Garbling [FOCS:AIK11, ICALP:IW14, EC:LL20] *(Generalization of LSSS)*

1. $\mathsf{Garble}(f, \sigma; \mathbf{r}) \rightarrow (L_1, \ldots, L_m)$
   - low-degree (affine) functions in *public* input $\mathbf{x}$ ("label functions")
   - coefficient vectors $(\mathbf{L}_1, \ldots, \mathbf{L}_m)$ encode *secret* input $\boldsymbol{\sigma}$ and randomness $\mathbf{r}$

Hide σ and r? ⇝ Hide these multiplications!

2. $\ell_1 = L_1(\mathbf{x}) = \boxed{\langle (1, \mathbf{x}), \mathbf{L}_1 \rangle}, \ldots, \ell_m = L_m(\mathbf{x}) = \boxed{\langle (1, \mathbf{x}), \mathbf{L}_m \rangle}$
   - $\ell_1, \ldots, \ell_m$ ("labels")



Inner-Product FE
$\mathbf{u}$
$\mathsf{ict}(\mathbf{u})$
$\mathbf{v}$
$\mathsf{isk}(\mathbf{v})$
$\langle \mathbf{u}, \mathbf{v} \rangle$

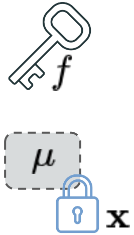
3. $\mathsf{Eval}(f, \mathbf{x}, \ell_1, \ldots, \ell_m) \rightarrow \sigma \cdot f(\mathbf{x})$
   - high degree in $\mathbf{x}$
   - security: $\ell_1, \ldots, \ell_m$ reveal nothing about $\boldsymbol{\sigma}$ beyond $\boldsymbol{\sigma} \cdot f(\mathbf{x})$

General Paradigm. ABE $\Longleftarrow$ IPFE $\circ$ Garbling

sk_f 🔑 f $\mathsf{isk}(\mathbf{L}_1), \ldots, \mathsf{isk}(\mathbf{L}_m)$ (w.r.t. σ)

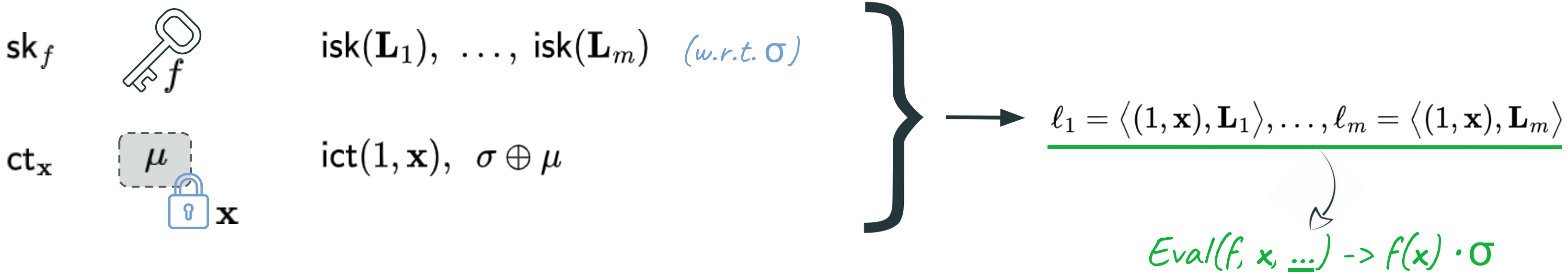
$\mathsf{ct}_{\mathbf{x}}$ $\boxed{\mu}$ 🔒 $\mathbf{x}$ $\mathsf{ict}(1, \mathbf{x}), \ \sigma \oplus \mu$

General Paradigm. ABE ⟸ IPFE ∘ Garbling



$$\left.\begin{array}{lll} \mathsf{sk}_f & & \mathsf{isk}(\mathbf{L}_1),\ \ldots,\ \mathsf{isk}(\mathbf{L}_m) \quad \textit{(w.r.t. } \sigma\textit{)} \\ \mathsf{ct}_{\mathbf{x}} & & \mathsf{ict}(1, \mathbf{x}),\ \ \sigma \oplus \mu \end{array}\right\} \longrightarrow \ell_1 = \langle (1, \mathbf{x}), \mathbf{L}_1 \rangle, \ldots, \ell_m = \langle (1, \mathbf{x}), \mathbf{L}_m \rangle$$

General Paradigm. ABE $\Leftarrow$ IPFE $\circ$ Garbling



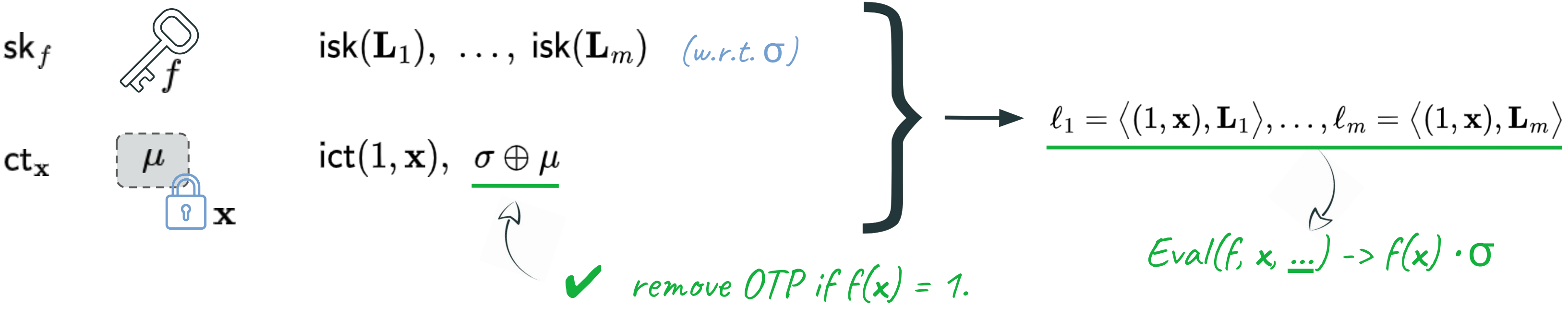
sk_f $\quad f \quad$ $\mathsf{isk}(\mathbf{L}_1), \ \ldots, \ \mathsf{isk}(\mathbf{L}_m)$ (w.r.t. σ)

$\mathsf{ct}_{\mathbf{x}}$ $\quad \mu \quad \mathbf{x}$ $\quad$ $\mathsf{ict}(1, \mathbf{x}), \ \sigma \oplus \mu$

$$\rightarrow \underline{\ell_1 = \langle (1, \mathbf{x}), \mathbf{L}_1 \rangle, \ldots, \ell_m = \langle (1, \mathbf{x}), \mathbf{L}_m \rangle}$$

Eval(f, x, $\underline{\ldots}$) -> f(x) $\cdot\, \sigma$

General Paradigm. ABE $\Longleftarrow$ IPFE $\circ$ Garbling



sk_f f $\mathsf{isk}(\mathbf{L}_1),\ \ldots,\ \mathsf{isk}(\mathbf{L}_m)$ (w.r.t. σ)

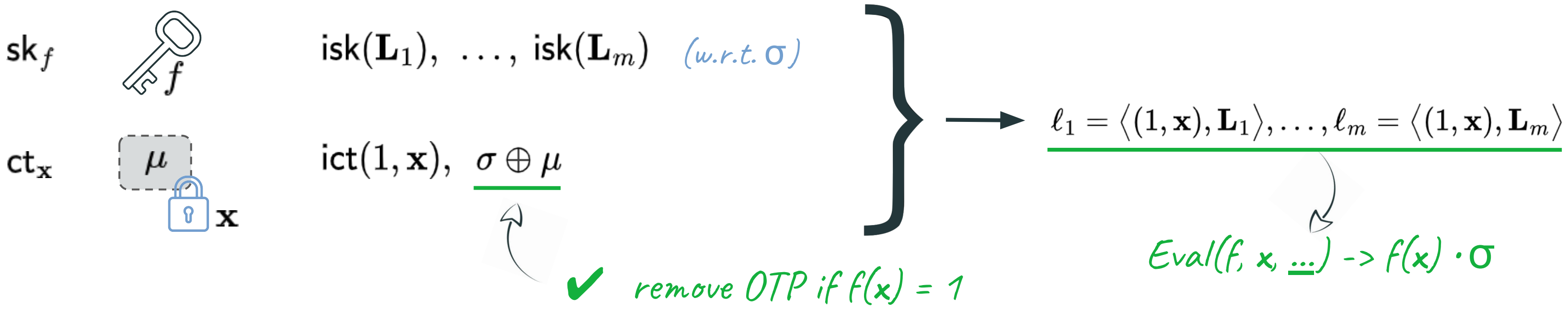
$\mathsf{ct}_{\mathbf{x}}$ μ $\mathbf{x}$ $\mathsf{ict}(1,\mathbf{x}),\ \sigma \oplus \mu$

$\ell_1 = \langle (1,\mathbf{x}), \mathbf{L}_1 \rangle, \ldots, \ell_m = \langle (1,\mathbf{x}), \mathbf{L}_m \rangle$

✔ remove OTP if f(x) = 1.

Eval(f, x, ...) -> f(x) · σ

General Paradigm. ABE $\Leftarrow$ IPFE $\circ$ Garbling



sk_f f $\mathsf{isk}(\mathbf{L}_1), \ldots, \mathsf{isk}(\mathbf{L}_m)$ (w.r.t. σ)

$\mathsf{ct}_\mathbf{x}$ μ $\mathbf{x}$ $\mathsf{ict}(1, \mathbf{x}),\ \underline{\sigma \oplus \mu}$

$\rightarrow \underline{\ell_1 = \langle (1, \mathbf{x}), \mathbf{L}_1 \rangle, \ldots, \ell_m = \langle (1, \mathbf{x}), \mathbf{L}_m \rangle}$

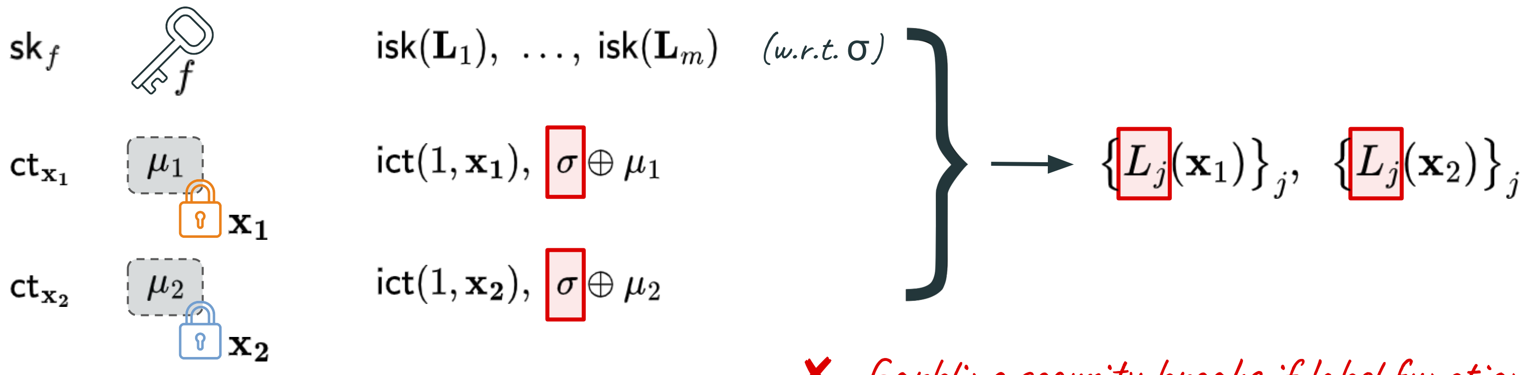
✔ remove OTP if f(x) = 1

Eval(f, x, ...) -> f(x) · σ

One-Time Security

1. IPFE ⇝ only labels revealed
2. garbling ⇝ only $\sigma \cdot f(\mathbf{x})$revealed
3. σ is OTP for μ when $f(\mathbf{x}) = 0$

General Paradigm. ABE $\Longleftarrow$ IPFE $\circ$ Garbling



sk_f f

$\mathsf{ct}_{\mathbf{x}_1}$ μ_1 $\mathbf{x}_1$

$\mathsf{ct}_{\mathbf{x}_2}$ μ_2 $\mathbf{x}_2$

$\mathsf{isk}(\mathbf{L}_1), \ \ldots, \ \mathsf{isk}(\mathbf{L}_m)$ *(w.r.t. σ)*

$\mathsf{ict}(1, \mathbf{x}_1), \ \boxed{\sigma} \oplus \mu_1$

$\mathsf{ict}(1, \mathbf{x}_2), \ \boxed{\sigma} \oplus \mu_2$

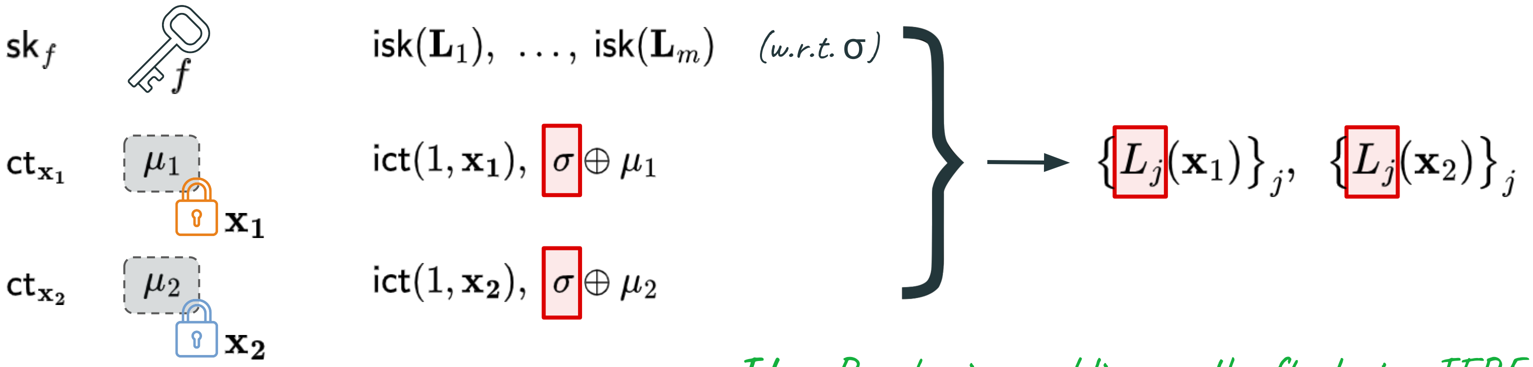
$\longrightarrow \{\boxed{L_j}(\mathbf{x}_1)\}_j, \ \{\boxed{L_j}(\mathbf{x}_2)\}_j$

✘ *Garbling security breaks if label functions are reused!*

One-Time Security

1. IPFE ⇝ only labels revealed
2. garbling ⇝ only $\sigma \cdot f(\mathbf{x})$revealed
3. σ is OTP for μ when $f(\mathbf{x}) = 0$

General Paradigm. ABE $\Longleftarrow$ IPFE $\circ$ Garbling



sk_f f

$\mathsf{ct}_{\mathbf{x}_1}$ μ_1 $\mathbf{x}_1$

$\mathsf{ct}_{\mathbf{x}_2}$ μ_2 $\mathbf{x}_2$

$\mathsf{isk}(\mathbf{L}_1),\ \ldots,\ \mathsf{isk}(\mathbf{L}_m)$ *(w.r.t. σ)*

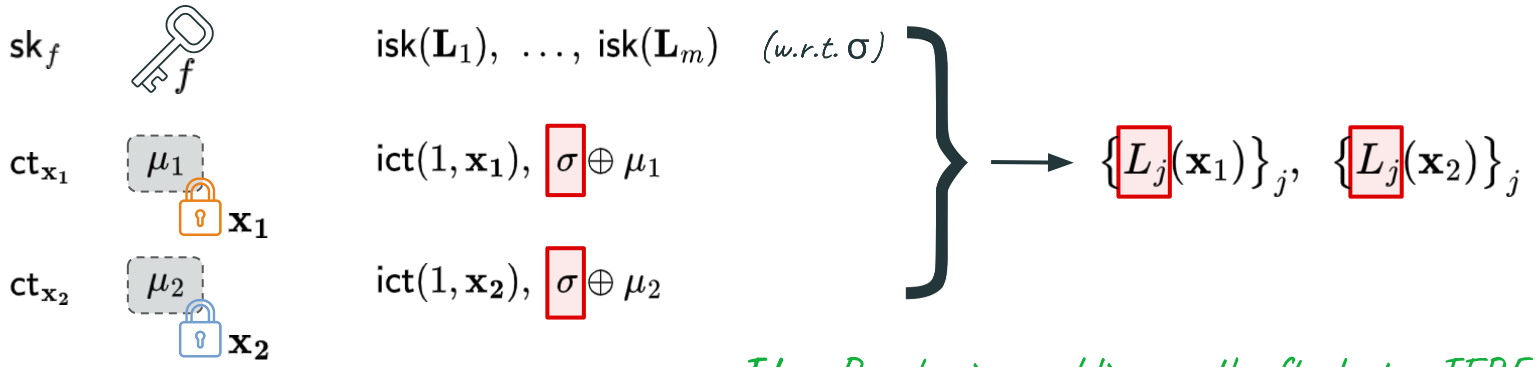
$\mathsf{ict}(1, \mathbf{x}_1),\ \boxed{\sigma} \oplus \mu_1$

$\mathsf{ict}(1, \mathbf{x}_2),\ \boxed{\sigma} \oplus \mu_2$

$$\left.\right\} \longrightarrow \{\boxed{L_j}(\mathbf{x}_1)\}_j,\ \ \{\boxed{L_j}(\mathbf{x}_2)\}_j$$

Idea. Randomize garbling on the fly during IFPE decryption.

General Paradigm. ABE $\Leftarrow$ IPFE $\circ$ Garbling



sk_f f

$\mathsf{isk}(\mathbf{L}_1),\ \ldots,\ \mathsf{isk}(\mathbf{L}_m)$ (w.r.t. σ)

$\mathsf{ct}_{\mathbf{x}_1}$ μ_1 $\mathbf{x}_1$

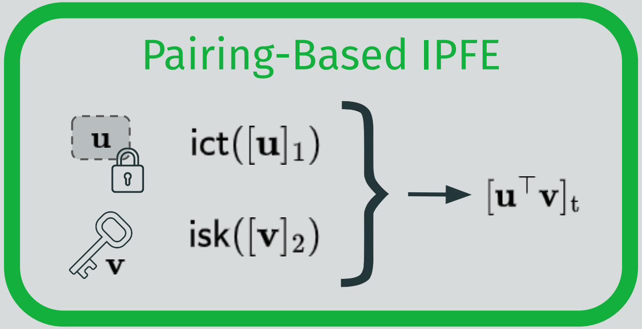
$\mathsf{ict}(1, \mathbf{x}_1),\ \boxed{\sigma} \oplus \mu_1$

$\mathsf{ct}_{\mathbf{x}_2}$ μ_2 $\mathbf{x}_2$

$\mathsf{ict}(1, \mathbf{x}_2),\ \boxed{\sigma} \oplus \mu_2$

$\longrightarrow \{\boxed{L_j}(\mathbf{x}_1)\}_j,\ \{\boxed{L_j}(\mathbf{x}_2)\}_j$

Idea. Randomize garbling on the fly during IFPE decryption.



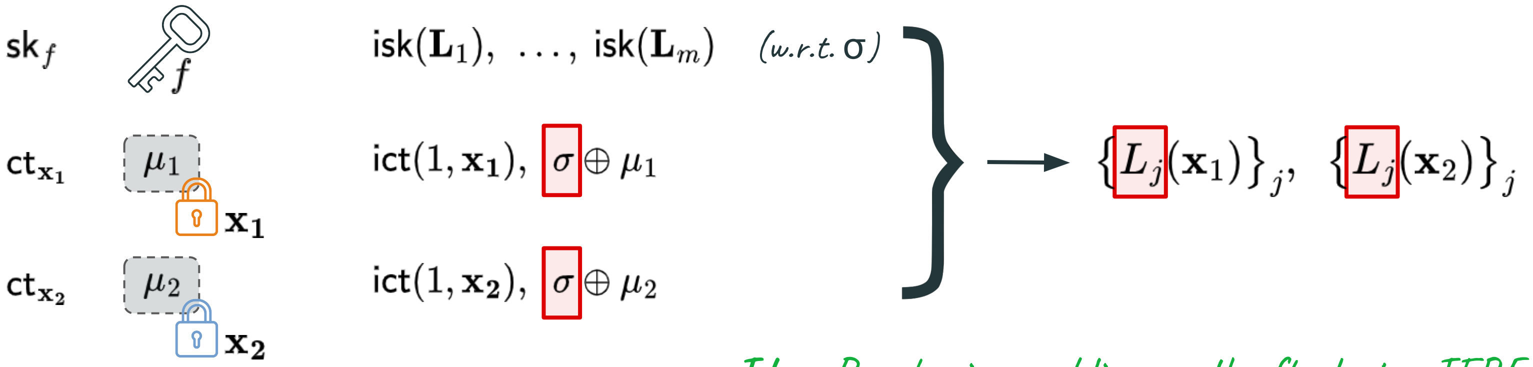
Pairing-Based IPFE

$\mathbf{u}$ $\mathsf{ict}([\mathbf{u}]_1)$

$\mathbf{v}$ $\mathsf{isk}([\mathbf{v}]_2)$

$\longrightarrow [\mathbf{u}^\top \mathbf{v}]_t$

General Paradigm. ABE $\Leftarrow$ IPFE $\circ$ Garbling



sk_f $\quad f \quad$ $\mathsf{isk}(\mathbf{L}_1), \ \ldots, \ \mathsf{isk}(\mathbf{L}_m)$ *(w.r.t. σ)*

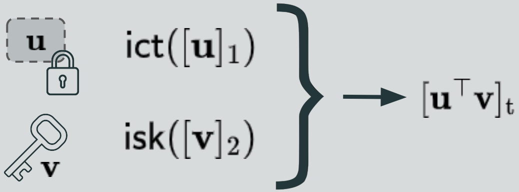
$\mathsf{ct}_{\mathbf{x}_1}$ $\quad \mu_1 \quad \mathbf{x_1} \quad$ $\mathsf{ict}(1, \mathbf{x_1}), \ \boxed{\sigma} \oplus \mu_1$

$\mathsf{ct}_{\mathbf{x}_2}$ $\quad \mu_2 \quad \mathbf{x_2} \quad$ $\mathsf{ict}(1, \mathbf{x_2}), \ \boxed{\sigma} \oplus \mu_2$

$\longrightarrow \{\boxed{L_j}(\mathbf{x}_1)\}_j, \ \{\boxed{L_j}(\mathbf{x}_2)\}_j$

Idea. Randomize garbling on the fly during IFPE decryption.

Pairing-Based IPFE



$\mathbf{u}$ $\quad \mathsf{ict}([\mathbf{u}]_1)$

$\mathbf{v}$ $\quad \mathsf{isk}([\mathbf{v}]_2)$

$\longrightarrow [\mathbf{u}^\top \mathbf{v}]_t$

Linearity Properties of Linear Garbling

- $\mathsf{Garble}(f, \sigma; \mathbf{r}) \to (L_1, \ldots, L_m)$
  linear in $(\sigma, \mathbf{r})$

General Paradigm. ABE $\Longleftarrow$ IPFE $\circ$ Garbling

sk_f f $\quad$ $\mathsf{isk}([\mathbf{L}_1]_2), \ \ldots, \ \mathsf{isk}([\mathbf{L}_m]_2), \ \mathsf{isk}'([\sigma, 1]_2)$

$\mathsf{ct}_{\mathbf{x}_1}$ μ_1 $\mathbf{x_1}$ $\quad$ $\mathsf{ict}([\boxed{s_1}(1, \mathbf{x_1})]_1)$, $\quad$ $\mathsf{ict}'([\boxed{s_1}, \mu_1]_1)$

$\mathsf{ct}_{\mathbf{x}_2}$ μ_2 $\mathbf{x_2}$ $\quad$ $\mathsf{ict}([\boxed{s_2}(1, \mathbf{x_2})]_1)$, $\quad$ $\mathsf{ict}'([\boxed{s_2}, \mu_2]_1)$

$$\longrightarrow \quad \begin{aligned} &\{[\boxed{s_1} \cdot L_j(\mathbf{x}_1)]_\mathsf{t}\}_j, \ [\boxed{s_1}\sigma + \mu_1]_\mathsf{t} \\ &\{[\boxed{s_2} \cdot L_j(\mathbf{x}_2)]_\mathsf{t}\}_j, \ [\boxed{s_2}\sigma + \mu_2]_\mathsf{t} \end{aligned}$$

Idea. Randomize garbling on the fly during IFPE decryption.

Pairing-Based IPFE

$\mathbf{u}$ $\quad$ $\mathsf{ict}([\mathbf{u}]_1)$

$\mathbf{v}$ $\quad$ $\mathsf{isk}([\mathbf{v}]_2)$

$\longrightarrow [\mathbf{u}^\top \mathbf{v}]_\mathsf{t}$

Linearity Properties of Linear Garbling

- $\mathsf{Garble}(f, \sigma; \mathbf{r}) \to (L_1, \ldots, L_m)$
  linear in $(\sigma, \mathbf{r})$

General Paradigm. ABE $\Longleftarrow$ IPFE $\circ$ Garbling

sk_f f $\quad \mathsf{isk}([\mathbf{L}_1]_2), \ \ldots, \ \mathsf{isk}([\mathbf{L}_m]_2), \ \mathsf{isk}'([\sigma, 1]_2)$

$\mathsf{ct}_{\mathbf{x}_1}$ μ_1 $\mathbf{x}_1$ $\quad \mathsf{ict}([\boxed{s_1}(1, \mathbf{x}_1)]_1), \quad \mathsf{ict}'([\boxed{s_1}, \mu_1]_1)$

$\mathsf{ct}_{\mathbf{x}_2}$ μ_2 $\mathbf{x}_2$ $\quad \mathsf{ict}([\boxed{s_2}(1, \mathbf{x}_2)]_1), \quad \mathsf{ict}'([\boxed{s_2}, \mu_2]_1)$

$\rightarrow$

Eval can be run in the exponent.

$$\{[\boxed{s_1} \cdot L_j(\mathbf{x}_1)]_{\mathrm{t}}\}_j, \quad [\boxed{s_1}\sigma + \mu_1]_{\mathrm{t}}$$

$$\{[\boxed{s_2} \cdot L_j(\mathbf{x}_2)]_{\mathrm{t}}\}_j, \quad [\boxed{s_2}\sigma + \mu_2]_{\mathrm{t}}$$

Idea. Randomize garbling on the fly during IFPE decryption.

Pairing-Based IPFE

$\mathbf{u}$ $\quad \mathsf{ict}([\mathbf{u}]_1)$

$\mathbf{v}$ $\quad \mathsf{isk}([\mathbf{v}]_2)$

$\rightarrow [\mathbf{u}^\top \mathbf{v}]_{\mathrm{t}}$

Linearity Properties of Linear Garbling

- $\mathsf{Garble}(f, \sigma; \mathbf{r}) \rightarrow (L_1, \ldots, L_m)$
  linear in $(\sigma, \mathbf{r})$
- $\mathsf{Eval}(f, \mathbf{x}, \ell_1, \ldots, \ell_m) \rightarrow \sigma \cdot f(\mathbf{x})$
  linear in $\ell_1, \ldots, \ell_m$

General Paradigm. $\mathsf{ABE} \Longleftarrow \mathsf{IPFE} \circ \mathsf{Garbling}$

sk_f $\quad f$ $\quad \mathsf{isk}([\mathbf{L}_1]_2),\ \ldots,\ \mathsf{isk}([\mathbf{L}_m]_2),\ \mathsf{isk}'([\sigma, 1]_2)$

$\mathsf{ct}_{\mathbf{x}_1}$ $\quad \mu_1$ $\quad \mathbf{x}_1$ $\quad \mathsf{ict}([\boxed{s_1}(1, \mathbf{x}_1)]_1),$ $\quad \mathsf{ict}'([\boxed{s_1}, \mu_1]_1)$

$\mathsf{ct}_{\mathbf{x}_2}$ $\quad \mu_2$ $\quad \mathbf{x}_2$ $\quad \mathsf{ict}([\boxed{s_2}(1, \mathbf{x}_2)]_1),$ $\quad \mathsf{ict}'([\boxed{s_2}, \mu_2]_1)$

$\longrightarrow$

Garbling looks fresh under DDH.

$$\{[\boxed{s_1} \cdot L_j(\mathbf{x}_1)]_{\mathsf{t}}\}_j,\ [\boxed{s_1}\sigma + \mu_1]_{\mathsf{t}}$$

$$\{[\boxed{s_2} \cdot L_j(\mathbf{x}_2)]_{\mathsf{t}}\}_j,\ [\boxed{s_2}\sigma + \mu_2]_{\mathsf{t}}$$

Idea. Randomize garbling on the fly during IFPE decryption.

Pairing-Based IPFE

$\mathbf{u}$ $\quad \mathsf{ict}([\mathbf{u}]_1)$

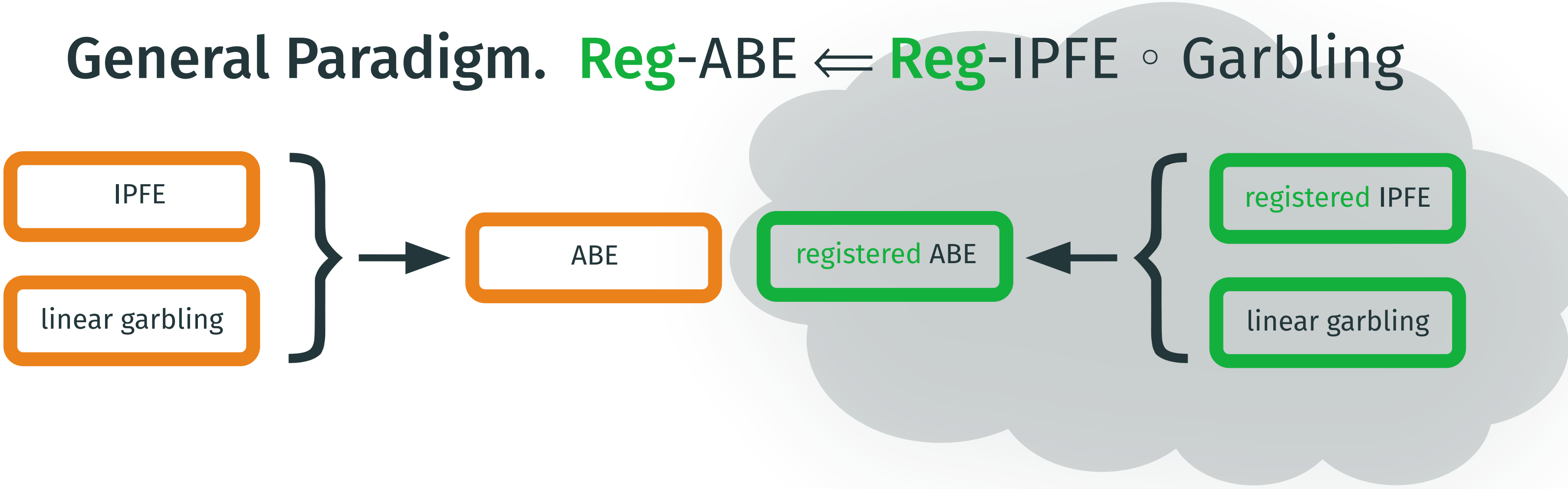
$\mathbf{v}$ $\quad \mathsf{isk}([\mathbf{v}]_2)$

$\longrightarrow [\mathbf{u}^\top \mathbf{v}]_{\mathsf{t}}$

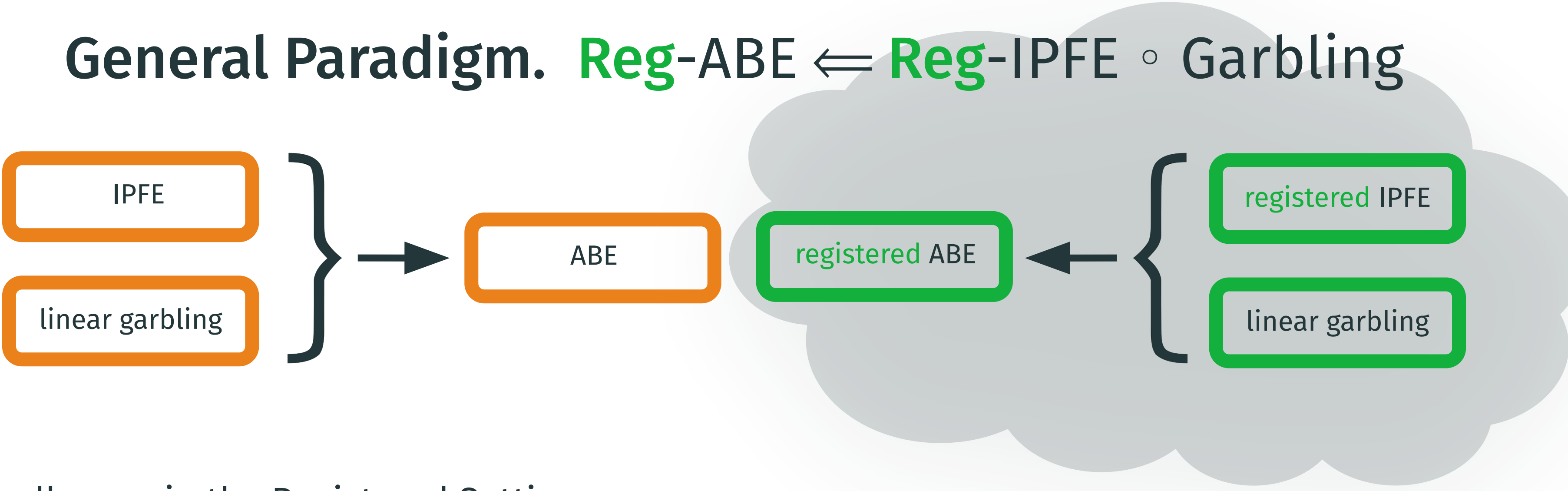
Linearity Properties of Linear Garbling

- $\mathsf{Garble}(f, \sigma; \mathbf{r}) \to (L_1, \ldots, L_m)$
  linear in $(\sigma, \mathbf{r})$
- $\mathsf{Eval}(f, \mathbf{x}, \ell_1, \ldots, \ell_m) \to \sigma \cdot f(\mathbf{x})$
  linear in $\ell_1, \ldots, \ell_m$

General Paradigm. Reg-ABE $\Longleftarrow$ Reg-IPFE $\circ$ Garbling



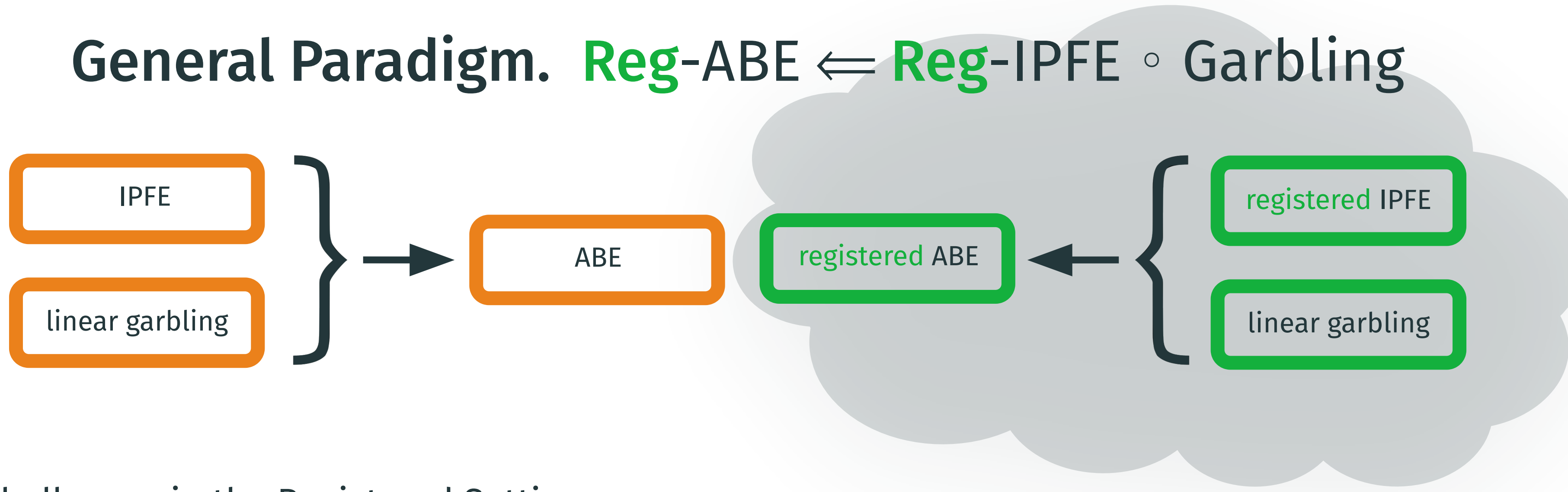
General Paradigm. Reg-ABE $\Leftarrow$ Reg-IPFE $\circ$ Garbling



Challenges in the Registered Setting.

(1) Registered IPFE supporting registration of vectors over group unknown

General Paradigm. Reg-ABE $\Leftarrow$ Reg-IPFE $\circ$ Garbling



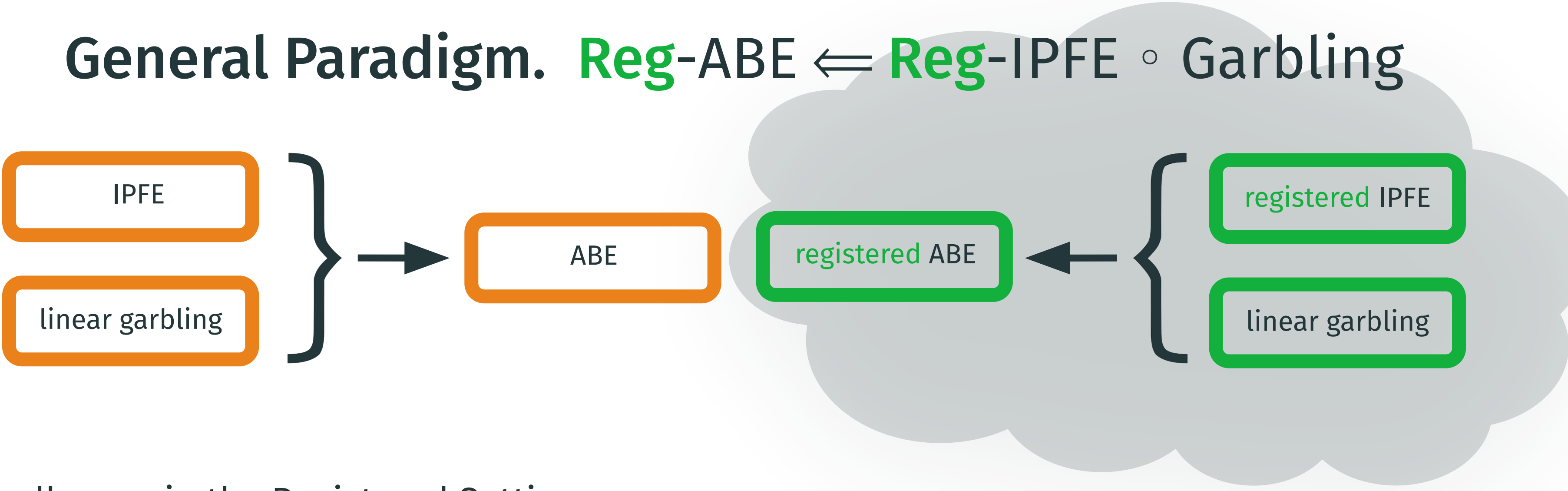
Challenges in the Registered Setting.

(1) Registered IPFE supporting registration of vectors over group unknown

(2) Sampling of user-specific randomness

- key generation performed by (potentially **malicious**) users
- aggregation is **deterministic**
- encryption time is polylogarithmic in number of users (**compactness**)

General Paradigm. Reg-ABE $\Leftarrow$ Reg-IPFE $\circ$ Garbling



Challenges in the Registered Setting.

(1) Registered IPFE supporting registration of vectors over group unknown

(2) Sampling of user-specific randomness

- key generation performed by (potentially **malicious**) users
- aggregation is **deterministic**
- encryption time is polylogarithmic in number of users (**compactness**)
- setup performed before user functions are known ⇝ decompose garbling procedure

Linearity to the Rescue

- divide garbling algorithm into two phases
  a. probabilistic offline phase: sample$(\sigma, \mathbf{r})$
  b. deterministic online phase: compute matrix $\widehat{\mathbf{L}} = \left(\widehat{\mathbf{L}}_1 \| \ldots \| \widehat{\mathbf{L}}_m\right)$ s.t. $\mathbf{L}_i = \left(\mathbf{I}_{1+|\mathbf{x}|} \otimes (\sigma, \mathbf{r})\right) \cdot \widehat{\mathbf{L}}_i$

Linearity Properties of Linear Garbling

- $\mathsf{Garble}(f, \sigma; \mathbf{r}) \to \mathbf{L} = (\mathbf{L}_1 \| \ldots \| \mathbf{L}_m)$ linear in $(\sigma, \mathbf{r})$
- $\boldsymbol{\ell} = (\ell_1, \ldots, \ell_m) = (1, \mathbf{x}) \cdot \mathbf{L}$ affine in $\mathbf{x}$

Linearity to the Rescue

- divide garbling algorithm into two phases
  a. probabilistic offline phase: sample$(\sigma, \mathbf{r})$
  b. deterministic online phase: compute matrix $\widehat{\mathbf{L}} = (\widehat{\mathbf{L}}_1 \| \dots \| \widehat{\mathbf{L}}_m)$ s.t. $\mathbf{L}_i = \big(\mathbf{I}_{1+|\mathbf{x}|} \otimes (\sigma, \mathbf{r})\big) \cdot \widehat{\mathbf{L}}_i$

$$\boldsymbol{\ell} = (1, \mathbf{x}) \cdot \mathbf{L} = (1, \mathbf{x}) \cdot \big(\mathbf{I}_{1+\mathbf{x}} \otimes (\sigma, \mathbf{r})\big) \cdot \widehat{\mathbf{L}} = \big((1, \mathbf{x}) \otimes (\sigma, \mathbf{r})\big) \cdot \widehat{\mathbf{L}}$$

Linearity Properties of Linear Garbling

- Garble$(f, \sigma; \mathbf{r}) \rightarrow \mathbf{L} = (\mathbf{L}_1 \| \dots \| \mathbf{L}_m)$ linear in $(\sigma, \mathbf{r})$
- $\boldsymbol{\ell} = (\ell_1, \dots, \ell_m) = (1, \mathbf{x}) \cdot \mathbf{L}$ affine in $\mathbf{x}$

Linearity to the Rescue

- divide garbling algorithm into two phases
  a. probabilistic offline phase: sample$(\sigma, \mathbf{r})$
  b. deterministic online phase: compute matrix $\widehat{\mathbf{L}} = (\widehat{\mathbf{L}}_1 \| \ldots \| \widehat{\mathbf{L}}_m)$ s.t. $\mathbf{L}_i = \big(\mathbf{I}_{1+|\mathbf{x}|} \otimes (\sigma, \mathbf{r})\big) \cdot \widehat{\mathbf{L}}_i$

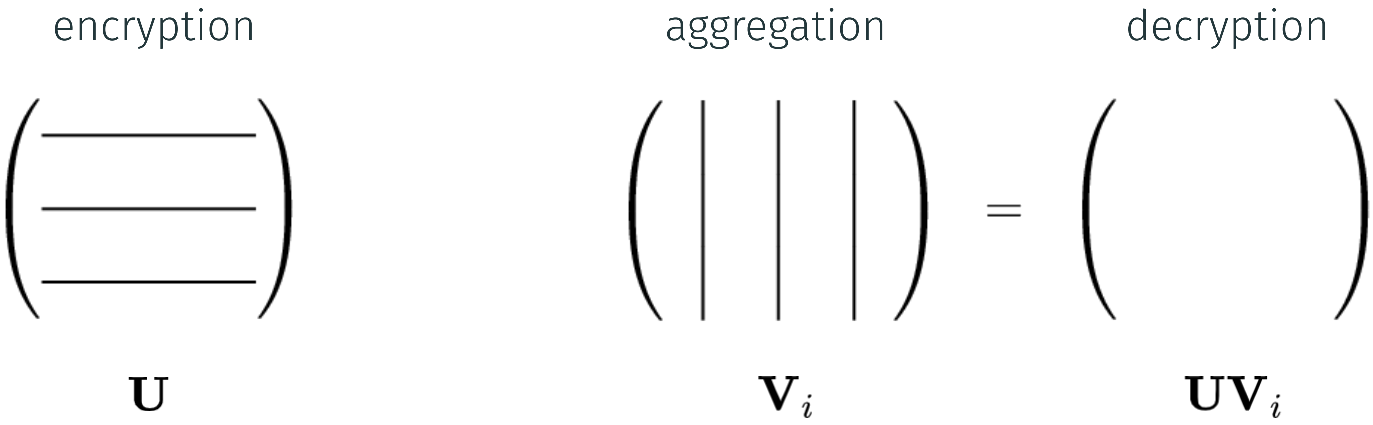
$$\boxed{\boldsymbol{\ell} = (1, \mathbf{x}) \cdot \mathbf{L} = (1, \mathbf{x}) \cdot \big(\mathbf{I}_{1+\mathbf{x}} \otimes (\sigma, \mathbf{r})\big) \cdot \widehat{\mathbf{L}} = \big((1, \mathbf{x}) \otimes (\sigma, \mathbf{r})\big) \cdot \widehat{\mathbf{L}}}$$

- run offline phase during setup, online phase during aggregation
  ⇝ we need **generalization of inner product functionality**

Linearity Properties of Linear Garbling

- Garble$(f, \sigma; \mathbf{r}) \rightarrow \mathbf{L} = (\mathbf{L}_1 \| \ldots \| \mathbf{L}_m)$ linear in $(\sigma, \mathbf{r})$
- $\boldsymbol{\ell} = (\ell_1, \ldots, \ell_m) = (1, \mathbf{x}) \cdot \mathbf{L}$ affine in $\mathbf{x}$

Reg-FE for IP (Batch Variant)



encryption

aggregation

decryption

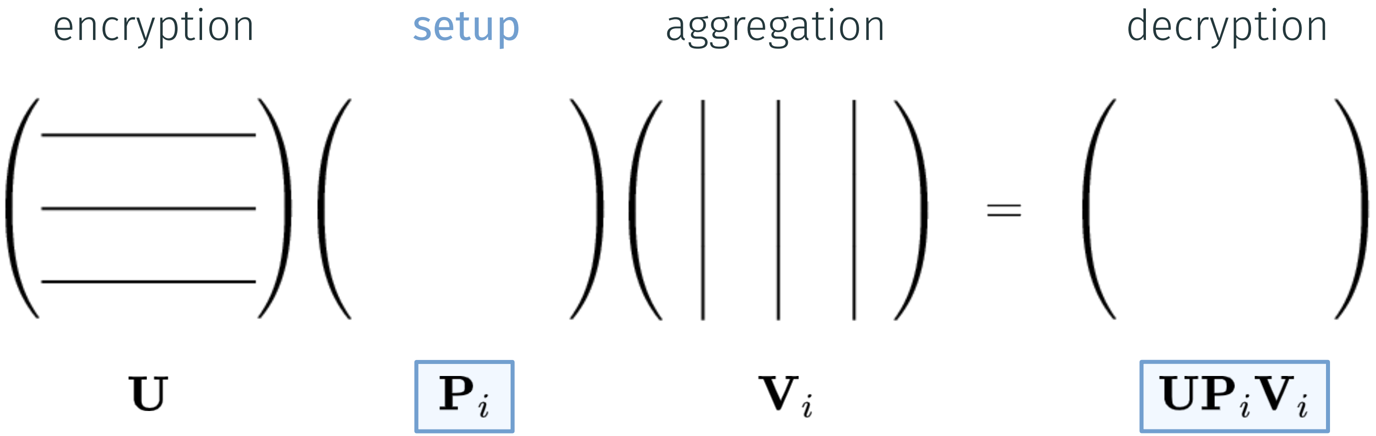
$\mathbf{U}$

$\mathbf{V}_i$

$=$

$\mathbf{U}\mathbf{V}_i$

Reg-FE for Pre-IP (Batch Variant)

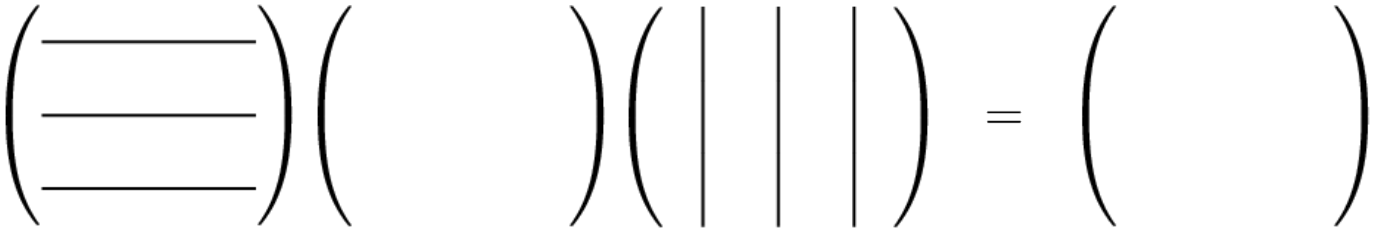


encryption setup aggregation decryption

$\mathbf{U}$ $\boxed{\mathbf{P}_i}$ $\mathbf{V}_i$ $=$ $\boxed{\mathbf{U}\mathbf{P}_i\mathbf{V}_i}$

Reg-FE for Pre-IP (Batch Variant)

encryption setup aggregation decryption



$\mathbf{U}$ $\boxed{\mathbf{P}_i}$ $\mathbf{V}_i$ $\boxed{\mathbf{U}\mathbf{P}_i\mathbf{V}_i}$

Theorem. Reg-FE for Pre-IP can be built from (bilateral) MDDH.

How to Pick the Matrices?

$$\begin{array}{ccccc} \text{encryption} & \text{setup} & \text{aggregation} & & \text{decryption} \\ \left(\boxed{\mu, s}, \boxed{((1,\mathbf{x}) \otimes s\mathbf{I}_{1+|r|})}\right) & \begin{pmatrix} \boxed{\begin{matrix}1\\ \sigma_i\end{matrix}} & \\ & \boxed{\mathbf{I}_{1+|\mathbf{x}|} \otimes (\sigma_i, \mathbf{r}_i)} \end{pmatrix} & \begin{pmatrix} 1 & \\ & \boxed{\widehat{\mathbf{L}}_i}\end{pmatrix} & = & \left(\boxed{\mu + s\sigma_i}, \boxed{((1,\mathbf{x})\otimes(s\sigma_i, s\mathbf{r}_i))\cdot \widehat{\mathbf{L}}_i}\right) \\ [\mathbf{U}]_1 & [\mathbf{P}_i]_2 & \mathbf{V}_i & & [\mathbf{U}\mathbf{P}_i\mathbf{V}_i]_t \end{array}$$

Formula for Garbling Labels. $\boxed{\boldsymbol{\ell} = (\ell_1, \ldots, \ell_m) = ((1,\mathbf{x}) \otimes (\sigma, \mathbf{r})) \cdot \widehat{\mathbf{L}}}$

How to Pick the Matrices?

$$\begin{array}{ccccc}
\text{encryption} & \text{setup} & \text{aggregation} & & \text{decryption} \\
\left(\boxed{\mu, s}, \boxed{((1,\mathbf{x}) \otimes s\mathbf{I}_{1+|r|})}\right) & \begin{pmatrix} \boxed{\begin{matrix}1\\ \sigma_i\end{matrix}} & \\ & \boxed{\mathbf{I}_{1+|\mathbf{x}|} \otimes (\sigma_i, \mathbf{r}_i)} \end{pmatrix} & \begin{pmatrix} 1 & \\ & \boxed{\widehat{\mathbf{L}}_i} \end{pmatrix} & = & \left(\boxed{\mu + s\sigma_i}, \boxed{((1,\mathbf{x}) \otimes (s\sigma_i, s\mathbf{r}_i)) \cdot \widehat{\mathbf{L}}_i}\right) \\
[\mathbf{U}]_1 & [\mathbf{P}_i]_2 & \mathbf{V}_i & & [\mathbf{U}\mathbf{P}_i\mathbf{V}_i]_t
\end{array}$$

Correctness. *RIPFE decryption yields* $\boxed{[\mu + s\sigma_i]_t}$, $\boxed{\left[((1,\mathbf{x}) \otimes (s\sigma_i, s\mathbf{r}_i)) \cdot \widehat{\mathbf{L}}_i\right]_t}$

Eval(...) -> $f_i(x) \cdot s\sigma_i$

Formula for Garbling Labels. $\boxed{\boldsymbol{\ell} = (\ell_1, \dots, \ell_m) = ((1,\mathbf{x}) \otimes (\sigma, \mathbf{r})) \cdot \widehat{\mathbf{L}}}$

How to Pick the Matrices?

$$\begin{array}{cccccc} \text{encryption} & \text{setup} & \text{aggregation} & & \text{decryption} \\ \left(\boxed{\mu, s}, \boxed{((1,\mathbf{x}) \otimes s\mathbf{I}_{1+|r|})}\right) & \begin{pmatrix} \boxed{\begin{matrix} 1 \\ \sigma_i \end{matrix}} & \\ & \boxed{\mathbf{I}_{1+|\mathbf{x}|} \otimes (\sigma_i, \mathbf{r}_i)} \end{pmatrix} & \begin{pmatrix} 1 & \\ & \boxed{\widehat{\mathbf{L}}_i} \end{pmatrix} & = & \left(\boxed{\mu + s\sigma_i}, \boxed{((1,\mathbf{x}) \otimes (s\sigma_i, s\mathbf{r}_i)) \cdot \widehat{\mathbf{L}}_i}\right) \\ [\mathbf{U}]_1 & [\mathbf{P}_i]_2 & \mathbf{V}_i & & [\mathbf{U}\mathbf{P}_i\mathbf{V}_i]_\mathsf{t} \end{array}$$

Security. *RIPFE leakage is* $\boxed{[\mu + s\overset{\$}{\sigma_i}]_\mathsf{t}}$, $\boxed{\left[((1,\mathbf{x}) \otimes (\overset{\$}{s\sigma_i}, \overset{\$}{s\mathbf{r}_i})) \cdot \widehat{\mathbf{L}}_i\right]_\mathsf{t}}$

indistinguishable from Sim(f, x, d <-$)

Formula for Garbling Labels. $\boxed{\boldsymbol{\ell} = (\ell_1, \ldots, \ell_m) = ((1,\mathbf{x}) \otimes (\sigma, \mathbf{r})) \cdot \widehat{\mathbf{L}}}$

How to Pick the Matrices?

encryption setup aggregation decryption

$$\left(\boxed{\mu, s}, \boxed{((1,\mathbf{x}) \otimes s\mathbf{I}_{1+|r|})}\right) \begin{pmatrix} \boxed{\begin{matrix}1\\ \sigma_i\end{matrix}} & \\ & \boxed{\mathbf{I}_{1+|\mathbf{x}|} \otimes (\sigma_i, \mathbf{r}_i)} \end{pmatrix} \begin{pmatrix} 1 & \\ & \boxed{\widehat{\mathbf{L}}_i} \end{pmatrix} = \left(\boxed{\mu + s\sigma_i}, \boxed{((1,\mathbf{x}) \otimes (s\sigma_i, s\mathbf{r}_i)) \cdot \widehat{\mathbf{L}}_i}\right)$$

$[\mathbf{U}]_1$ $[\mathbf{P}_i]_2$ $\mathbf{V}_i$ $[\mathbf{U}\mathbf{P}_i\mathbf{V}_i]_t$

$$\boxed{[\mu + s\overset{\$}{\sigma_i}]_t,} \quad \boxed{\left[((1,\mathbf{x}) \otimes (s\overset{\$}{\sigma_i}, s\overset{\$}{\mathbf{r}_i})) \cdot \widehat{\mathbf{L}}_i\right]_t}$$

indistinguishable from Sim(f, x, d <-$)

What about Turing machines?

Problem: shape of L and r depends on input length, runtime and space
-> study concrete garbling schemes

Formula for Garbling Labels. $\boxed{\boldsymbol{\ell} = (\ell_1, \dots, \ell_m) = ((1,\mathbf{x}) \otimes (\sigma, \mathbf{r})) \cdot \widehat{\mathbf{L}}}$

Generalization to Reg-FE

- so far, we used σ_1 = pad for fixed message μ (and σ_0 not used at all)

Linear Garbling

$\mathsf{Garble}(f, \sigma_0, \sigma_1; \mathbf{r}) \to \mathbf{L} = (\mathbf{L}_1 \| \dots \| \mathbf{L}_m)$

$\mathsf{Eval}(f, \mathbf{x}, \ell := (1, \mathbf{x}) \cdot \mathbf{L}) \to d$ s.t. $d = \sigma_1 f(\mathbf{x}) + \sigma_0$

Generalization to Reg-FE

- so far, we used σ_1 = pad for fixed message μ (and σ_0 not used at all)
- more general we can
  - encode data in σ_1

    ⇝ attribute-weighted sums functionalities [C:AGW20]
  - use σ_0 as masking term for other Reg-FE functionalities

    ⇝ attribute-based functionalities (AB-AWS, AB-QF)

Linear Garbling

$\mathsf{Garble}(f, \sigma_0, \sigma_1; \mathbf{r}) \to \mathbf{L} = (\mathbf{L}_1 \| \ldots \| \mathbf{L}_m)$

$\mathsf{Eval}(f, \mathbf{x}, \ell := (1, \mathbf{x}) \cdot \mathbf{L}) \to d$ s.t. $d = \sigma_1 f(\mathbf{x}) + \sigma_0$

Generalization to Reg-FE

- so far, we used σ_1 = pad for fixed message μ (and σ_0 not used at all)
- more general we can
  - encode data in σ_1
    ⇝ attribute-weighted sums functionalities [C:AGW20]
  - use σ_0 as masking term for other Reg-FE functionalities
    ⇝ attribute-based functionalities (AB-AWS, AB-QF)
- this yields Reg-FE instantiations for many functionalities known for pairing-based FEs (exception: *unbounded* linear and quadratic functions [EC:T23])

Linear Garbling

$\mathsf{Garble}(f, \sigma_0, \sigma_1; \mathbf{r}) \rightarrow \mathbf{L} = (\mathbf{L}_1 \| \ldots \| \mathbf{L}_m)$

$\mathsf{Eval}(f, \mathbf{x}, \ell := (1, \mathbf{x}) \cdot \mathbf{L}) \rightarrow d$ s.t. $d = \sigma_1 f(\mathbf{x}) + \sigma_0$

Existing Reg-FE beyond Predicates

Work	Function Class	Assumption	Remarks
[AC:FFM$^+$23, AC:DPY24]	general	iO, SSB hash functions	
[AC:DPY24]	AB-IP	GGM	LSSS access policies
[AC:BLM$^+$24]	IP, weak QF	q-type, GGM	
[EC:ZLZ$^+$24]	IP, QF	bilateral MDDH	
[EPRINT:PS25]	AB-AWS	bilateral MDDH	ABPs on public inputs
[this work]	**AB-AWS, AB-QF**	**bilateral MDDH**	**ABPs or logspace TMs on public inputs**

Existing Reg-FE beyond Predicates

Work	Function Class	Assumption	Remarks
[AC:FFM$^+$23, AC:DPY24]	general	iO, SSB hash functions	
[AC:DPY24]	AB-IP	GGM	LSSS access policies
[AC:BLM$^+$24]	IP, weak QF	q-type, GGM	
[EC:ZLZ$^+$24]	IP, QF	bilateral MDDH	
[EPRINT:PS25]	AB-AWS	bilateral MDDH	ABPs on public inputs
[this work]	**AB-AWS, AB-QF**	**bilateral MDDH**	**ABPs or logspace TMs on public inputs**

previously, logspace TMs unknown even for Reg-ABE

Conclusion

- adapt general paradigm for ABE and FE: plain ⇝ registered setting
- registered analogs of many pairing-based ABEs and FEs, e.g.

  Reg-ABE for ABPs [AC:ZZGQ23] and logspace TMs (↔ [EC:LL20])

  Reg-FE for attribute-based quadratic functions (↔ [TCC:W20])

  Reg-FE for (attribute-based) attribute-weighted sums (↔ [AC:DP21, AC:DPT22, C:ATY23])

Conclusion

- adapt general paradigm for ABE and FE: plain ⇝ registered setting
- registered analogs of many pairing-based ABEs and FEs, e.g.

  Reg-ABE for ABPs [AC:ZZGQ23] and logspace TMs (↔ [EC:LL20])

  Reg-FE for attribute-based quadratic functions (↔ [TCC:W20])

  Reg-FE for (attribute-based) attribute-weighted sums (↔ [AC:DP21, AC:DPT22, C:ATY23])

ia.cr/2025/2207

Thank you! :-)