

Multi-User Computation Over Encrypted Data

Ngoc Ky Nguyen

PhD Thesis Defense

Supervisors: David Pointcheval and Duong Hieu Phan

Outline

- Introduction & Main Techniques Overview
 - Motivation and Context: Functional Encryption, Research Questions
 - Preliminaries: Definitions, Technical Tools
 - ★ Overview of new techniques
 - Contributions:
 - ★ From Thesis: 4 main projects (3 in this talk)
 - ★ Others: 2 different topics + 2 extensions
- Controlling Functional Keys
- Optimal Security Notion
- From Secret-Key to Public-Key Encryption Schemes
- Conclusion and Future Works

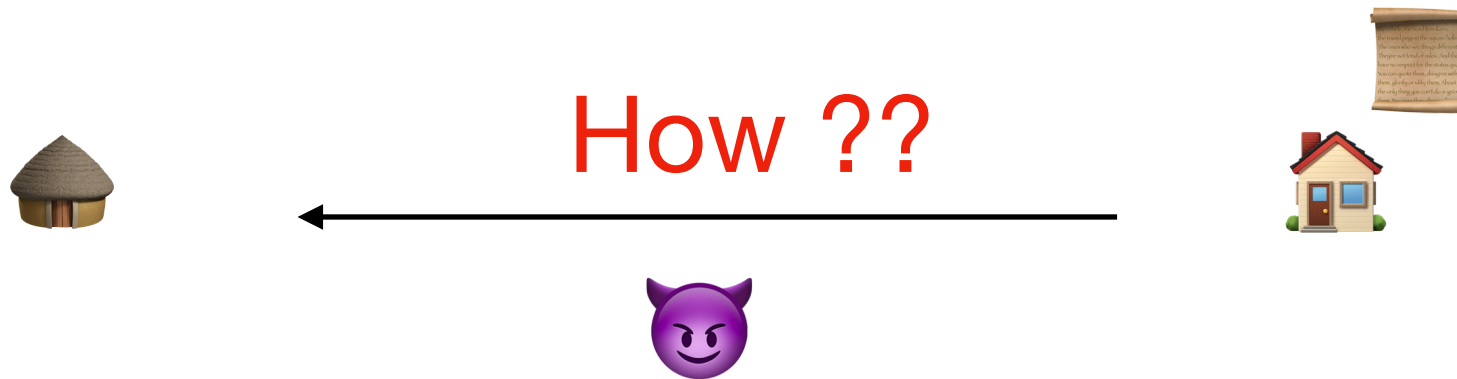
Context

Cryptography - Secure Communication



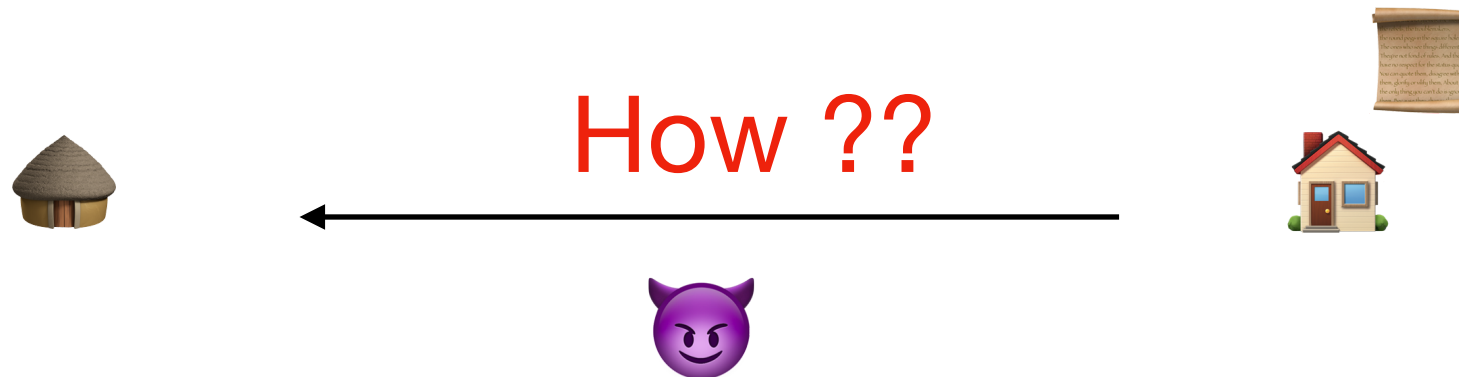
Context

Cryptography - Secure Communication



Context

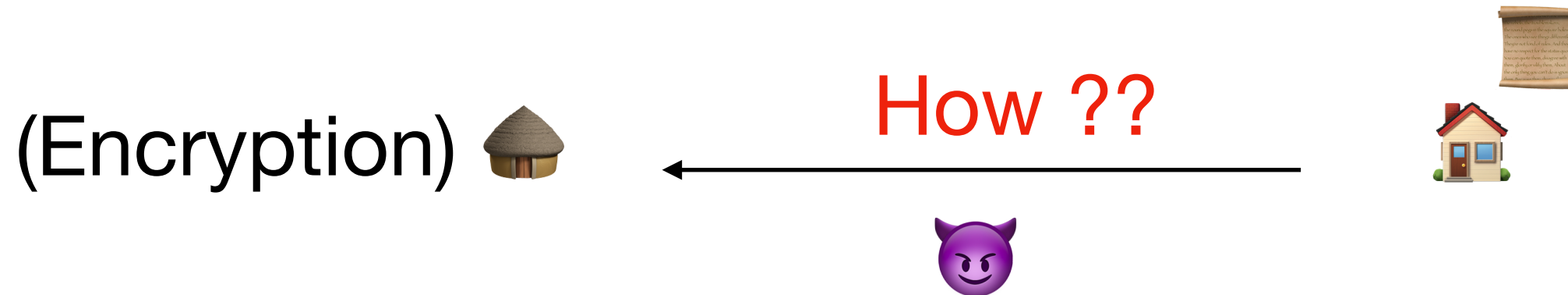
Cryptography - Secure Communication



😈 : observing the communication channel

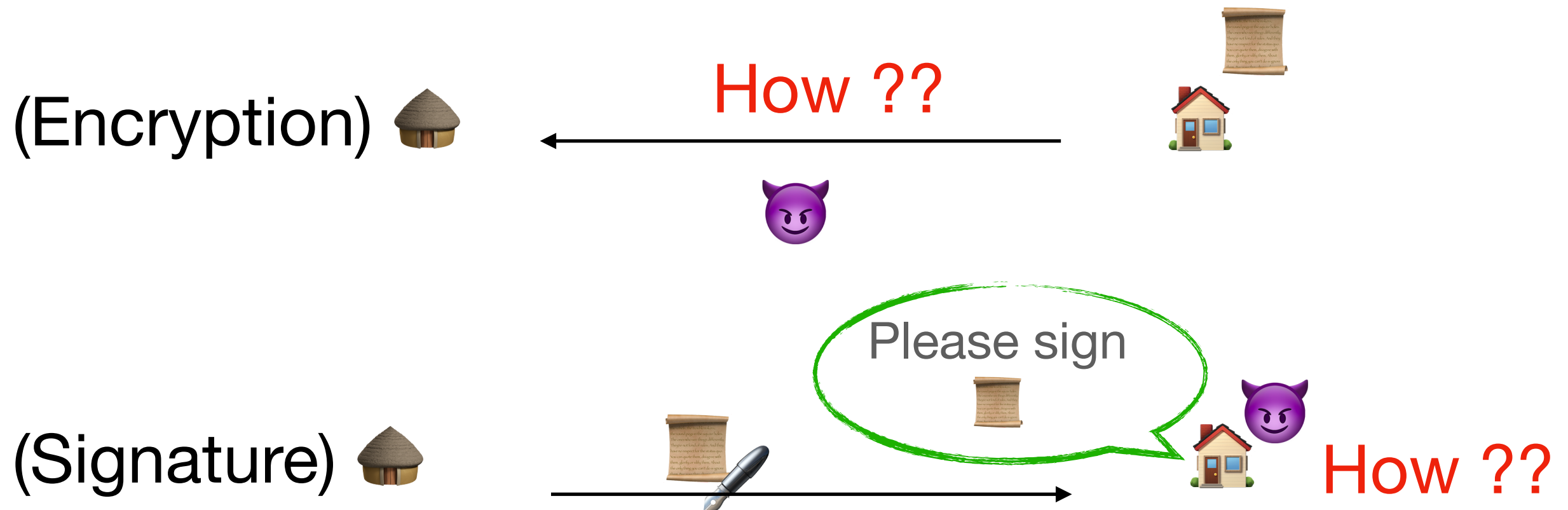
Context

Cryptography - Secure Communication, and More



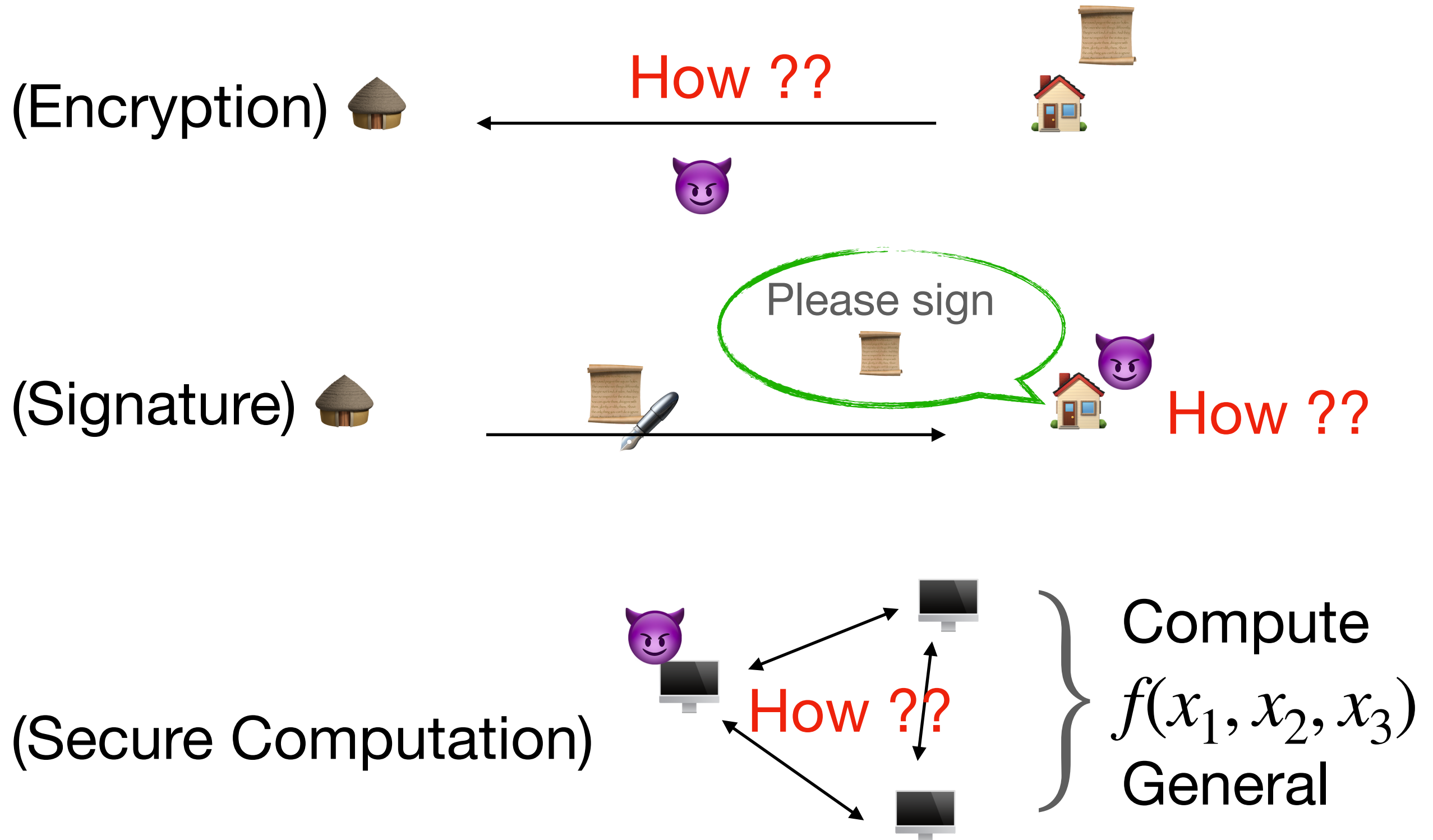
Context

Cryptography - Secure Communication, and More



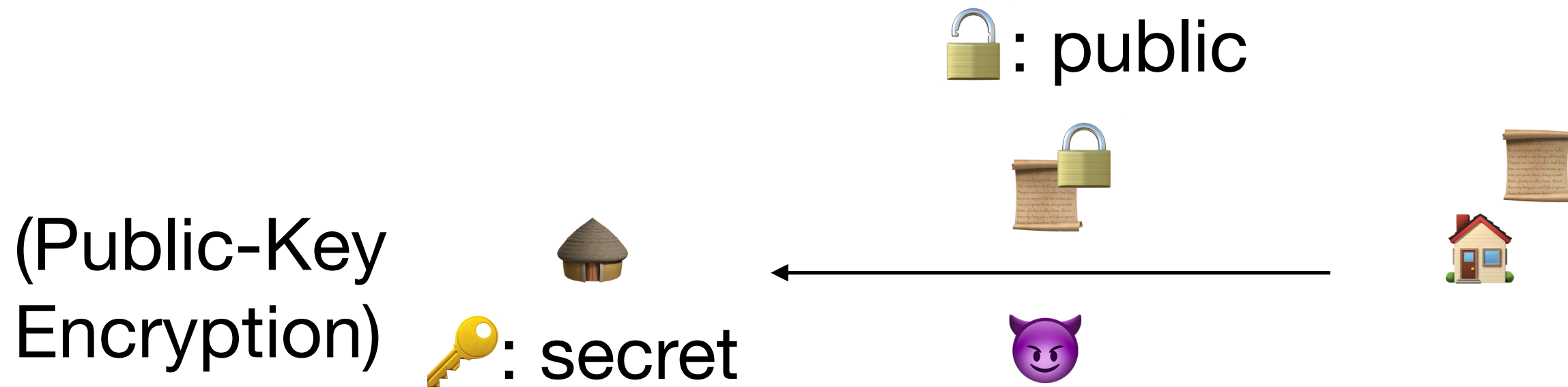
Context

Cryptography - Secure Communication, and More



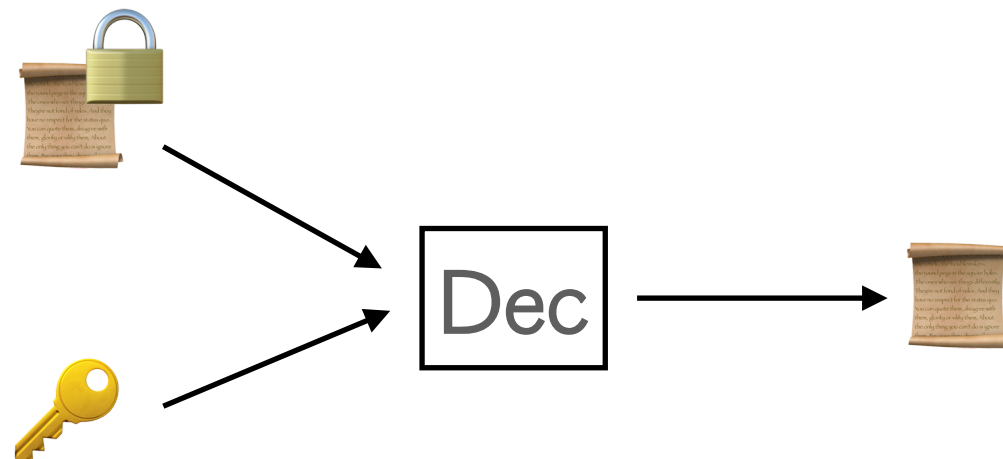
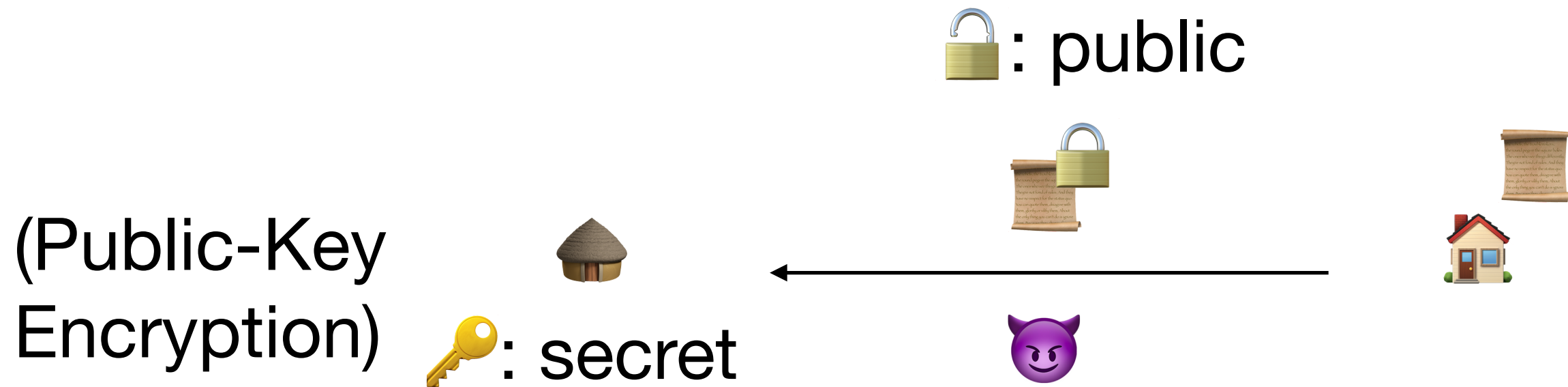
Context

Cryptography - Secure Communication, Publicly



Context

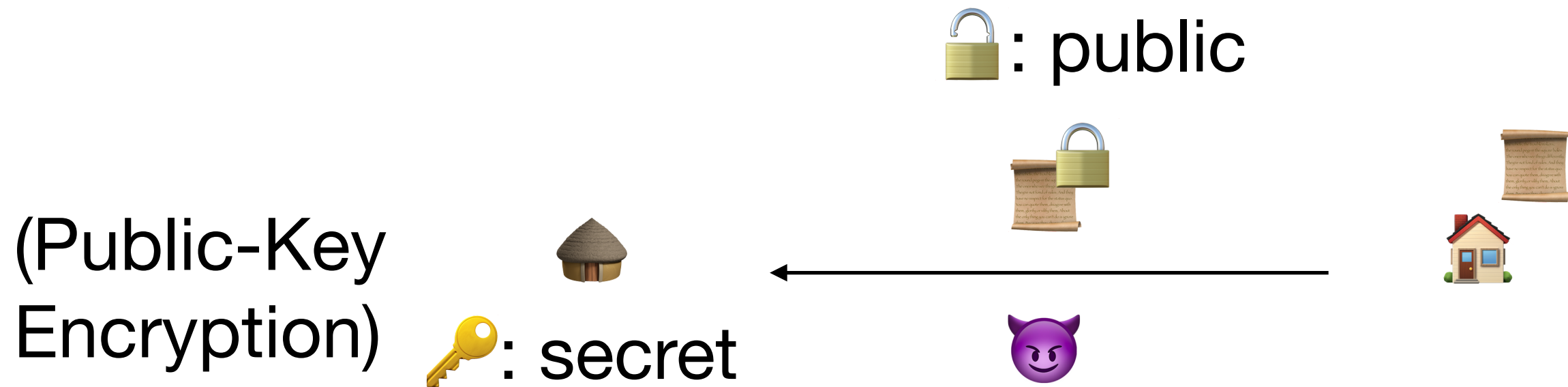
Cryptography - Secure Communication, Publicly



Decrypt “All ”
Or “Nothing”

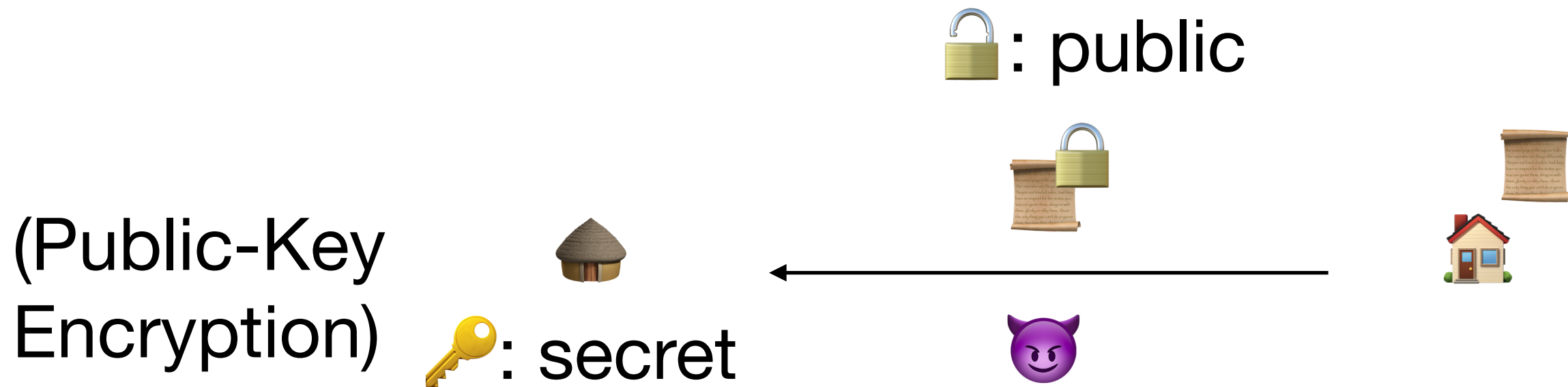
Context

Cryptography - Secure Communication, Publicly



Context

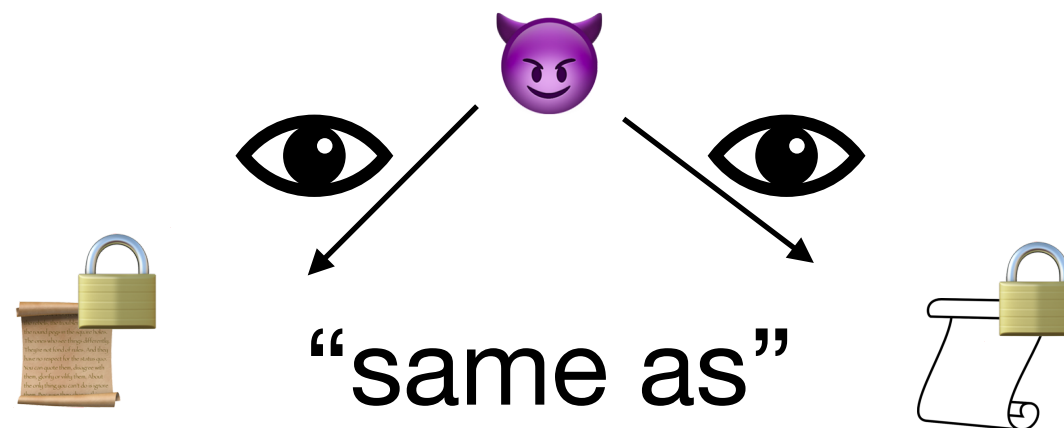
Cryptography - Secure Communication, Publicly



[Goldwasser-Micali'84]

“Nothing”

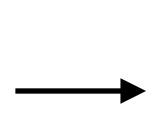
Semantic Security



Context

Cryptography - Secure Communication, Publicly with Control

(Identity-Based Encryption)



secret



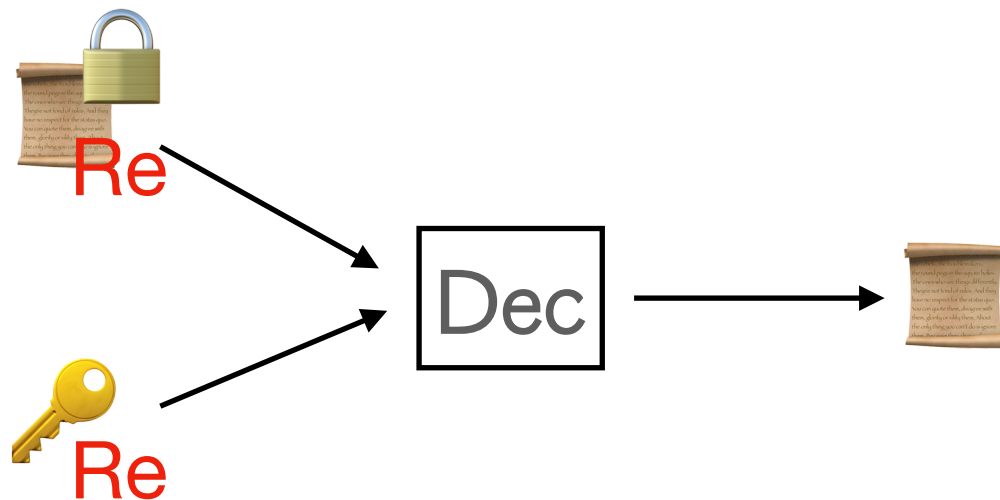
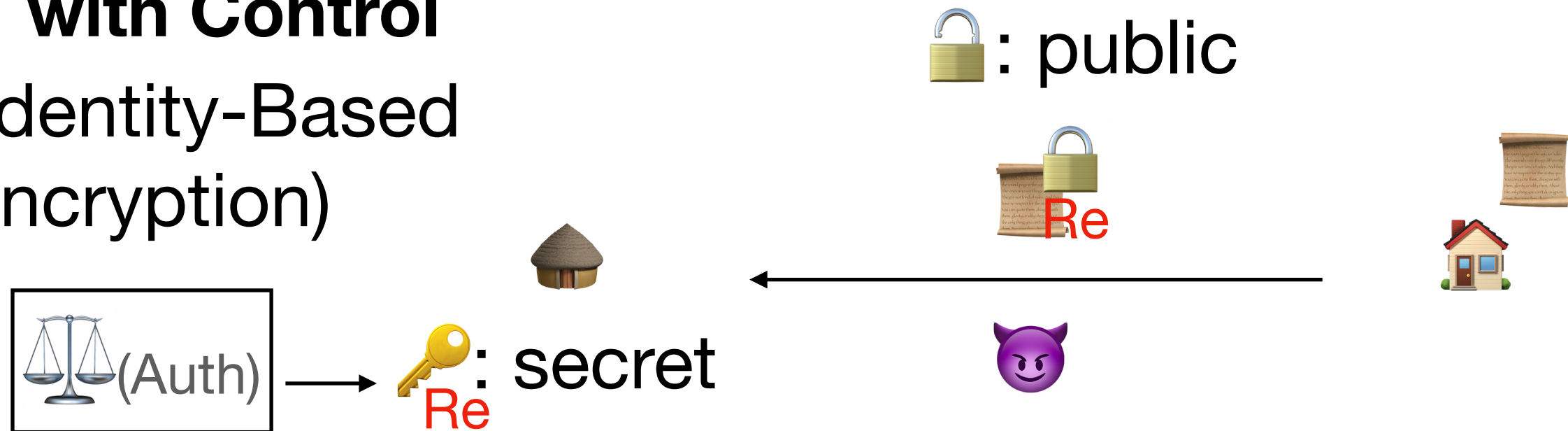
: public



Context

Cryptography - Secure Communication, Publicly with Control

(Identity-Based Encryption)

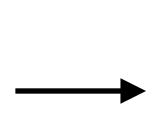


Decrypt “All ”
Or “Nothing”

Context

Cryptography - Secure Communication, Publicly with Control

(Identity-Based Encryption)



secret



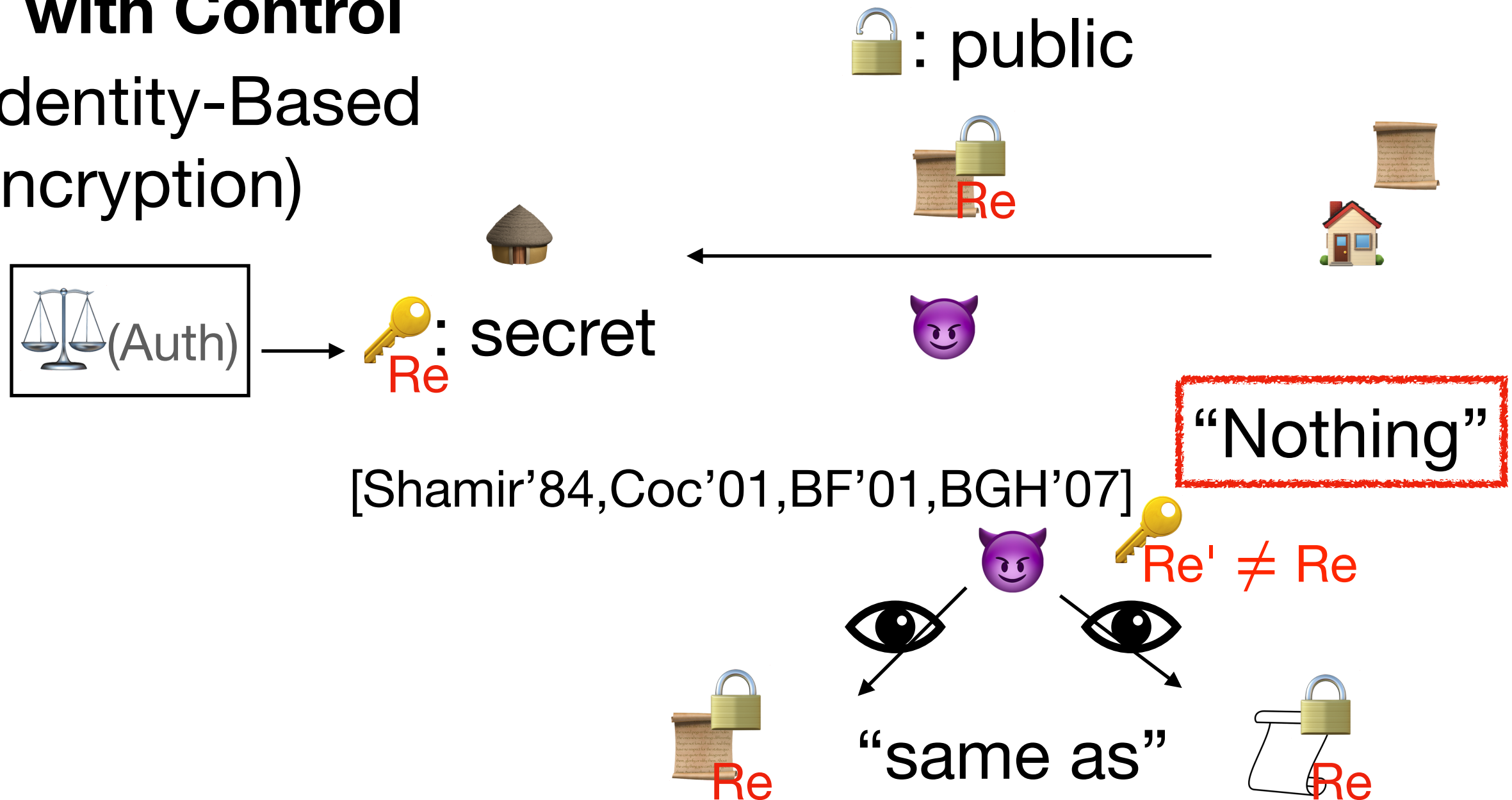
: public



Context

Cryptography - Secure Communication, Publicly with Control

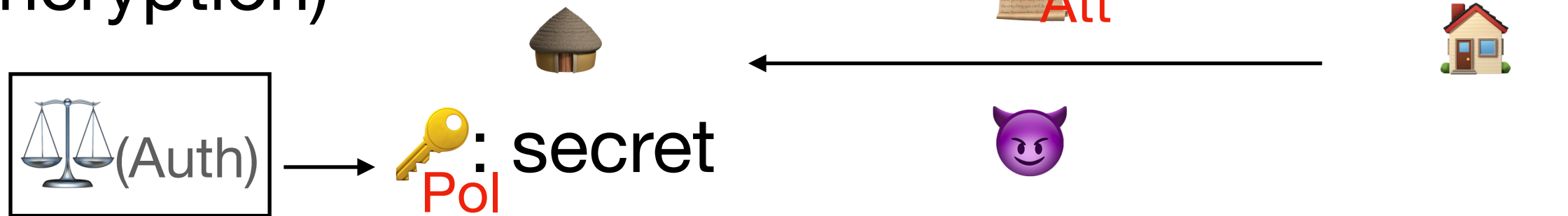
(Identity-Based Encryption)



Context

Cryptography - Secure Communication, Publicly with More Control

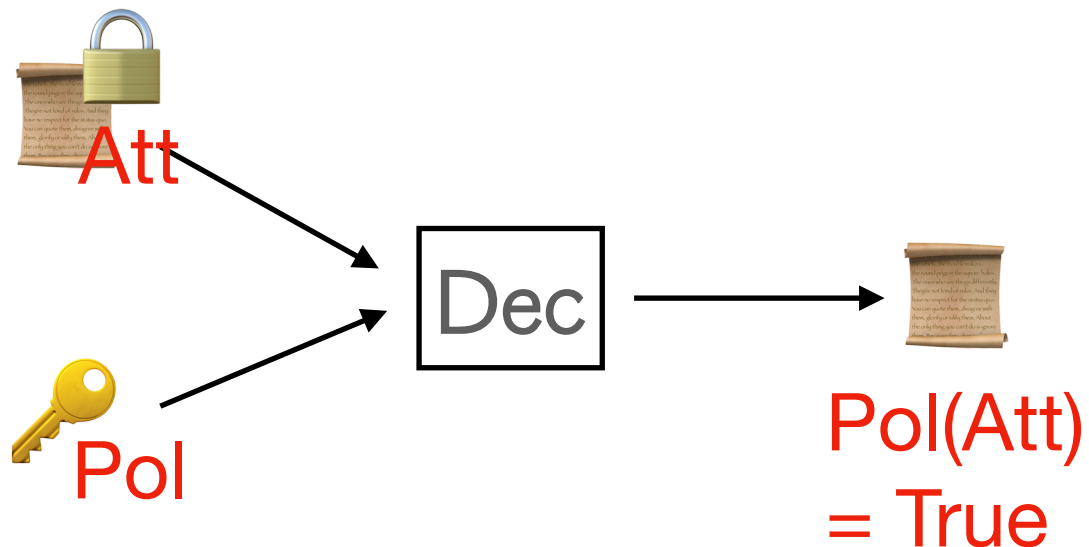
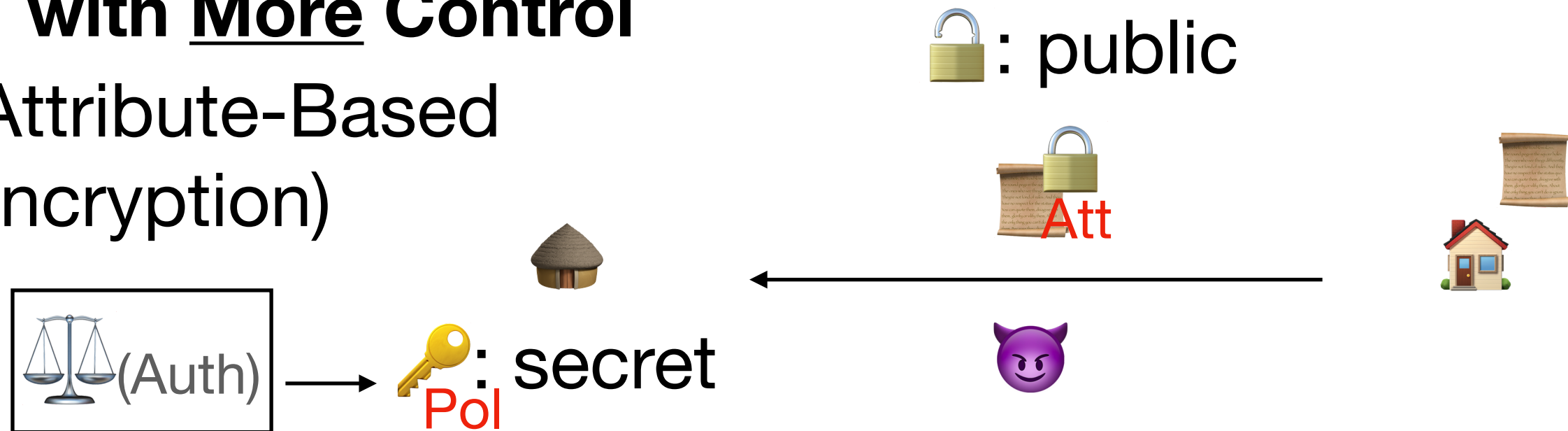
(Attribute-Based Encryption)



Context

Cryptography - Secure Communication, Publicly with More Control

(Attribute-Based Encryption)



Key-Policy

Decrypt “All ”
Or “Nothing”

Context

Cryptography - Secure Communication, Publicly with More Control

(Attribute-Based Encryption)



secret



: public



Context

Cryptography - Secure Communication, Publicly with More Control

(Attribute-Based Encryption)



→  : secret
Pol

[SW'05, GPSW'06]

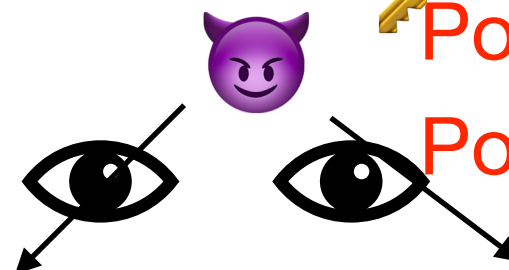
 : public



“Nothing”

 **Pol(Att-0) = False**

Pol(Att-1) = False



“same as”



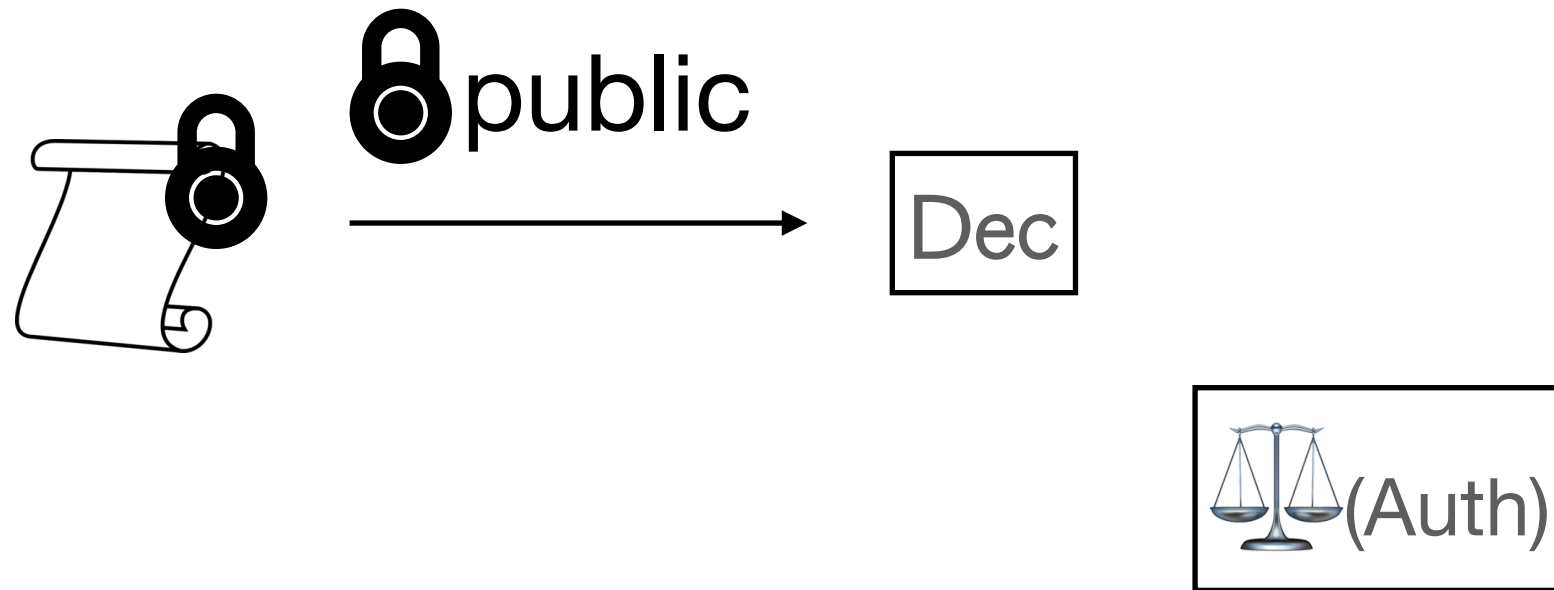
Att-0



Att-1

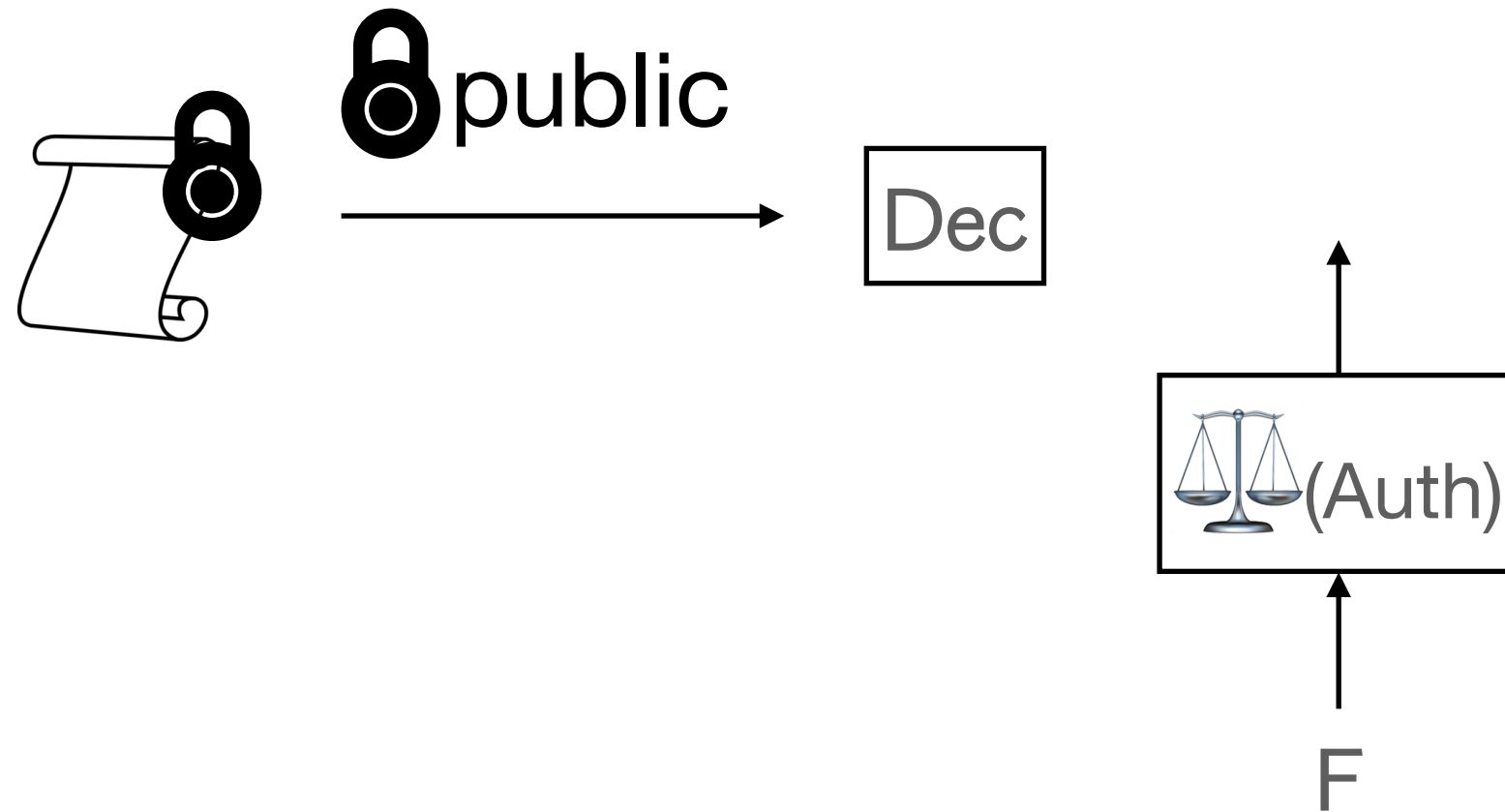
Our Story - The starting point

Functional Encryption (FE) [O'N10,BSW11]



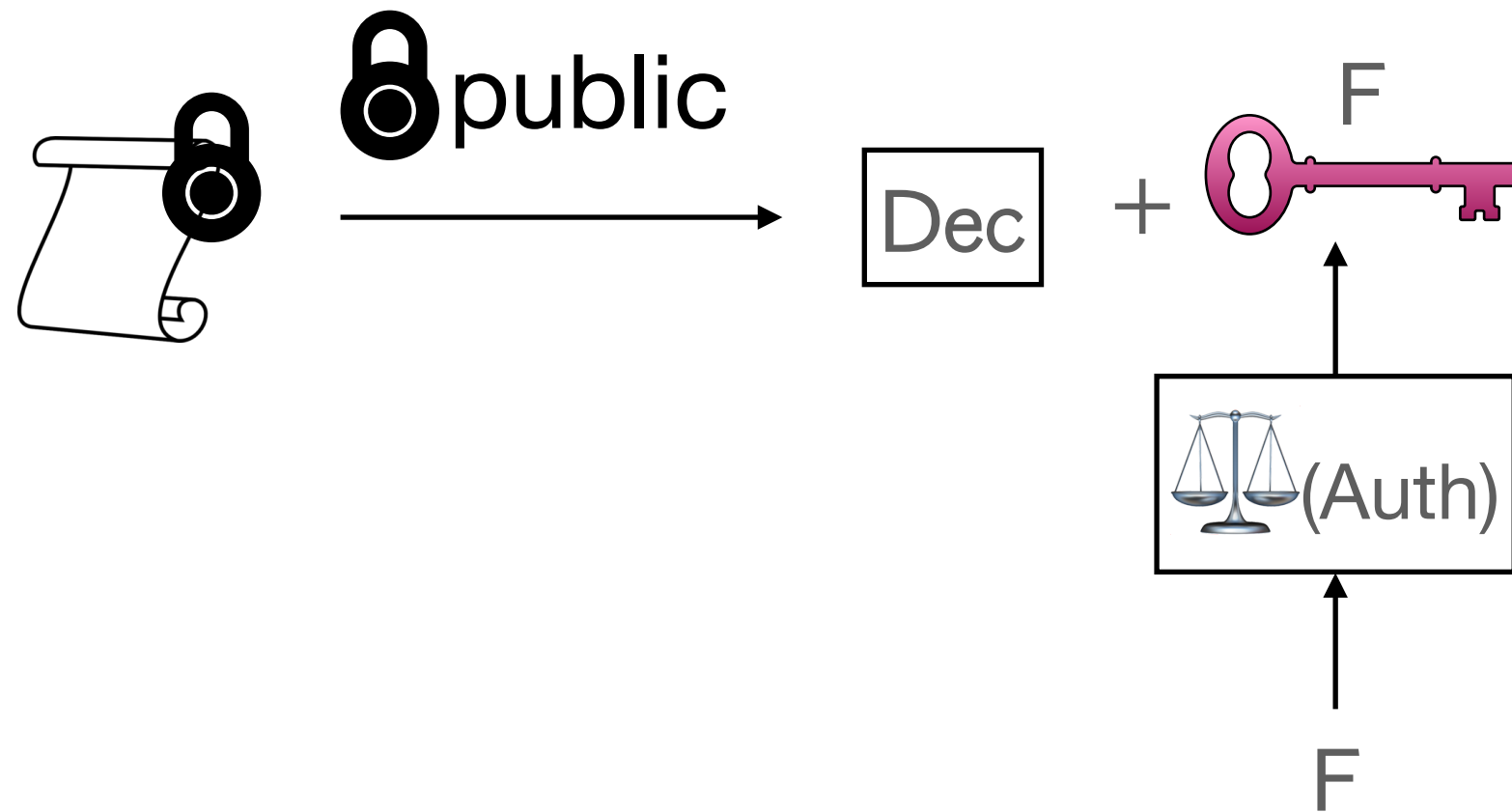
Our Story - The starting point

Functional Encryption (FE) [O'N10,BSW11]



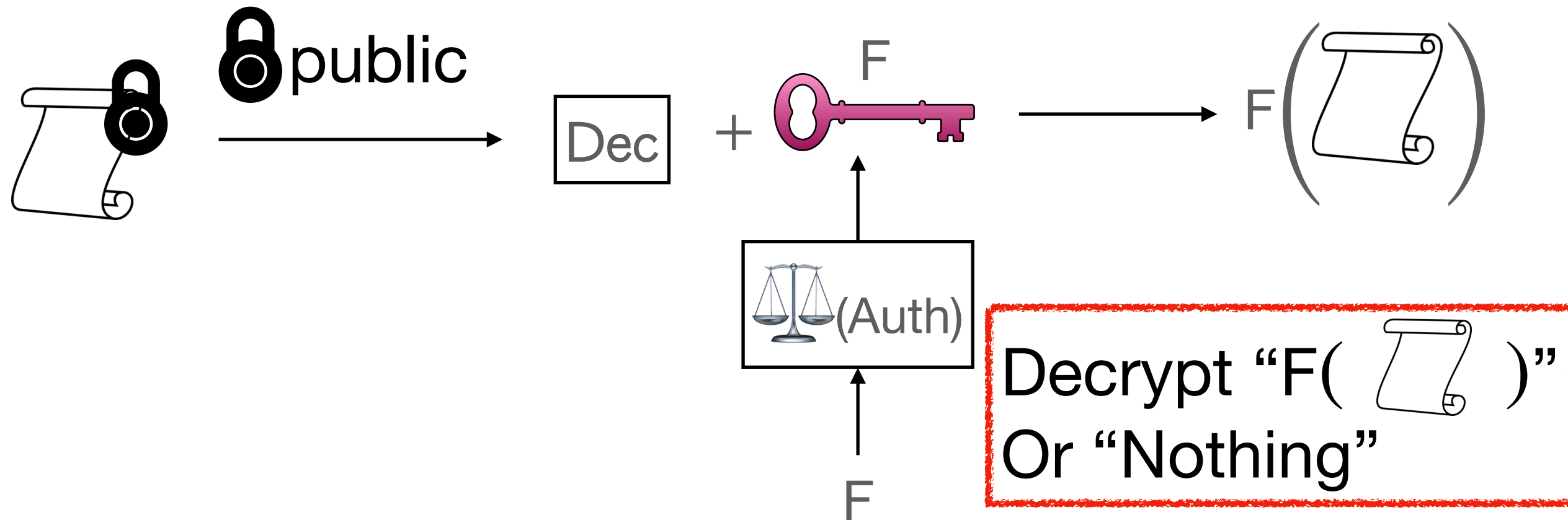
Our Story - The starting point

Functional Encryption (FE) [O'N10,BSW11]



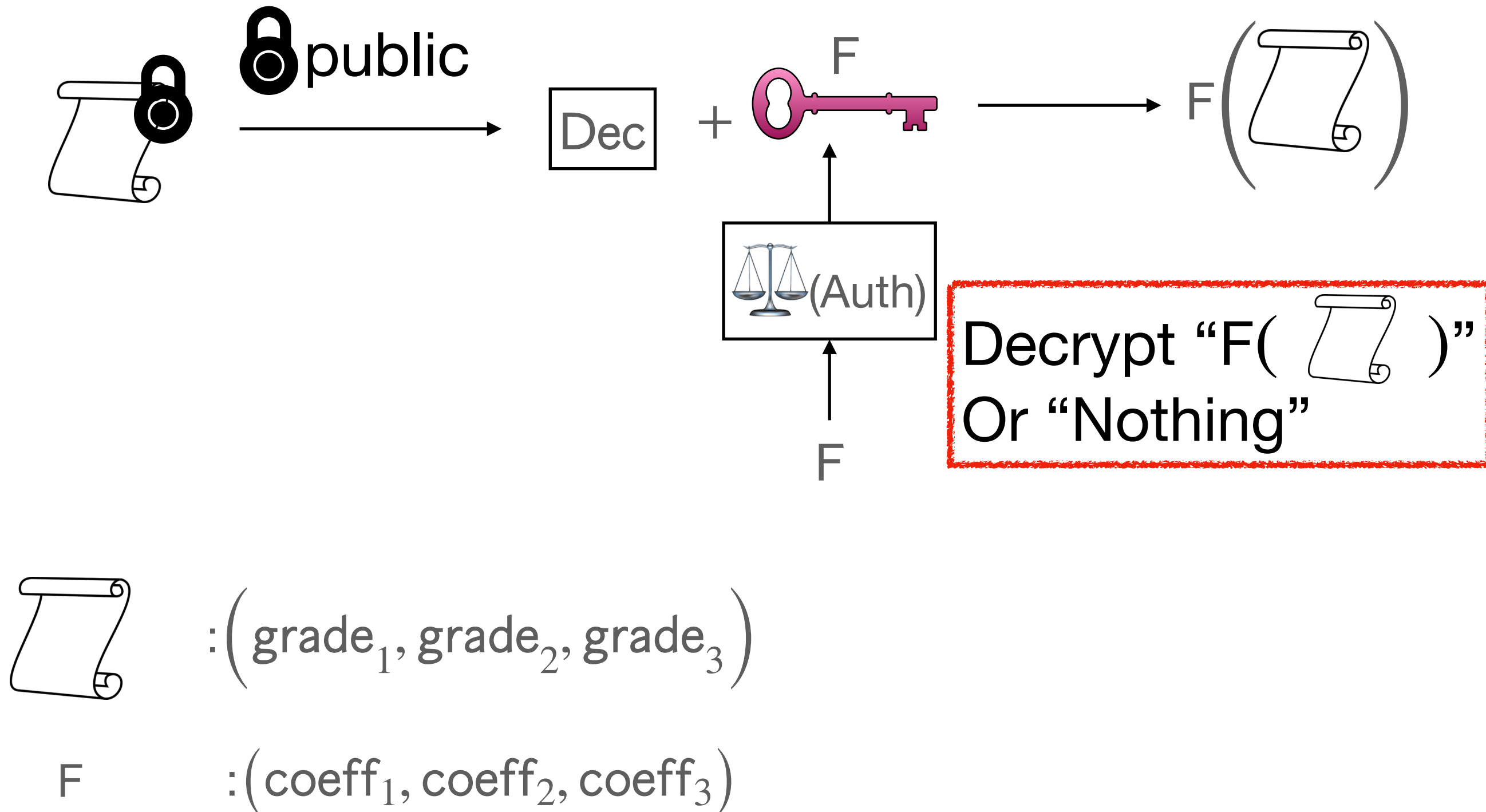
Our Story - The starting point

Functional Encryption (FE) [O’N10,BSW11]



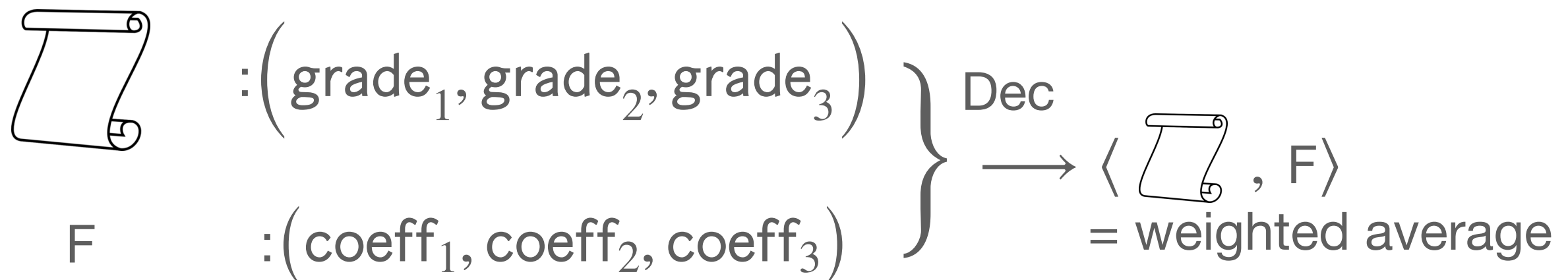
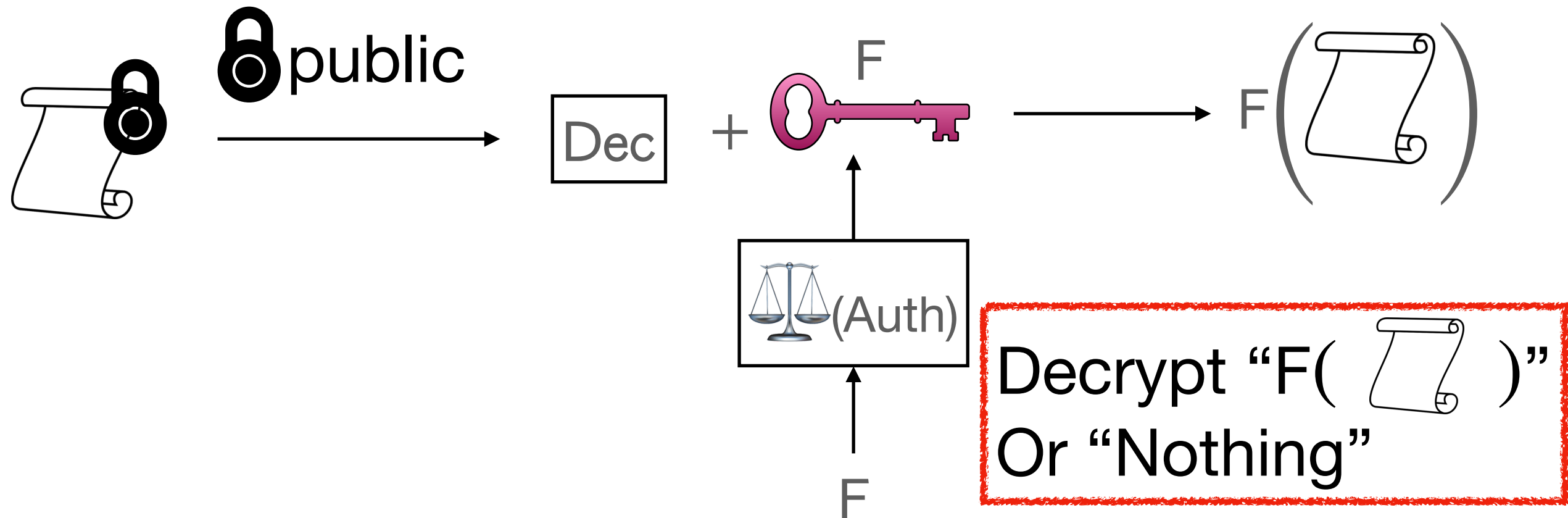
Our Story - The starting point

Functional Encryption (FE) [O’N10,BSW11]



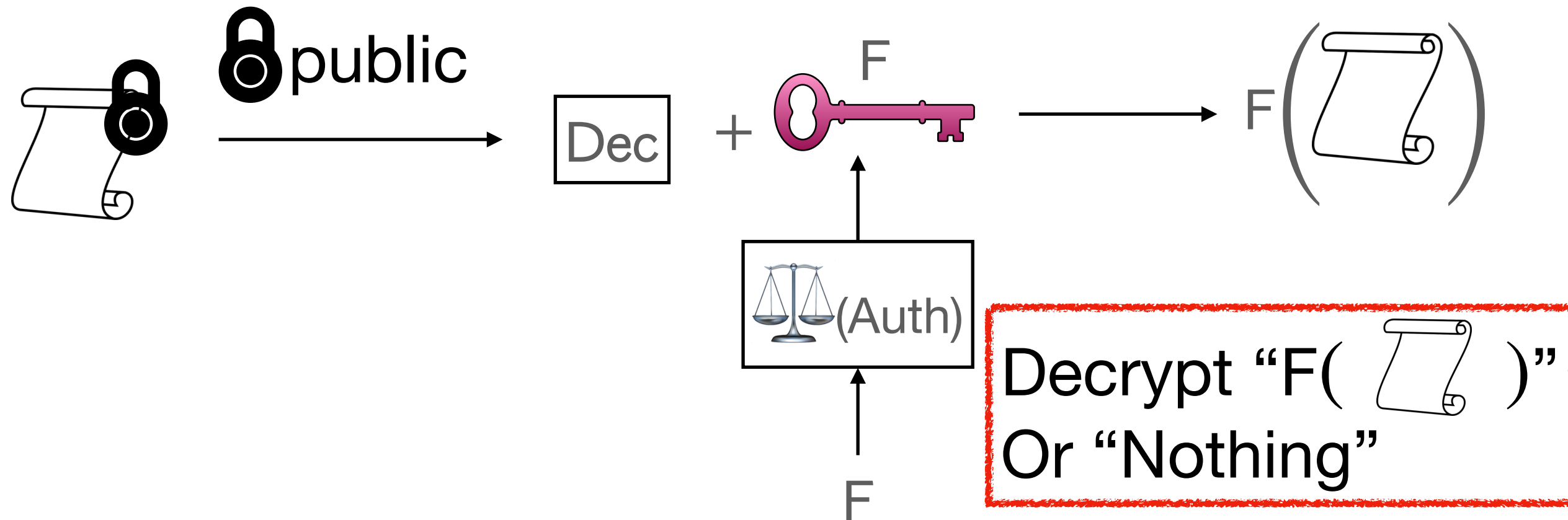
Our Story - The starting point

Functional Encryption (FE) [O’N10,BSW11]



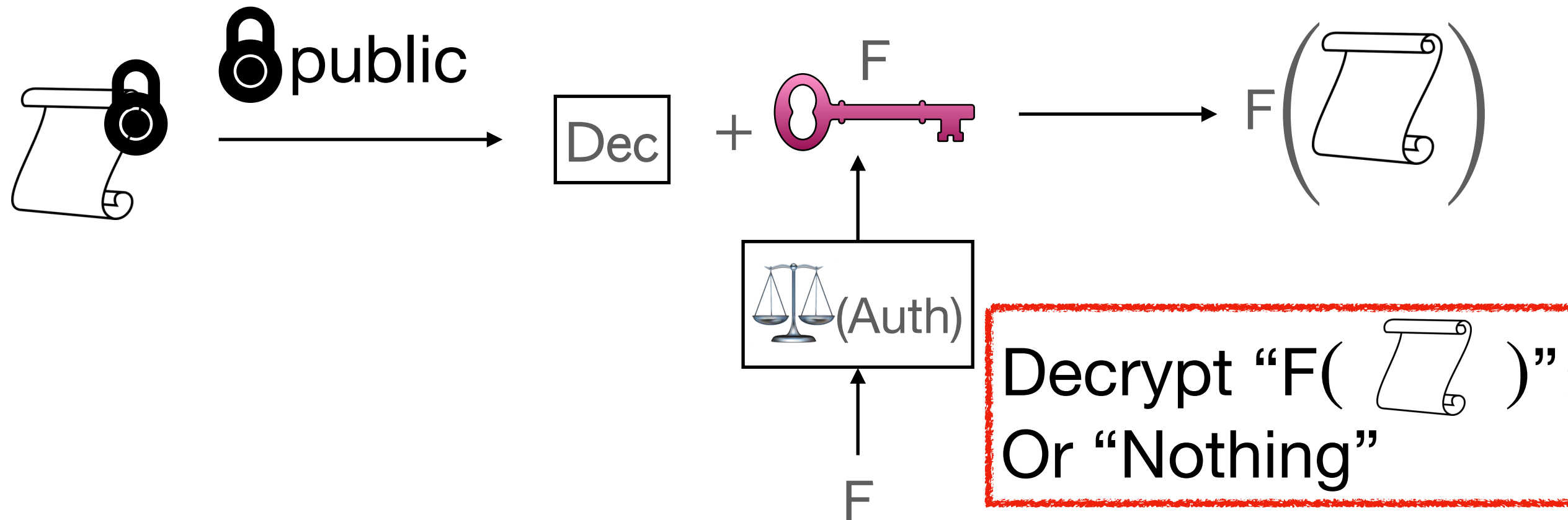
Our Story - The starting point

Functional Encryption (FE) [O’N10,BSW11]



Our Story - The starting point

Functional Encryption (FE) [O'N10,BSW11]



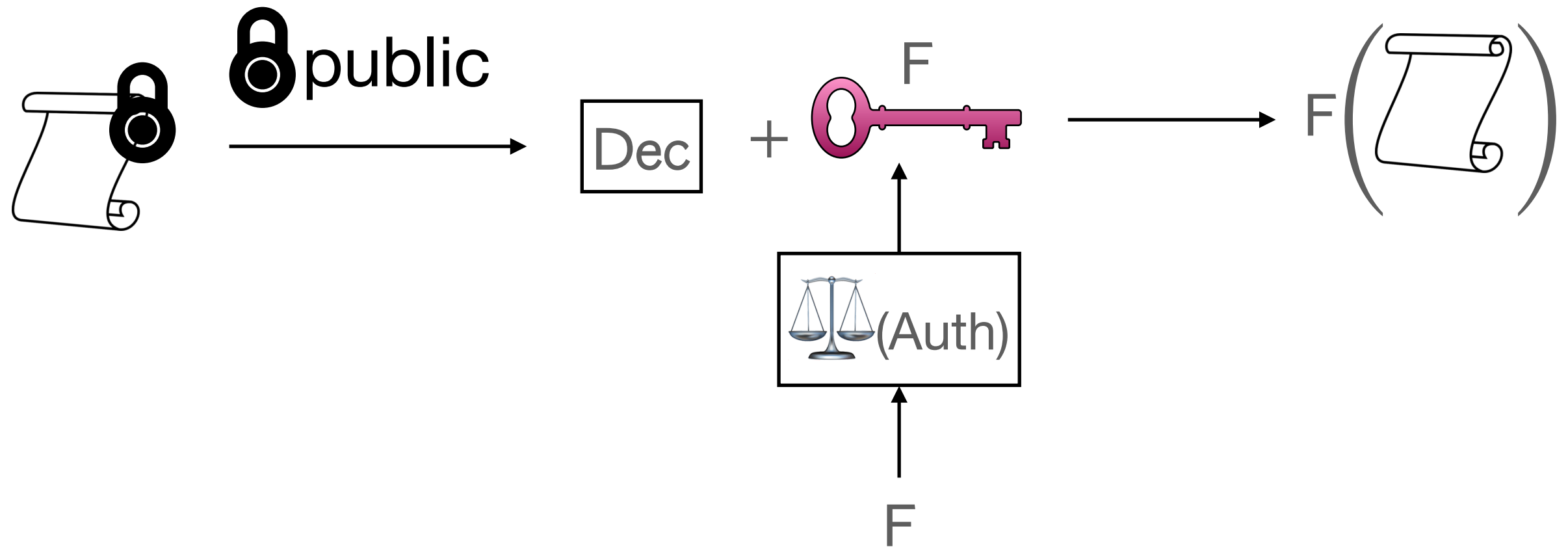
[ABDP'15]

FE computing inner-product: $\left\langle \text{document icon}, F \right\rangle$

—> Follow-ups: [ALS16,ABDP16,BBL17,CLT18] and more

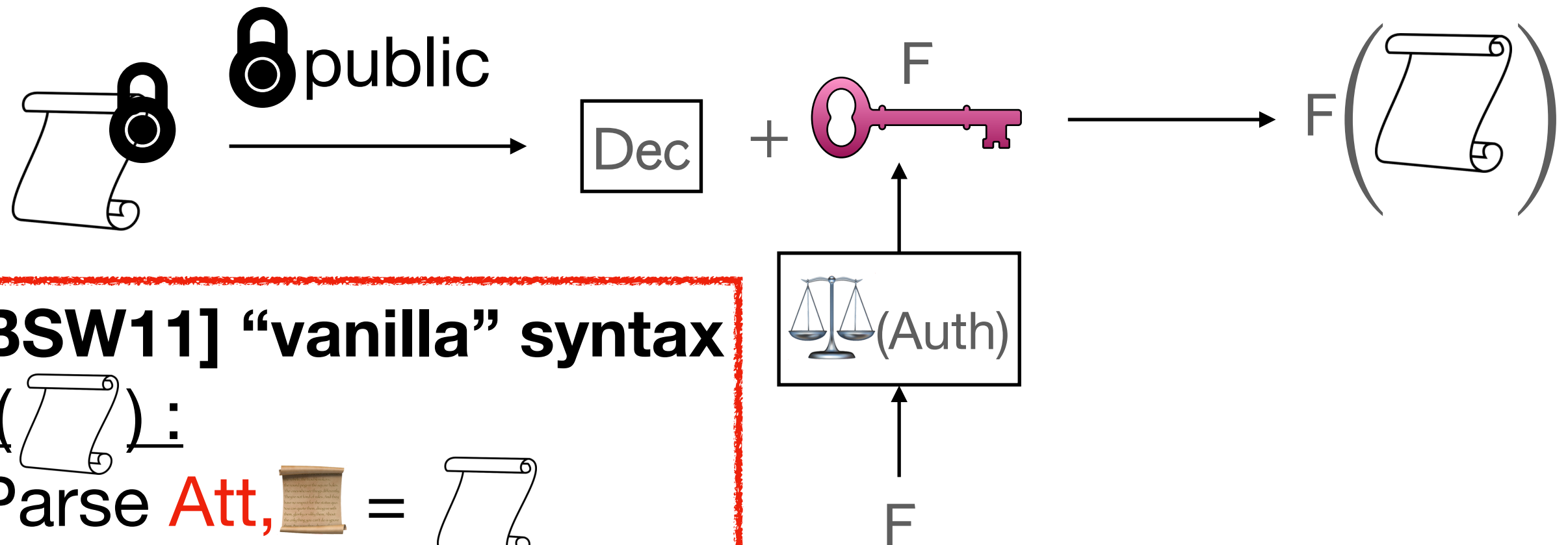
Our Story - The starting point

FE [O'N10,BSW11]: Capturing ABE ?



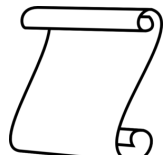
Our Story - The starting point

FE [O'N10,BSW11]: Capturing ABE ?



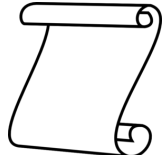
[BSW11] “vanilla” syntax

$F(\text{document icon})$:

Parse **Att**,  = 

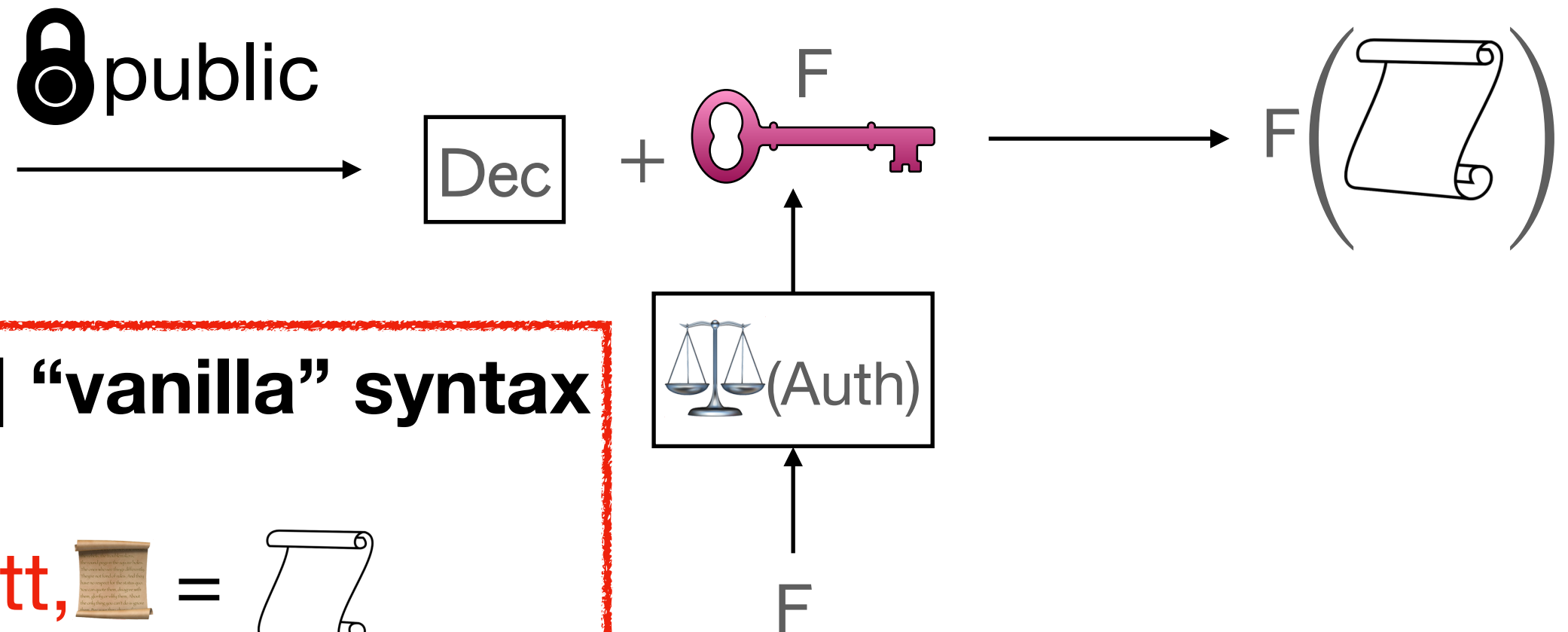
If **Pol(Att) = True** then 

Else “Nothing”

Att,  =  $\xrightarrow{(\text{Enc})}$ 

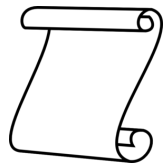
Our Story - The starting point

FE [O’N10,BSW11]: Capturing ABE ?



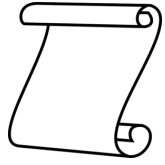
[BSW11] “vanilla” syntax

$F(\text{scroll icon})$:

Parse **Att**,  = 

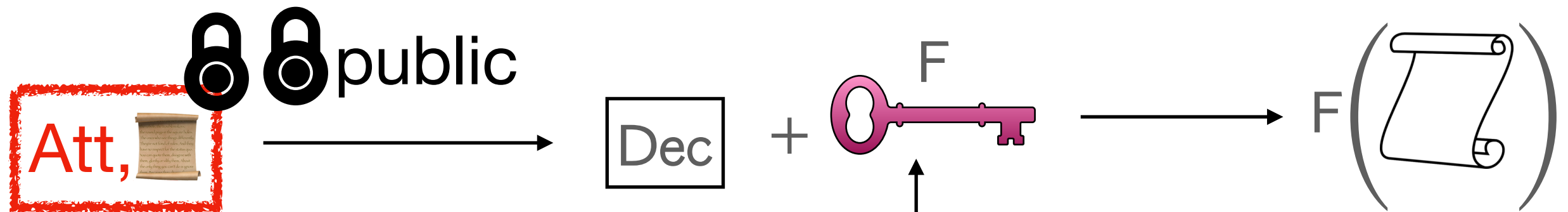
If **Pol(Att) = True** then 

Else “Nothing”

Att,  =  $\xrightarrow{(\text{Enc})}$ 

Our Story - The starting point

FE [O'N10,BSW11]: Capturing ABE ?



[BSW11] “vanilla” syntax

$F(\text{Dec})$:

Parse Att, =

If $\text{Pol}(\text{Att}) = \text{True}$ then

Else “Nothing”

Att, = $\xrightarrow{(\text{Enc})}$



Our Story - The starting point

FE [O'N10,BSW11]: Capturing ABE ?



[BSW11] “vanilla” syntax

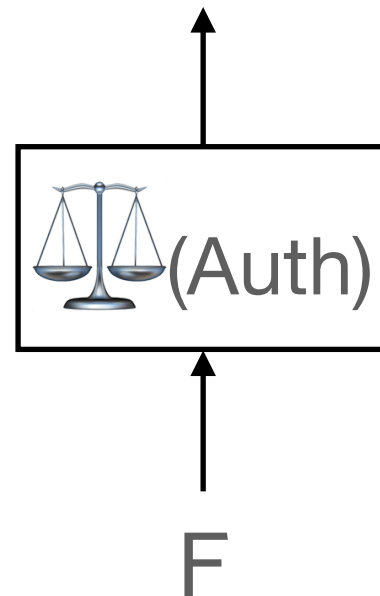
$F(\text{scroll icon})$:

Parse Att, =

If $\text{Pol}(\text{Att}) = \text{True}$ then

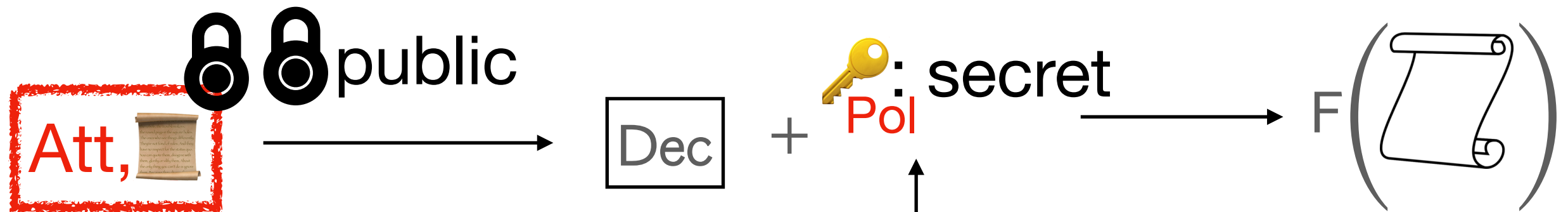
Else “Nothing”

Att, = $\xrightarrow{(\text{Enc})}$



Our Story - The starting point

FE [O'N10,BSW11]: Capturing ABE ?



[BSW11] “vanilla” syntax

$F(\text{scroll icon})$:

Parse Att, =

If Pol(Att) = True then

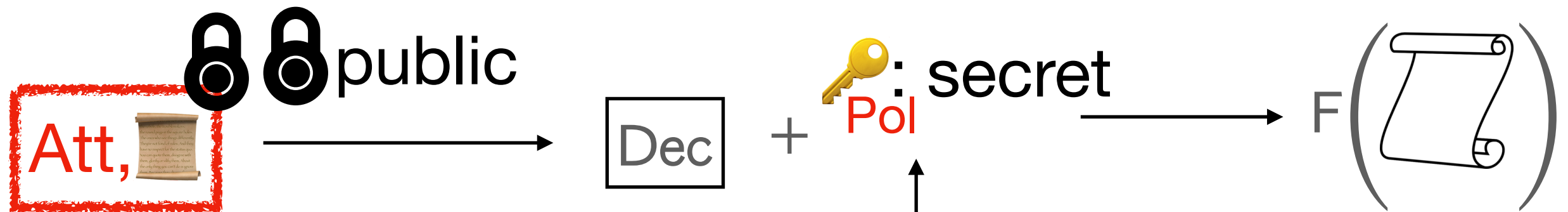
Else “Nothing”

Att, = $\xrightarrow{(\text{Enc})}$



Our Story - The starting point

FE [O'N10,BSW11]: Capturing ABE ?



[BSW11] “vanilla” syntax

$F(\text{scroll})$:

Parse **Att**, =

If **Pol(Att) = True** then

Else “Nothing”

Att, = $\xrightarrow{(\text{Enc})}$

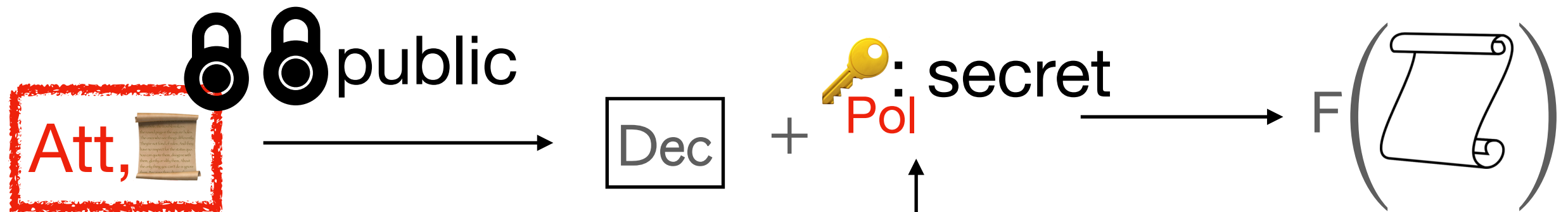


F

Decrypt “ $F(\text{scroll})$ ”
Or “Nothing”

Our Story - The starting point

FE [O’N10,BSW11]: Capturing ABE ?



[BSW11] “vanilla” syntax

$F(\text{scroll icon})$:

Parse Att, =

If Pol(Att) = True then

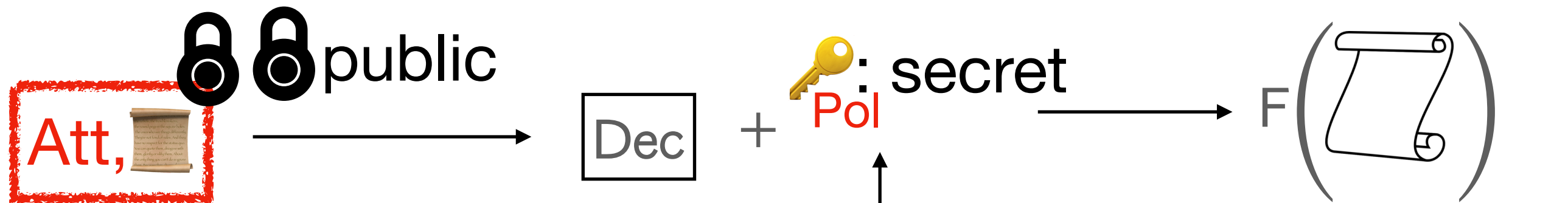
Else “Nothing”

Att, = $\xrightarrow{(\text{Enc})}$



Our Story - The starting point

FE [O'N10,BSW11]: Capturing ABE ?



[BSW11] “vanilla” syntax

$F(\text{scroll})$:

Parse $\text{Att, scroll} = \text{scroll}$

If $\text{Pol}(\text{Att}) = \text{True}$ then scroll

Else “Nothing”

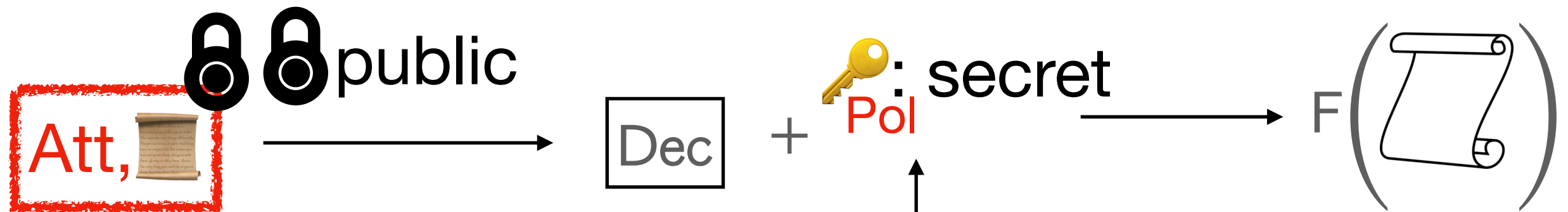
$\text{Att, scroll} = \text{scroll} \xrightarrow{(\text{Enc})} \text{scroll with lock}$



F
Decrypt $\text{Att, scroll with lock} \rightarrow \text{“All scroll”}$
If $\text{Pol}(\text{Att}) = \text{True}$
Or “Nothing”

Our Story - The starting point

FE [O'N10,BSW11]: Capturing ABE ?



[BSW11] “vanilla” syntax

$F(\text{$):

Parse Att, =

If Pol(Att) = True then

Else “Nothing”

Att, = $\xrightarrow{(\text{Enc})}$

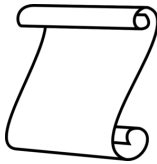


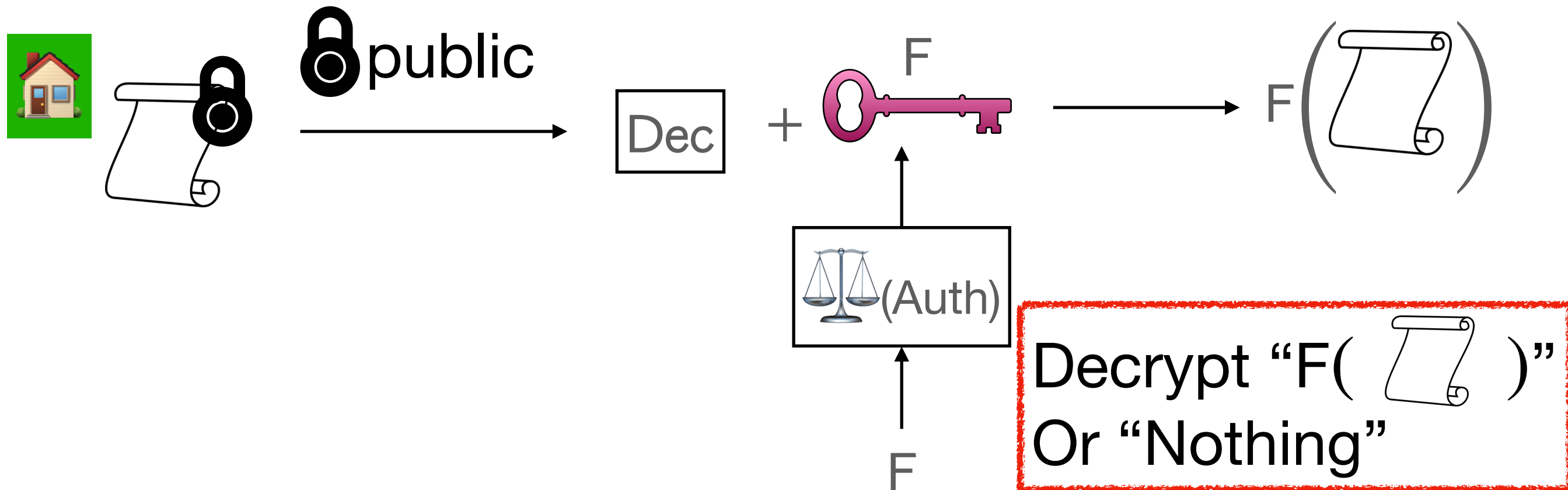
F

Syntactical Problem:

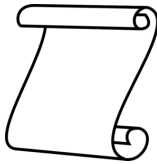
NOT allow public-Att

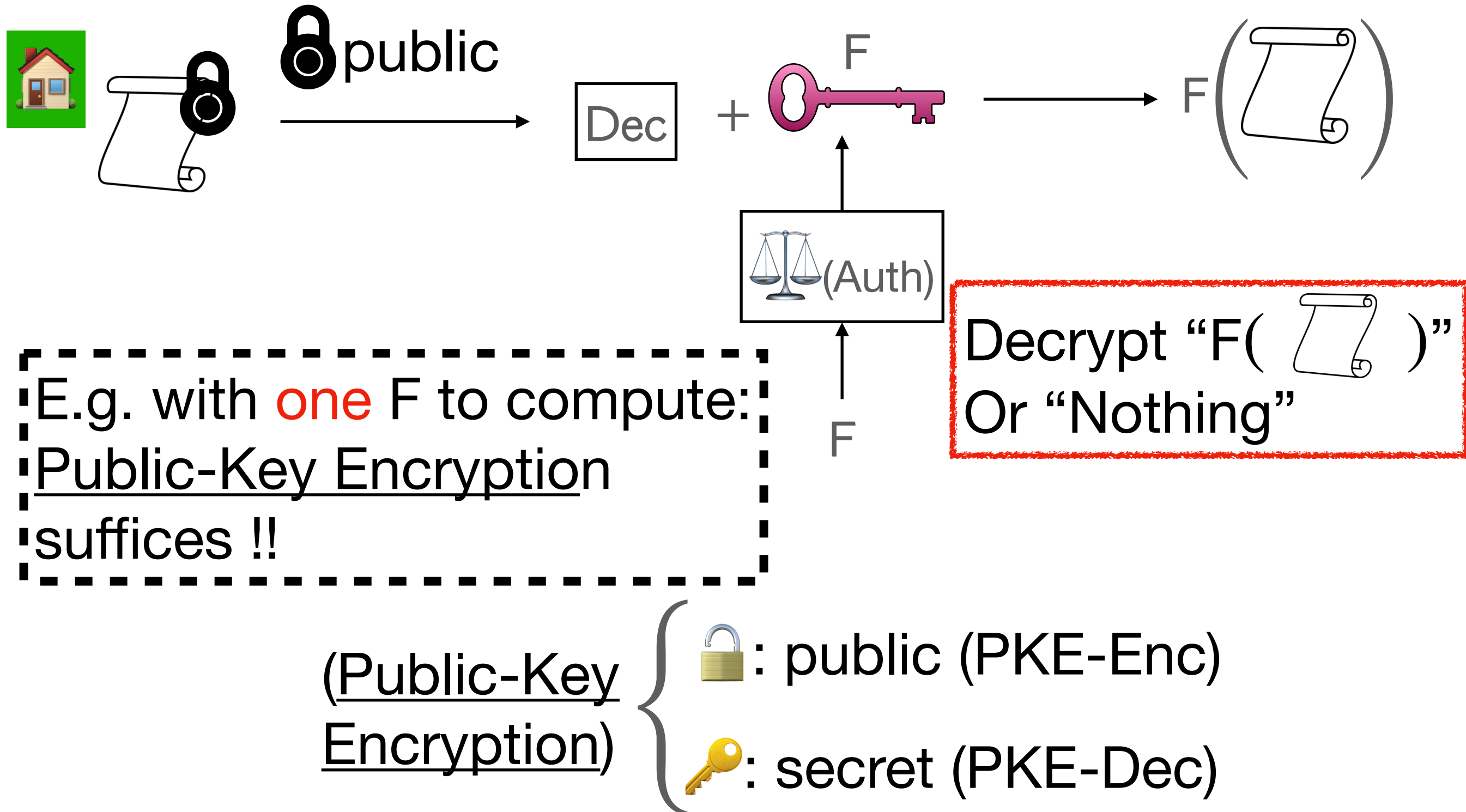
Next in the Story

FE - When  comes from 1 source

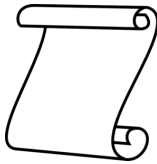


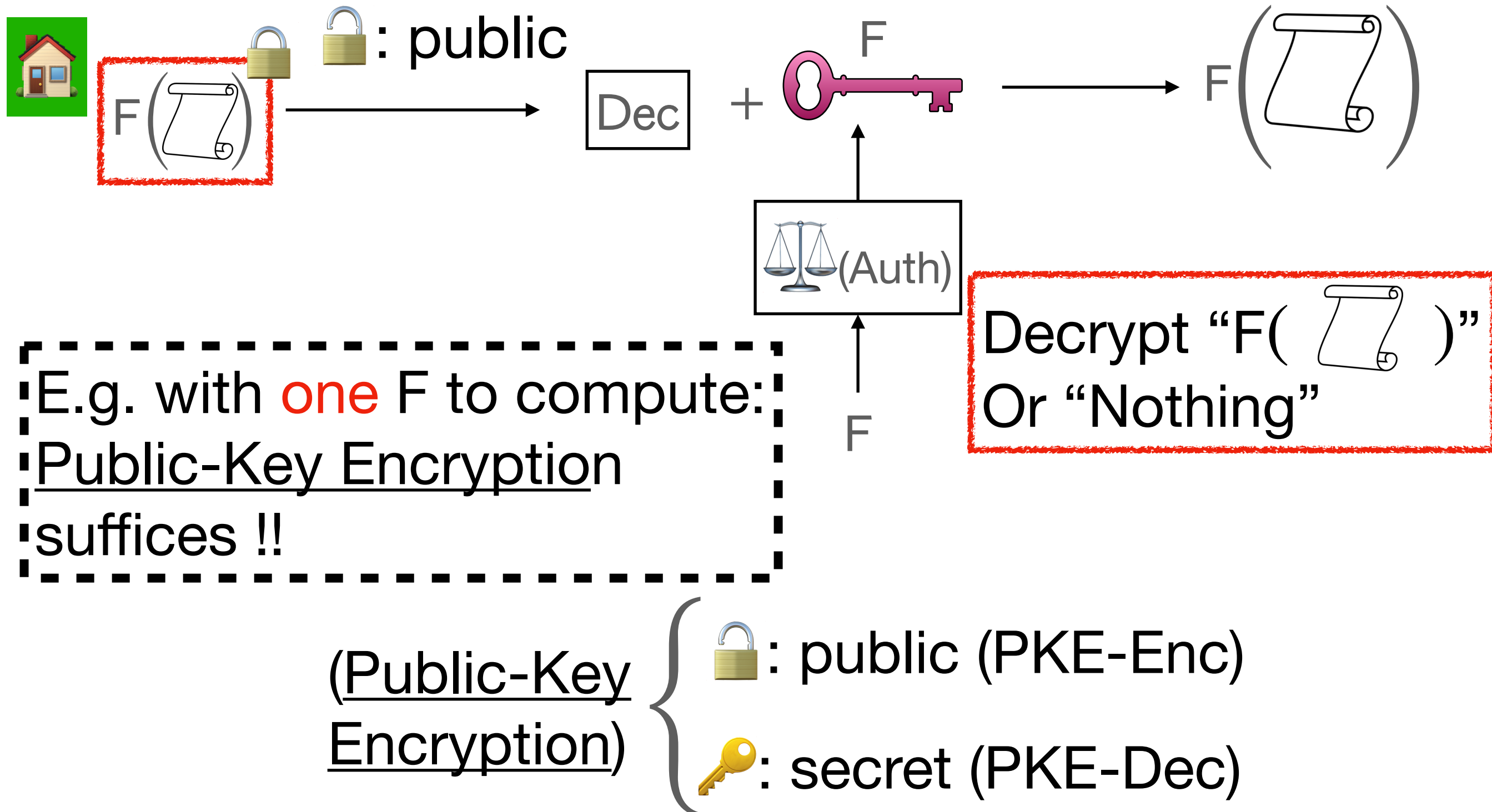
Next in the Story

FE - When  comes from 1 source

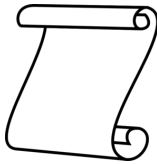


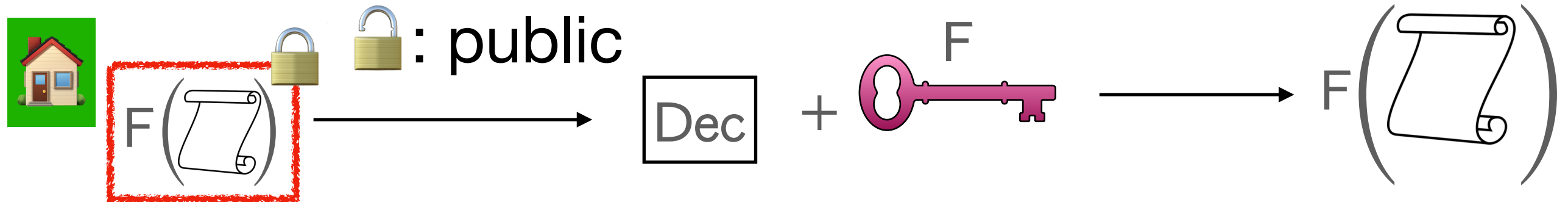
Next in the Story

FE - When  comes from 1 source



Next in the Story

FE - When  comes from 1 source

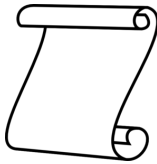


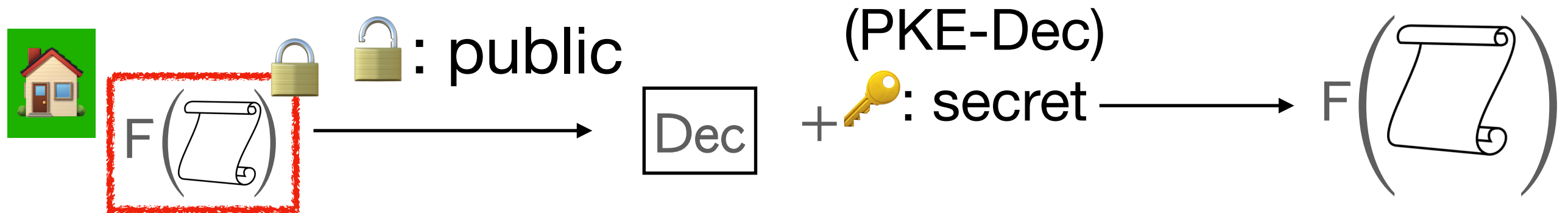
E.g. with **one** F to compute:
Public-Key Encryption
suffices !!

Decrypt " $F(\text{scroll})$ "
Or "Nothing"

(Public-Key Encryption) {  : public (PKE-Enc)
 : secret (PKE-Dec)

Next in the Story

FE - When  comes from 1 source

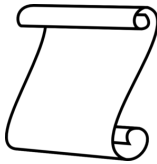


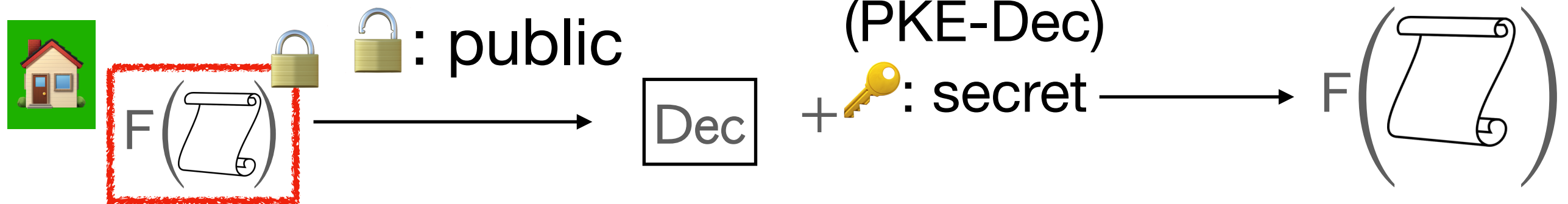
E.g. with **one** F to compute:
Public-Key Encryption
suffices !!

Decrypt “ $F(\text{scroll})$ ”
Or “Nothing”

(Public-Key Encryption) {  : public (PKE-Enc)
 : secret (PKE-Dec)

Next in the Story

FE - When  comes from 1 source

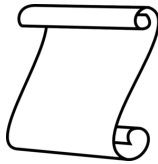


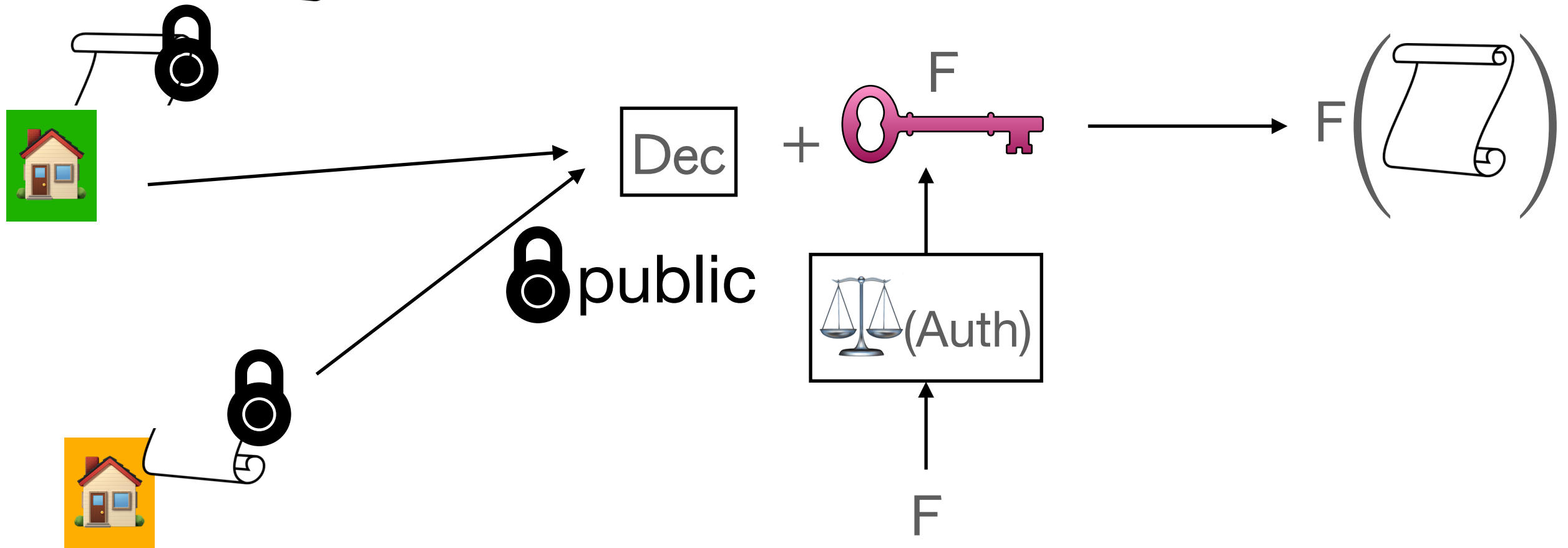
E.g. with **one** F to compute:
Public-Key Encryption
suffices !!

Decrypt “ $F(\text{scroll})$ ”
Or “Nothing”

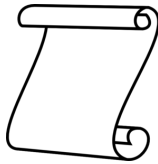
(Public-Key Encryption) {  : public (PKE-Enc)
 : secret (PKE-Dec)

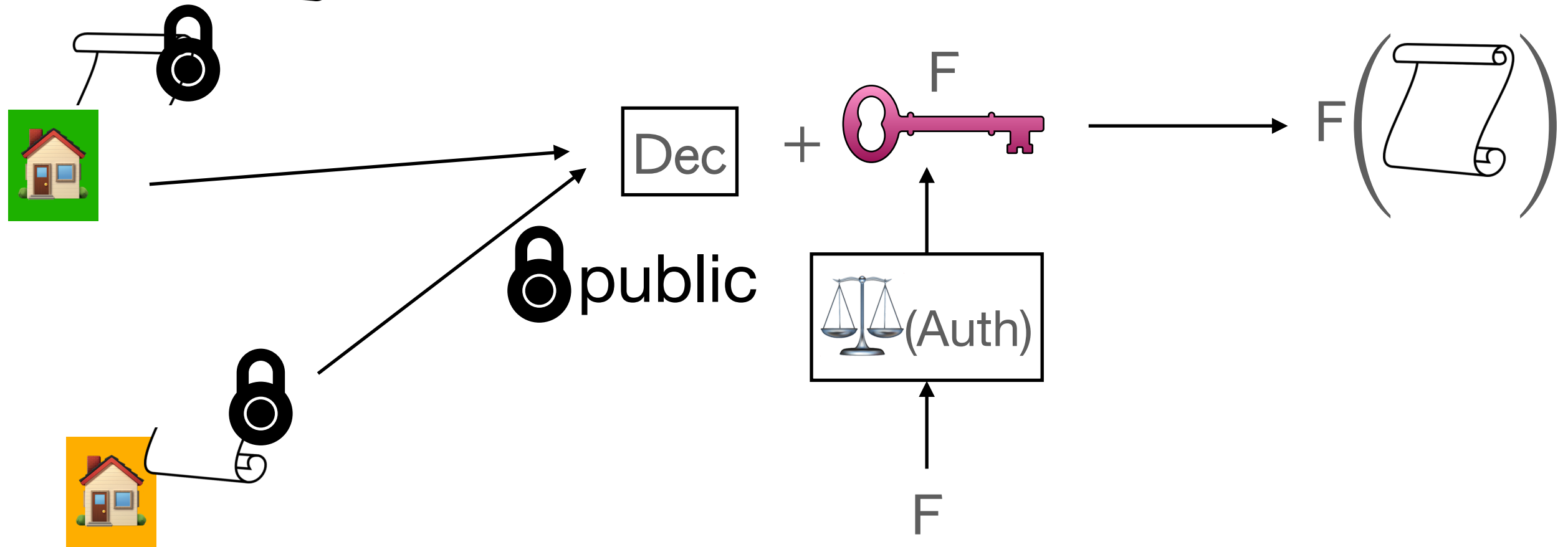
Next in the Story

And if  comes from ≥ 2 sources



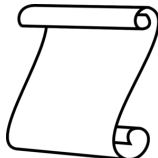
Next in the Story

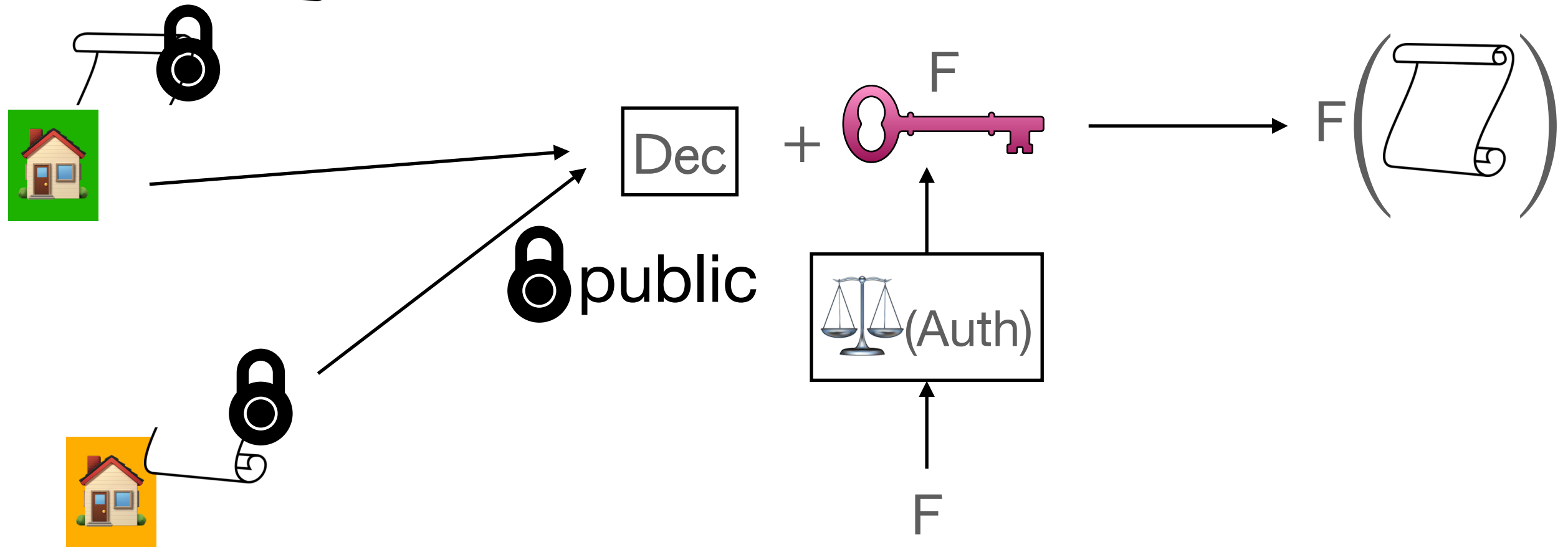
And if  comes from ≥ 2 sources



! Even only **one** F to compute:
! Public-Key Encryption
! Is NOT enough !!

Next in the Story

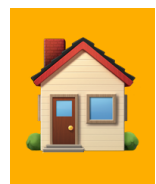
And if  comes from ≥ 2 sources



Even only **one** F to compute:
Public-Key Encryption
Is NOT enough !!



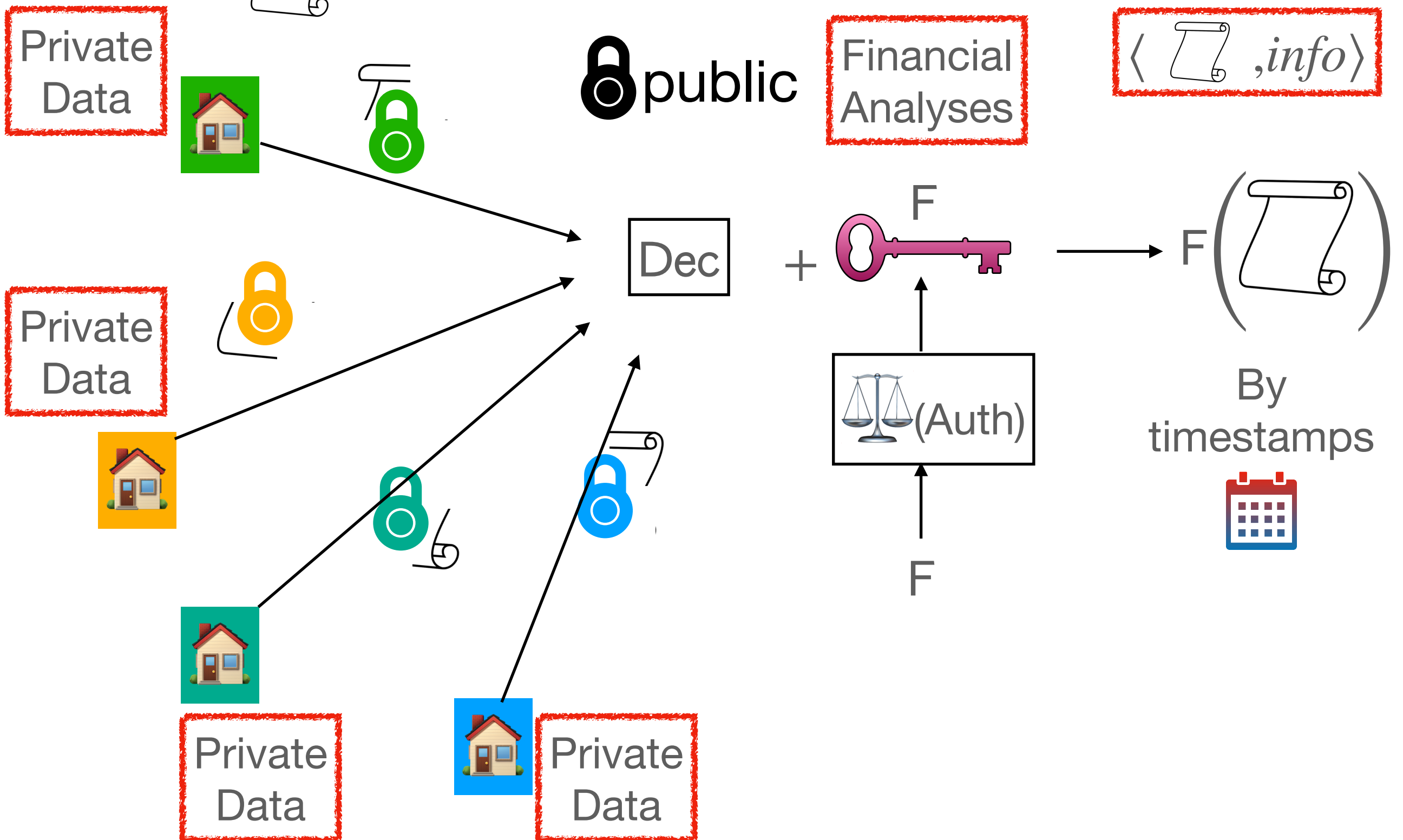
$\neq$



Independent

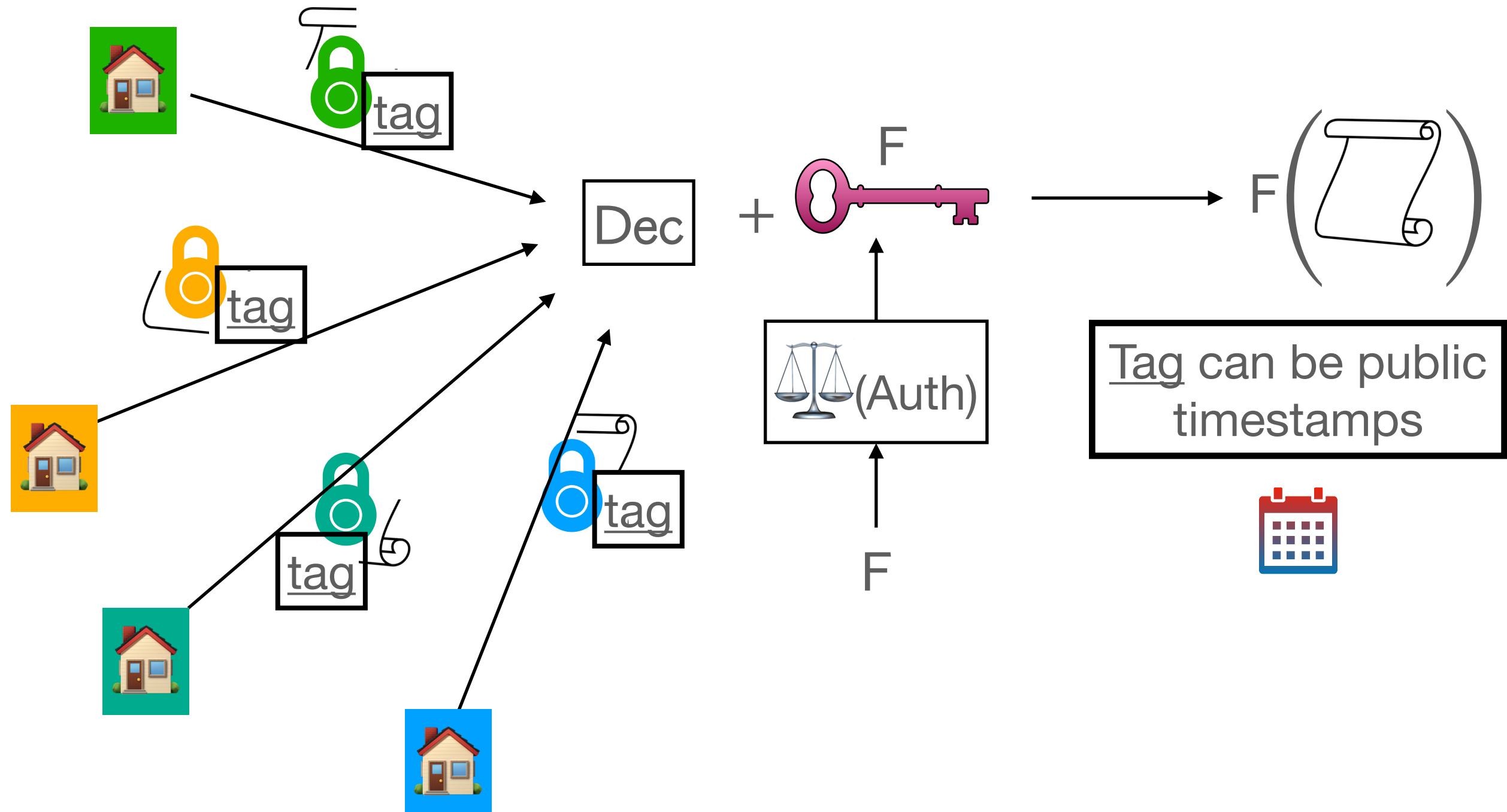
Example

And if  comes from ≥ 2 sources



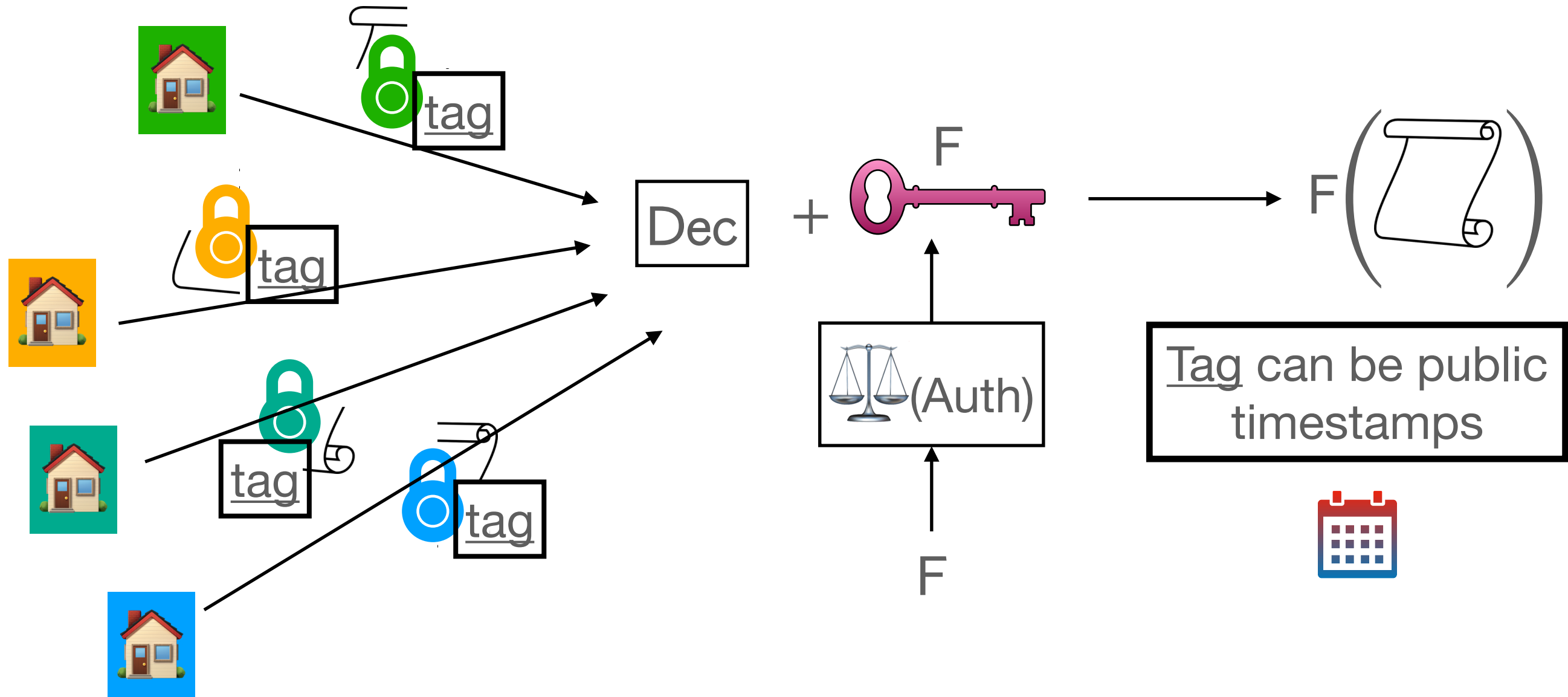
Next in the Story

Multi-Client Functional Encryption (MCFE) [GGG+14, CDGPP18]



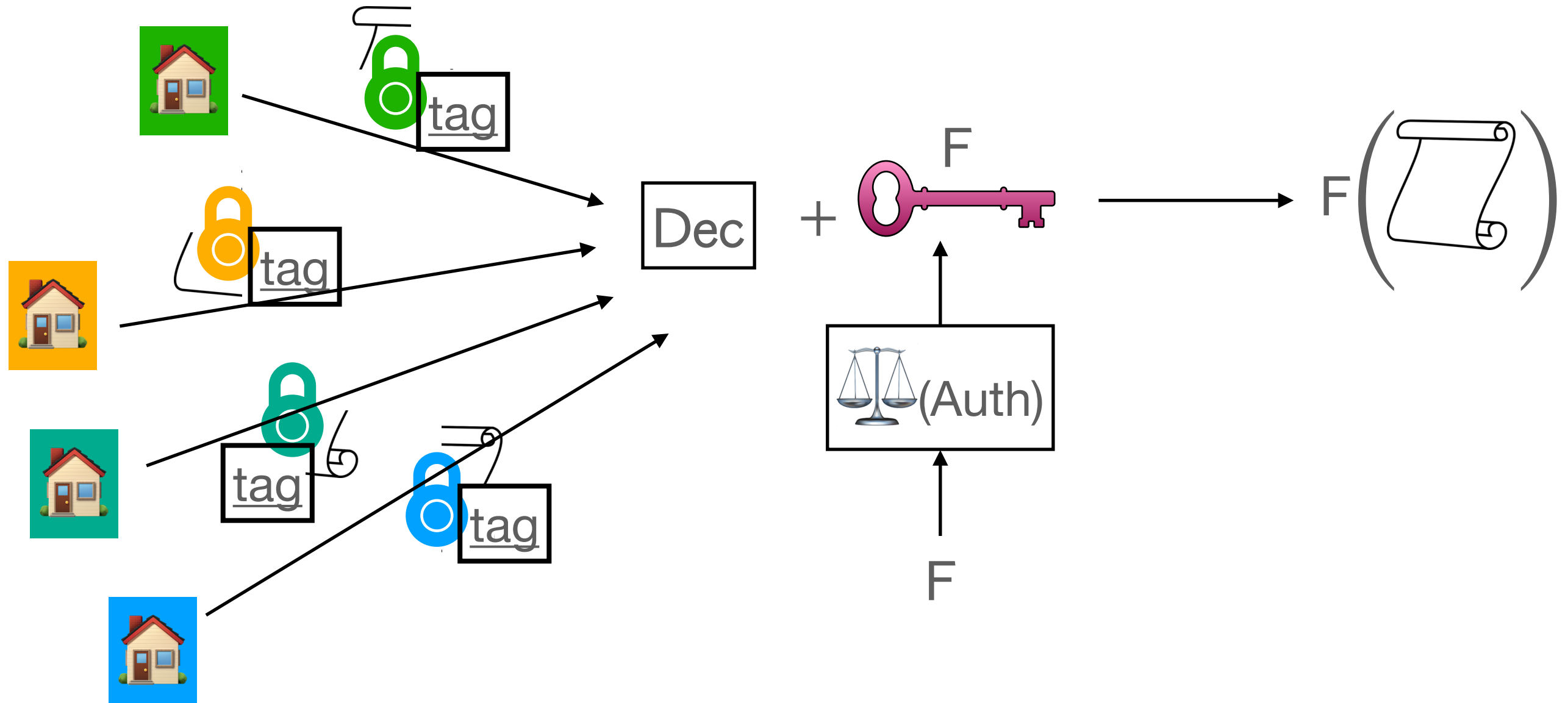
Next in the Story

MCFE [GGG+14,CDGPP18] - How to Encrypt?



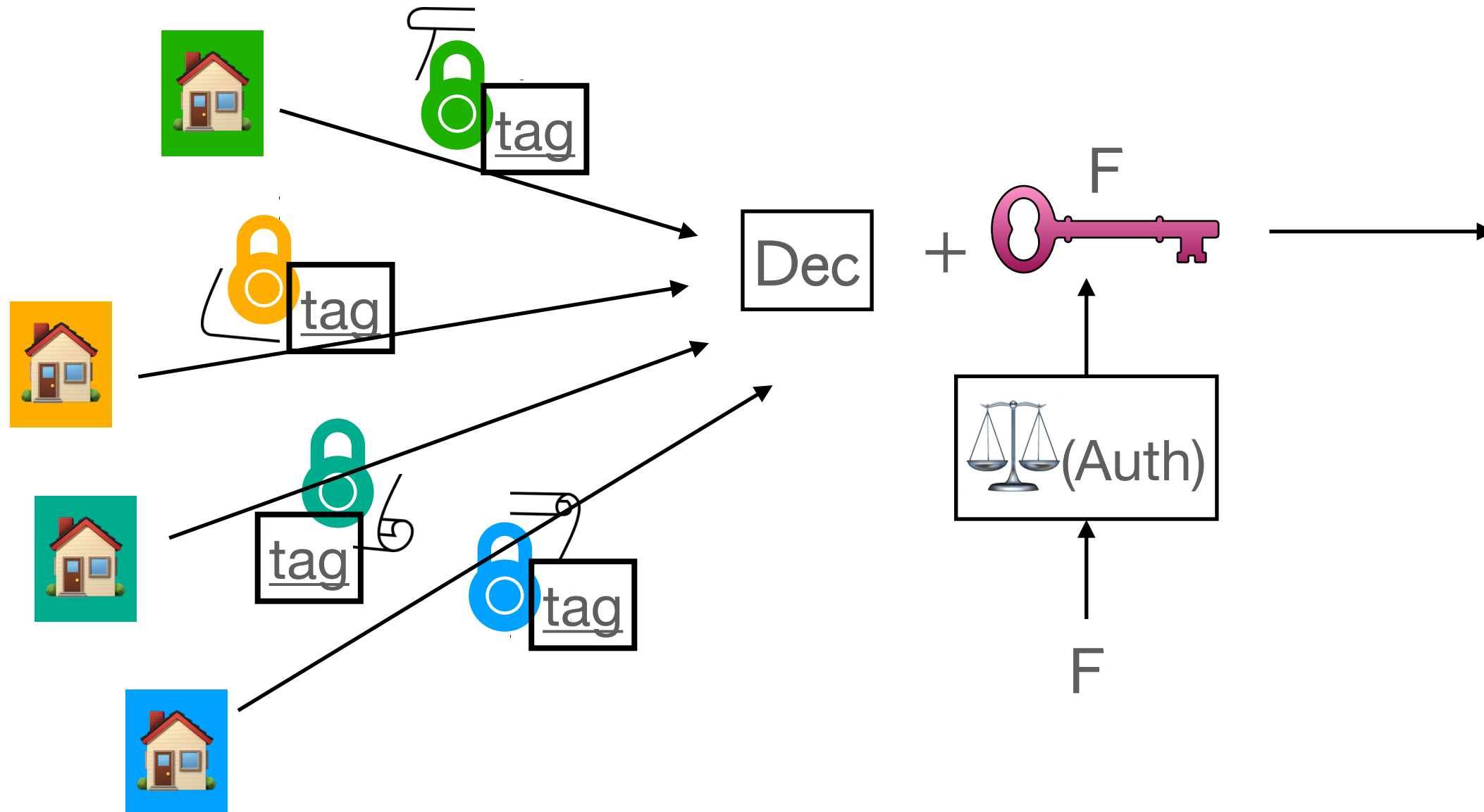
Next in the Story

MCFE [GGG+14,CDGPP18] - How to Encrypt?



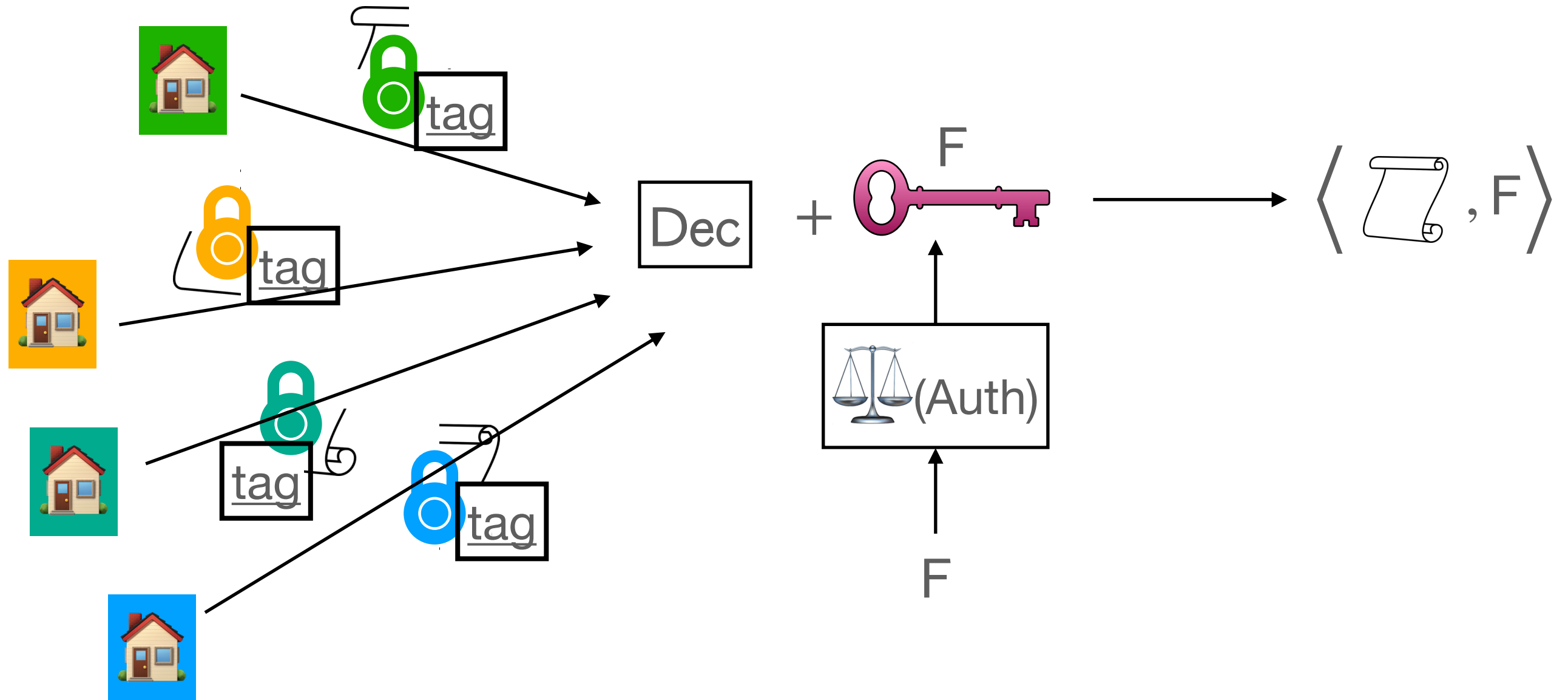
Next in the Story

MCFE [GGG+14,CDGPP18] - How to Encrypt?



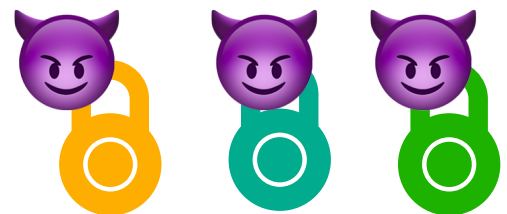
Next in the Story

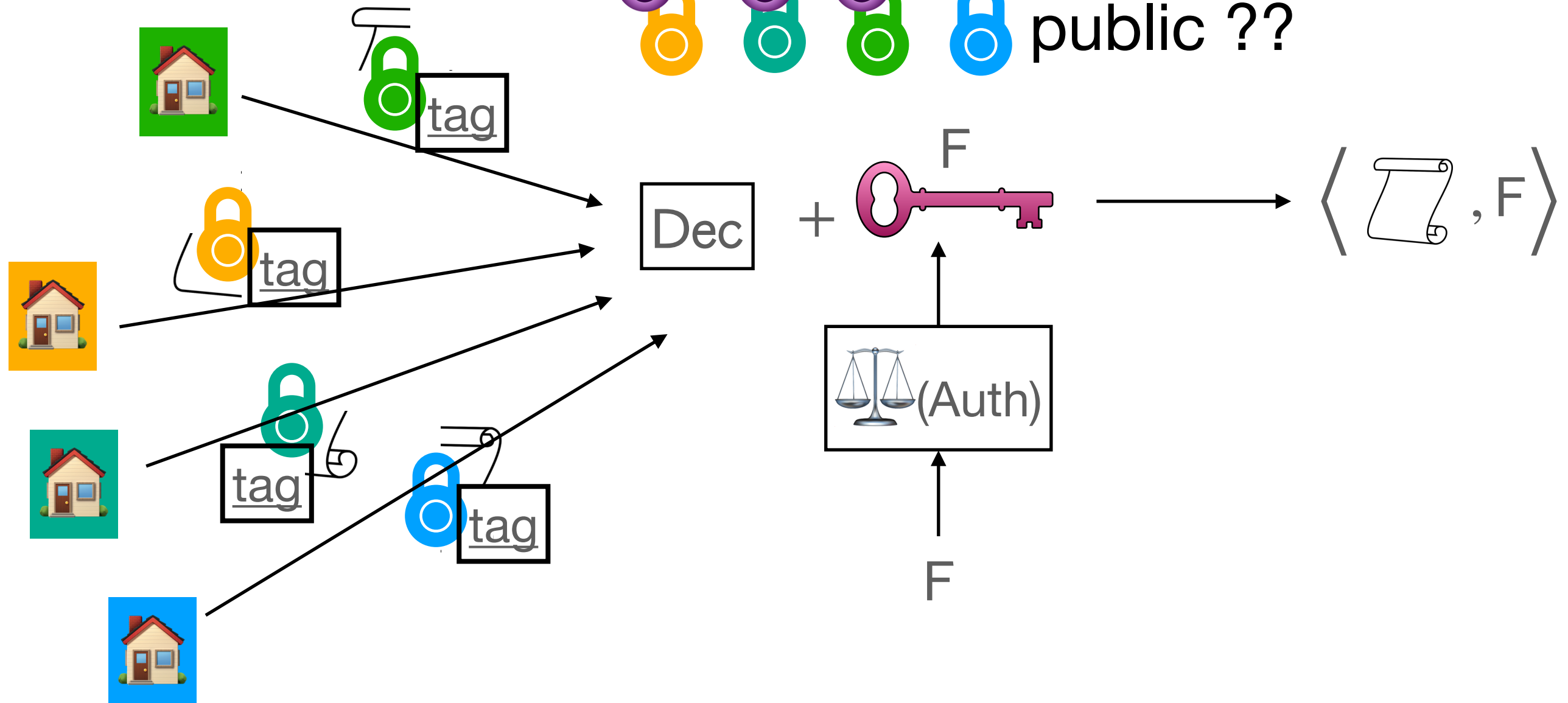
MCFE [GGG+14,CDGPP18] - How to Encrypt?



Next in the Story

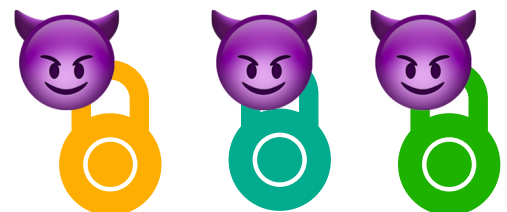
MCFE [GGG+14,CDGPP18] - How to Encrypt?

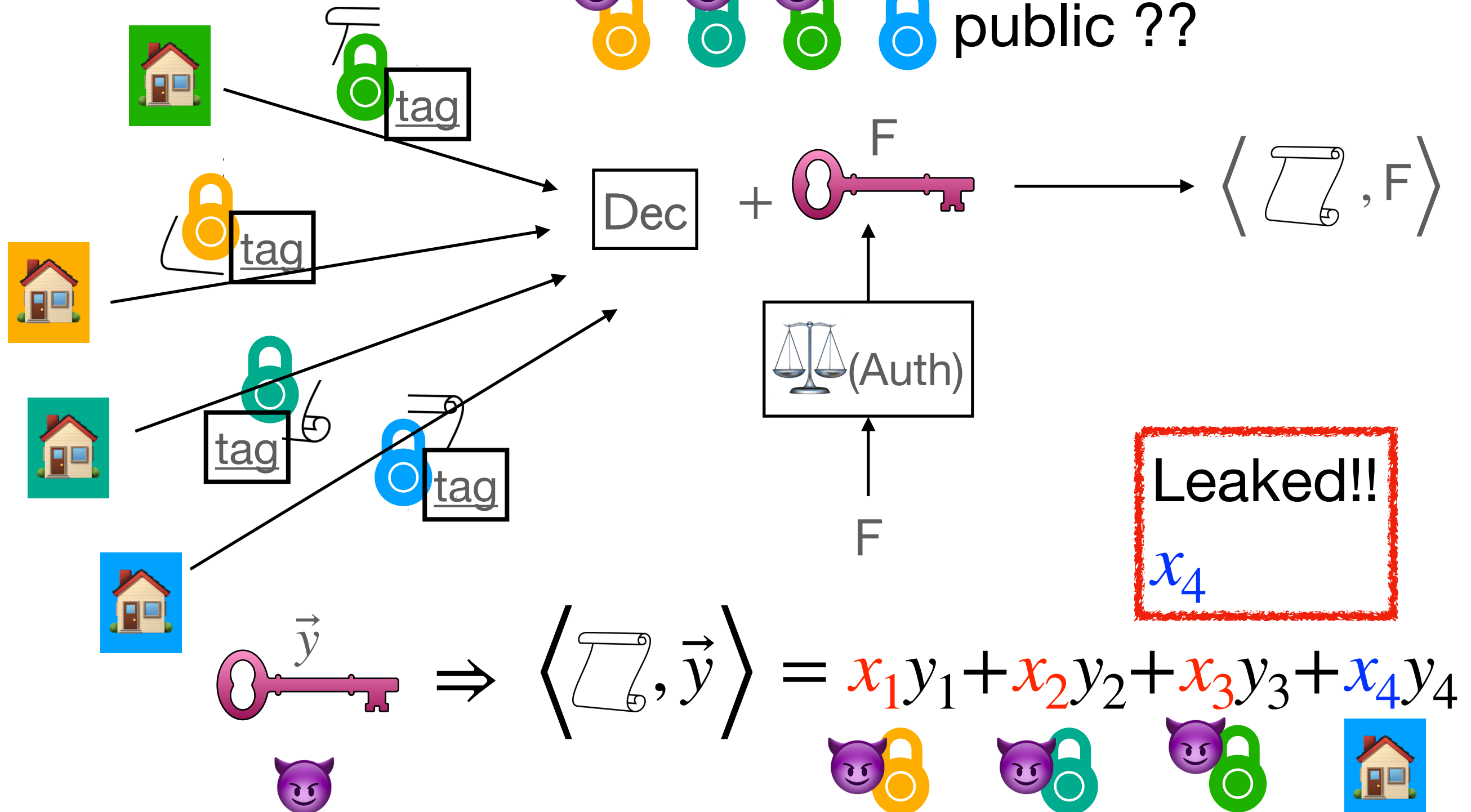
 public ??



Next in the Story

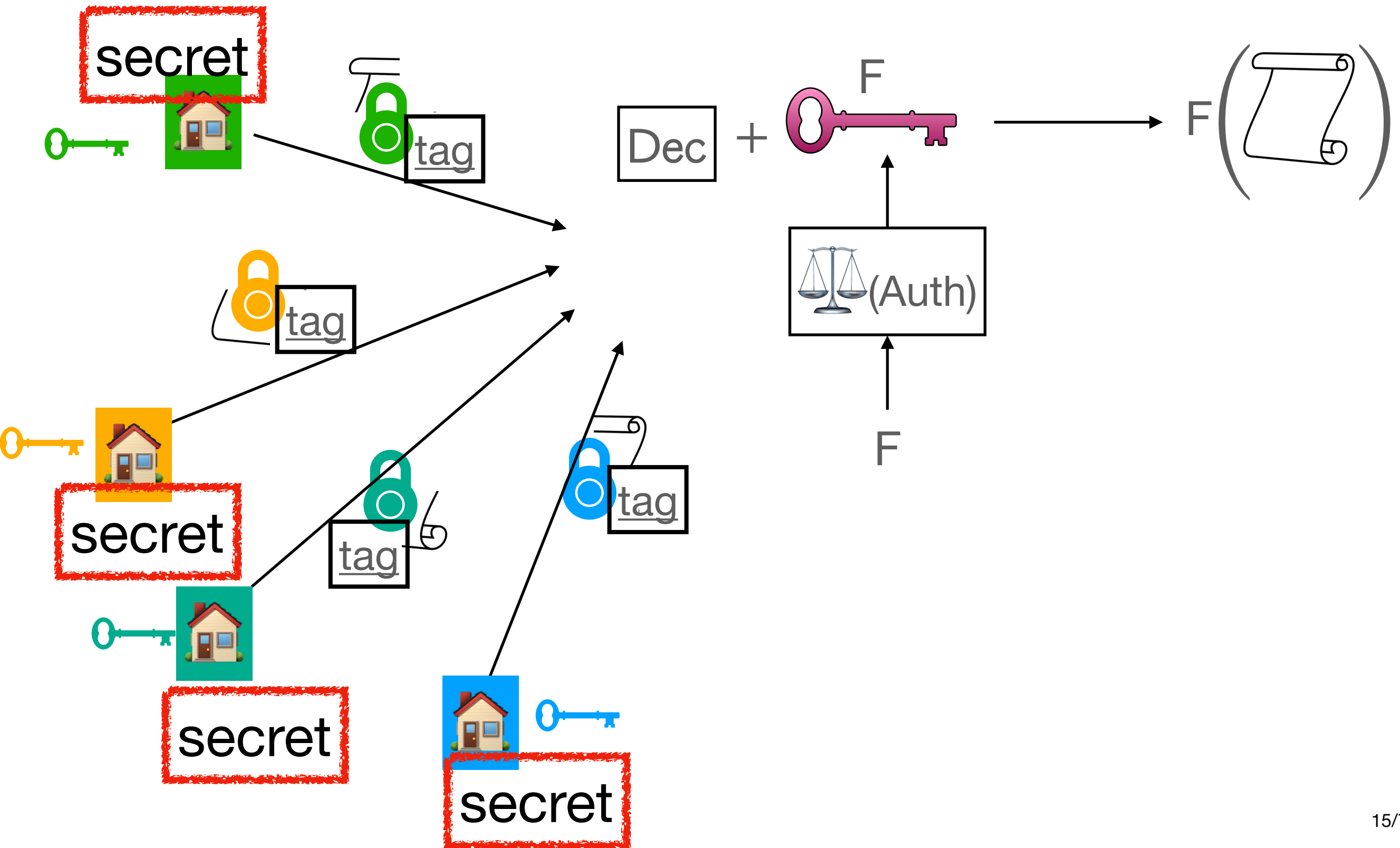
MCFE [GGG+14,CDGPP18] - How to Encrypt?

 public ??



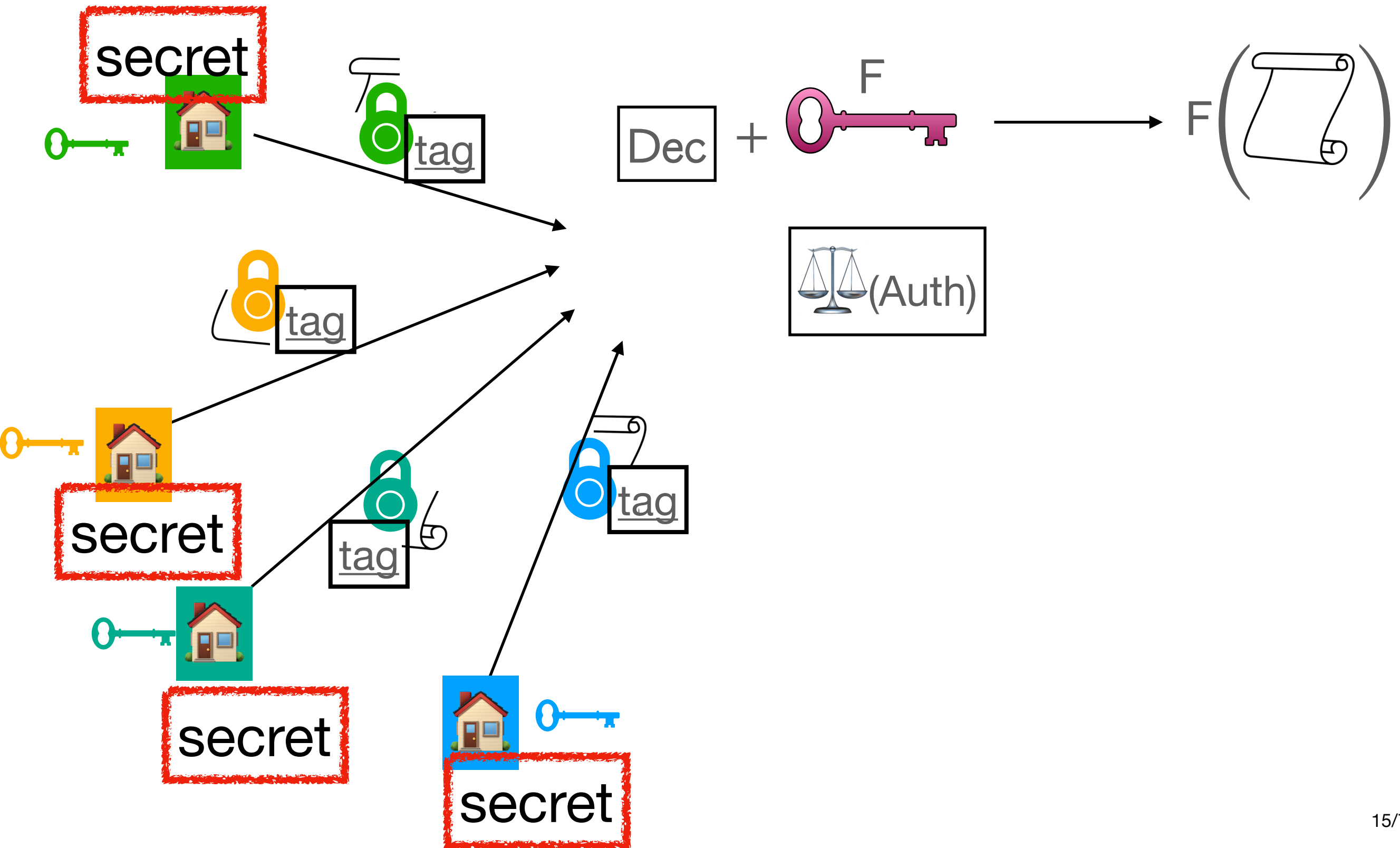
Our Story - Changes in Model

MCFE [GGG+14,CDGPP18] - **Secret** Encryption Keys



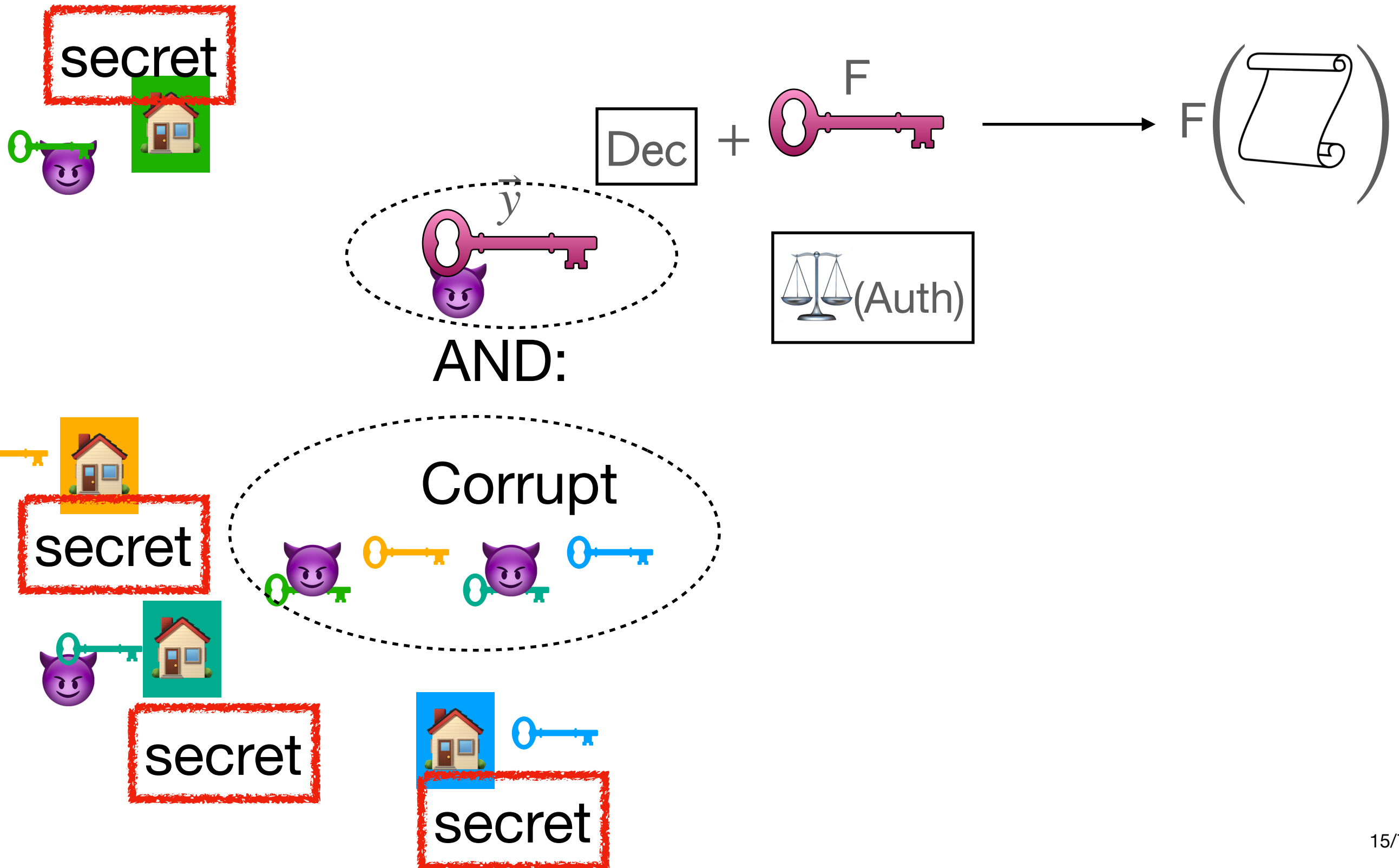
Our Story - Changes in Model

MCFE [GGG+14,CDGPP18] - **Secret** Encryption Keys



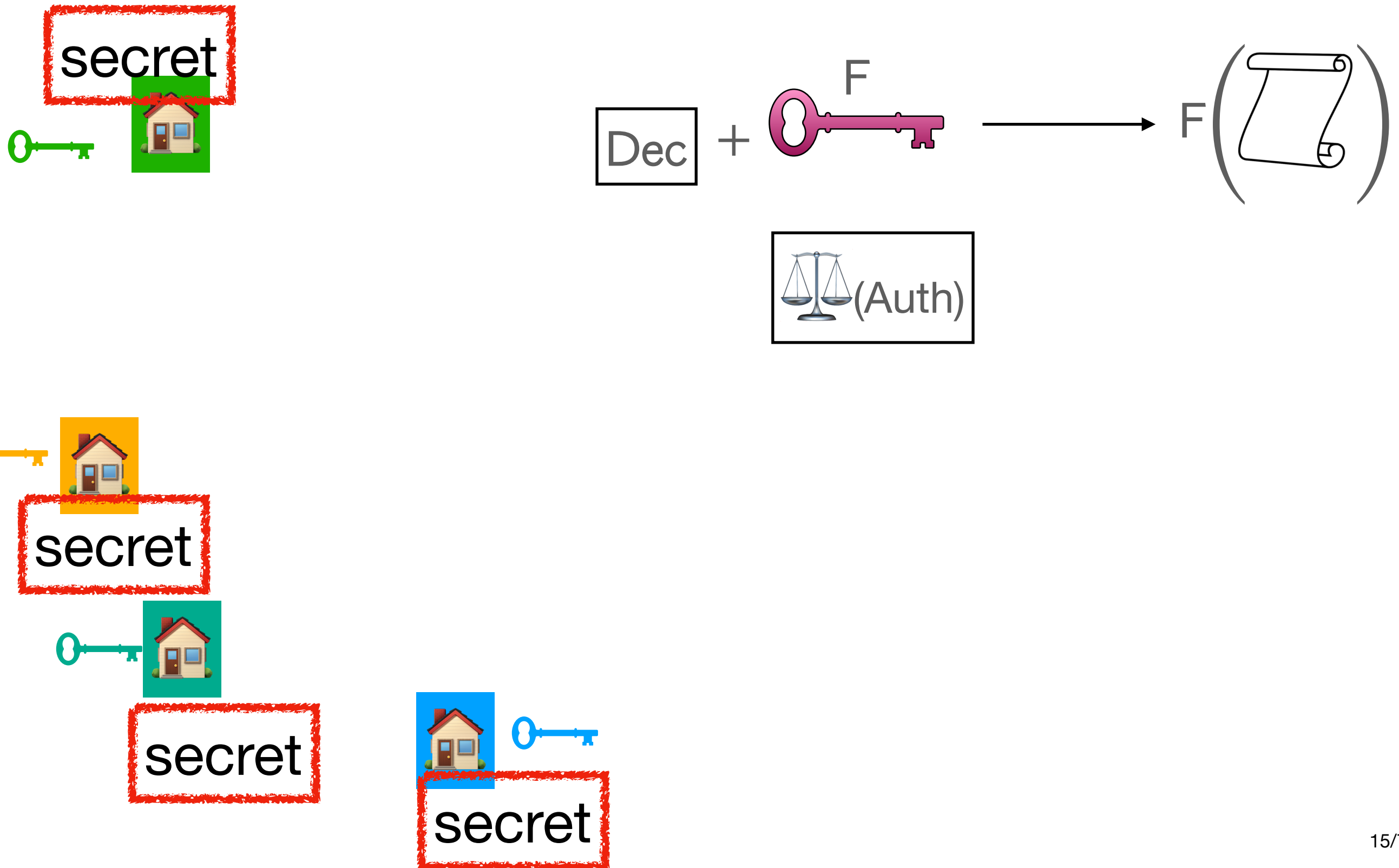
Our Story - Changes in Model

MCFE [GGG+14,CDGPP18] - **Secret** Encryption Keys



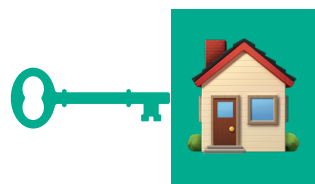
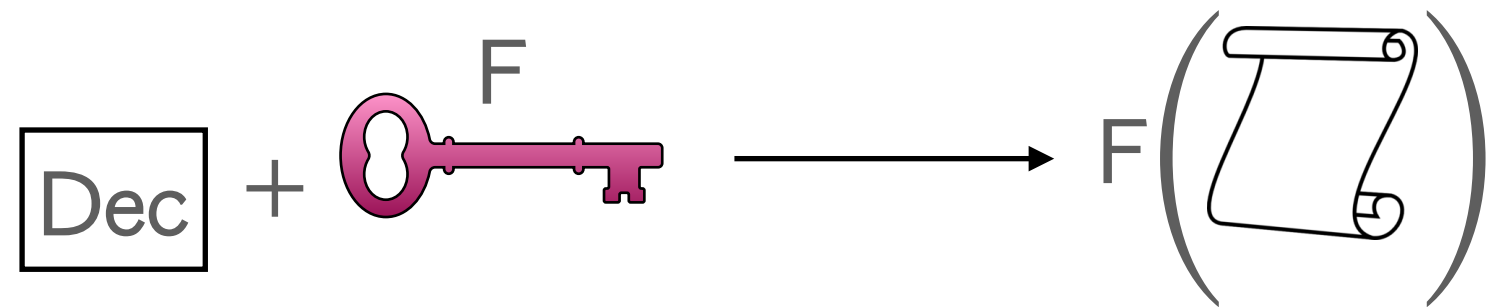
Our Story - Changes in Model

MCFE [GGG+14,CDGPP18] - **Secret** Encryption Keys



Our Story - Changes in Model

MCFE [GGG+14,CDGPP18] - **Secret** Encryption Keys



[GGG+14,CDGPP18]

“vanilla” syntax

$\underline{F}(\text{scroll})$:

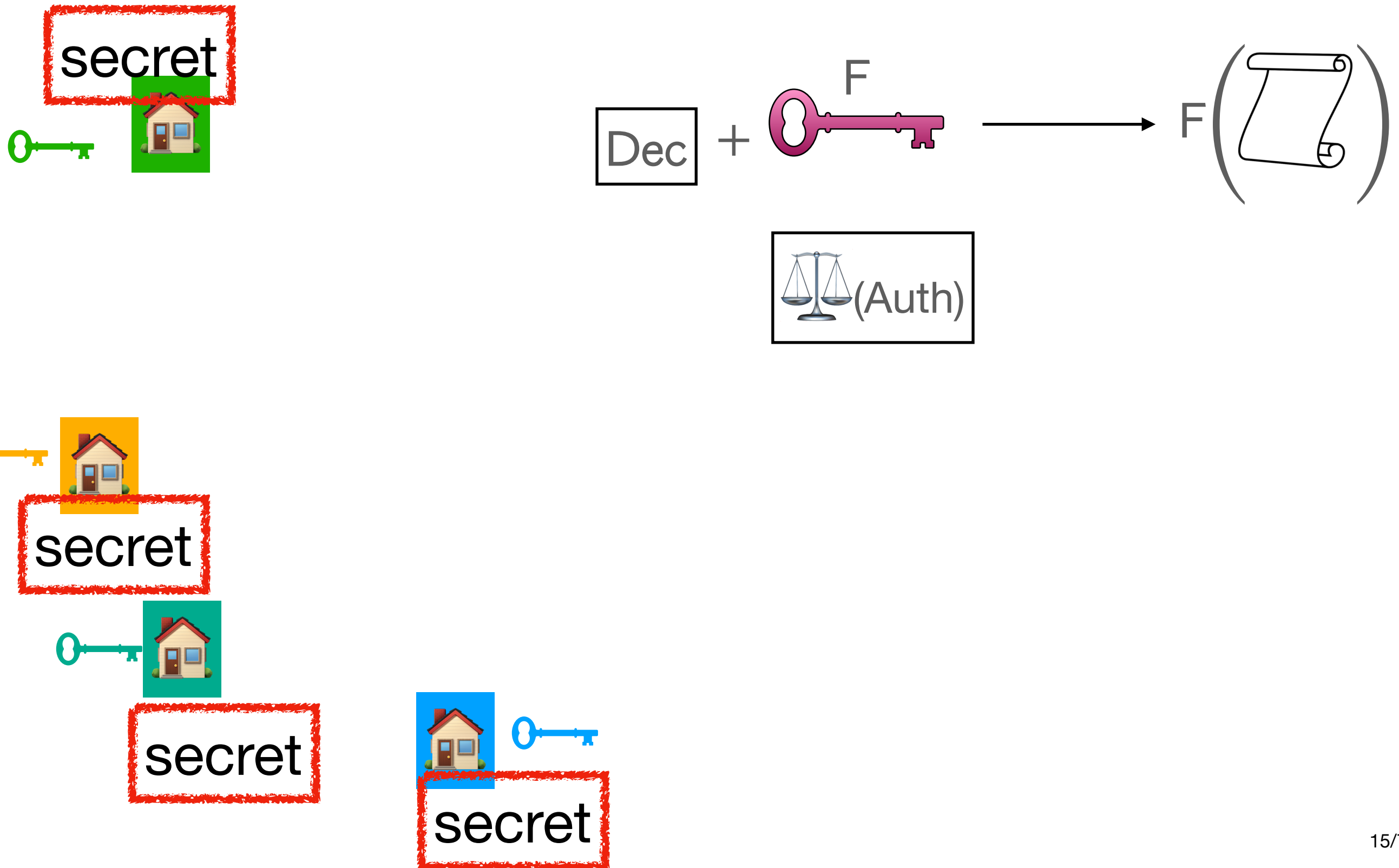
Parse **Att**,  =

If **Pol(Att) = True** then 

Else “Nothing”

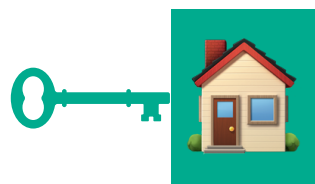
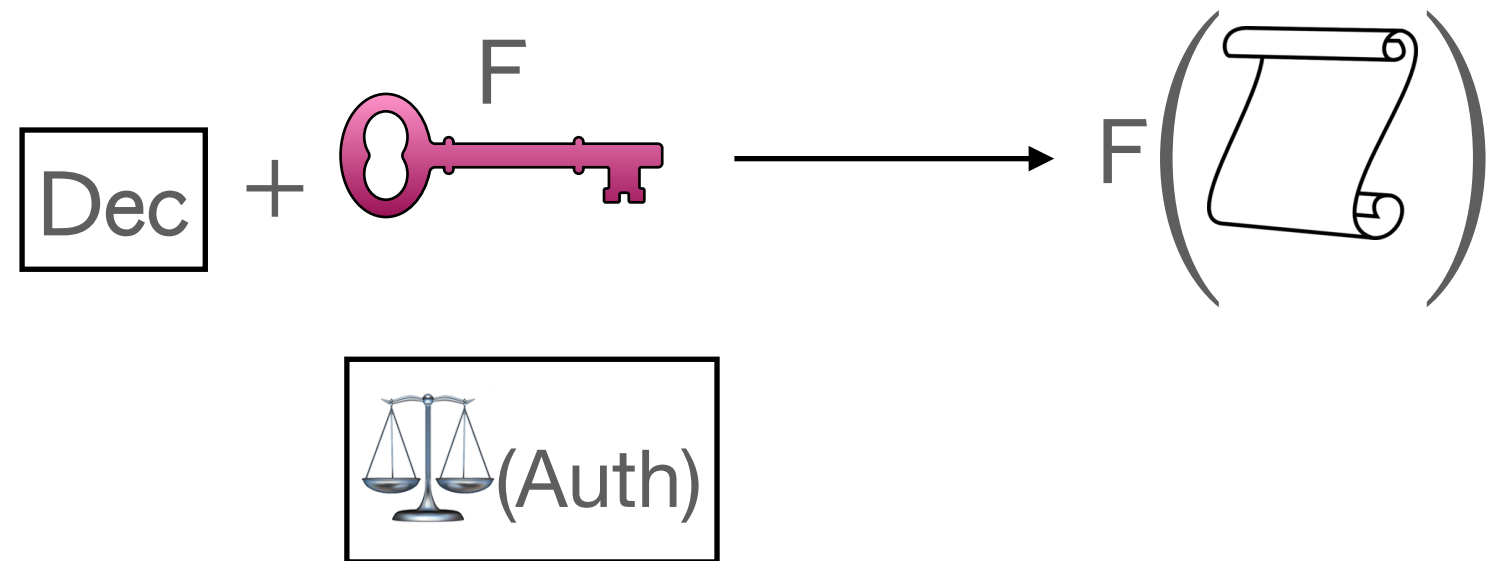
Our Story - Changes in Model

MCFE [GGG+14,CDGPP18] - **Secret** Encryption Keys



Our Story - Changes in Model

MCFE [GGG+14,CDGPP18] - **Secret** Encryption Keys



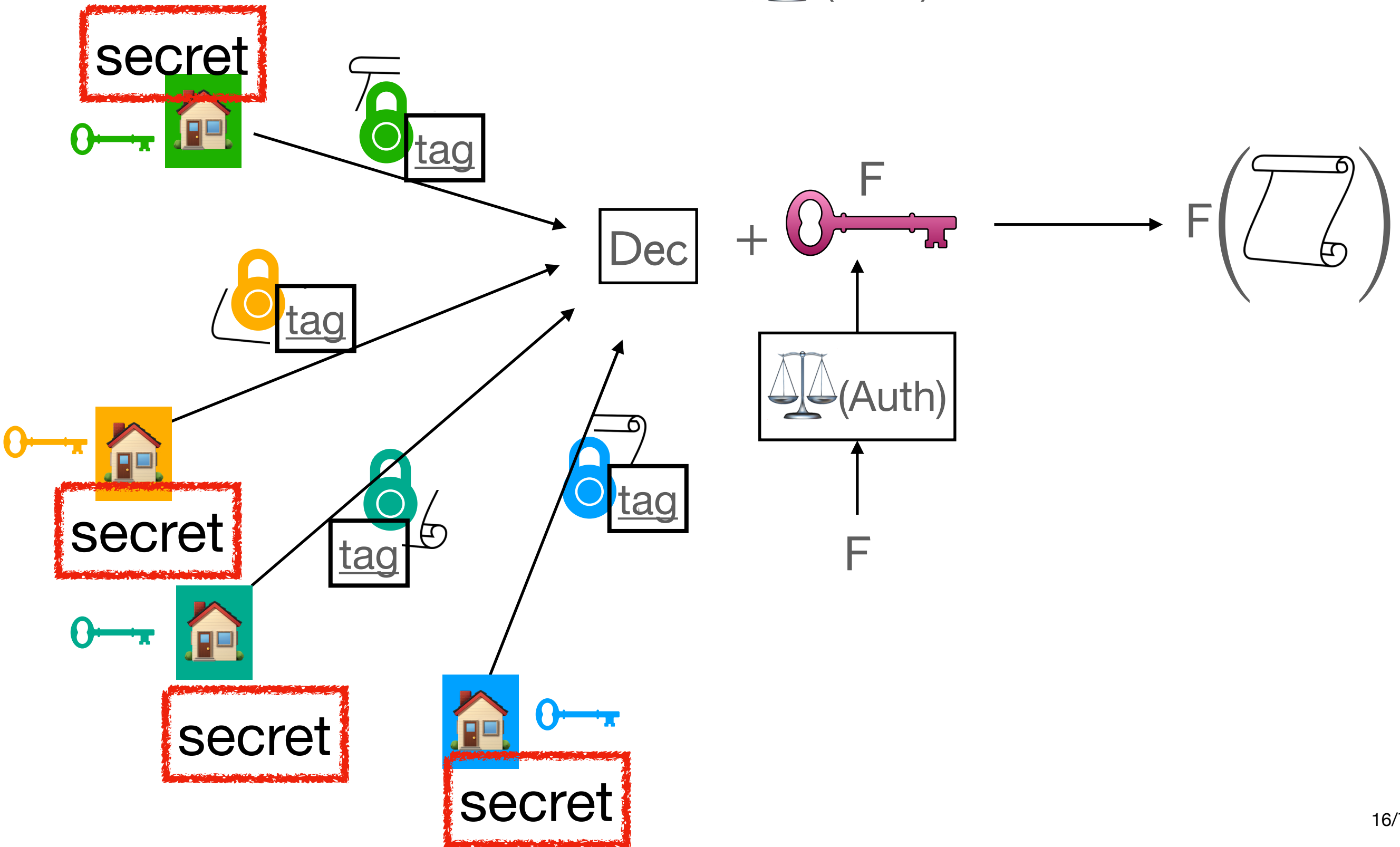
🤔 Similar

Syntactical Problem:

NOT allow public-**Att**

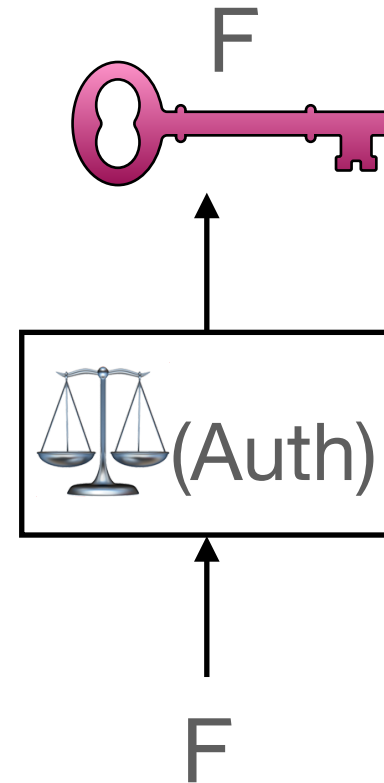
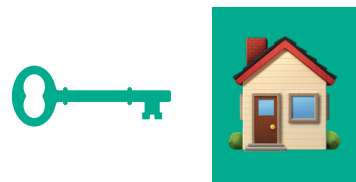
Next in Our Story

Trust model - The Role of  (Auth)



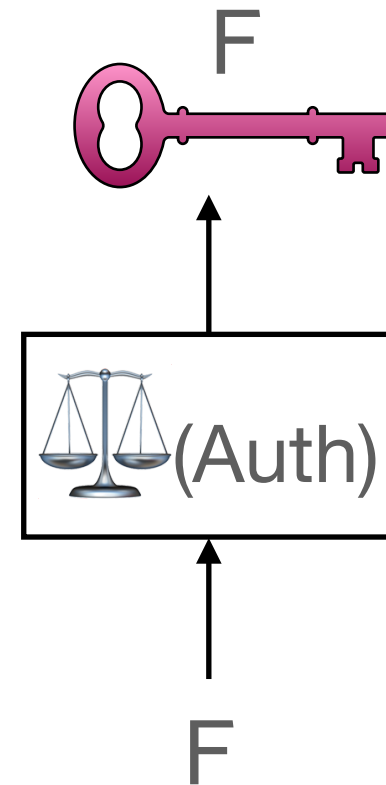
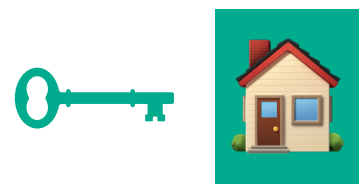
Next in Our Story

Trust model - The Role of



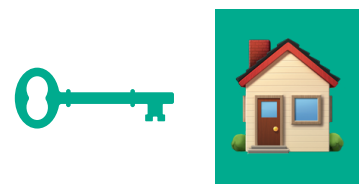
Next in Our Story

Trust model - The Role of

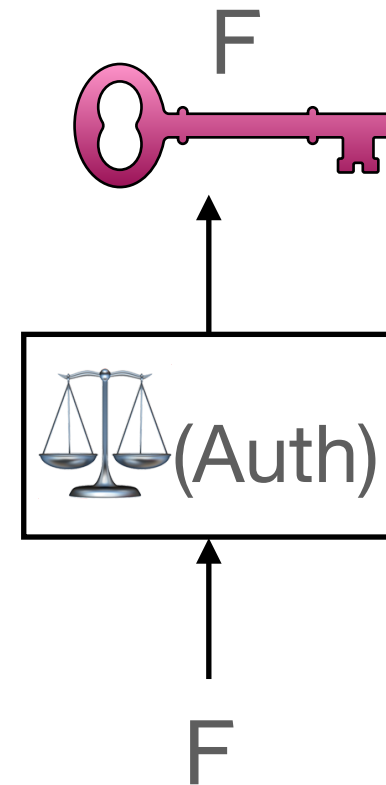


Next in Our Story

Trust model - The Role of

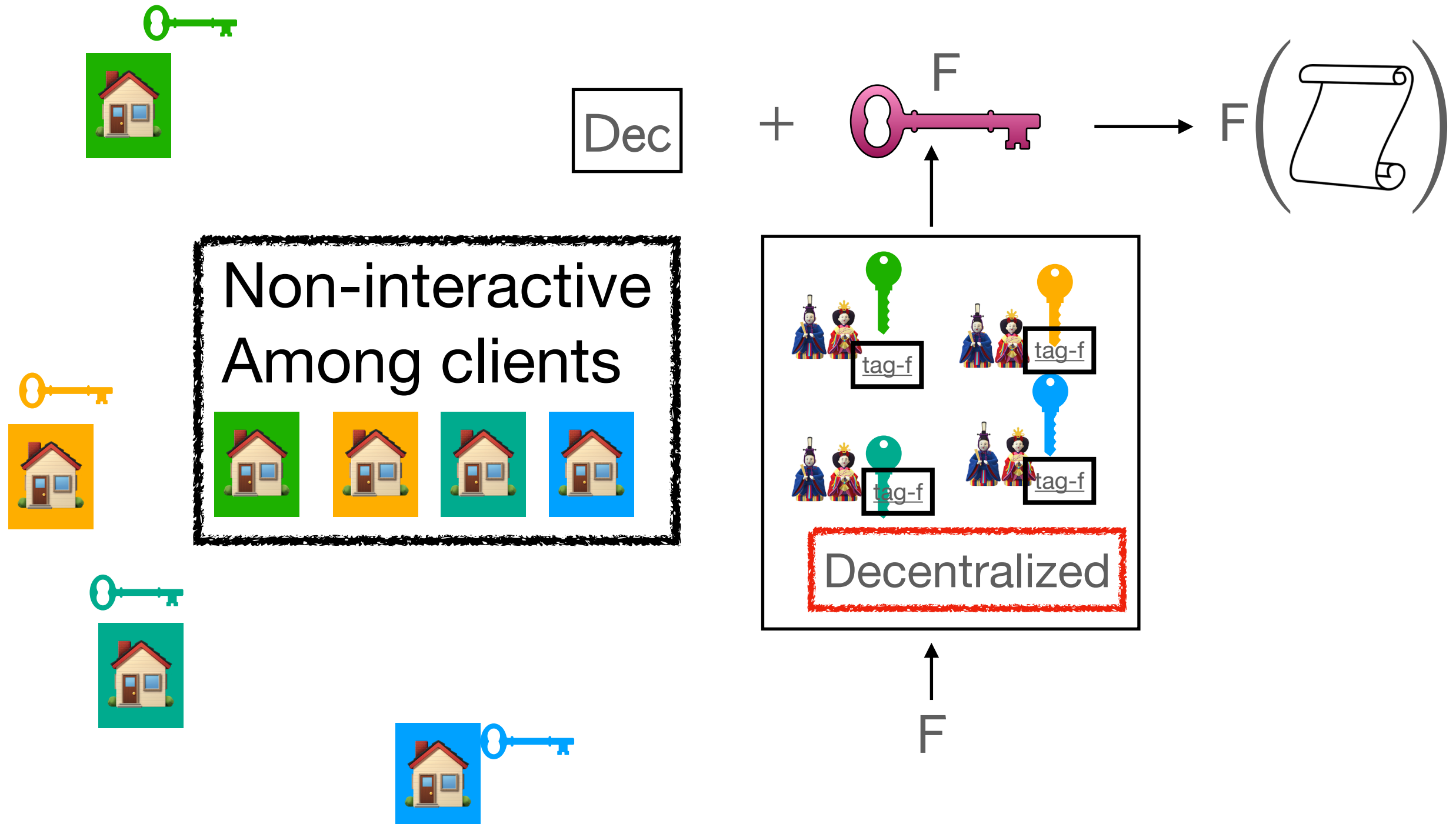


Non-interactive
Among clients

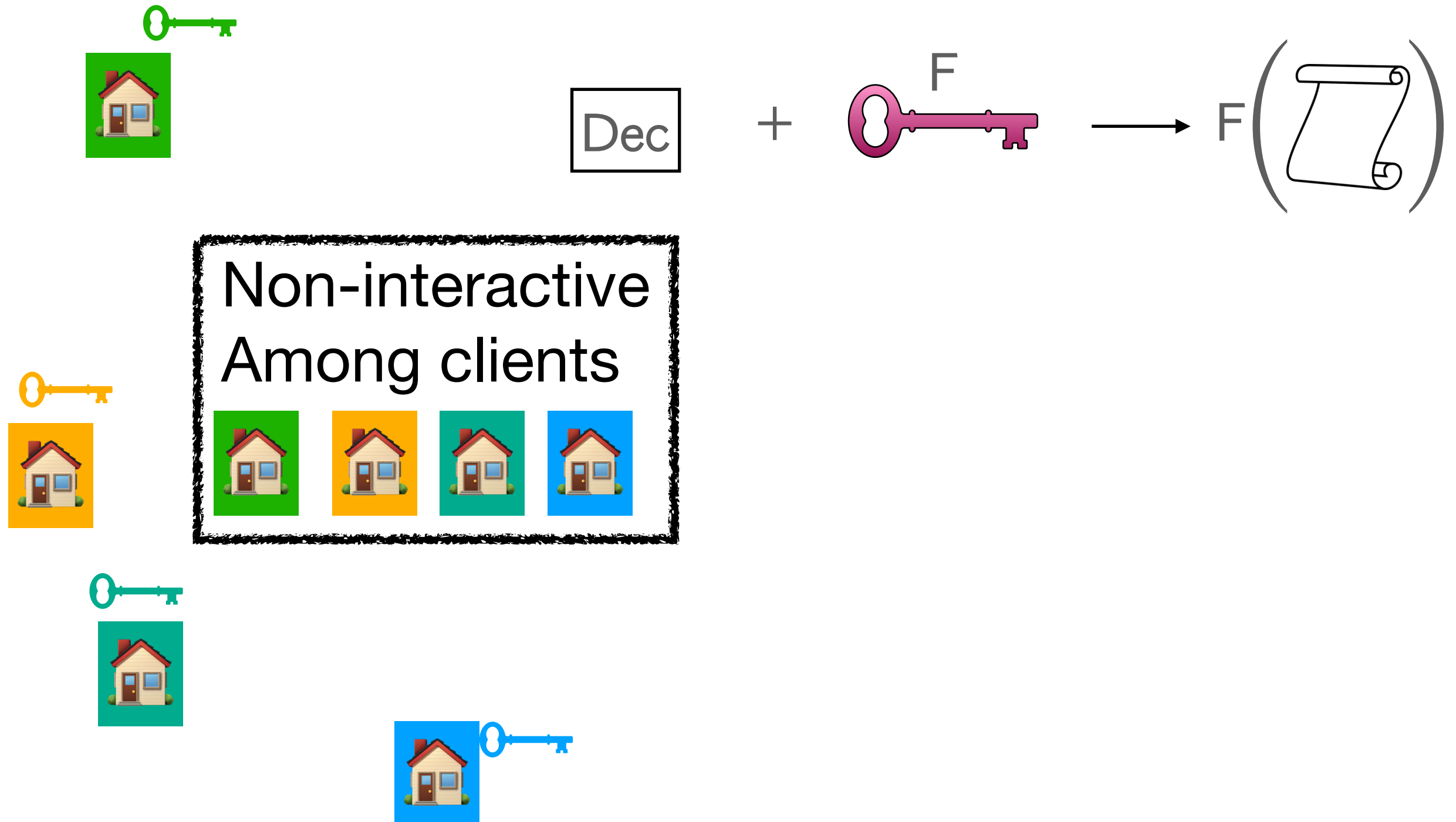


Clients do
NOT want to
trust  (Auth) ??

Decentralized Multi-Client Functional Encryption (DMCFE) [CDGPP'18] - Decentralizing KeyGen

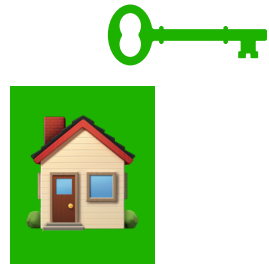


Decentralized Multi-Client Functional Encryption (DMCFE) [CDGPP'18] - Decentralizing KeyGen



Decentralized Multi-Client Functional Encryption (DMCFE)

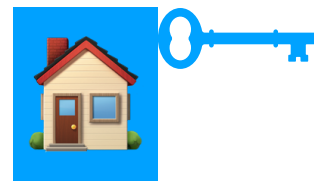
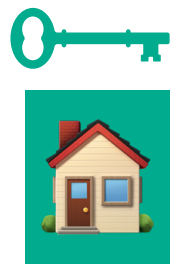
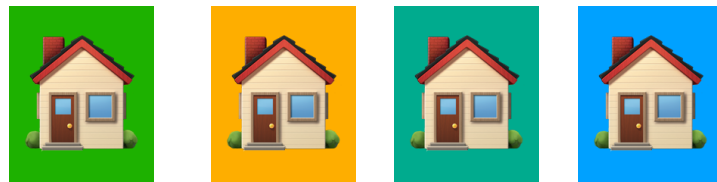
[CDGPP'18] - Decentralizing KeyGen



Dec

$$+ \text{key}^F \longrightarrow F\left(\text{document}\right)$$

Non-interactive
Among clients



[CDGPP'18]

DMCFE computing

inner-product: $\langle \text{document}, F \rangle$

—> Follow-ups:

[CDG+18],[ACF18],

[ABKW19],[ABG19],[LT19],

[CDG+20],[ACGU20],

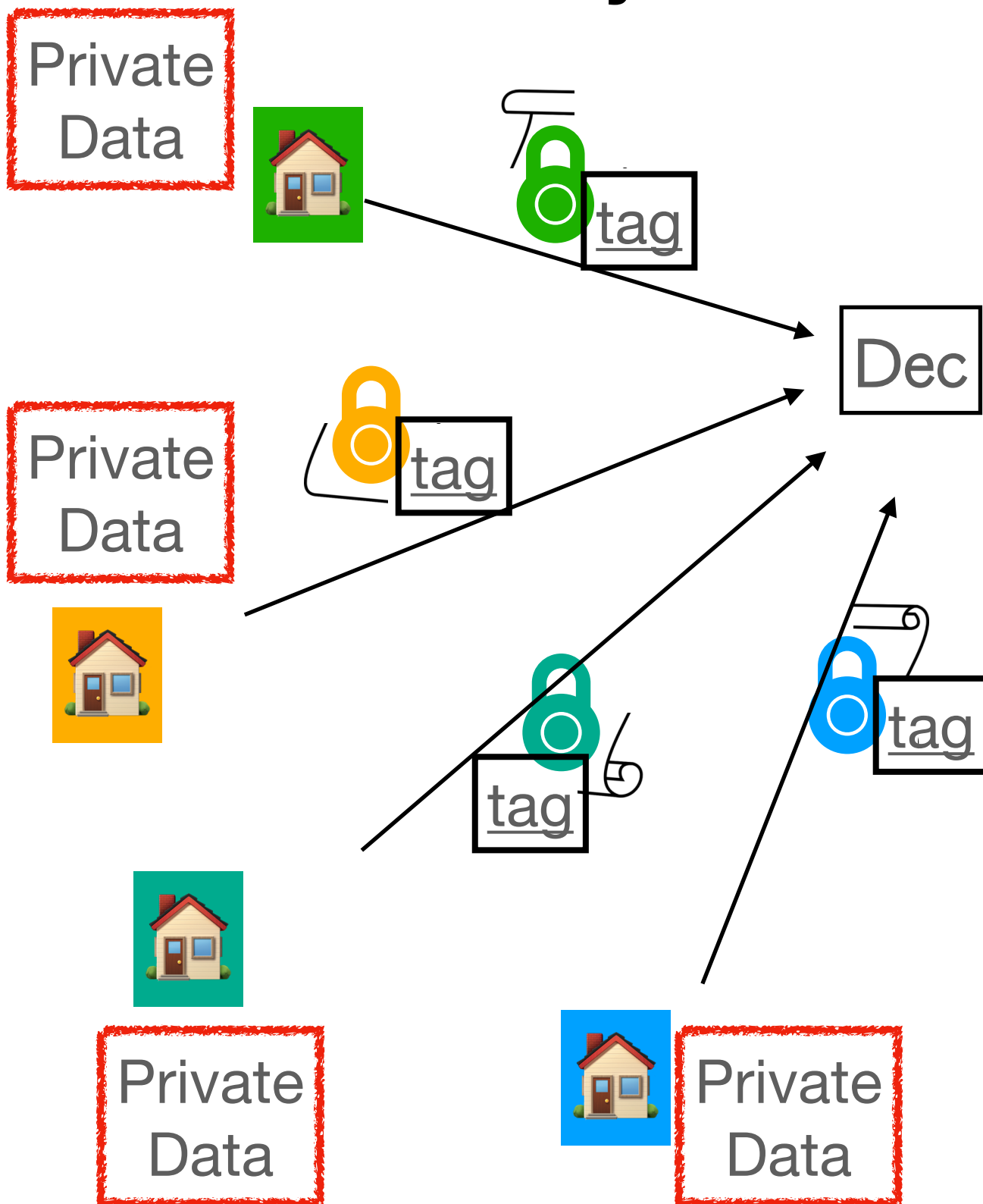
[AGT21],

[SV23],[NPP23],

[NPS24], and still counting..

Use-case

Financial Analysis



Use-case

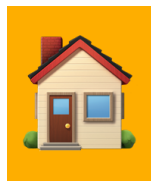
Financial Analysis

Private
Data

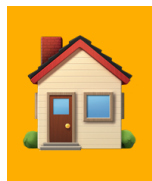


Dec

Private
Data



Protect individual
Private data



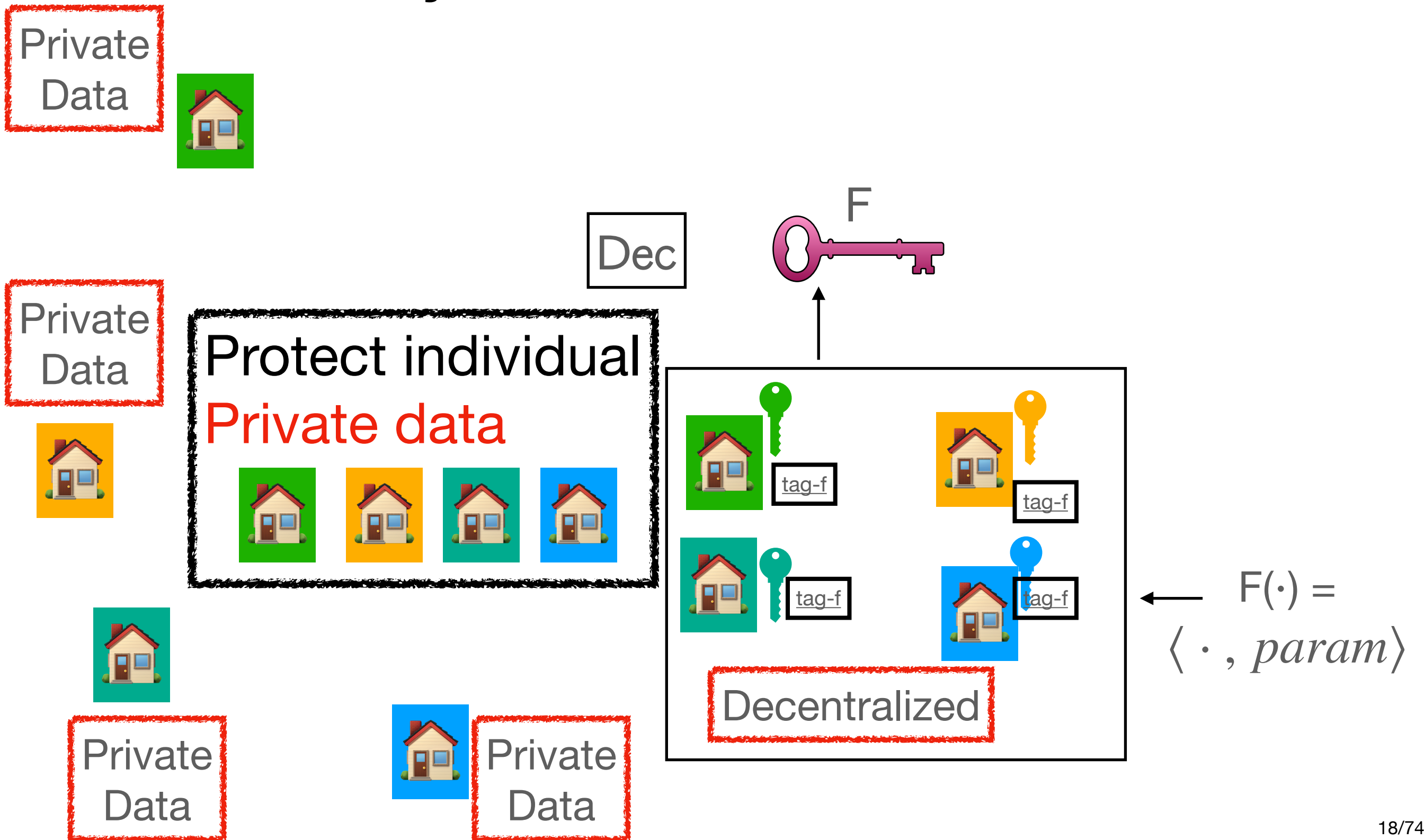
Private
Data



Private
Data

Use-case

Financial Analysis



Use-case

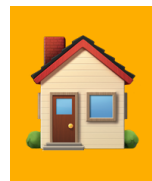
Financial Analysis

Private
Data

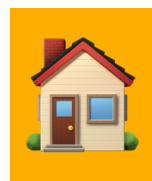
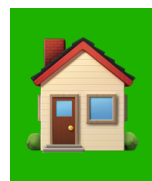


Dec

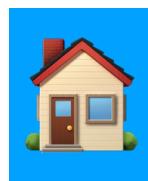
Private
Data



Protect individual
Private data

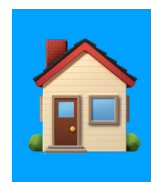
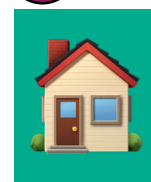
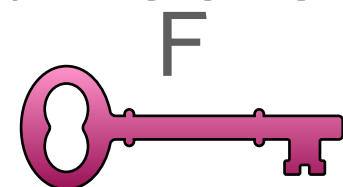


Private
Data



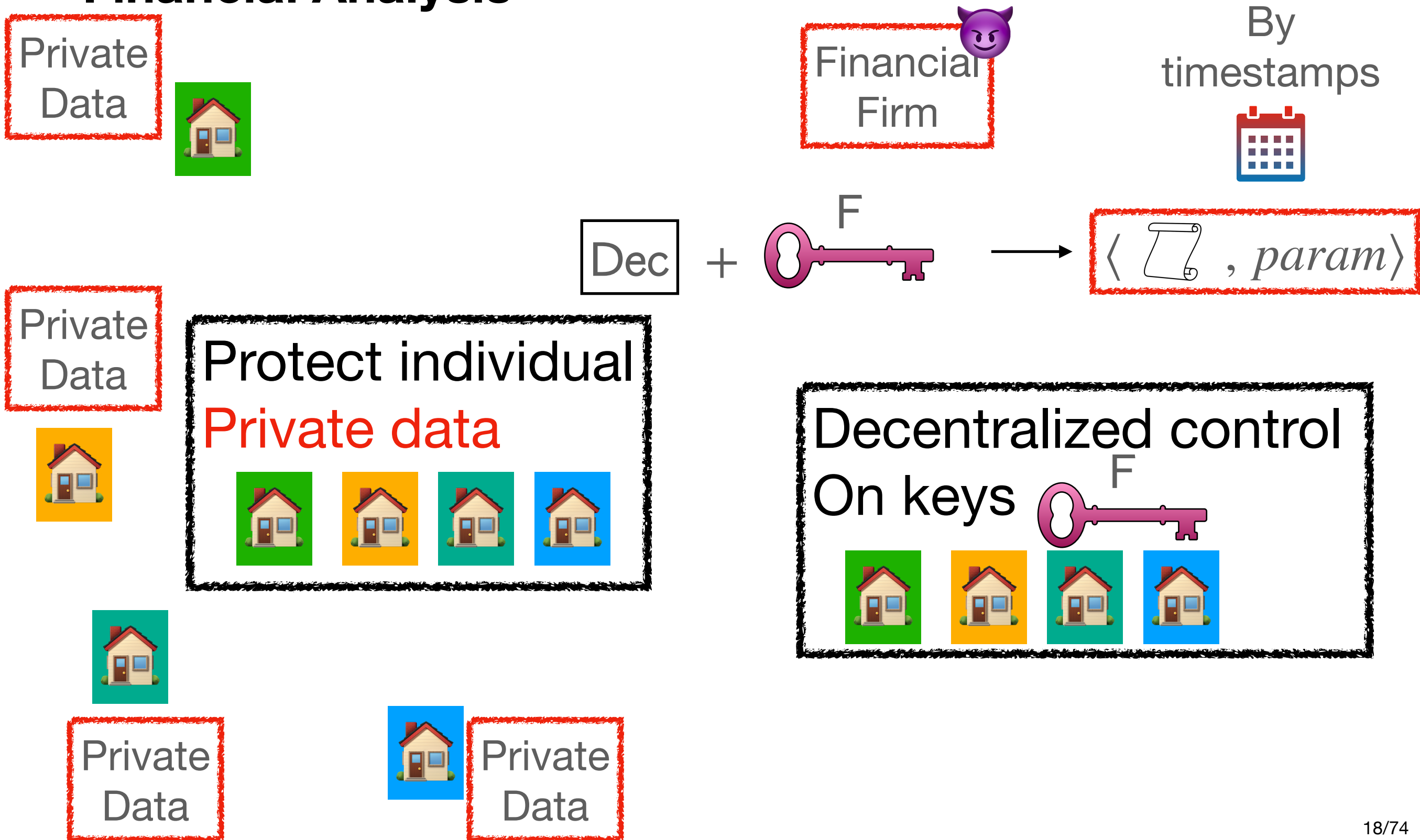
Private
Data

Decentralized control
On keys ^F



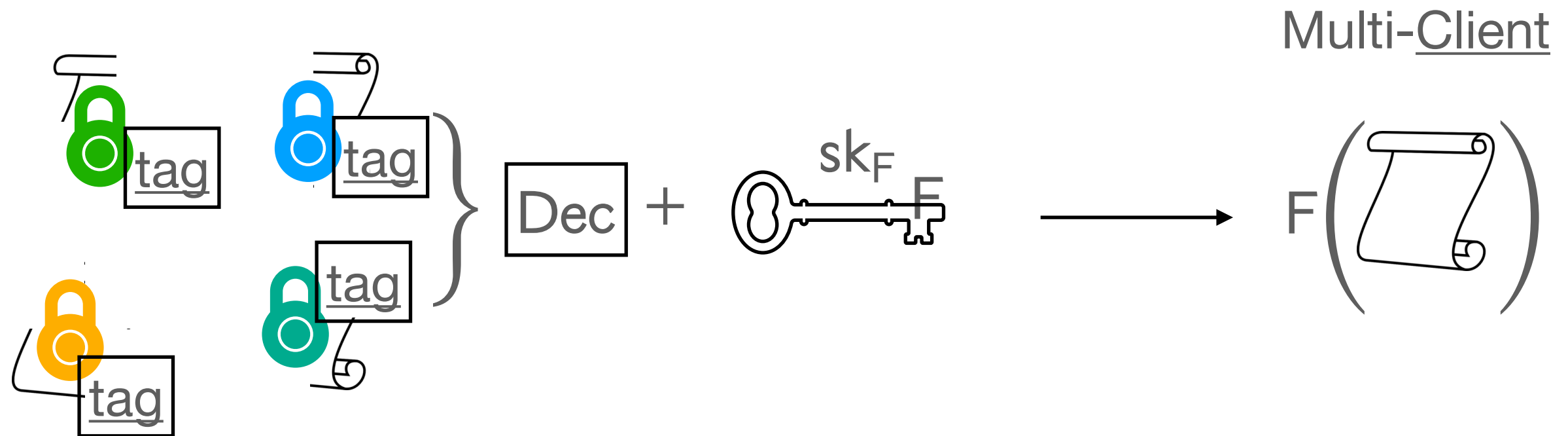
Use-case

Financial Analysis



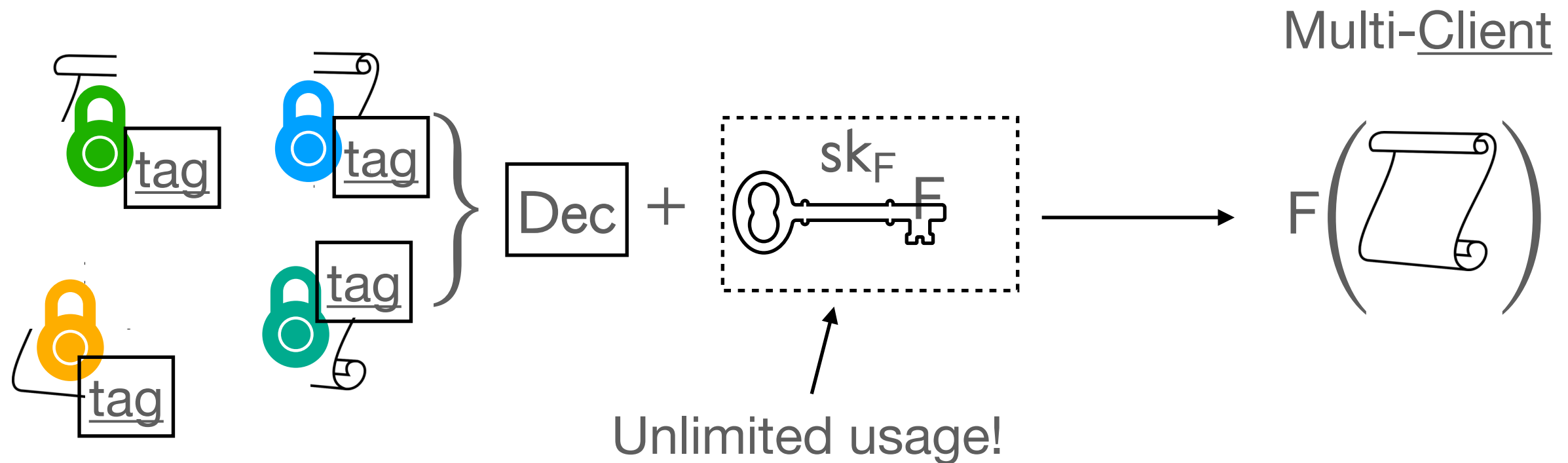
Motivation - Usage of Keys

MCFE [GGG+14, GKL+13, CDGPP18]



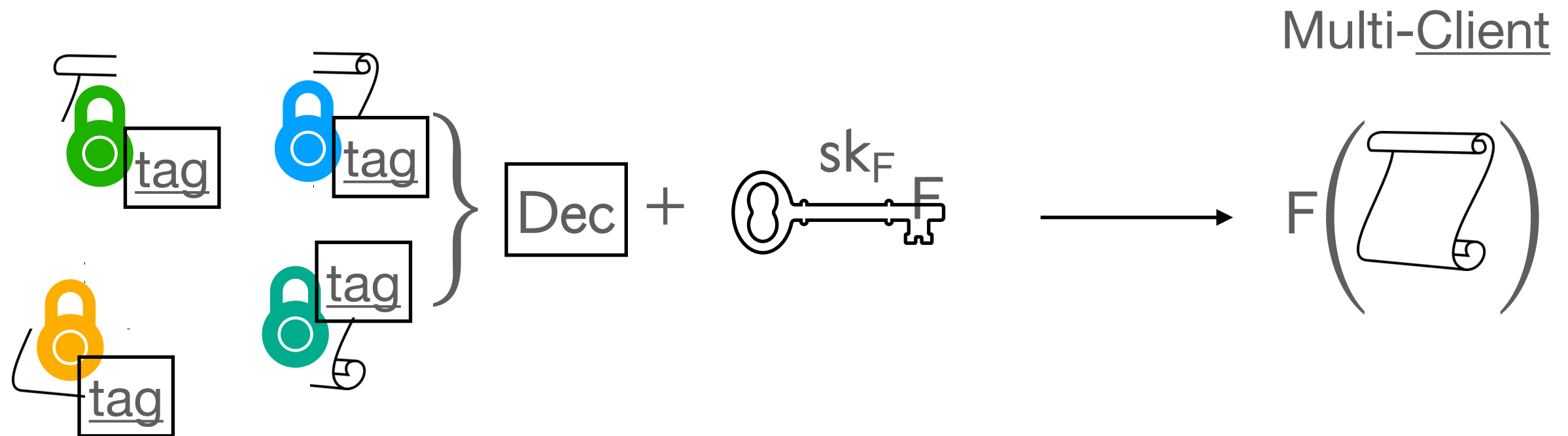
Motivation - Usage of Keys

MCFE [GGG+14, GKL+13, CDGPP18]



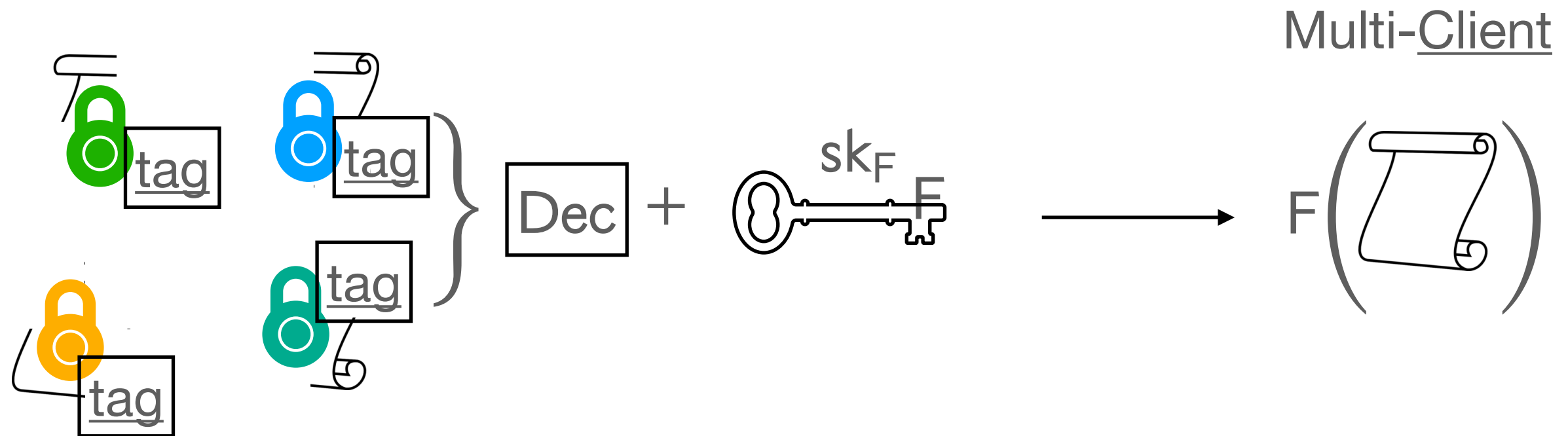
Motivation - Usage of Keys

MCFE [GGG+14, GKL+13, CDGPP18]



Motivation - Usage of Keys

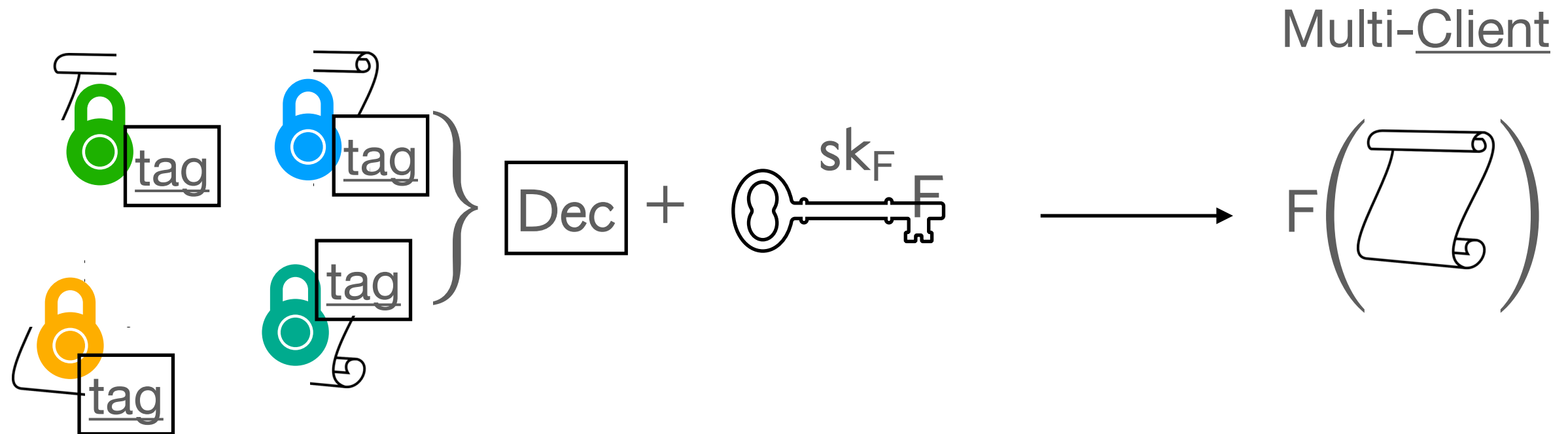
MCFE [GGG+14, GKL+13, CDGPP18]



[ACGU20]: control keys
In Multi-Input FE [GGG+14] { **Secret** Encryption Keys
Only **one** encryptor 


Motivation - Usage of Keys

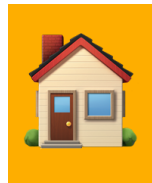
MCFE [GGG+14, GKL+13, CDGPP18]



 **Efficiency Problem**

Research Question (1)

Controlling Decryption Keys in MCFE



Heart() = 1

Heart() = 0

Spade() = 0

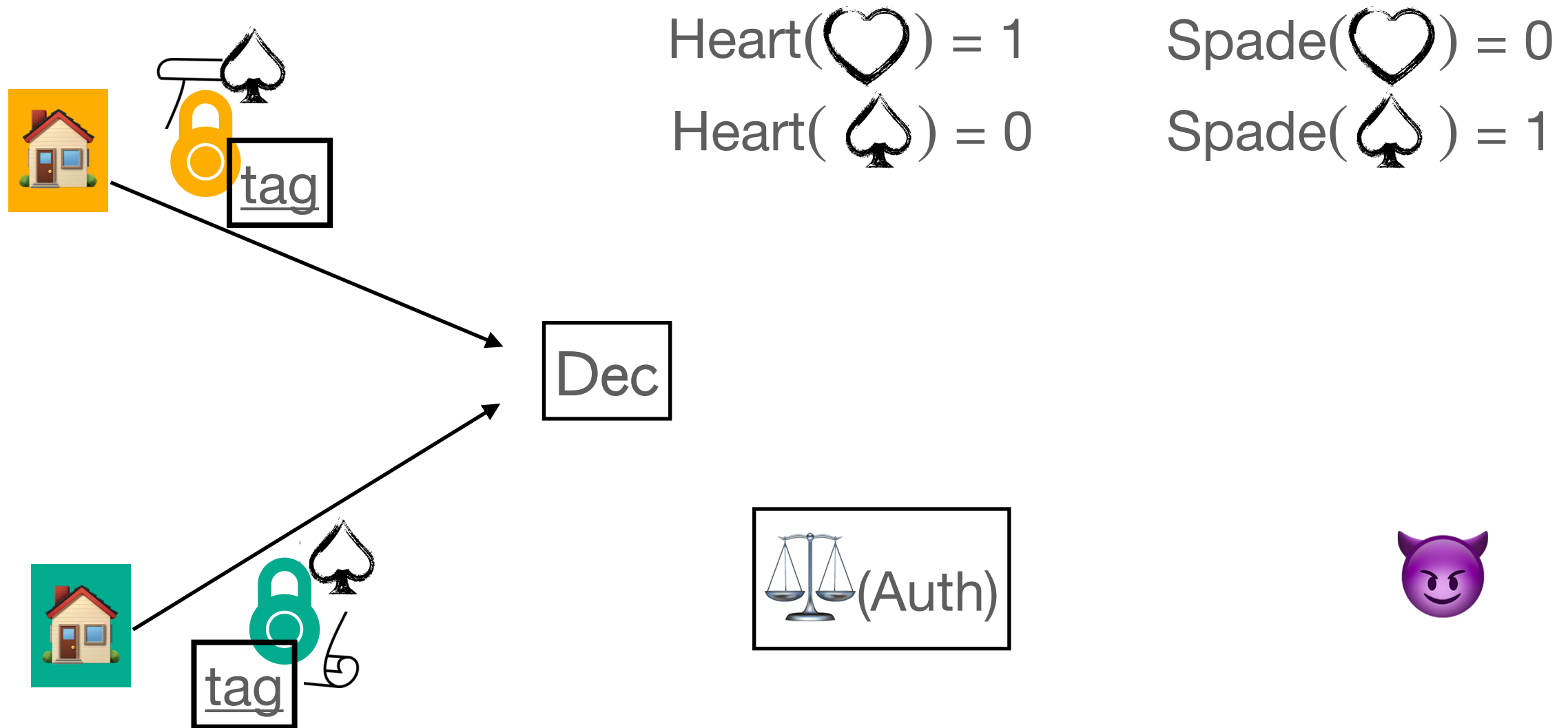
Spade() = 1

Dec



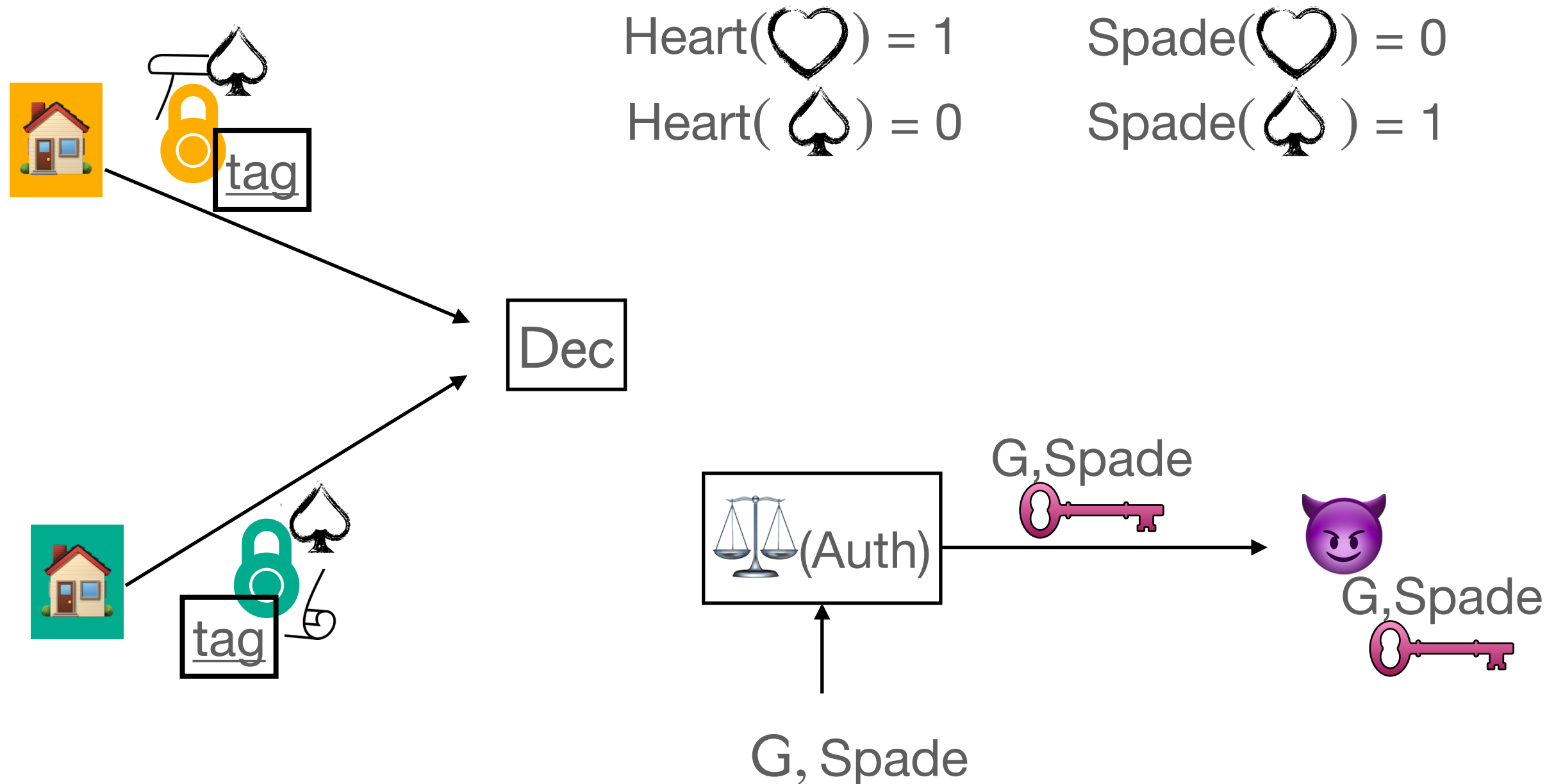
Research Question (1)

Controlling Decryption Keys in MCFE



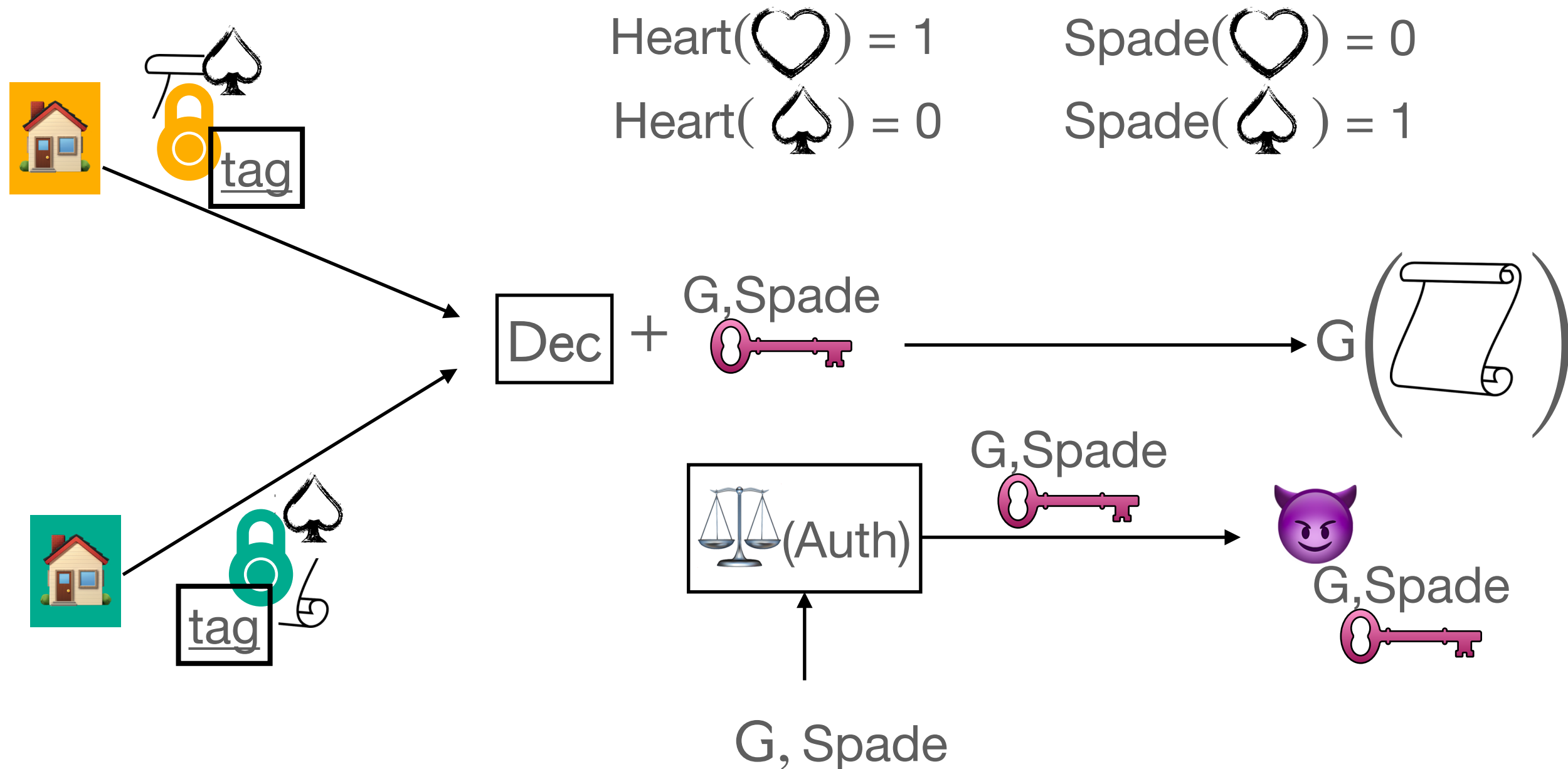
Research Question (1)

Controlling Decryption Keys in MCFE



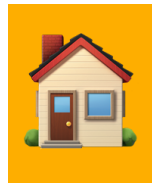
Research Question (1)

Controlling Decryption Keys in MCFE



Research Question (1)

Controlling Decryption Keys in MCFE



Heart() = 1

Heart() = 0

Spade() = 0

Spade() = 1

Dec



Research Question (1)

Controlling Decryption Keys in MCFE



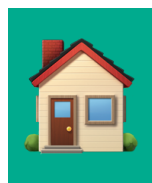
Heart(♥) = 1

Heart(♠) = 0

Spade(♥) = 0

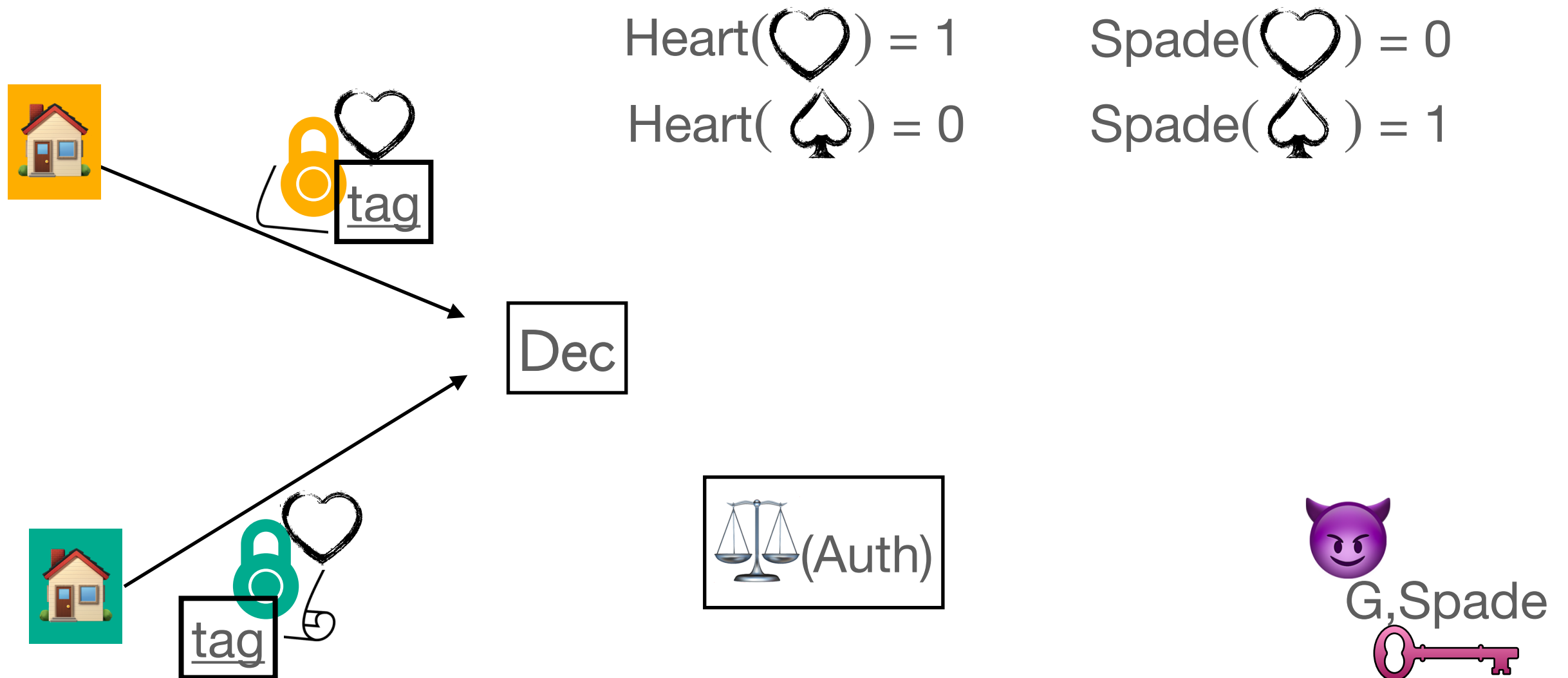
Spade(♠) = 1

Dec



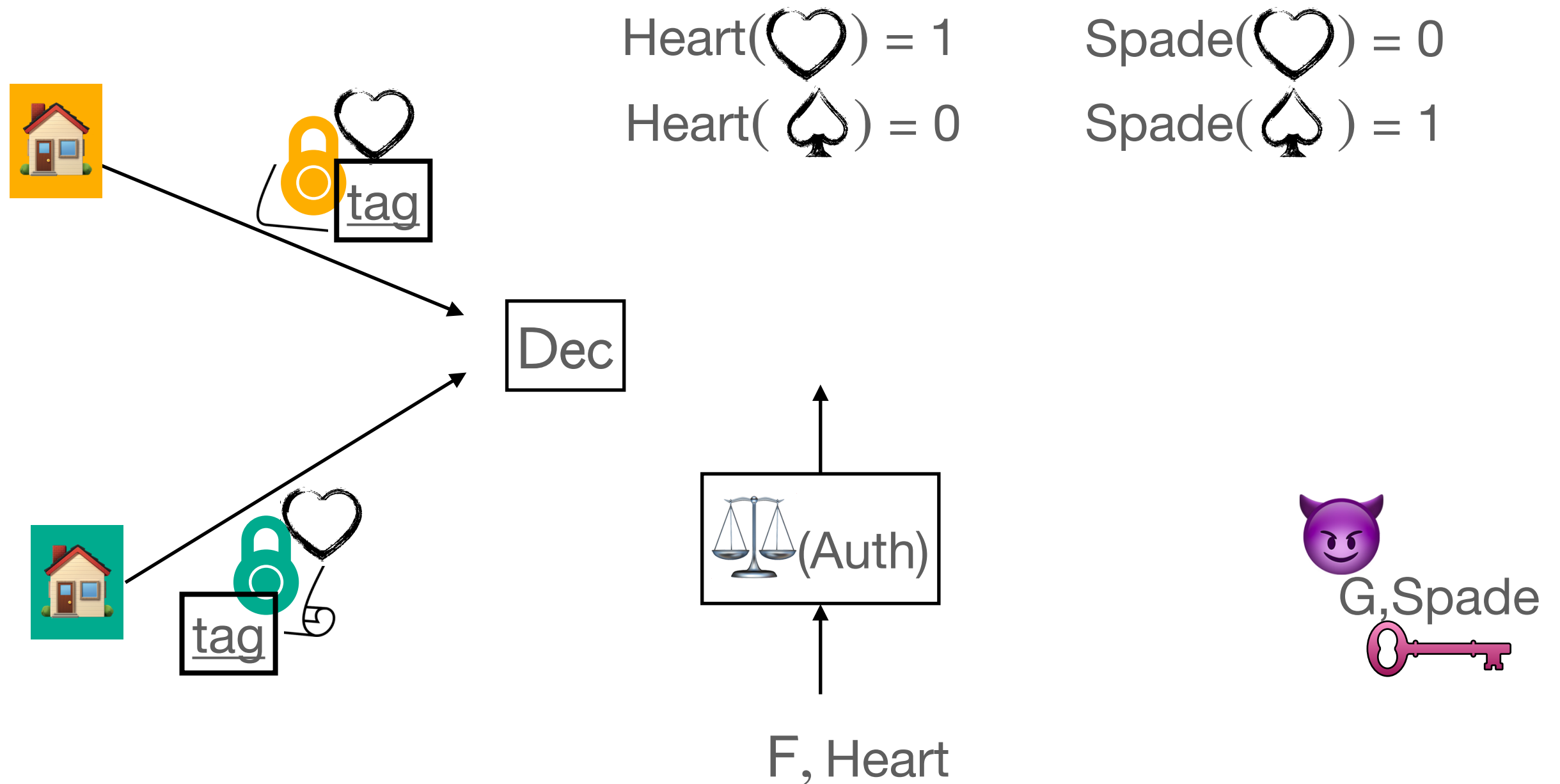
Research Question (1)

Controlling Decryption Keys in MCFE



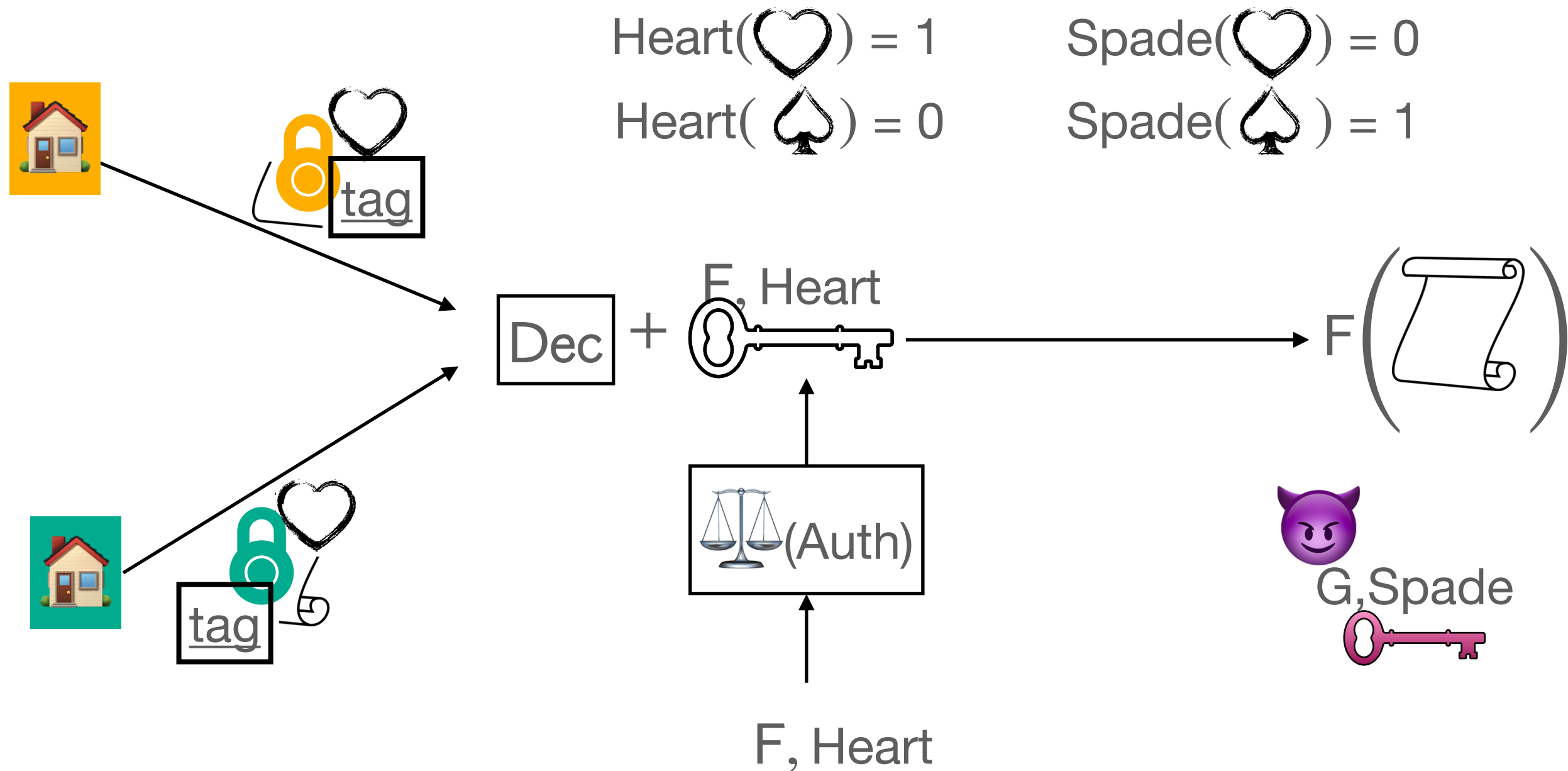
Research Question (1)

Controlling Decryption Keys in MCFE



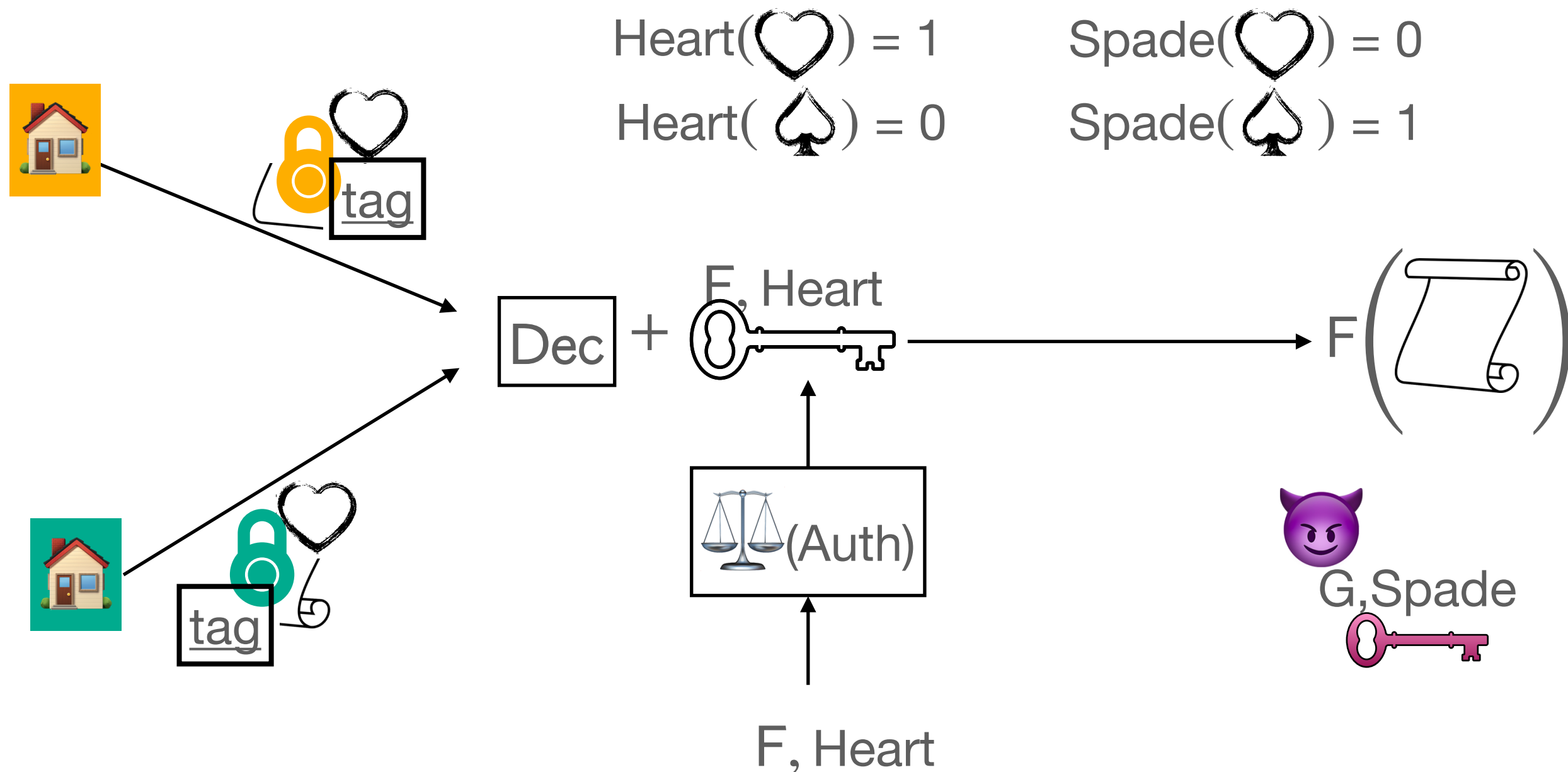
Research Question (1)

Controlling Decryption Keys in MCFE



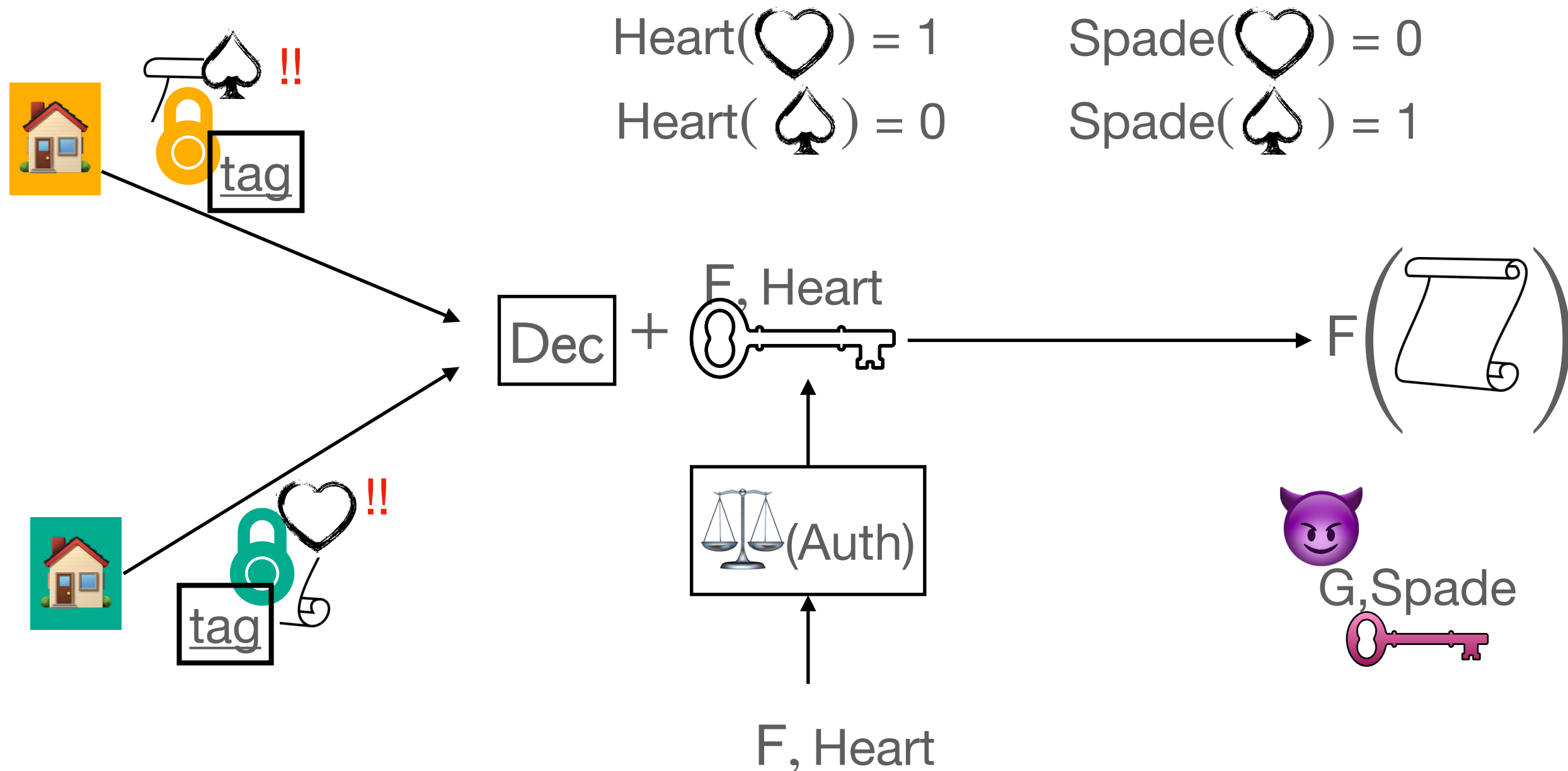
Research Question (1)

Controlling Decryption Keys in MCFE



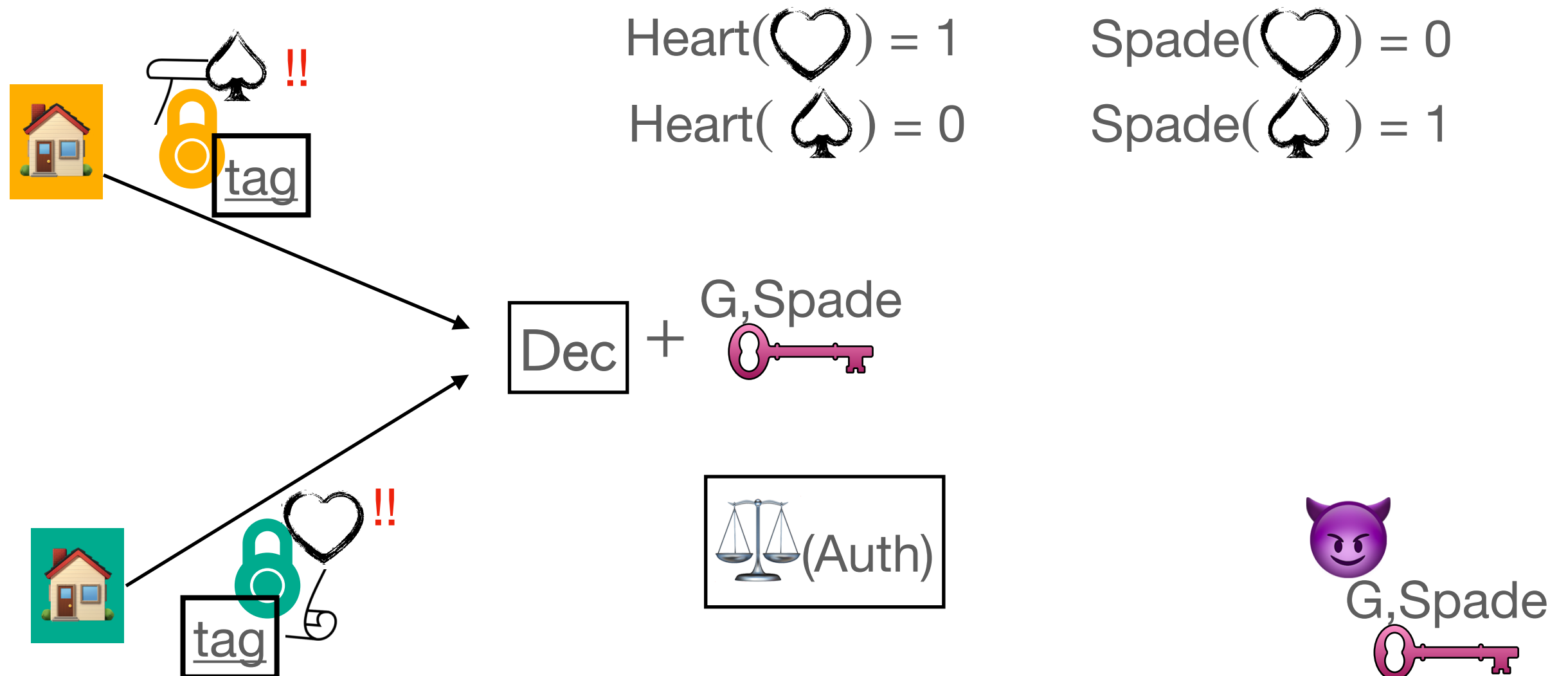
Research Question (1)

Controlling Decryption Keys in MCFE



Research Question (1)

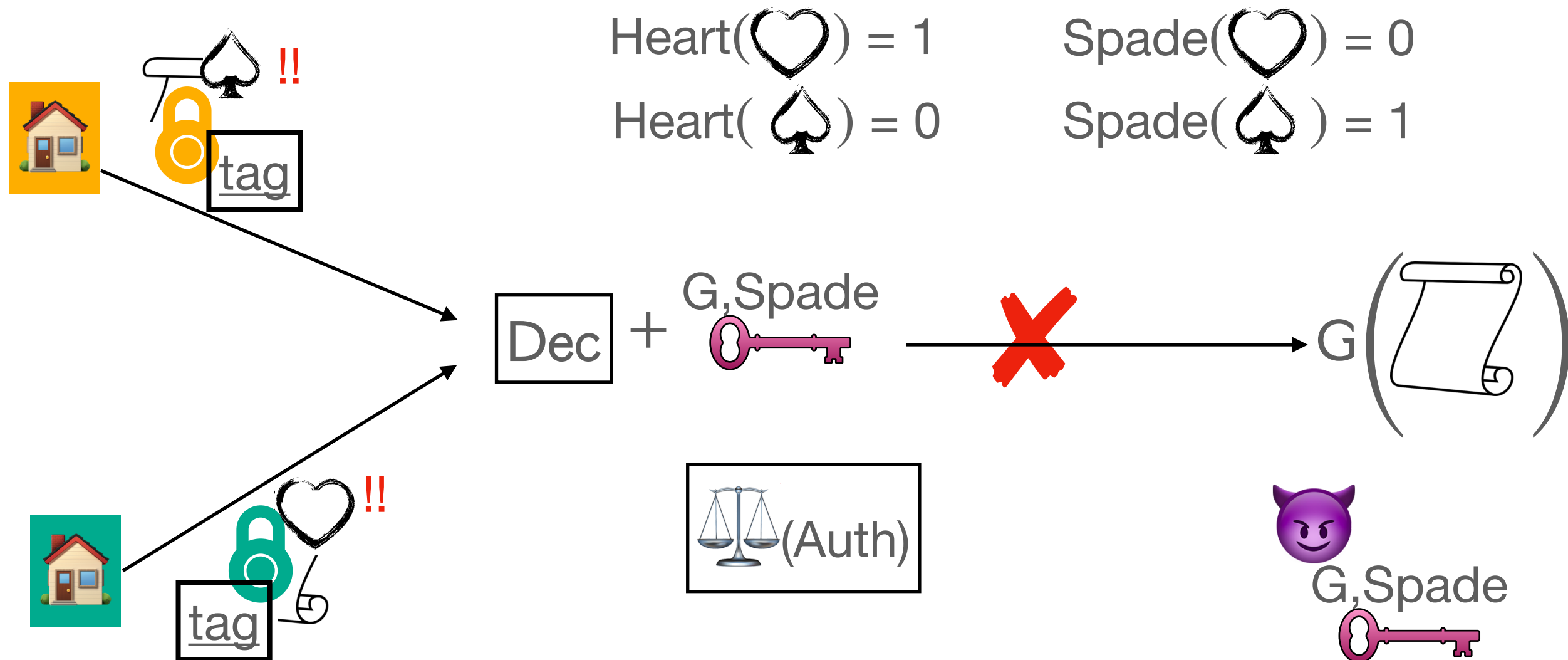
Controlling Decryption Keys in MCFE



Tag, ♠, ♥ : can be public

Research Question (1)

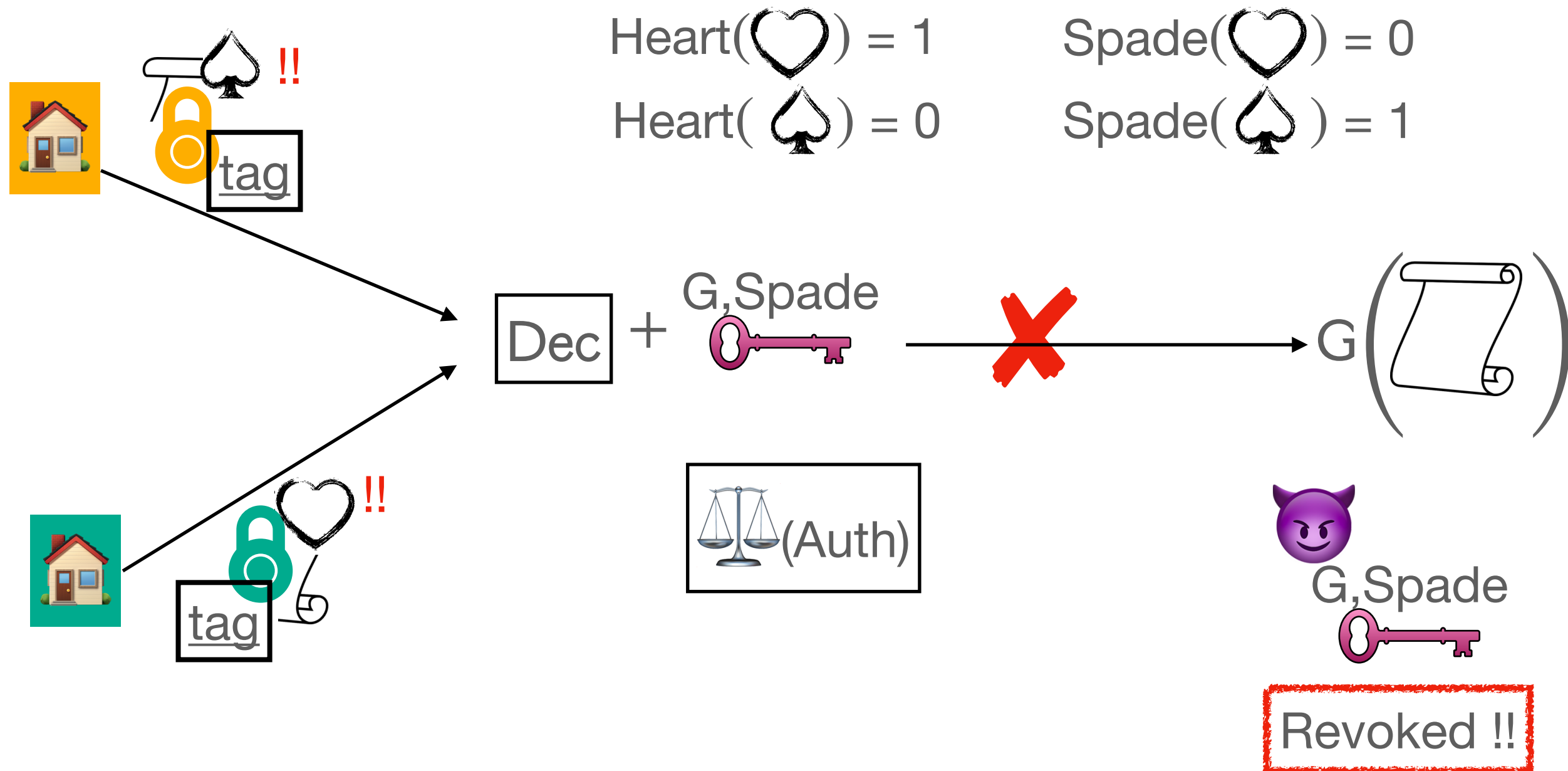
Controlling Decryption Keys in MCFE



Tag, ♠, ♥ : can be public

Research Question (1)

Controlling Decryption Keys in MCFE



Tag, ♠, ♥ : can be public

Research Question (1) - Our Result

Multi-Client with Fine-Grained Access Control [NPP22]

Multi-Client Functional Encryption
Schemes for Inner Products,
With Access Control using Linear Secret
Sharing Schemes

Research Question (1) - Our Result

Multi-Client with Fine-Grained Access Control [NPP22]

**Multi-Client Functional Encryption
Schemes for Inner Products,
With Access Control using Linear Secret
Sharing Schemes**



First definitions for MCFE with key control

Research Question (1) - Our Result

Multi-Client with Fine-Grained Access Control [NPP22]

Multi-Client Functional Encryption
Schemes for Inner Products,
With Access Control using Linear Secret
Sharing Schemes



First definitions for MCFE with key control



Concrete MCFE for Inner-Products $\langle \cdot, param \rangle^{F(\cdot) =}$

Research Question (1) - Our Result

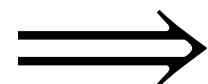
Multi-Client with Fine-Grained Access Control [NPP22]

Multi-Client Functional Encryption
Schemes for Inner Products,
With Access Control using Linear Secret
Sharing Schemes

✓ First definitions for MCFE with key control

✓ Concrete MCFE for Inner-Products $F(\cdot) = \langle \cdot, param \rangle$

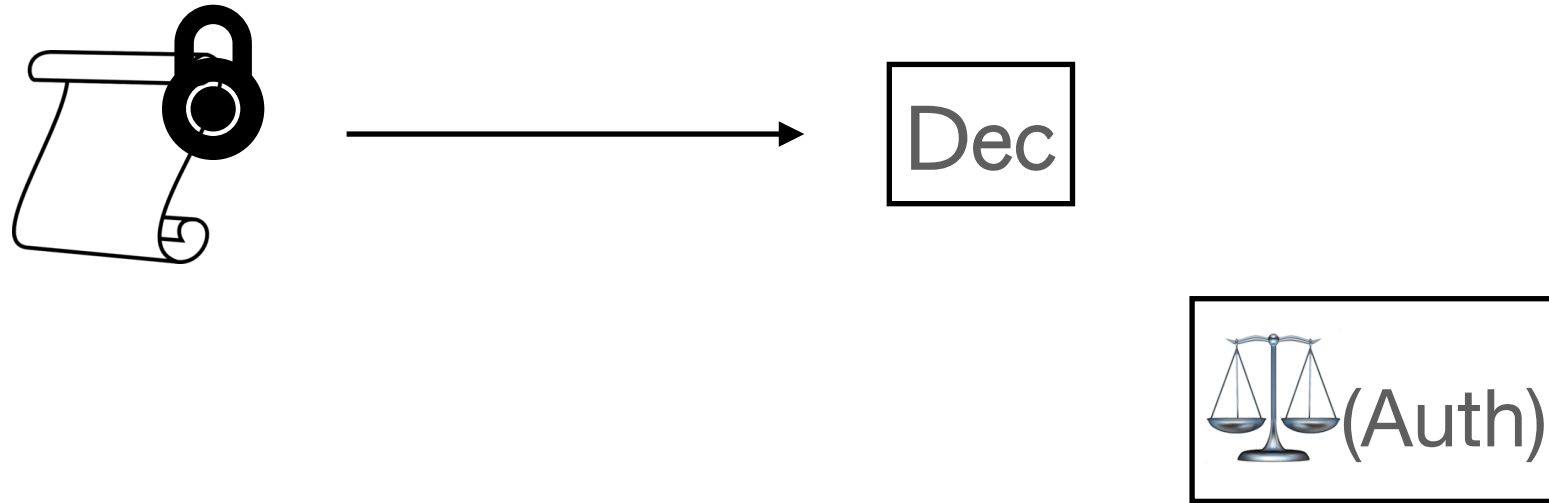
Transformed to
Multi-Input FE



More efficient than ✓
[ACGU20]

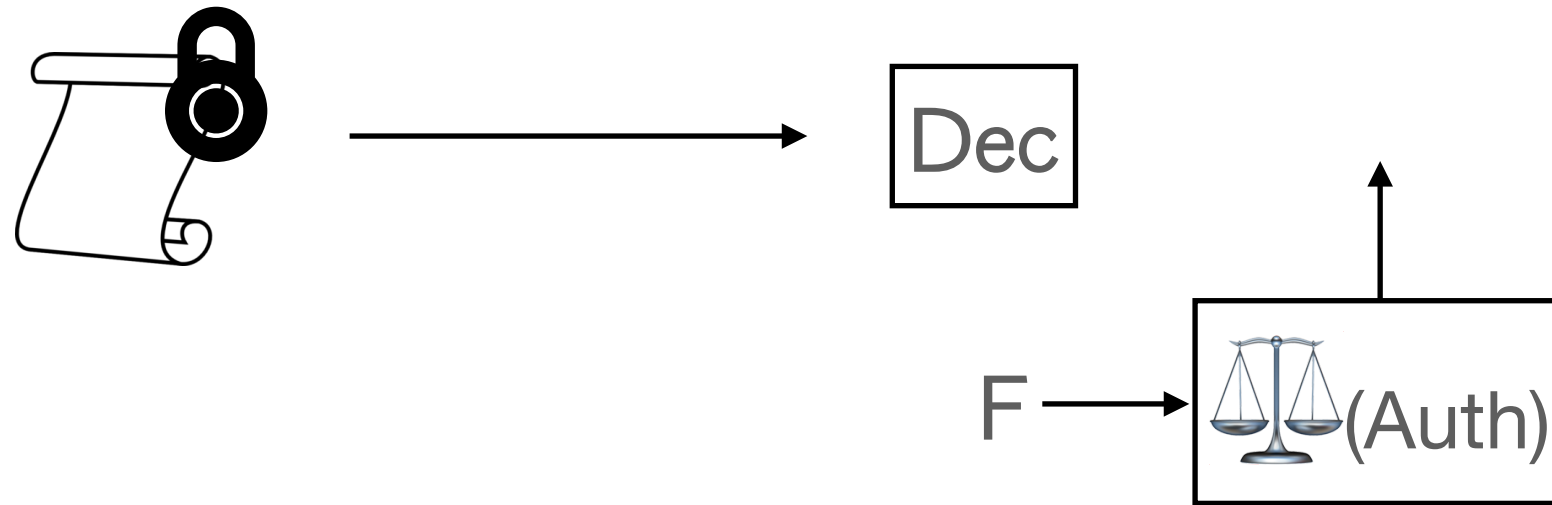
Motivation - Security Model

How do we define a “secure” FE ?



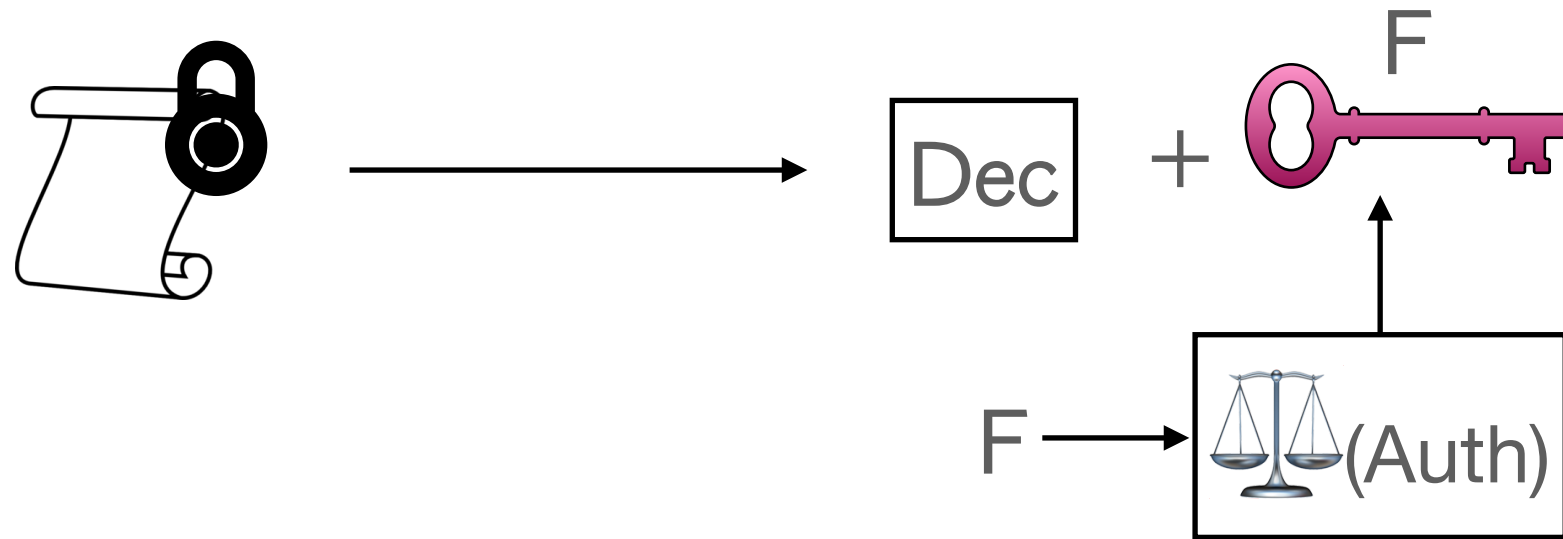
Motivation - Security Model

How do we define a “secure” FE ?



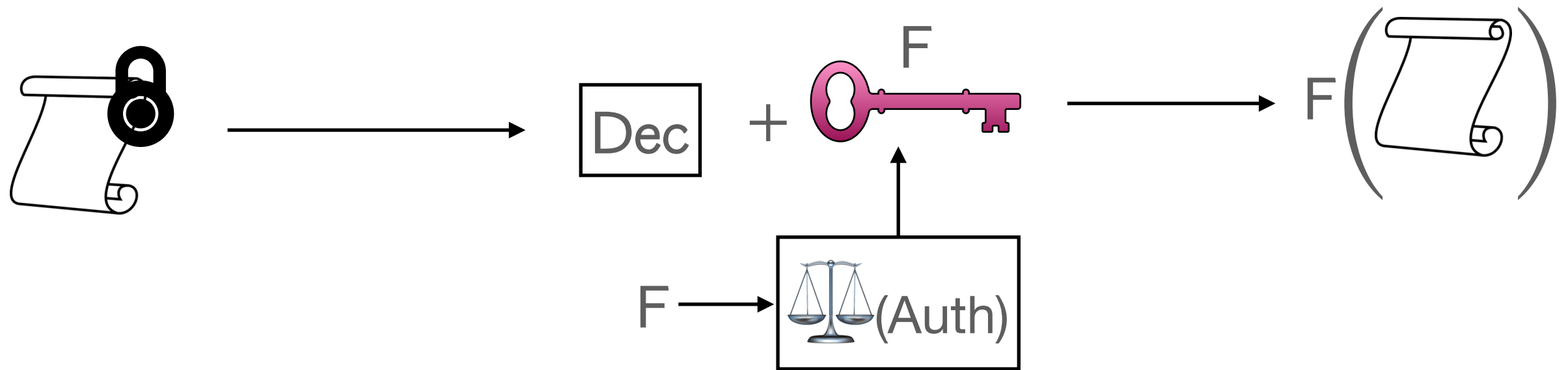
Motivation - Security Model

How do we define a “secure” FE ?



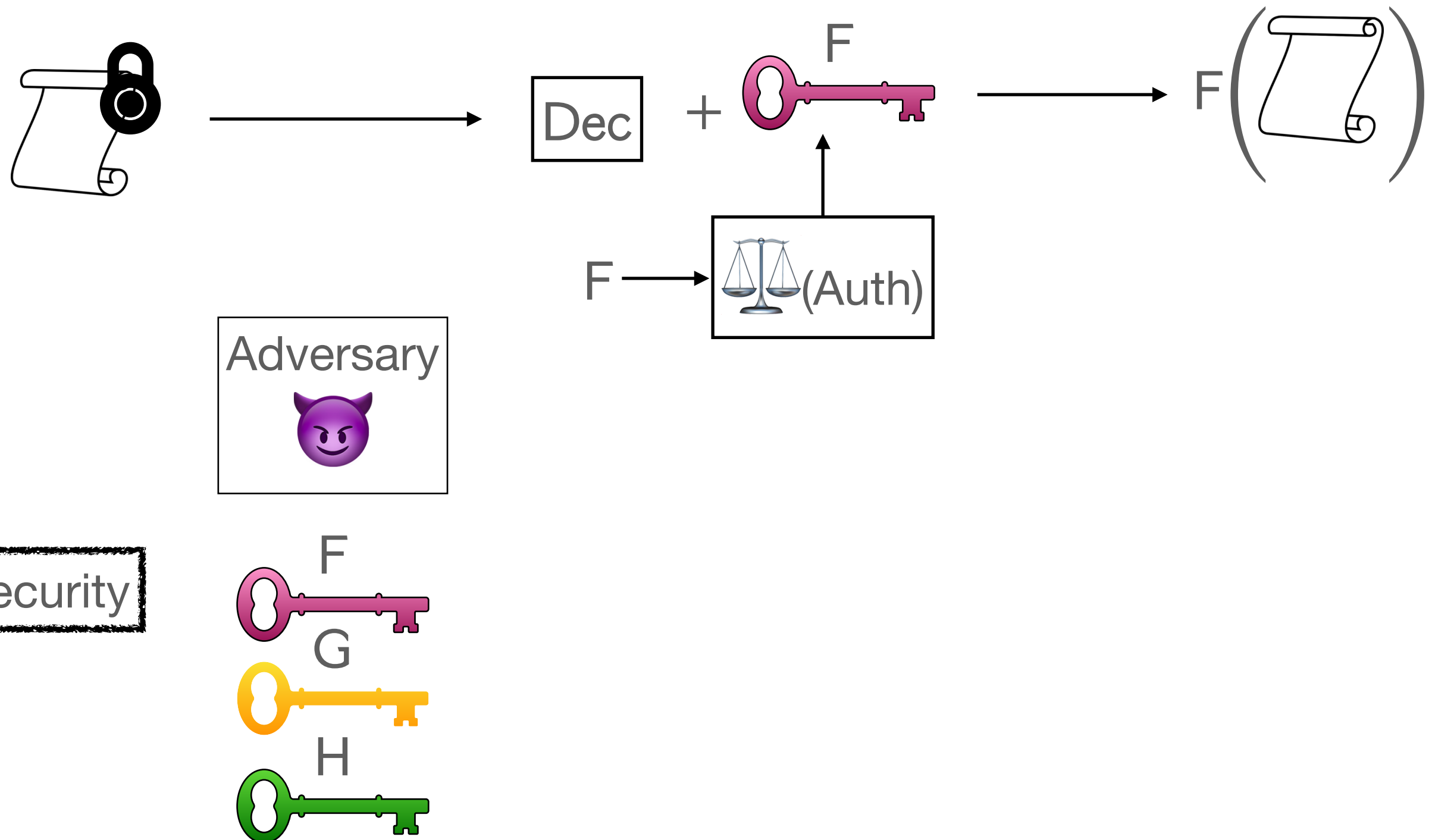
Motivation - Security Model

How do we define a “secure” FE ?



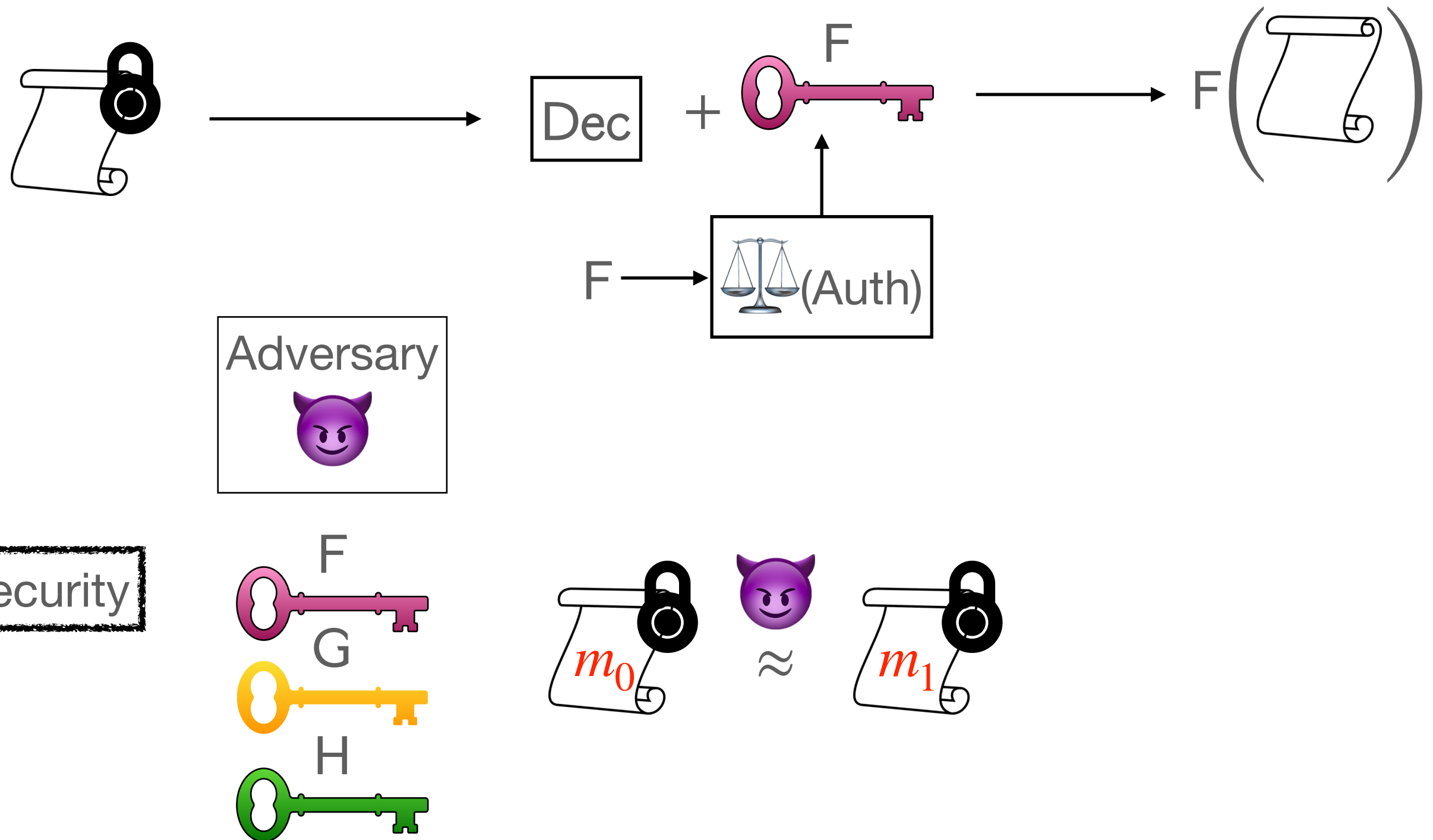
Motivation - Security Model

How do we define a “secure” FE ?



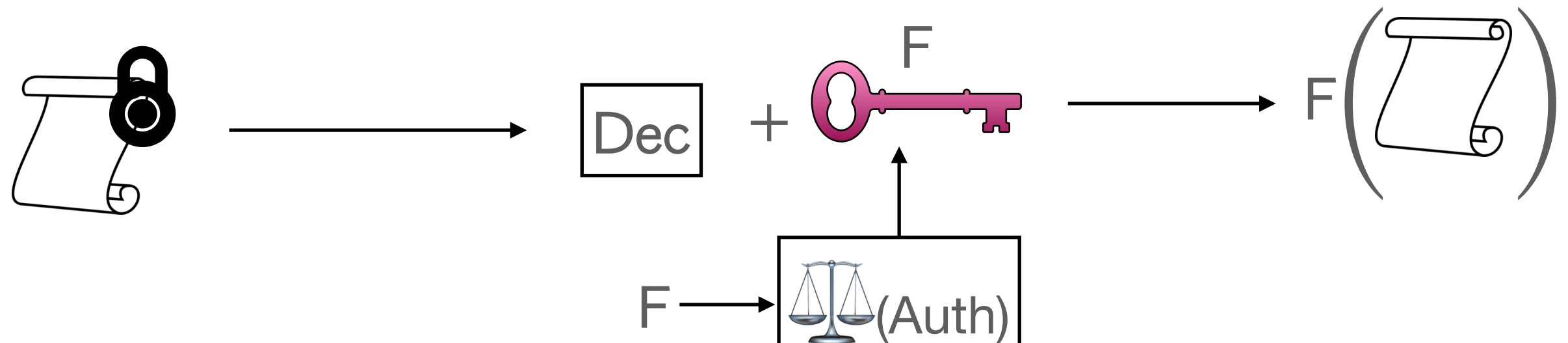
Motivation - Security Model

How do we define a “secure” FE ?



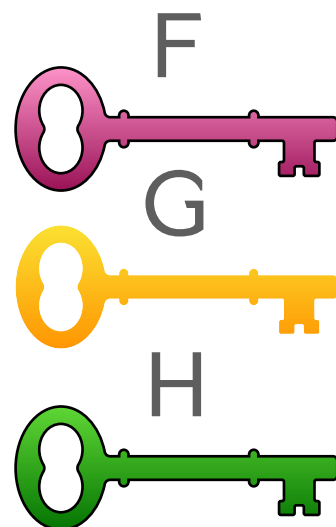
Motivation - Security Model

How do we define a “secure” FE ?



! Admissible

Security



$$F(m_0) = F(m_1)$$

$$G(m_0) = G(m_1)$$

$$H(m_0) = H(m_1)$$

Motivation - Security Model

How do we define a “secure” MCFE ?

Challenger: $b \xleftarrow{\$} \{0,1\}$



Chall

Enc

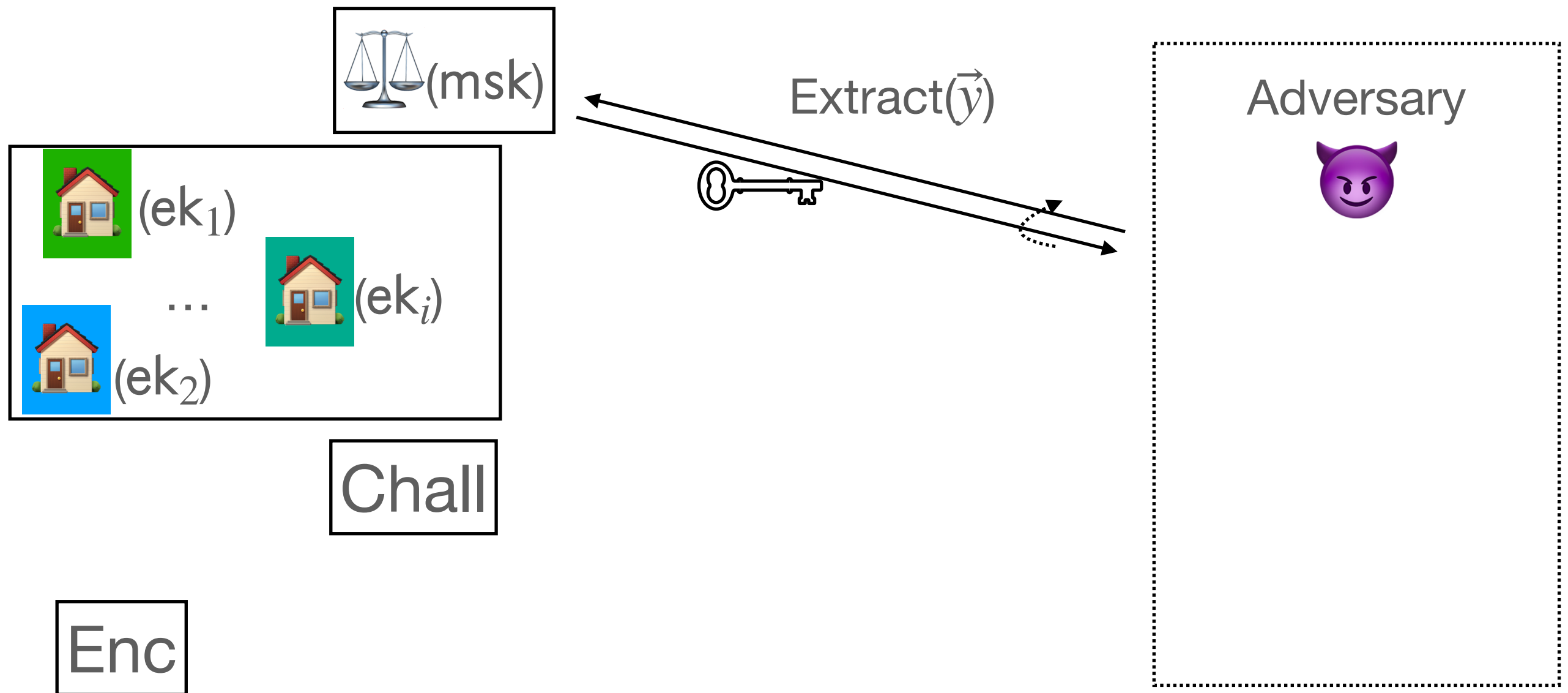
Adversary



Motivation - Security Model

How do we define a “secure” MCFE ?

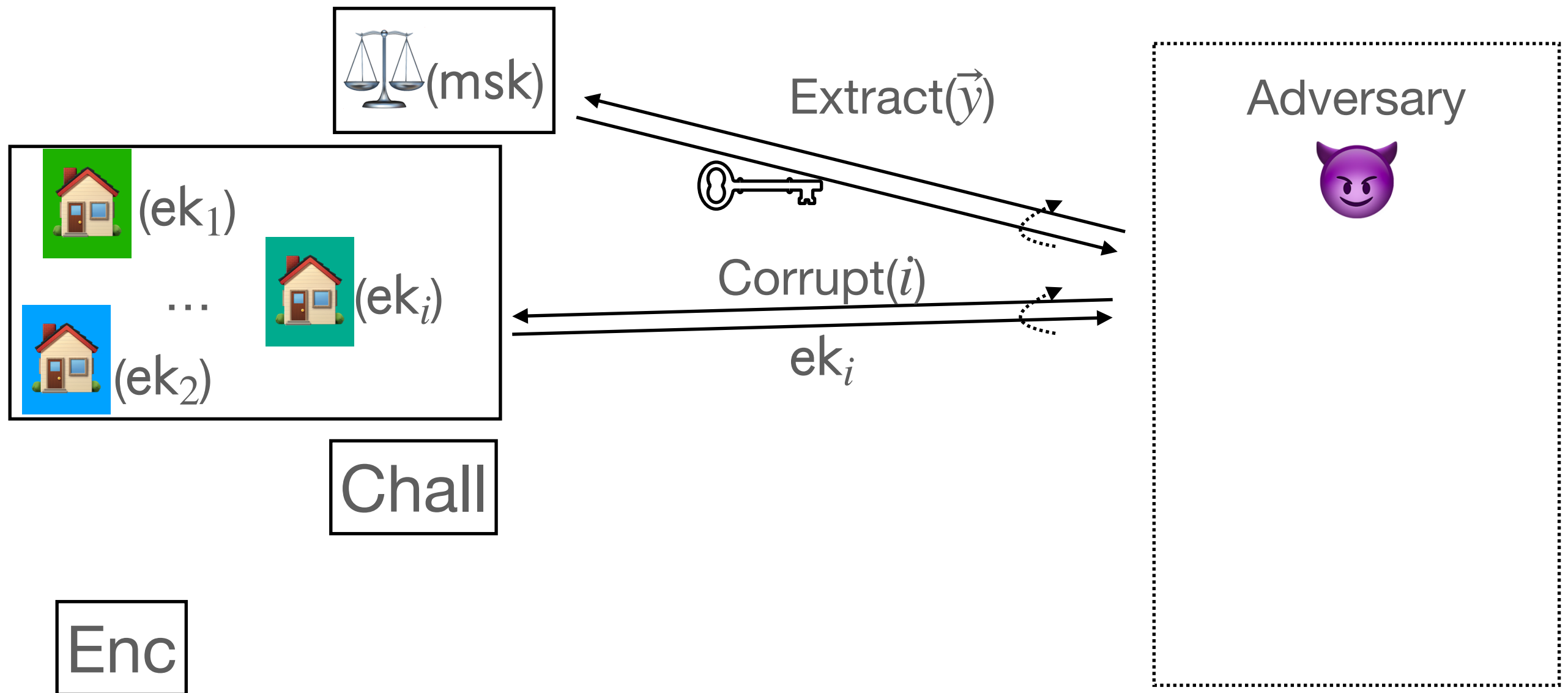
Challenger: $b \leftarrow^{\$} \{0,1\}$



Motivation - Security Model

How do we define a “secure” MCFE ?

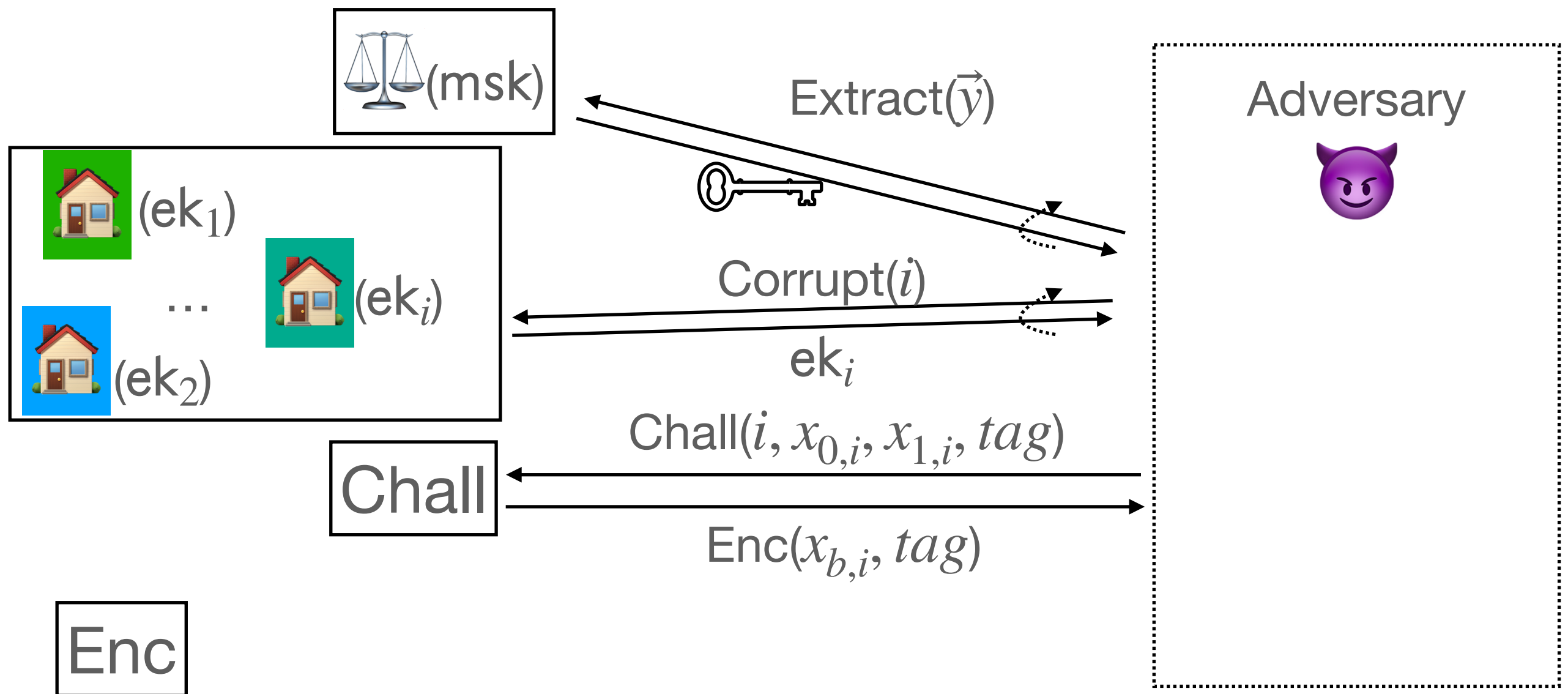
Challenger: $b \leftarrow^{\$} \{0,1\}$



Motivation - Security Model

How do we define a “secure” MCFE ?

Challenger: $b \xleftarrow{\$} \{0,1\}$

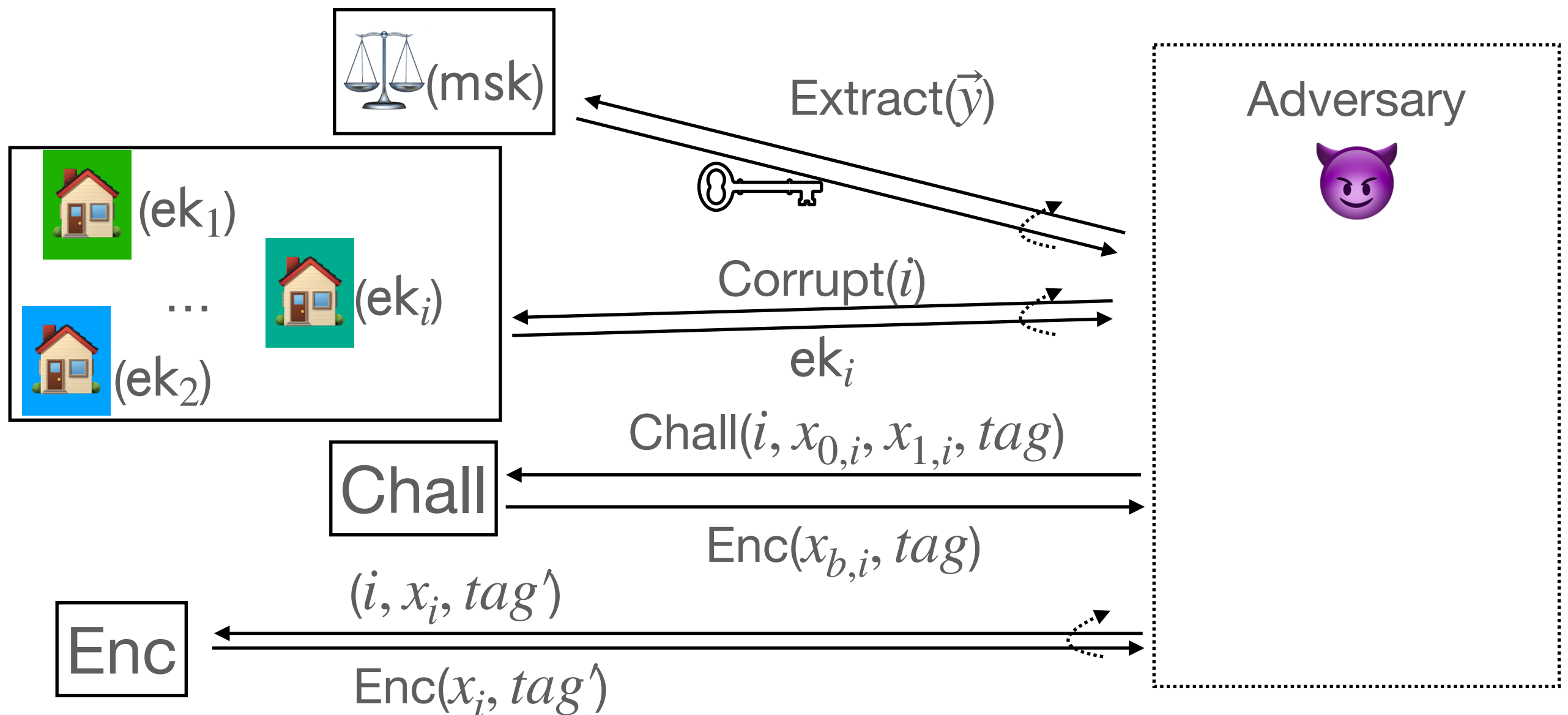


Secret Keys: **Enc** + 1 chall security $\Rightarrow$ multi-chall security

Motivation - Security Model

How do we define a “secure” MCFE ?

Challenger: $b \xleftarrow{\$} \{0,1\}$



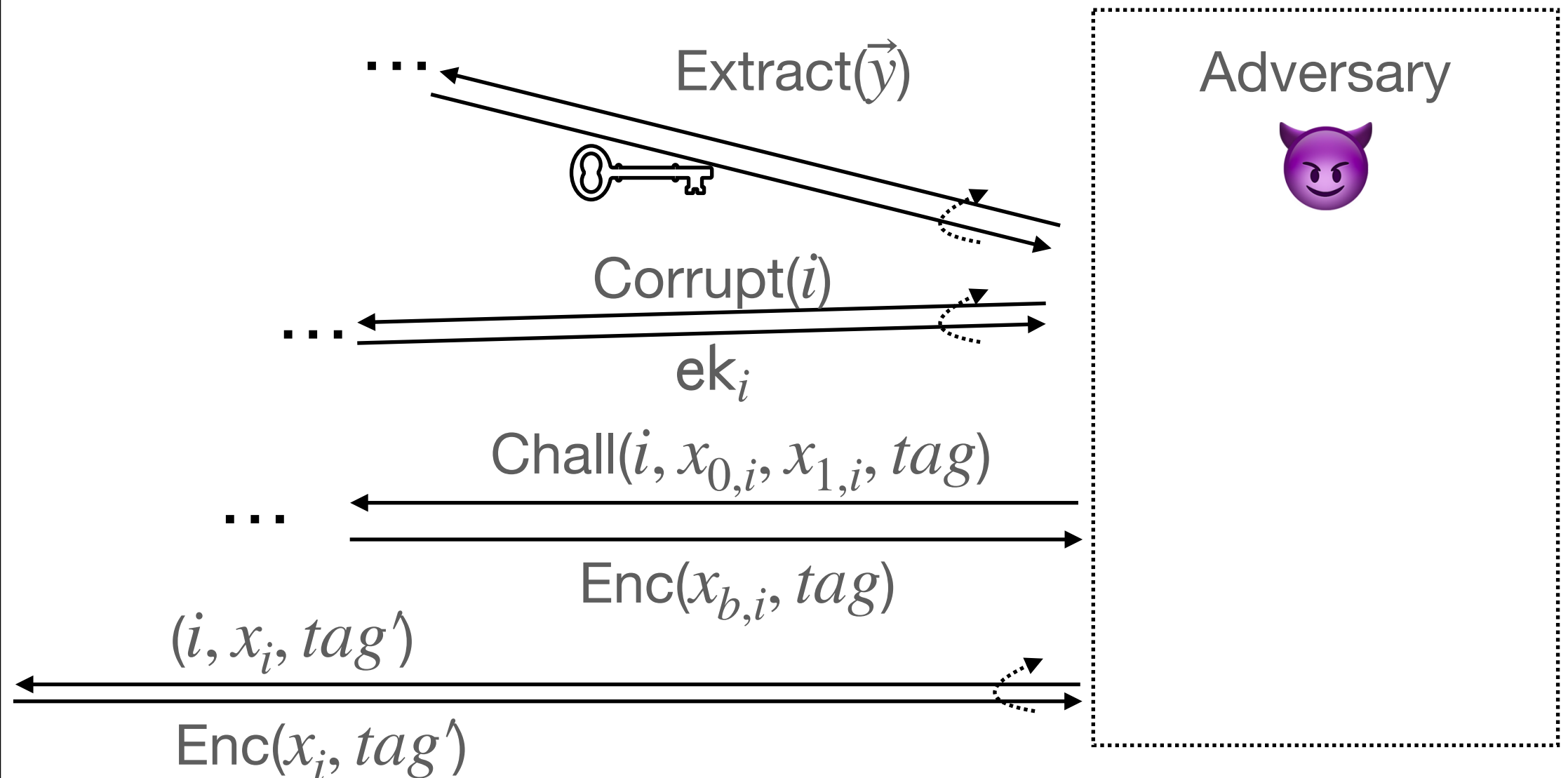
Secret Keys: **Enc** + 1 chall security $\Rightarrow$ multi-chall security

Motivation - Security Model

How do we define a “secure” MCFE ?

Challenger

Challenger: $b \leftarrow^{\$} \{0,1\}$



Motivation - Security Model

How do we define a “secure” MCFE ?

Win if $b' = b$

$b' \in \{0,1\}$

Adversary



Extract($\vec{y}$)



Corrupt(i)

ek_i

Chall($i, x_{0,i}, x_{1,i}, tag$)

Enc($x_{b,i}, tag$)

(i, x_i, tag')

Enc(x_i, tag')

Challenger

Challenger: $b \leftarrow^{\$} \{0,1\}$

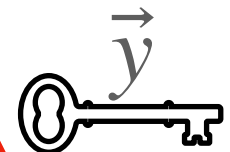
Motivation - Security Model

How do we define a “secure” MCFE ?

Win if $b' = b$

$b' \in \{0,1\}$

Adversary



$$\langle \vec{x}_0 - \vec{x}_1, \vec{y} \rangle = 0$$



Extract($\vec{y}$)



Corrupt(i)

ek_i

Chall($i, x_{0,i}, x_{1,i}, tag$)

Enc($x_{b,i}, tag$)

(i, x_i, tag')

Enc(x_i, tag')

Challenger

Challenger: $b \leftarrow^{\$} \{0,1\}$

Motivation - Security Model

How do we define a “secure” MCFE ?

Win if $b' = b$

$b' \in \{0,1\}$

Adversary



[CDGPP'18]

Corrupt(i)

$\Rightarrow x_{0,i} = x_{1,i}$

🤔 Justified?

Challenger

Challenger: $b \leftarrow^{\$} \{0,1\}$

Extract($\vec{y}$)



Corrupt(i)

ek_i

Chall($i, x_{0,i}, x_{1,i}, tag$)

Enc($x_{b,i}, tag$)

(i, x_i, tag')

Enc(x_i, tag')

Motivation - Security Model

A “secure” MCFE - the case of Inner Products

Adversary



[CDGPP'18]

Corrupt(i)

$\Rightarrow x_{0,i} = x_{1,i}$

🤔 Justified?

Motivation - Security Model

A “secure” MCFE - the case of Inner Products

[CDGPP'18]

Corrupt(i)

$$\Rightarrow x_{0,i} = x_{1,i}$$

—> Follow-ups with **same** constraint:

[ABKW19],[ABG19],[LT19],

[CDG+20],[AGT21],[SV23]

Adversary



[CDGPP'18]

Corrupt(i)

$$\Rightarrow x_{0,i} = x_{1,i}$$



Justified?

Motivation - Security Model

A “secure” MCFE - the case of Inner Products

[CDGPP'18]

Corrupt(i)

$$\Rightarrow x_{0,i} = x_{1,i}$$

—> Follow-ups with **same** constraint:
[ABKW19],[ABG19],[LT19],
[CDG+20],[AGT21],[SV23]

Adversary



[CDGPP'18]

Corrupt(i)

$$\Rightarrow x_{0,i} = x_{1,i}$$

🤔 Justified?

From [CDGPP18]:

MCFE is **Secret** Encryption Keys

⇒ Usually deterministic

⇒

Corrupt(i)

$$\Rightarrow x_{0,i} = x_{1,i}$$

Motivation - Security Model

A “secure” MCFE - the case of Inner Products

[CDGPP'18]

Corrupt(i)

$$\Rightarrow x_{0,i} = x_{1,i}$$

—> Follow-ups with **same** constraint:

[ABKW19],[ABG19],[LT19],

[CDG+20],[AGT21],[SV23]

Adversary



[CDGPP'18]

Corrupt(i)

$$\Rightarrow x_{0,i} = x_{1,i}$$



Justified?

Motivation - Security Model

A “secure” MCFE - the case of Inner Products

[CDGPP'18]

Corrupt(i)

$$\Rightarrow x_{0,i} = x_{1,i}$$

—> Follow-ups with **same** constraint:
[ABKW19],[ABG19],[LT19],
[CDG+20],[AGT21],[SV23]

Adversary



[CDGPP'18]

Corrupt(i)

$$\Rightarrow x_{0,i} = x_{1,i}$$

🤔 Justified?



🤔 Modeling Problem:

Why can't we do better?

(/i.e., weaker condition

$\Rightarrow$ considering stronger attacker

$\Rightarrow$ stronger security guarantee)

Research Question (2) - Our Result

Improved Security Notion for (D)MCFE

Optimal Security Notion for Decentralized Multi-Client Functional Encryption [NPP'23]

Adversary

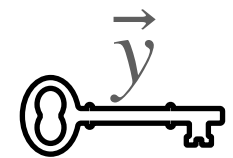


Research Question (2) - Our Result

Improved Security Notion for (D)MCFE

Optimal Security Notion for Decentralized Multi-Client Functional Encryption [NPP'23]

Adversary



$$\langle \vec{x}_0 - \vec{x}_1, \vec{y} \rangle = 0$$



Research Question (2) - Our Result

Improved Security Notion for (D)MCFE

Optimal Security Notion for Decentralized Multi-Client Functional Encryption [NPP'23]

Adversary



Corrupt(i)

$\Rightarrow x_{0,i} = x_{1,i}$
or $y_i = 0$

Research Question (2) - Our Result

Improved Security Notion for (D)MCFE

Optimal Security Notion for Decentralized Multi-Client Functional Encryption [NPP'23]

Adversary



Corrupt(i)

$\Rightarrow x_{0,i} = x_{1,i}$
or $y_i = 0$

+ Optimality
proof : this is
the least we
require 🎉

Research Question (2) - Our Result

Improved Security Notion for (D)MCFE

Optimal Security Notion for Decentralized Multi-Client Functional Encryption [NPP'23]

✓ **First to Refine Security of DMCFE**
After [CDGPP18]

Adversary



Corrupt(i)

$\Rightarrow x_{0,i} = x_{1,i}$
or $y_i = 0$

+ Optimality
proof : this is
the least we
require 🎉

Research Question (2) - Our Result

Improved Security Notion for (D)MCFE

Optimal Security Notion for Decentralized Multi-Client Functional Encryption [NPP'23]

✓ First to Refine Security of DMCFE After [CDGPP18]

✓ First to give Proving Framework For Optimal Security of DMCFE

Adversary



Corrupt(i)

$\Rightarrow x_{0,i} = x_{1,i}$
or $y_i = 0$

+ Optimality
proof : this is
the least we
require 🎉

Research Question (2) - Our Result

Improved Security Notion for (D)MCFE

Optimal Security Notion for Decentralized Multi-Client Functional Encryption [NPP'23]

✓ First to Refine Security of DMCFE After [CDGPP18]

✓ First to give Proving Framework For Optimal Security of DMCFE

✓ Concrete DMCFE for Inner-Products, With New Stronger Security

$$F(\cdot) = \langle \cdot, param \rangle$$

Adversary



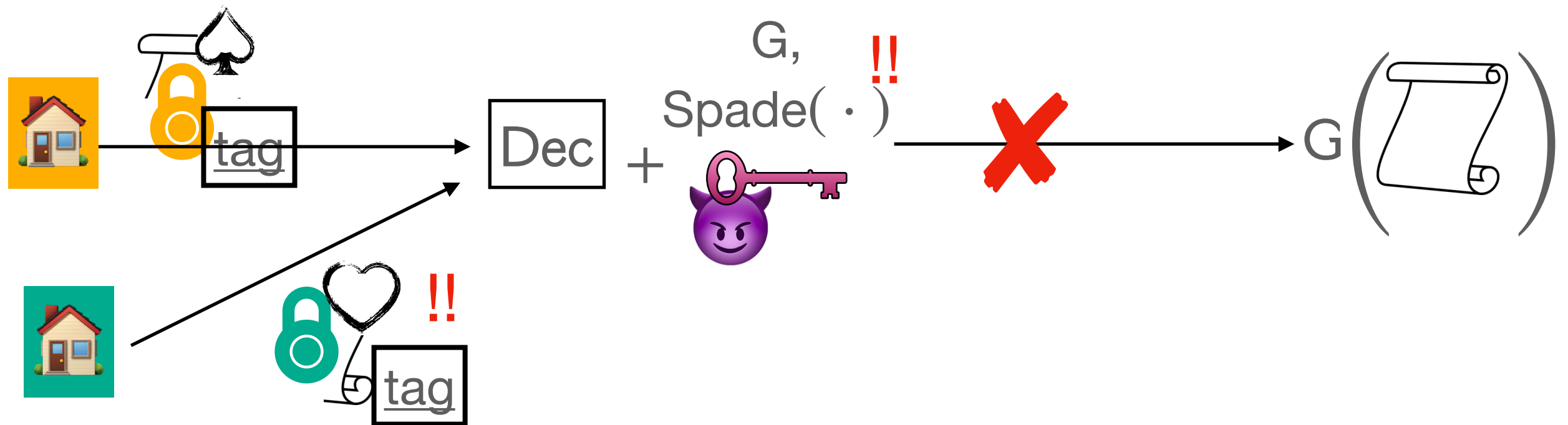
Corrupt(i)

$$\Rightarrow x_{0,i} = x_{1,i} \text{ or } y_i = 0$$

+ Optimality proof : this is the least we require 🎉

Research Question (3) - Combining All Together

Controlling Keys in the Optimal Setting



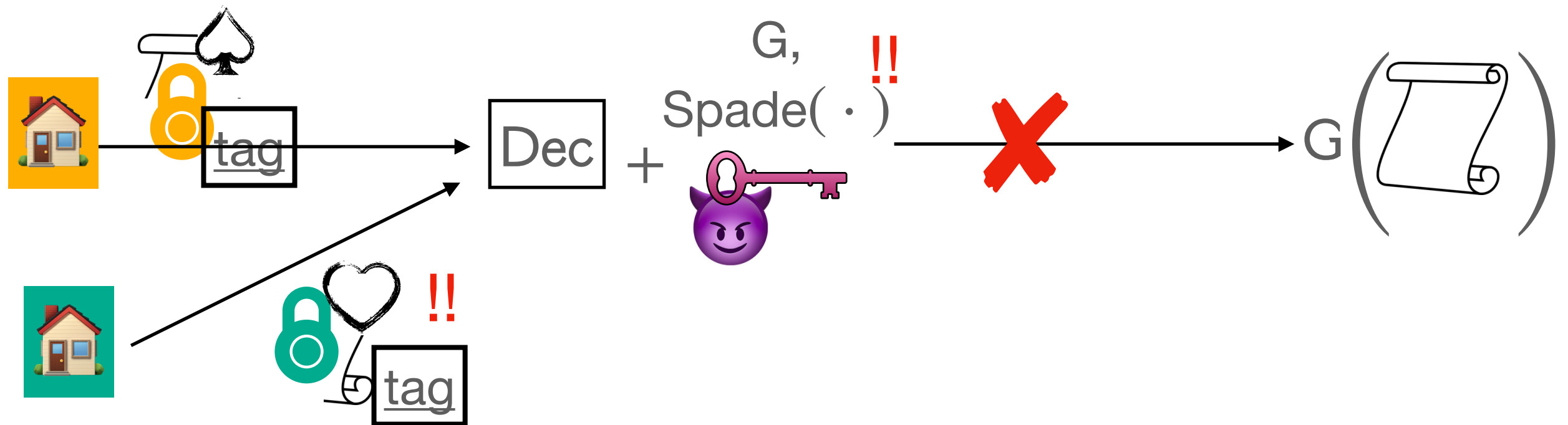
🤔 public-key FE [BSW11],
secret-key MCFE [GGG+14,
CDG+18]

Syntactical Problem:

NOT allow public-Att

Research Question (3) - Combining All Together

Controlling Keys in the Optimal Setting



Tag, ♠, ♥ :
can be public

🤔 public-key FE [BSW11],
secret-key MCFE [GGG+14,
CDG+18]

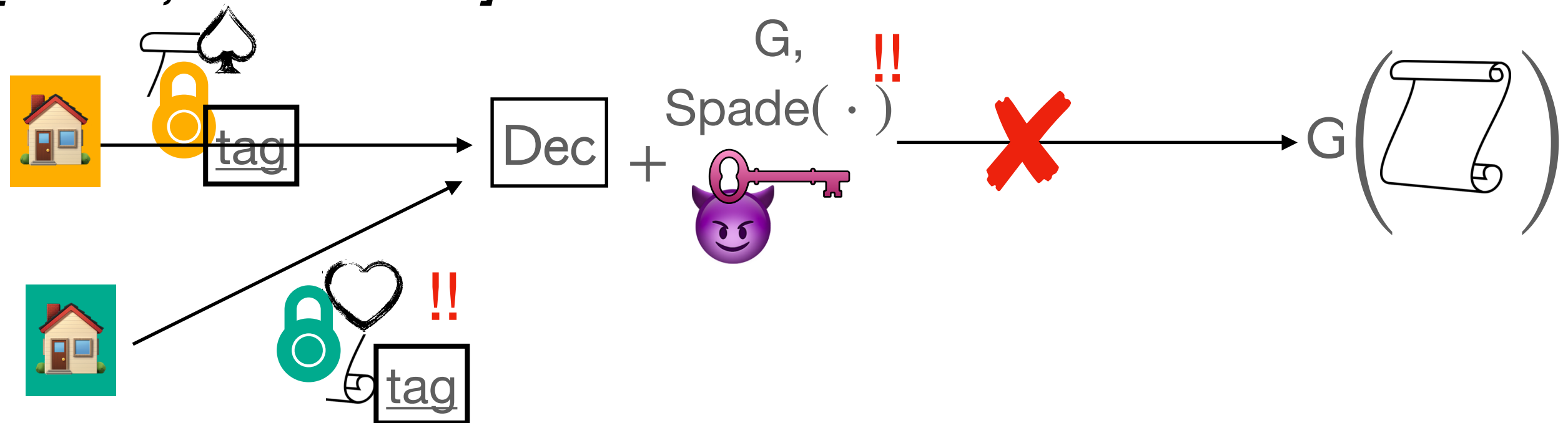
Syntactical Problem:

NOT allow public-Att

Research Question (3) - Our Result

Controlling Keys in the Optimal Setting

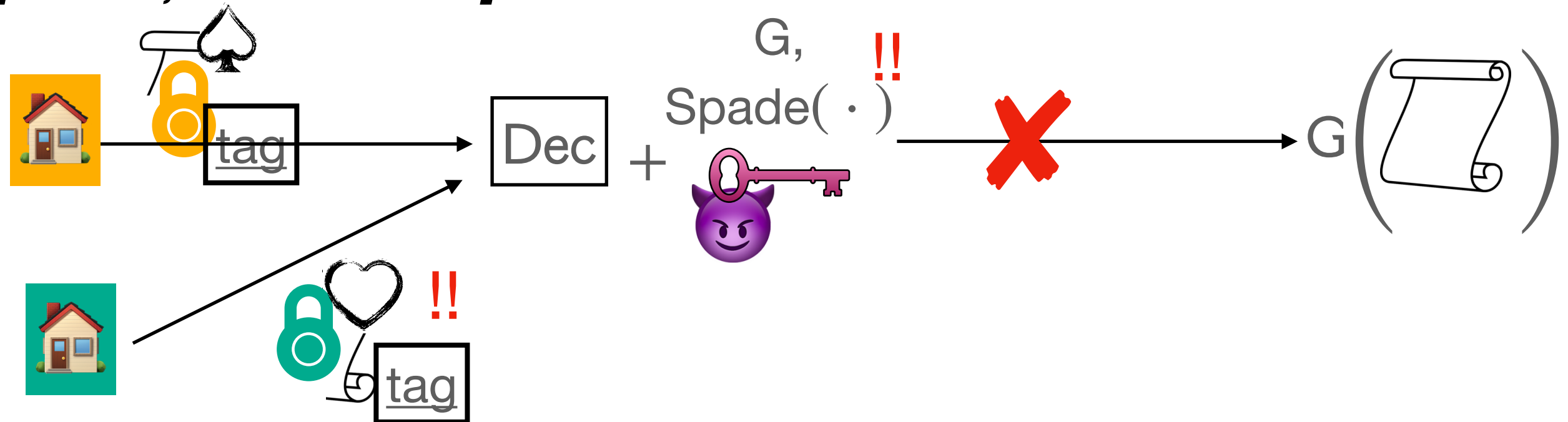
Multi-Client Functional Encryption with Public Inputs and Strong Security
[NPP'24, in submission]



Research Question (3) - Our Result

Controlling Keys in the Optimal Setting

Multi-Client Functional Encryption with Public Inputs and Strong Security
[NPP'24, in submission]

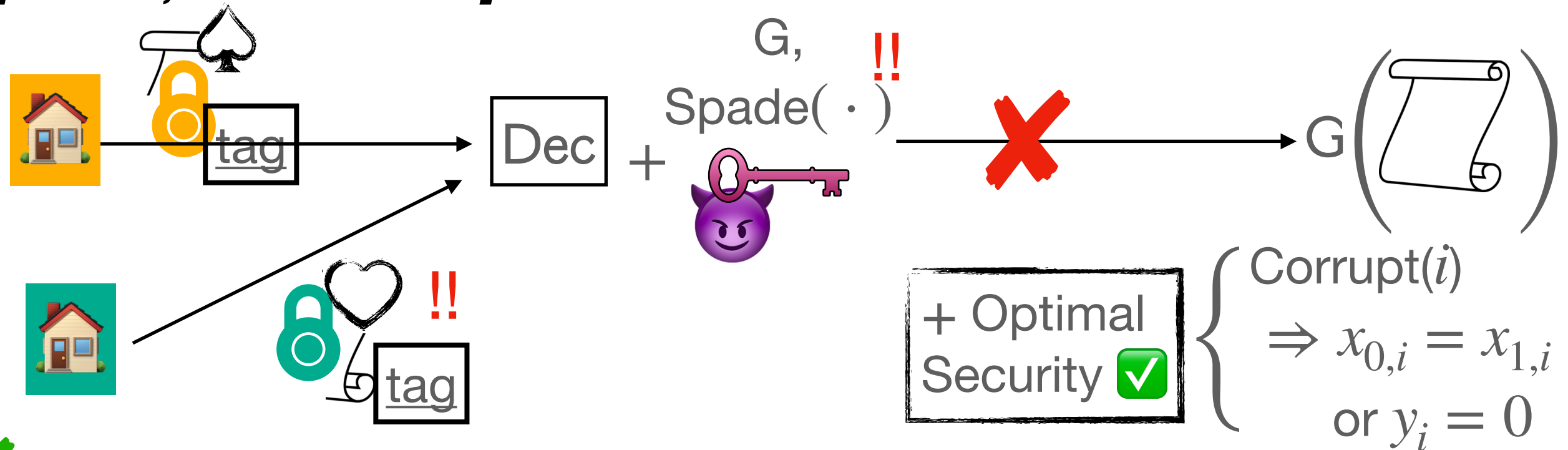


✓ Refine Syntax of MCFE:
With **public**-inputs

Research Question (3) - Our Result

Controlling Keys in the Optimal Setting

Multi-Client Functional Encryption with Public Inputs and Strong Security
[NPP'24, in submission]

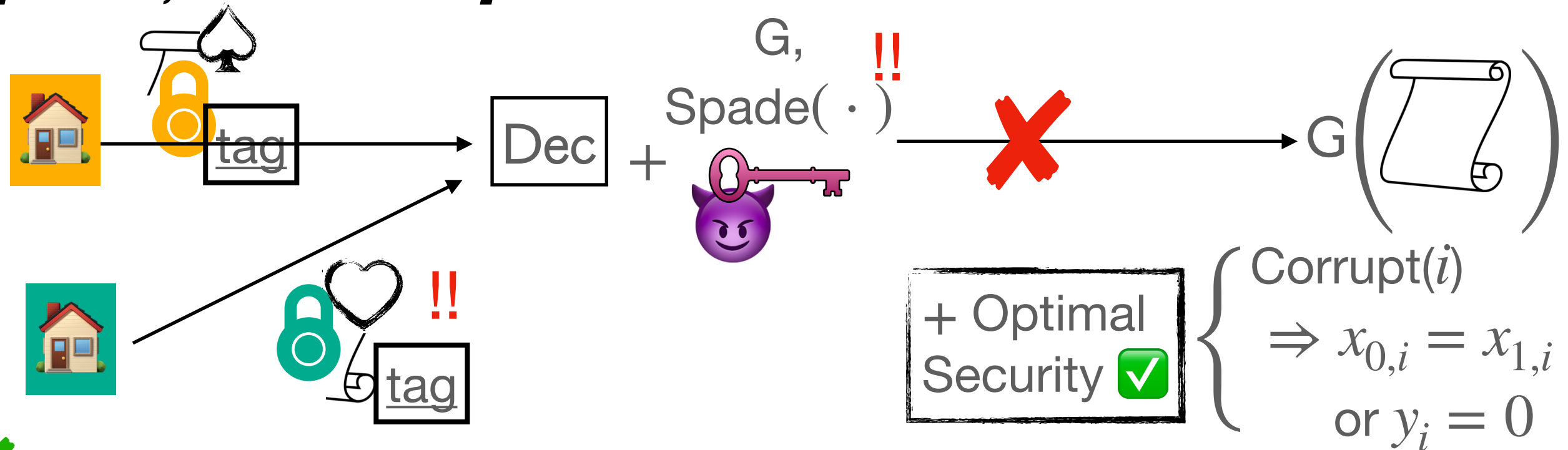


✓ Refine Syntax of MCFE:
With **public**-inputs

Research Question (3) - Our Result

Controlling Keys in the Optimal Setting

Multi-Client Functional Encryption with Public Inputs and Strong Security
[NPP'24, in submission]



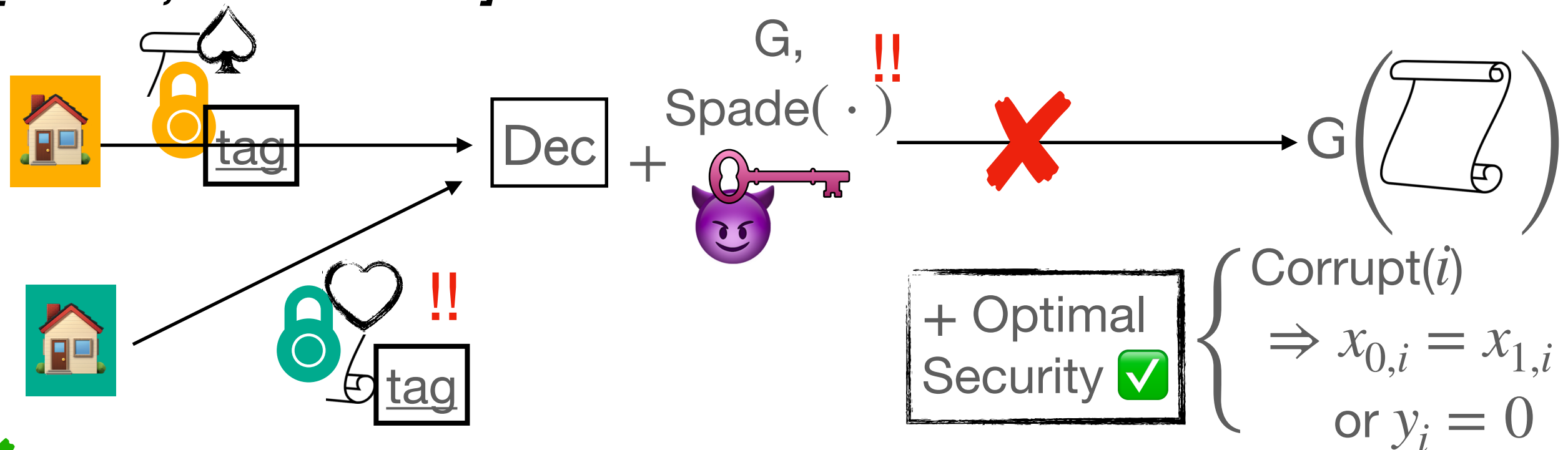
✓ Refine Syntax of MCFE:
With **public**-inputs

✓ Identify Implications:
MCFE + optimal security
 $\Rightarrow$ **public**-key FE

Research Question (3) - Our Result

Controlling Keys in the Optimal Setting

Multi-Client Functional Encryption with Public Inputs and Strong Security
[NPP'24, in submission]



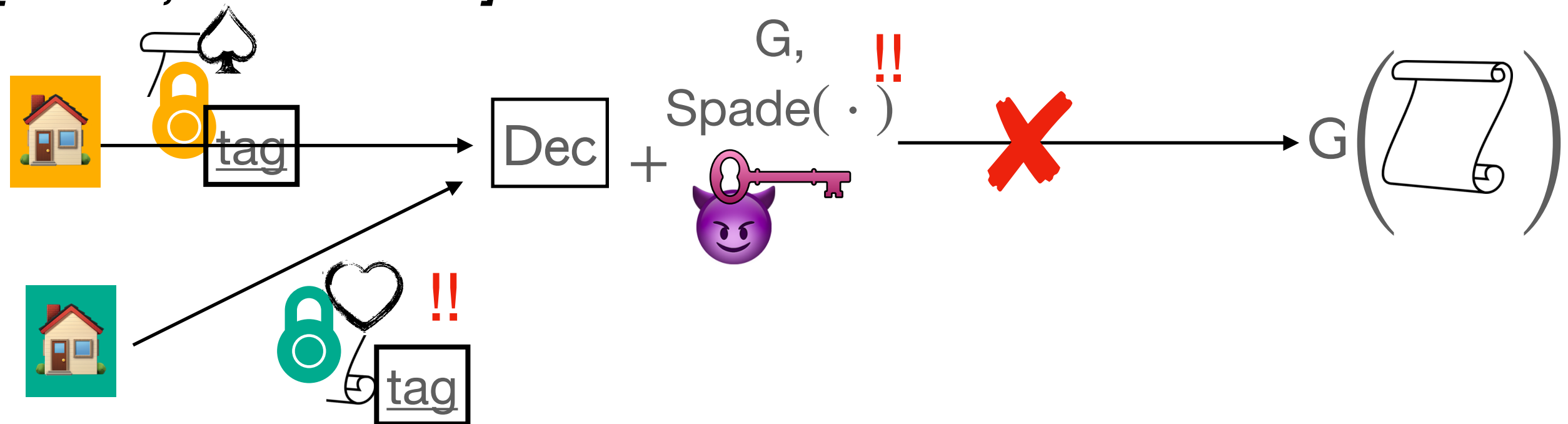
- ✓ Refine Syntax of MCFE:
With **public**-inputs
- ✓ Identify Implications:
MCFE + optimal security
 $\Rightarrow$ **public**-key FE

$\Rightarrow$ Key-Policy
Attribute-based
Encryption,
Public- or **private**-att 🎉

Research Question (3) - Our Result

Controlling Keys in the Optimal Setting

Multi-Client Functional Encryption with Public Inputs and Strong Security
[NPP'24, in submission]



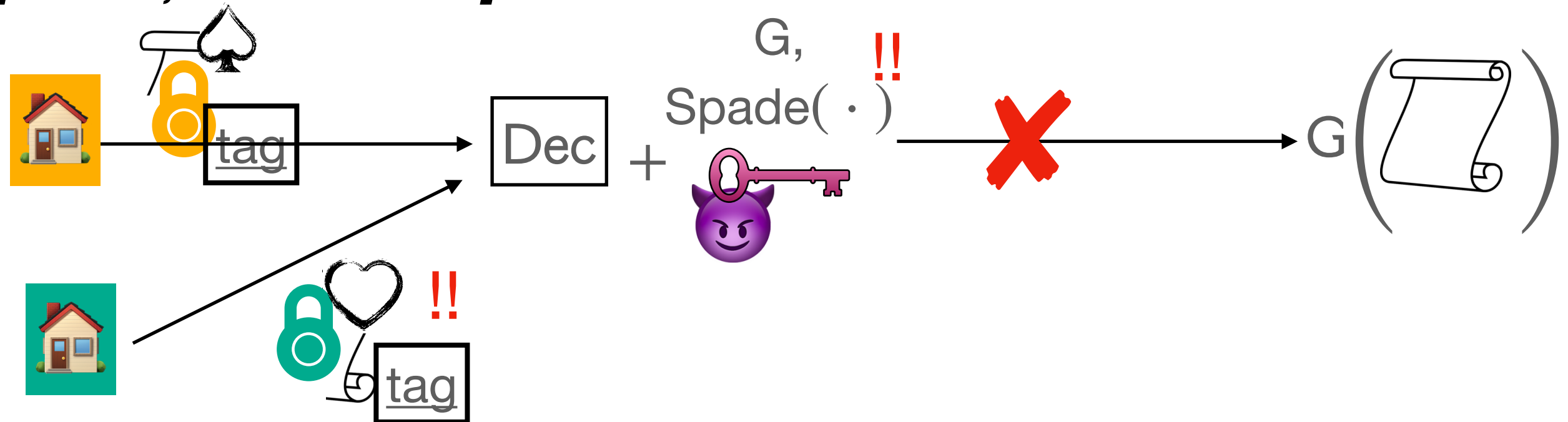
- ✓ Refine Syntax of MCFE:
With **public**-inputs
 - ✓ Identify Implications:
MCFE + optimal security
- ⇒ **public**-key FE

Key-Policy
Attribute-based
Encryption,
Public- or **private**-att 🎉

Research Question (3) - Our Result

Controlling Keys in the Optimal Setting

Multi-Client Functional Encryption with Public Inputs and Strong Security
[NPP'24, in submission]

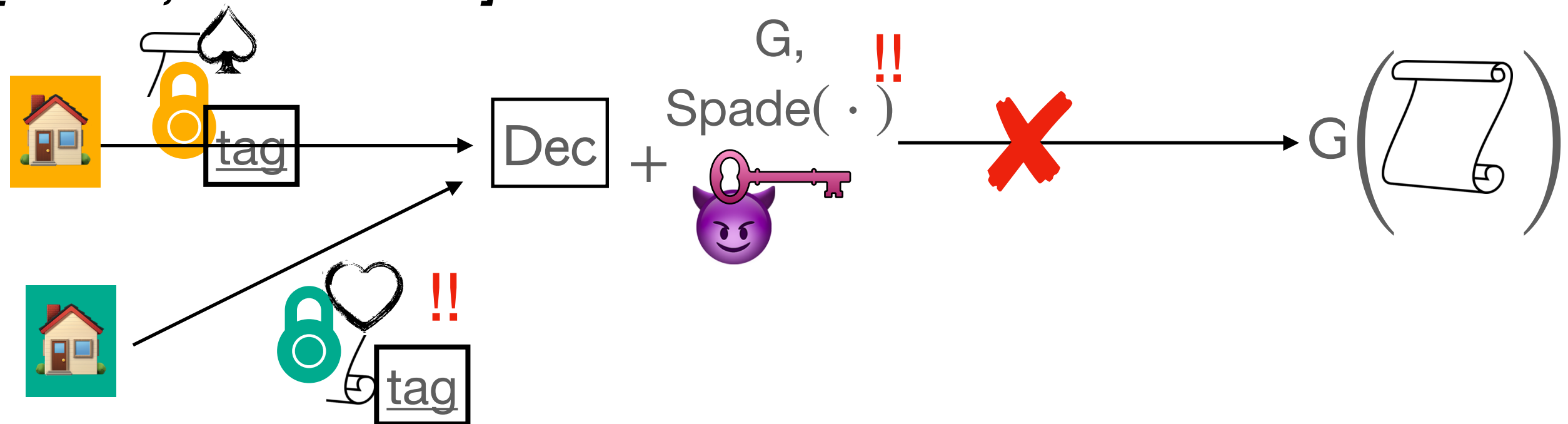


- ✓ Refine Syntax of MCFE:
With **public**-inputs
- ✓ Identify Implications:
MCFE + optimal security
⇒ **public**-key FE

Research Question (3) - Our Result

Controlling Keys in the Optimal Setting

Multi-Client Functional Encryption with Public Inputs and Strong Security
[NPP'24, in submission]



- ✓ Refine Syntax of MCFE:
With **public**-inputs
- ✓ Identify Implications:
MCFE + optimal security
⇒ **public**-key FE

✓ Concret MCFE for
Inner-Products,
Stronger Security
Improving
[NPP22, NPP23, ATY23]

Technical Tools

Dual Pairing Vector Spaces [Okamoto, Takashima'10,12]

Prime-order (additive) bilinear group: $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t, g_1, g_2, g_t, e, q)$, $N \in \mathbb{N}$

 Public vectors  Private vectors

$$e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_t$$
$$(\llbracket a \rrbracket_1, \llbracket b \rrbracket_2) \mapsto \llbracket ab \rrbracket_t$$

$$\mathcal{B} = \begin{pmatrix} \text{Public vectors} \\ \dots \\ \text{Private vectors} \end{pmatrix} \in GL_N(\mathbb{Z}_q)$$

We consider random $\mathcal{B}$ throughout

Technical Tools

Dual Pairing Vector Spaces [Okamoto, Takashima'10,12]

$(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t, g_1, g_2, g_t, e, q), N \in \mathbb{N}$

 Public
vectors  Private
vectors

$$\mathcal{B} = \begin{pmatrix} \text{Public vectors} \\ \dots \\ \text{Private vectors} \end{pmatrix} ; (\mathcal{B}^{-1})^T = \begin{pmatrix} \text{Private vectors} \\ \dots \\ \text{Public vectors} \end{pmatrix} \in GL_N(\mathbb{Z}_q)$$

Technical Tools

Dual Pairing Vector Spaces [Okamoto, Takashima'10,12]

$$(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t, g_1, g_2, g_t, e, q), N \in \mathbb{N}$$

 Public vectors
  Private vectors

$$\begin{array}{ccc}
 \mathcal{B} = \begin{pmatrix} \text{Public} \\ \dots \\ \text{Private} \end{pmatrix} ; (\mathcal{B}^{-1})^\top = \begin{pmatrix} \text{Private} \\ \dots \\ \text{Public} \end{pmatrix} \in GL_N(\mathbb{Z}_q) & & \\
 \downarrow \cdot g_1 & & \downarrow \cdot g_2 \\
 \mathbf{B} = \begin{bmatrix} \text{Public} \\ \dots \\ \text{Private} \end{bmatrix}_1 \in \mathbb{G}_1^{N \times N} ; \mathbf{B}^* = \begin{bmatrix} \text{Private} \\ \dots \\ \text{Public} \end{bmatrix}_2 \in \mathbb{G}_2^{N \times N} & &
 \end{array}$$

Technical Tools

Dual Pairing Vector Spaces [Okamoto, Takashima'10,12]

$(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t, g_1, g_2, g_t, e, q), N \in \mathbb{N}$

 Public vectors  Private vectors

$$\mathbf{B} = \begin{bmatrix} \text{Public vectors} \\ \vdots \\ \text{Private vectors} \end{bmatrix}_1 ; \mathbf{B}^* = \begin{bmatrix} \text{Private vectors} \\ \vdots \\ \text{Public vectors} \end{bmatrix}_2$$

Technical Tools

Dual Pairing Vector Spaces [Okamoto, Takashima'10,12]

$(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t, g_1, g_2, g_t, e, q), N \in \mathbb{N}$

 Public vectors  Private vectors

$$\mathbf{B} = \begin{bmatrix} \text{Public} \\ \vdots \\ \text{Private} \end{bmatrix}_1 ; \quad \mathbf{B}^* = \begin{bmatrix} \text{Private} \\ \vdots \\ \text{Public} \end{bmatrix}_2$$

$(x_1, \dots, x_n) \cdot$

$(x_1 \mid \dots \mid x_n)_{\mathbf{B}} \in \mathbb{G}_1^N$

Technical Tools

Dual Pairing Vector Spaces [Okamoto, Takashima'10,12]

$(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t, g_1, g_2, g_t, e, q), N \in \mathbb{N}$

 Public vectors  Private vectors

$$\mathbf{B} = \begin{bmatrix} \text{Public} \\ \vdots \\ \text{Private} \end{bmatrix}_1 ; \mathbf{B}^* = \begin{bmatrix} \text{Private} \\ \vdots \\ \text{Public} \end{bmatrix}_2$$

$(x_1, \dots, x_n) \cdot$

$$(x_1 \mid \dots \mid x_n)_{\mathbf{B}} \in \mathbb{G}_1^N$$

$(y_1, \dots, y_n) \cdot$

$$(y_1 \mid \dots \mid y_n)_{\mathbf{B}^*} \in \mathbb{G}_2^N$$

Technical Tools

Dual Pairing Vector Spaces [Okamoto, Takashima'10,12]

$$(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t, g_1, g_2, g_t, e, q), N \in \mathbb{N}$$

 Public vectors
  Private vectors

$$\mathbf{B} = \begin{bmatrix} \text{Public} \\ \vdots \\ \text{Private} \end{bmatrix}_1 ; \mathbf{B}^* = \begin{bmatrix} \text{Private} \\ \vdots \\ \text{Public} \end{bmatrix}_2$$

$$(x_1, \dots, x_n) \cdot$$

$$(x_1 \mid \dots \mid x_n)_{\mathbf{B}} \in \mathbb{G}_1^N$$

$$(y_1, \dots, y_n) \cdot$$

$$(y_1 \mid \dots \mid y_n)_{\mathbf{B}^*} \in \mathbb{G}_2^N$$

$\times$

$$\llbracket x_1 y_1 + \dots + x_n y_n \rrbracket_t$$

$$e(\llbracket a \rrbracket_1, \llbracket b \rrbracket_2) \mapsto \llbracket ab \rrbracket_t$$

Basis Changes For Random Bases

 Public vectors  Private vectors

Formal basis change

$$(x_1, x_2)_{\mathbf{U}} \quad \mathbf{U} = \begin{bmatrix} \text{blue} \\ \text{black} \end{bmatrix}_1$$

$$(y_1, y_2)_{\mathbf{U}^*} \quad \mathbf{U}^* = \begin{bmatrix} \text{black} \\ \text{blue} \end{bmatrix}_2$$

Basis Changes For Random Bases

 Public vectors
  Private vectors

$$(x_1, x_2)_U \quad U = \begin{bmatrix} \text{Private} \\ \text{Public} \end{bmatrix}_1$$



$$B = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}$$

$$B = \begin{bmatrix} \text{Private} \\ \text{Public} \end{bmatrix}_1$$

$$(x_1, x_2 - x_1)_B$$

Formal basis change

$$(y_1, y_2)_{U^*} \quad U^* = \begin{bmatrix} \text{Public} \\ \text{Private} \end{bmatrix}_2$$

$$(B^{-1})^T = \begin{bmatrix} 1 & 0 \\ -1 & 1 \end{bmatrix}$$

$$B^* = \begin{bmatrix} \text{Public} \\ \text{Private} \end{bmatrix}_2$$

$$(y_1 + y_2, y_2)_{B^*}$$

Basis Changes For Random Bases

 Public vectors
  Private vectors

Formal basis change

$$(x_1, x_2)_U \quad \mathbf{U} = \begin{bmatrix} \text{Private} \\ \text{Public} \end{bmatrix}_1$$

$$(y_1, y_2)_{U^*} \quad \mathbf{U}^* = \begin{bmatrix} \text{Public} \\ \text{Private} \end{bmatrix}_2$$



$$B = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}$$

$$(B^{-1})^T = \begin{bmatrix} 1 & 0 \\ -1 & 1 \end{bmatrix}$$

$$\mathbf{B} = \begin{bmatrix} \text{Private} \\ \text{Public} \end{bmatrix}_1$$

$$\mathbf{B}^* = \begin{bmatrix} \text{Public} \\ \text{Private} \end{bmatrix}_2$$

$$(x_1, x_2 - x_1)_B$$

$$(y_1 + y_2, y_2)_{B^*}$$

All vectors change
 From $\mathbf{U}$ to $\mathbf{B}$

All vectors change
 From $\mathbf{U}^*$ to $\mathbf{B}^*$

Basis Changes For Random Bases

 Public vectors  Private vectors

Formal basis change

$$(x_1, x_2)_{\mathbf{U}} \quad \mathbf{U} = \begin{bmatrix} \text{blue} \\ \text{black} \end{bmatrix}_1$$

$$(y_1, y_2)_{\mathbf{U}^*} \quad \mathbf{U}^* = \begin{bmatrix} \text{black} \\ \text{blue} \end{bmatrix}_2$$

Basis Changes For Random Bases

 Public vectors
  Private vectors

Formal basis change

$$(x_1, x_2)_U \quad U = \begin{bmatrix} \text{Private} \\ \text{Public} \end{bmatrix}_1$$

$$(y_1, y_2)_{U^*} \quad U^* = \begin{bmatrix} \text{Public} \\ \text{Private} \end{bmatrix}_2$$



$$B = \begin{bmatrix} \text{Private} \\ \text{Public} \end{bmatrix}_1$$

$$B^* = \begin{bmatrix} \text{Public} \\ \text{Private} \end{bmatrix}_2$$

$$(x_1, x_2 - x_1)_B$$

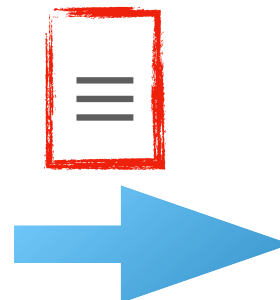
$$(y_1 + y_2, y_2)_{B^*}$$



 Public vectors ;

$$(x_1, x_2)_U$$

$$(y_1, y_2)_{U^*}$$



$$(x_1, x_2 - x_1)_B$$

$$(y_1 + y_2, y_2)_{B^*}$$

 Public Vectors

Basis Changes For Random Bases

From computational hardness

 Public vectors  Private vectors

DDH-tuple $[(a, b, c)]_1$

$$\begin{matrix} (b, c)_U + \\ (\text{rnd}, 0)_B \end{matrix}$$

$$\mathbf{U} = \begin{bmatrix} \text{blue} \\ \text{black} \end{bmatrix}_1;$$

$$(y_1, 0)_{U^*}$$

$$\mathbf{U}^* = \begin{bmatrix} \text{black} \\ \text{blue} \end{bmatrix}_2$$

Basis Changes For Random Bases

From computational hardness

 Public vectors
  Private vectors

DDH-tuple $[(a, b, c)]_1$

$$(b, c)_U + (rnd, 0)_B \quad U = \begin{bmatrix} \text{Private} \\ \text{Public} \end{bmatrix}_1;$$


 $B = \begin{bmatrix} 1 & a \\ 0 & 1 \end{bmatrix} \cdot$
 $B = \begin{bmatrix} \text{Public} \\ \text{Public} \end{bmatrix}_1$

$$(b + rnd, c - ab)_B$$

$$(y_1, 0)_{U^*} \quad U^* = \begin{bmatrix} \text{Public} \\ \text{Private} \end{bmatrix}_2$$

$(B^{-1})^T = \begin{bmatrix} 1 & 0 \\ -a & 1 \end{bmatrix} \cdot$
 $B^* = \begin{bmatrix} \text{Public} \\ \text{Private} \end{bmatrix}_2$

$$(y_1 + a \cdot 0, 0)_{B^*} = (y_1, 0)_{B^*}$$

Basis Changes For Random Bases

From computational hardness

 Public vectors  Private vectors

DDH-tuple $[(a, b, c)]_1$

$$\boxed{(b, c)_U + (rnd, 0)_B} \quad \mathbf{U} = \begin{bmatrix} \text{blue box} \\ \text{black box} \end{bmatrix}_1;$$

↓

$$B = \begin{bmatrix} 1 & a \\ 0 & 1 \end{bmatrix} \cdot \mathbf{B} = \begin{bmatrix} \text{black box} \\ \text{black box} \end{bmatrix}_1$$

↓

$$\boxed{(b + rnd, c - ab)_B}$$

Some can be written in $\mathbf{B}$
Some change from $\mathbf{U}$ to $\mathbf{B}$

$$\boxed{(y_1, 0)_{U^*}} \quad \mathbf{U}^* = \begin{bmatrix} \text{black box} \\ \text{blue box} \end{bmatrix}_2$$

↓

$$(B^{-1})^T = \begin{bmatrix} 1 & 0 \\ -a & 1 \end{bmatrix} \cdot \mathbf{B}^* = \begin{bmatrix} \text{black box} \\ \text{blue box} \end{bmatrix}_2$$

↓

$$\boxed{(y_1 + a \cdot 0, 0)_{B^*} = (y_1, 0)_{B^*}}$$

All vectors
From $\mathbf{U}^*$ to $\mathbf{B}^*$

Basis Changes For Random Bases

From computational hardness

 Public vectors
  Private vectors

DDH-tuple $[(a, b, c)]_1$

$$(b, c)_U + (\text{rnd}, 0)_B \quad U = \begin{bmatrix} \text{blue} \\ \text{black} \end{bmatrix}_1;$$

↓ $B = \begin{bmatrix} 1 & a \\ 0 & 1 \end{bmatrix} \cdot$

$$B = \begin{bmatrix} \text{black} \\ \text{black} \end{bmatrix}_1$$

$$(b + \text{rnd}, c - ab)_B$$

Some can be written in **B**
Some change from **U** to **B**

$$(y_1, 0)_{U^*} \quad U^* = \begin{bmatrix} \text{black} \\ \text{blue} \end{bmatrix}_2$$

$(B^{-1})^T = \begin{bmatrix} 1 & 0 \\ -a & 1 \end{bmatrix} \cdot$

$$B^* = \begin{bmatrix} \text{black} \\ \text{blue} \end{bmatrix}_2$$

$$(y_1 + a \cdot 0, 0)_{B^*} = (y_1, 0)_{B^*}$$

All vectors
From **U*** to **B***

We don't have $[a]_2$

Basis Changes For Random Bases

From computational hardness

 Public vectors
  Private vectors

DDH-tuple $[(a, b, c)]_1$

$$(b, c)_U + (rnd, 0)_B \quad U = \begin{bmatrix} \text{Private} \\ \text{Public} \end{bmatrix}_1;$$


 $B = \begin{bmatrix} 1 & a \\ 0 & 1 \end{bmatrix} \cdot$
 $B = \begin{bmatrix} \text{Public} \\ \text{Public} \end{bmatrix}_1$

$$(b + rnd, c - ab)_B$$

$$(y_1, 0)_{U^*} \quad U^* = \begin{bmatrix} \text{Public} \\ \text{Private} \end{bmatrix}_2$$

$(B^{-1})^T = \begin{bmatrix} 1 & 0 \\ -a & 1 \end{bmatrix} \cdot$
 $B^* = \begin{bmatrix} \text{Public} \\ \text{Private} \end{bmatrix}_2$

$$(y_1 + a \cdot 0, 0)_{B^*} = (y_1, 0)_{B^*}$$

Basis Changes For Random Bases

From computational hardness

 Public vectors  Private vectors

DDH-tuple $[(a, b, c)]_1$

$$\begin{matrix} (b, c)_U + \\ (\text{rnd}, 0)_B \end{matrix}$$

$$\mathbf{U} = \begin{bmatrix} \text{blue} \\ \text{black} \end{bmatrix}_1;$$

$$(y_1, 0)_{U^*}$$

$$\mathbf{U}^* = \begin{bmatrix} \text{black} \\ \text{blue} \end{bmatrix}_2$$

Basis Changes For Random Bases

From computational hardness

 Public vectors  Private vectors

DDH-tuple $[(a, b, c)]_1$ 

$$(b, c)_U + (rnd, 0)_B$$

$$U = \begin{bmatrix} \text{blue box} \\ \text{black box} \end{bmatrix}_1;$$

$$(y_1, 0)_{U^*}$$

$$U^* = \begin{bmatrix} \text{black box} \\ \text{blue box} \end{bmatrix}_2$$

$$B = \begin{bmatrix} \text{black box} \\ \text{black box} \end{bmatrix}_1$$

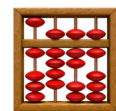
$$(y_1, 0)_{B^*}$$

$$B^* = \begin{bmatrix} \text{black box} \\ \text{blue box} \end{bmatrix}_2$$

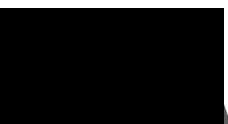
$$(b + rnd, c - ab)_B$$

$$c - ab = 0$$

$$\approx_c c - ab$$

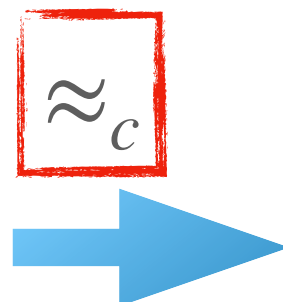


is rand,

 Public vectors ;

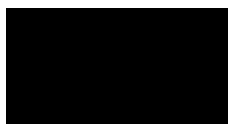
$$(rnd, 0)_B$$

$$(y_1, 0)_{B^*}$$



$$(rnd, rand.)_B$$

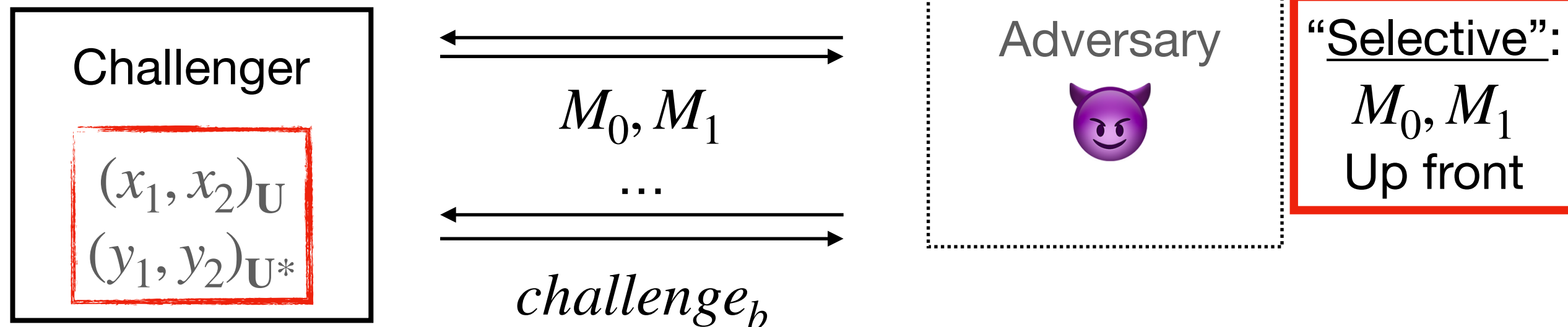
$$(y_1, 0)_{B^*}$$

 Public vectors

A Remark on Security Proofs

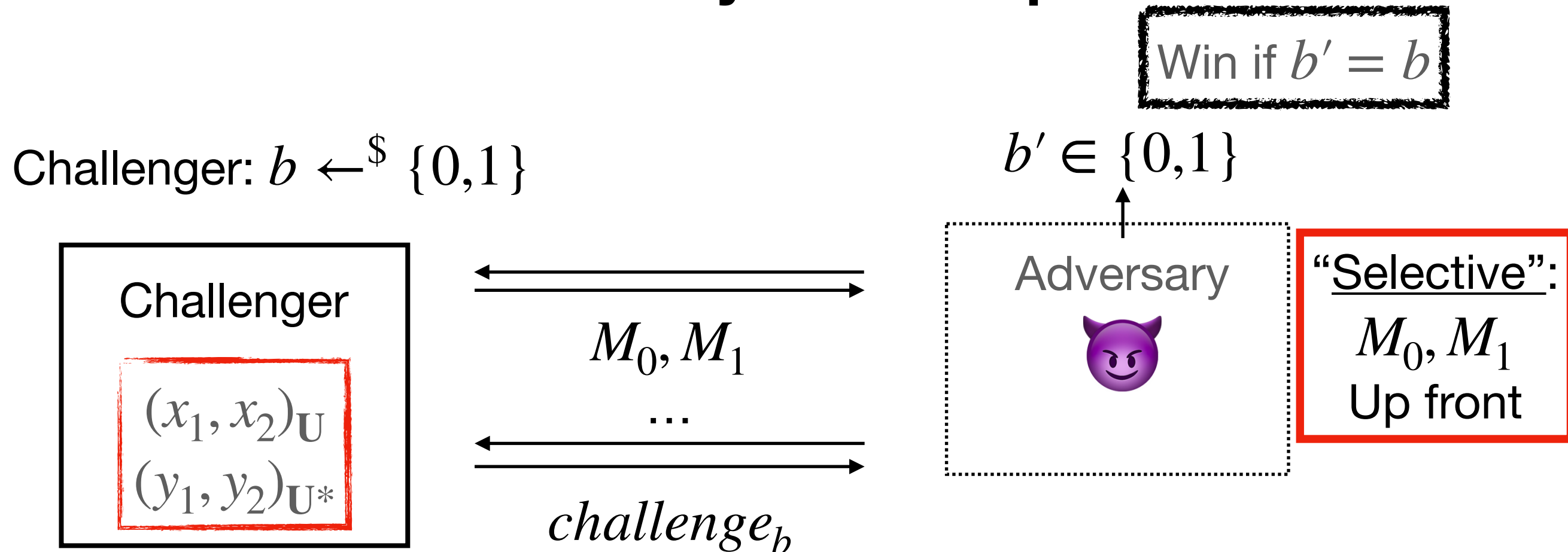
How We Proved Security - DPVS Specific

Challenger: $b \leftarrow^{\$} \{0,1\}$



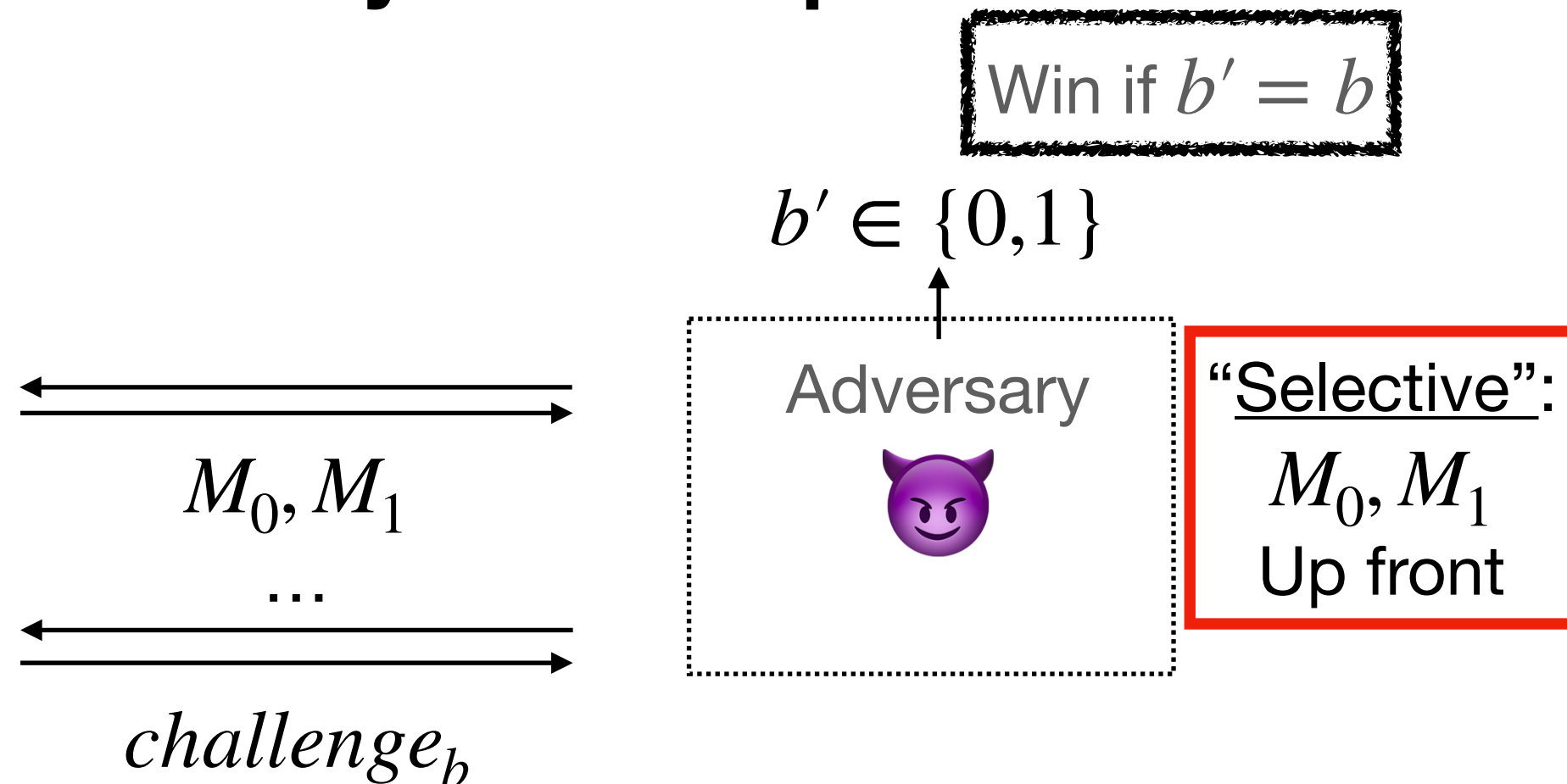
A Remark on Security Proofs

How We Proved Security - DPVS Specific



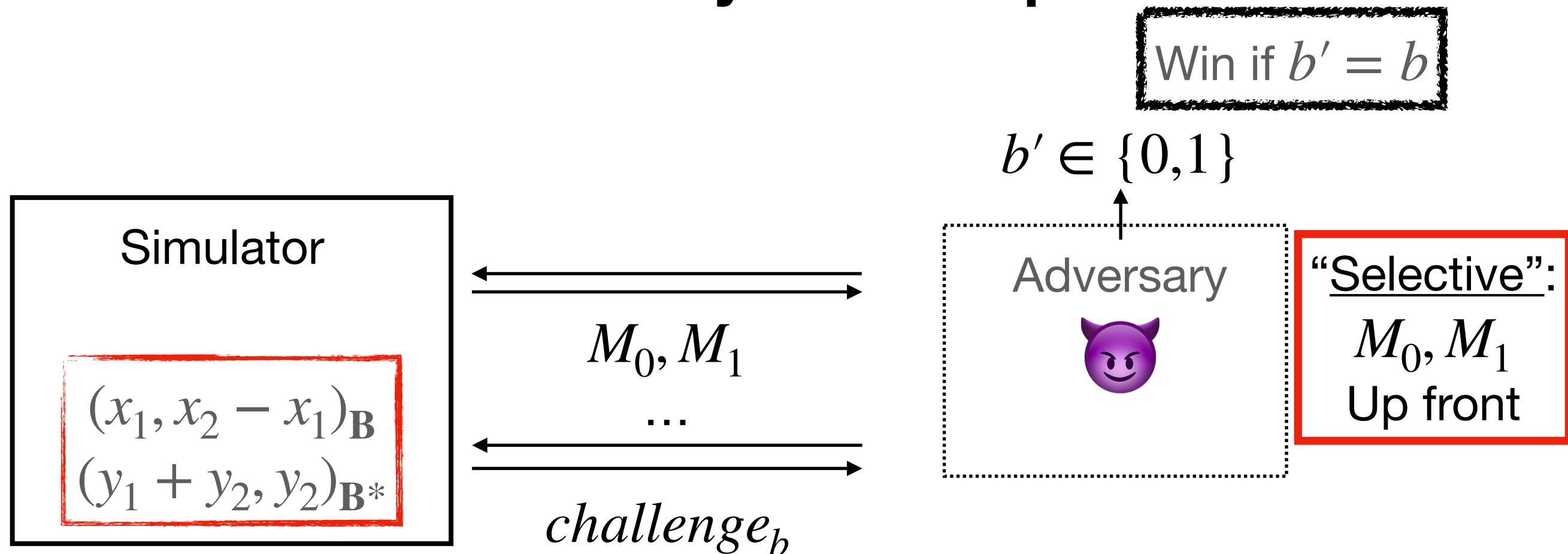
A Remark on Security Proofs

How We Proved Security - DPVS Specific



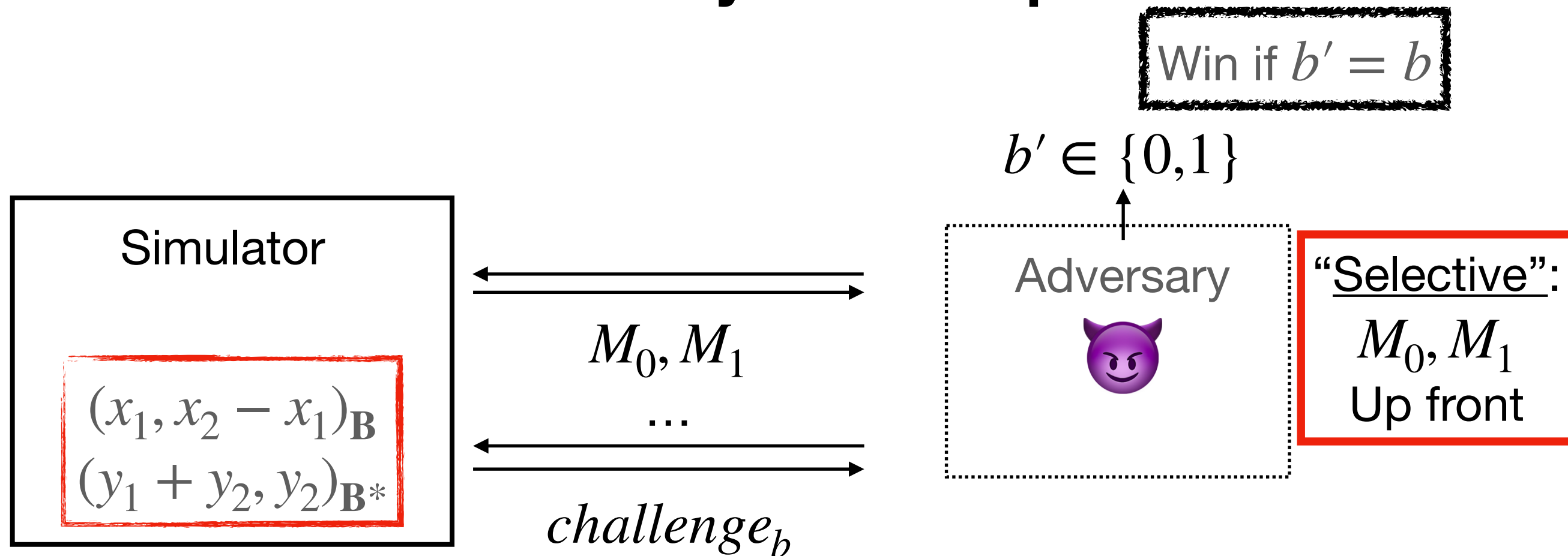
A Remark on Security Proofs

How We Proved Security - DPVS Specific

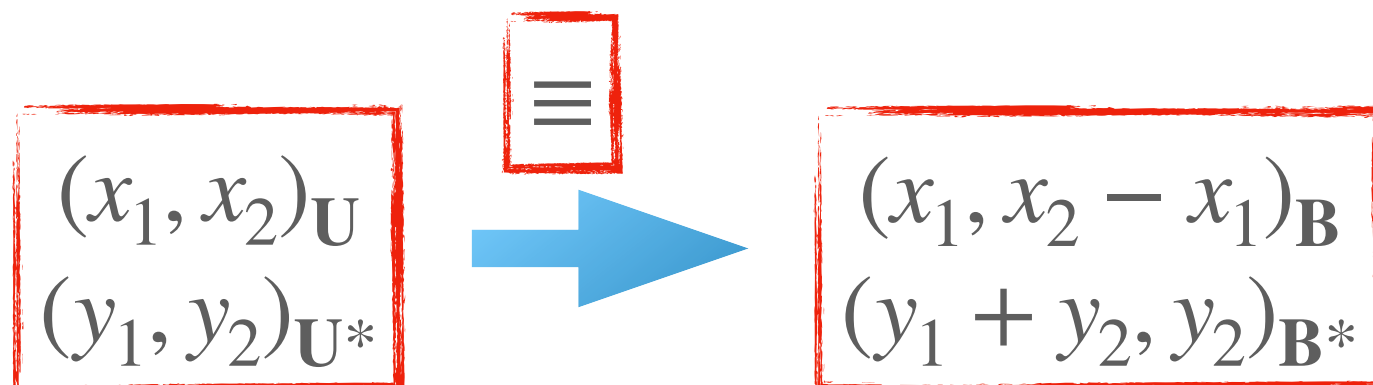


A Remark on Security Proofs

How We Proved Security - DPVS Specific

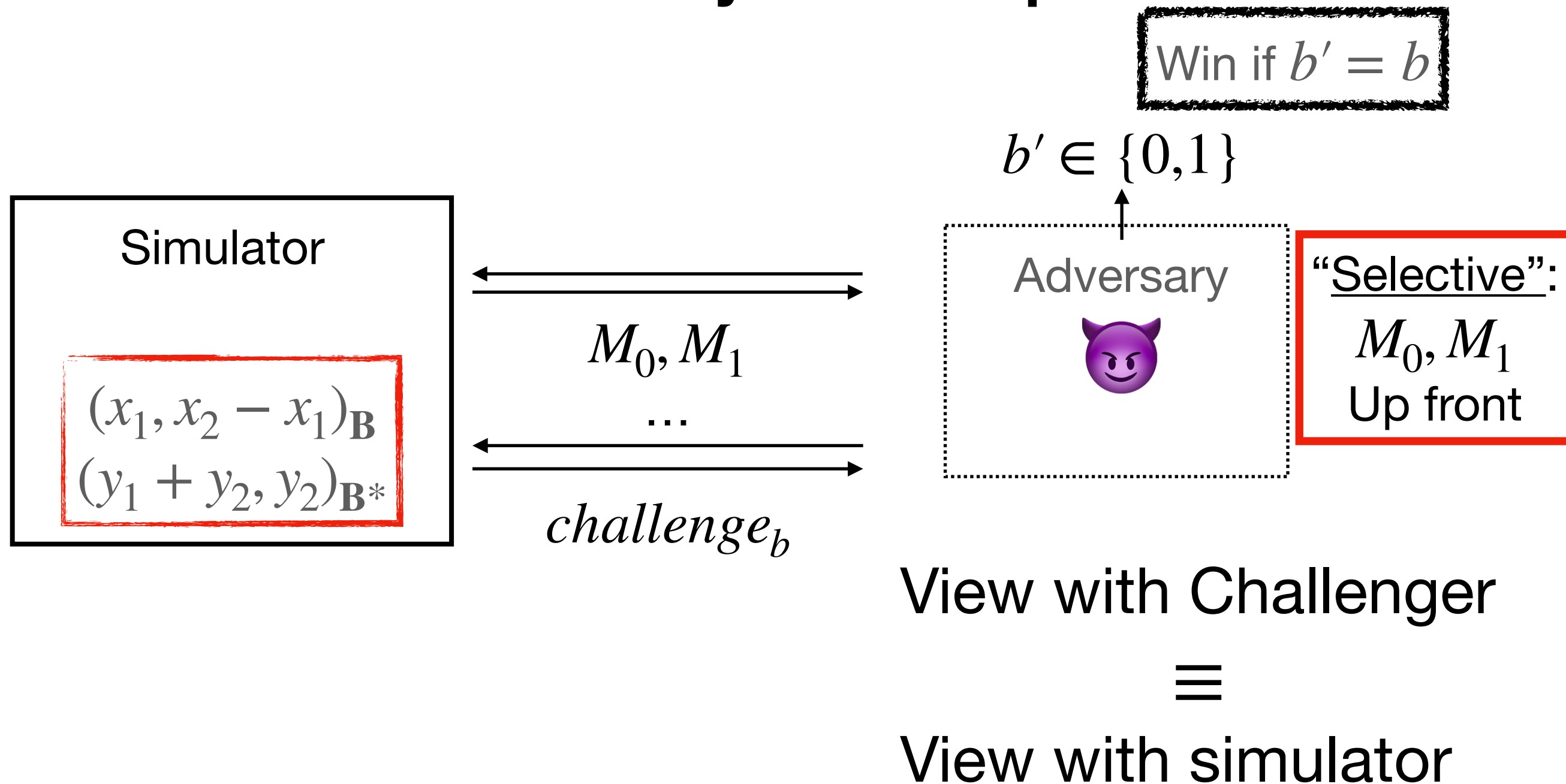


Formal change



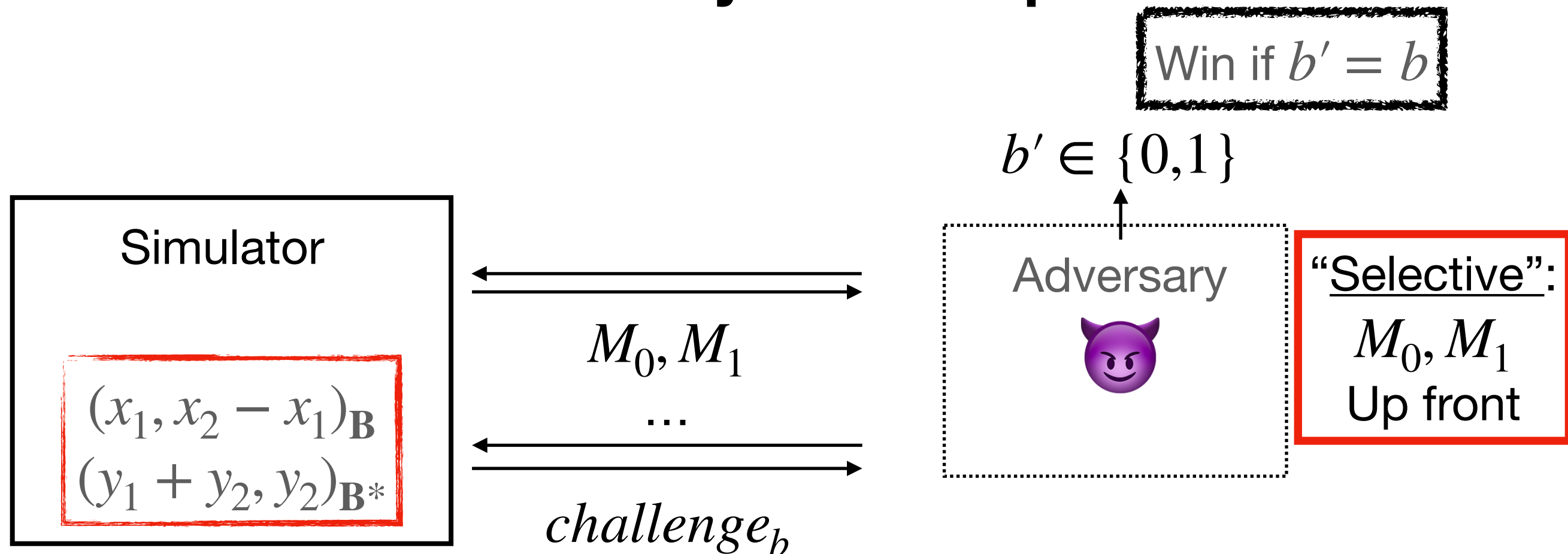
A Remark on Security Proofs

How We Proved Security - DPVS Specific



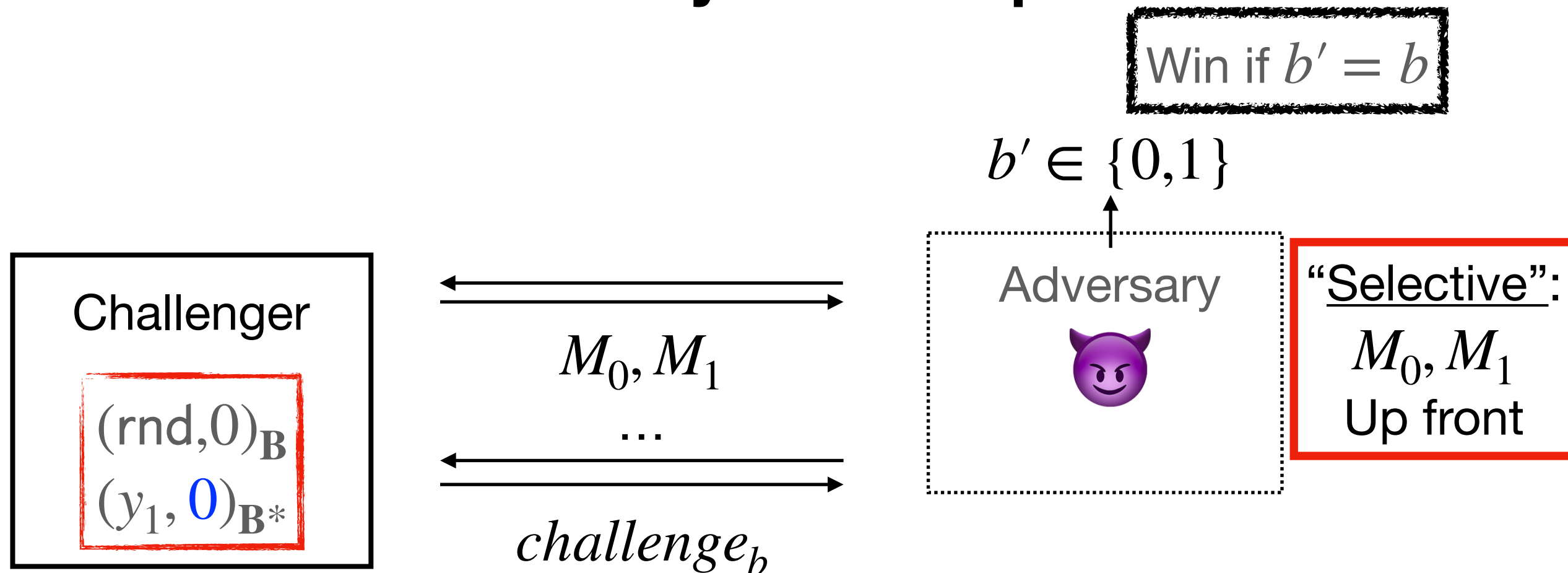
A Remark on Security Proofs

How We Proved Security - DPVS Specific



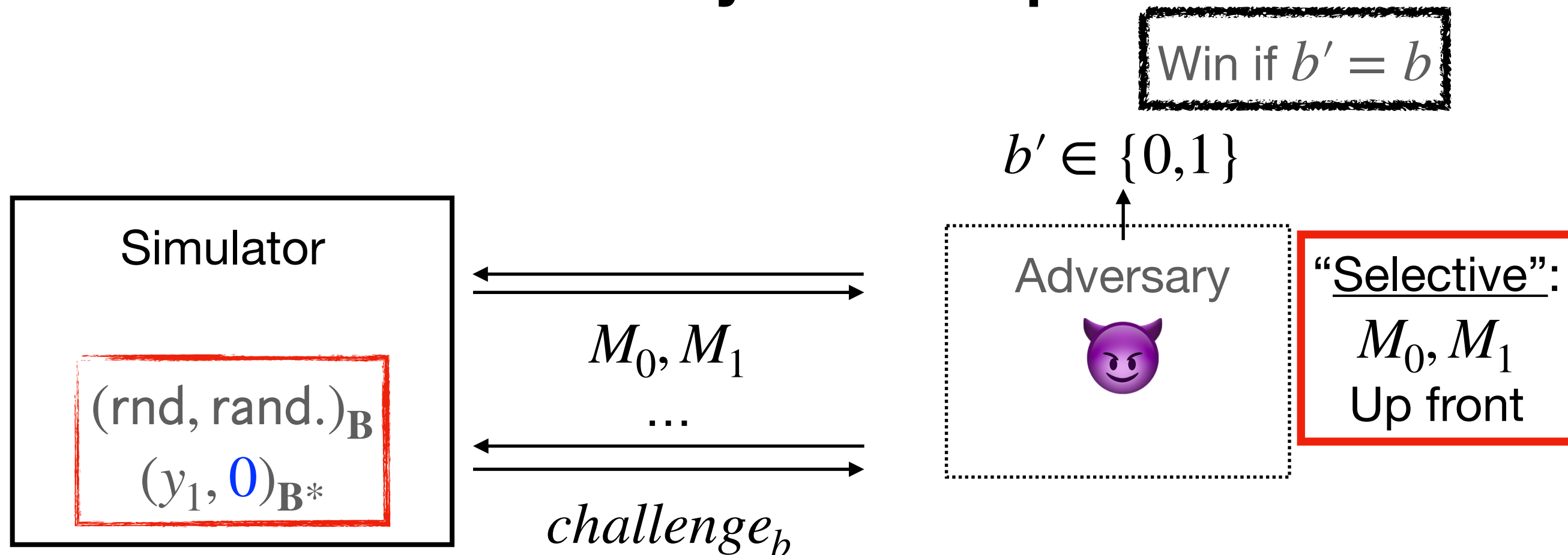
A Remark on Security Proofs

How We Proved Security - DPVS Specific



A Remark on Security Proofs

How We Proved Security - DPVS Specific



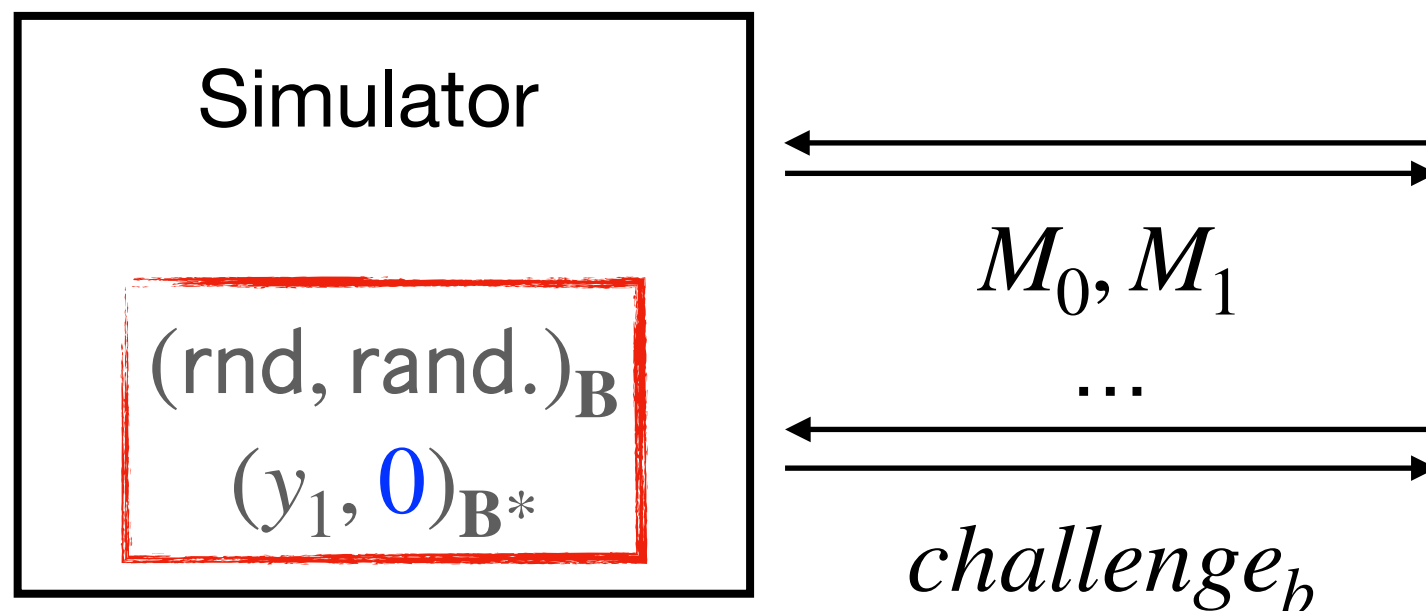
A Remark on Security Proofs

How We Proved Security - DPVS Specific

DDH-tuple $[(a, b, c)]_1$

Win if $b' = b$

$b' \in \{0,1\}$

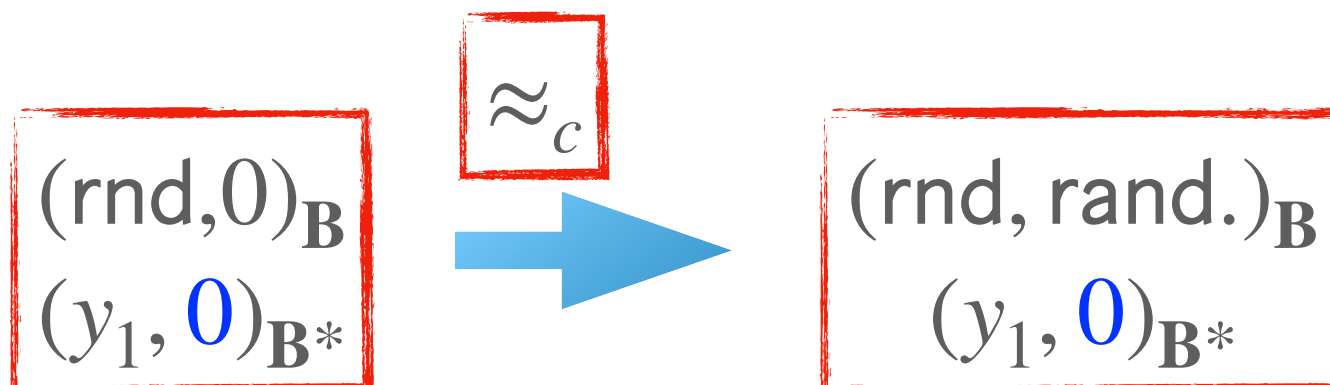


Adversary

“Selective”:

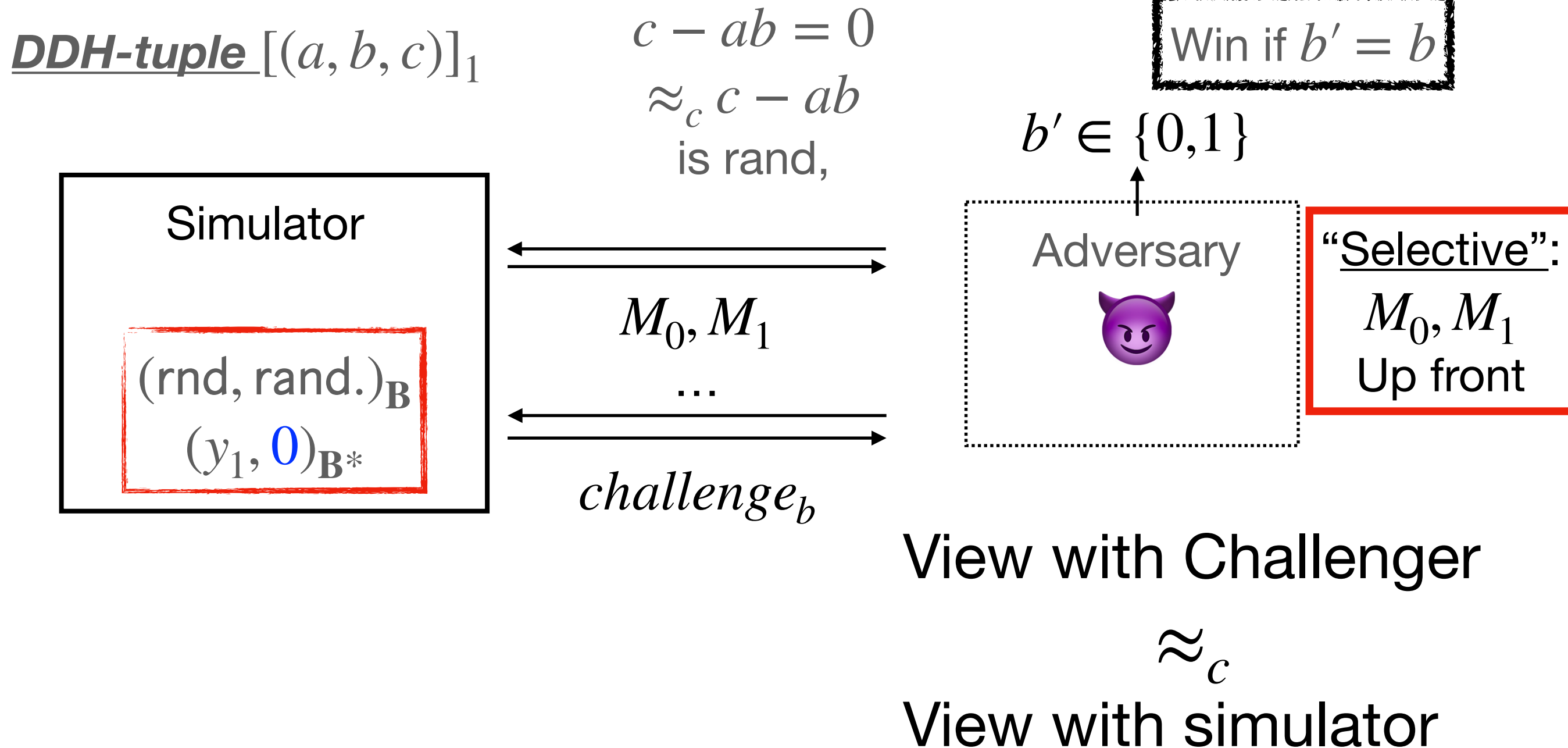
M_0, M_1
Up front

Computational change

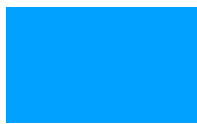


A Remark on Security Proofs

How We Proved Security - DPVS Specific



New Techniques: “Compress-and-Duplicate”

DPVS Specific - Used in [NPP22]  Public vectors  Private vectors

$$\mathcal{B} = \left[\begin{array}{ccc|ccc} \text{Private} & \vec{0} & & & & \\ \hline \vec{0} & \dots & \vec{0} & & \text{Private} & \\ \hline \vec{0} & \vec{0} & & \text{Private} & & \vec{0} \end{array} \right]$$

$$\mathbf{U} = \begin{bmatrix} \text{Private} & \text{Public} \\ \text{Public} & \text{Private} \end{bmatrix}_1 \mathbf{U}^* = \begin{bmatrix} \text{Public} & \text{Private} \\ \text{Private} & \text{Public} \end{bmatrix}_2$$

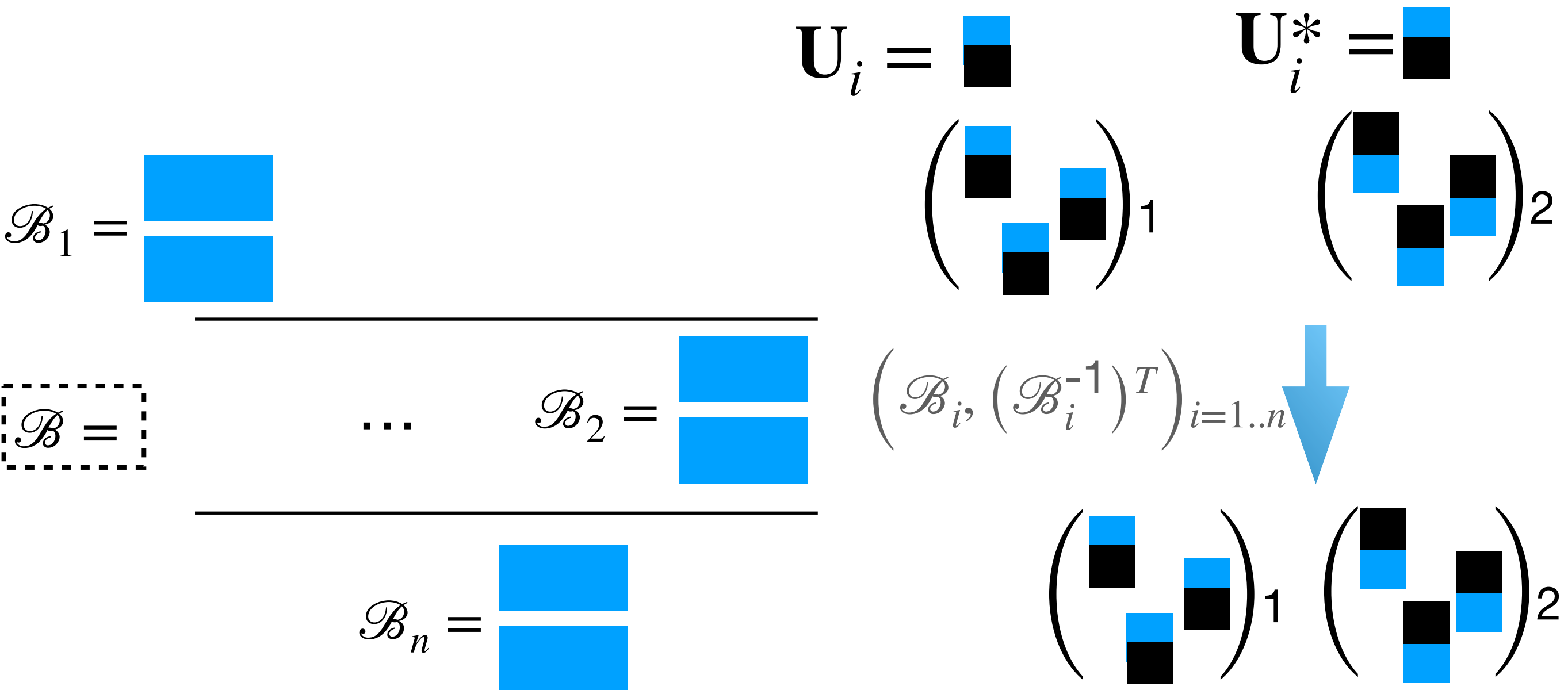
$\mathcal{B}, (\mathcal{B}^{-1})^T \downarrow$

$$\mathbf{B} = \begin{bmatrix} \text{Private} & \text{Public} \\ \text{Public} & \text{Private} \end{bmatrix}_1 \mathbf{B}^* = \begin{bmatrix} \text{Public} & \text{Private} \\ \text{Private} & \text{Public} \end{bmatrix}_2$$

Condition: $\mathcal{B}$ is some row permuted diagonal (block-)matrix

New Techniques: “Compress-and-Duplicate”

In a nutshell: **Duplicate** = “parallelization” within DPVS



Idea: 1 big global by $\mathcal{B}, (\mathcal{B}^{-1})^T$

“=” n smaller on *independent* blocks $\mathcal{B}_i, (\mathcal{B}_i^{-1})^T$

New Techniques: “Compress-and-Duplicate”

In a nutshell: **Compress** = “Block-indexing” in DPVS

$$\mathcal{B} = \left[\begin{array}{ccc|ccc} \text{Blue} & \vec{0} & & & & \\ \text{Blue} & & & & & \\ \hline \vec{0} & \dots & \vec{0} & & \text{Blue} & \\ \hline \vec{0} & \vec{0} & & \text{Blue} & & \vec{0} \end{array} \right]$$

$$\mathbf{U} = \begin{bmatrix} \text{Blue} & \text{Blue} & \text{Blue} \\ \text{Black} & \text{Black} & \text{Black} \end{bmatrix}_1 \mathbf{U}^* = \begin{bmatrix} \text{Black} & \text{Black} & \text{Black} \\ \text{Blue} & \text{Blue} & \text{Blue} \end{bmatrix}_2$$

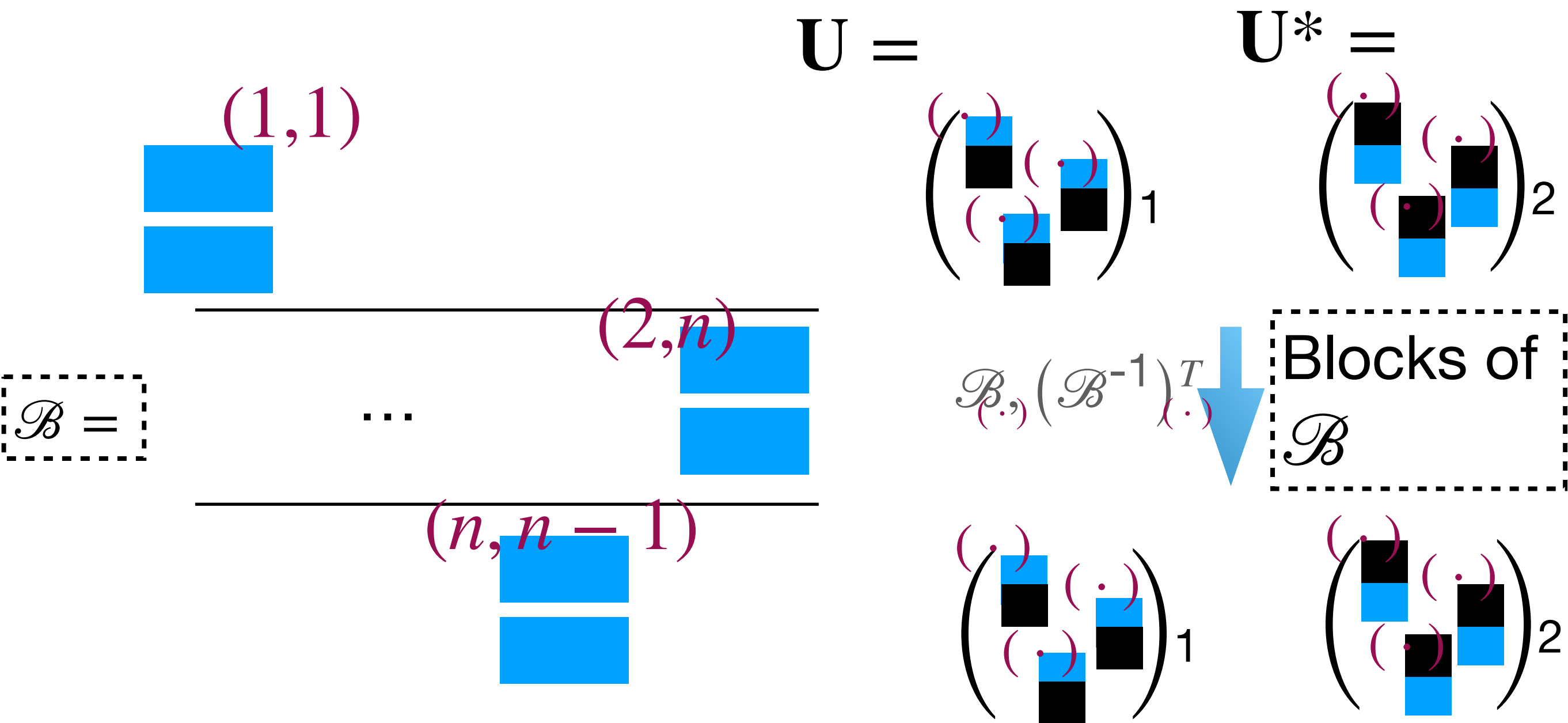
$\mathcal{B}, (\mathcal{B}^{-1})^T \downarrow$

$$\mathbf{B} = \begin{bmatrix} \text{Blue} & \text{Blue} & \text{Blue} \\ \text{Black} & \text{Black} & \text{Black} \end{bmatrix}_1 \mathbf{B}^* = \begin{bmatrix} \text{Black} & \text{Black} & \text{Black} \\ \text{Blue} & \text{Blue} & \text{Blue} \end{bmatrix}_2$$

Condition: $\mathcal{B}$ is some row permuted diagonal (block-)matrix

New Techniques: “Compress-and-Duplicate”

In a nutshell: **Compress** = “Block-indexing” in DPVS



Idea: indexing blocks of $\mathcal{B}$, use **1 block** instead of all at once
 No-matching indices ($\mathcal{B}$ -block, U -block) $\Rightarrow$ Random results

New Techniques: “Compress-and-Duplicate”

In a nutshell: **Compress** = “Independent-Block-indexing”

e.g. change bases

By blocks indexed $(2, n)$

$$\mathbf{U} = \text{Diag-block} \left(\begin{bmatrix} n, 0 \\ 0, -2 \end{bmatrix}^1, \begin{array}{c} \text{blue} \\ \text{black} \end{array} \right)$$

$$\mathcal{B} = \text{Diag-block} \left(\begin{bmatrix} 2, 0 \\ 0, n \end{bmatrix}, \begin{array}{c} \text{blue} \\ \text{blue} \end{array} \right)$$

$$\mathbf{U} = \begin{pmatrix} \begin{pmatrix} \cdot \end{pmatrix} \begin{pmatrix} \cdot \end{pmatrix} \\ \begin{pmatrix} \cdot \end{pmatrix} \begin{pmatrix} \cdot \end{pmatrix} \end{pmatrix}_1 \quad \begin{pmatrix} \begin{pmatrix} \cdot \end{pmatrix} \begin{pmatrix} \cdot \end{pmatrix} \\ \begin{pmatrix} \cdot \end{pmatrix} \begin{pmatrix} \cdot \end{pmatrix} \end{pmatrix}_2$$

$\mathcal{B}_{(\cdot)} (\mathcal{B}^{-1})_{(\cdot)}^T$ 

$$\begin{pmatrix} \begin{pmatrix} \cdot \end{pmatrix} \begin{pmatrix} \cdot \end{pmatrix} \\ \begin{pmatrix} \cdot \end{pmatrix} \begin{pmatrix} \cdot \end{pmatrix} \end{pmatrix}_1 \quad \begin{pmatrix} \begin{pmatrix} \cdot \end{pmatrix} \begin{pmatrix} \cdot \end{pmatrix} \\ \begin{pmatrix} \cdot \end{pmatrix} \begin{pmatrix} \cdot \end{pmatrix} \end{pmatrix}_2$$

New Techniques: “Compress-and-Duplicate”

In a nutshell: **Compress** = “Independent-Block-indexing”

e.g. change bases

By blocks indexed $(2, n)$

$$\mathbf{U} = \text{Diag-block} \left(\begin{bmatrix} n, 0 \\ 0, -2 \end{bmatrix} \mathbf{1}, \begin{array}{c} \text{blue} \\ \text{black} \end{array} \right)$$

$$\mathcal{B} = \text{Diag-block} \left(\begin{bmatrix} 2, 0 \\ 0, n \end{bmatrix}, \begin{array}{c} \text{blue} \\ \text{blue} \end{array} \right)$$

$$\mathbf{B} = \mathcal{B} \cdot \begin{array}{c} \text{blue} \\ \text{black} \end{array} = \text{Diag-block} \left(\begin{bmatrix} 2n, 0 \\ 0, -2n \end{bmatrix} \mathbf{1}, \begin{array}{c} \text{blue} \\ \text{blue} \end{array} \cdot \begin{array}{c} \text{blue} \\ \text{black} \end{array} \right)$$

New Techniques: “Compress-and-Duplicate”

In a nutshell: **Compress** = “Independent-Block-indexing”

e.g. change bases

By blocks indexed $(2,n)$

U

$\mathcal{B}$

$$\mathbf{B} = \mathcal{B} \cdot \begin{array}{|c|} \hline \text{blue} \\ \hline \text{black} \\ \hline \end{array} = \text{Diag-block} \left(\begin{bmatrix} 2n & 0 \\ 0 & -2n \end{bmatrix}^1, \begin{array}{|c|} \hline \text{blue} \\ \hline \text{blue} \\ \hline \end{array} \cdot \begin{array}{|c|} \hline \text{blue} \\ \hline \text{black} \\ \hline \end{array} \right)$$

New Techniques: “Compress-and-Duplicate”

In a nutshell: **Compress** = “Independent-Block-indexing”

e.g. change bases

By blocks indexed $(2,n)$

U

Encode a vector $\vec{x}$
By $(\text{rand}, \text{rand}, \vec{x}) \cdot \mathbf{B}$

$$= \left[2n\text{rand}, -2n\text{rand} \quad , \vec{x} \cdot \begin{bmatrix} \text{blue} & \text{blue} \\ \text{blue} & \text{black} \end{bmatrix} \right]_1$$

Dually, in $\mathbf{B}^*$ we can
encode $\vec{y}$ by

$$= \left[-2n\widehat{\text{rand}}, 2n\widehat{\text{rand}} \quad , \vec{y} \cdot \begin{bmatrix} \text{blue} & \text{black} \\ \text{blue} & \text{blue} \end{bmatrix} \right]_2$$

$\mathcal{B}$

$$\mathbf{B} = \mathcal{B} \cdot \begin{bmatrix} \text{blue} & \text{black} \end{bmatrix} = \text{Diag-block} \left(\begin{bmatrix} 2n & 0 \\ 0 & -2n \end{bmatrix}_1, \begin{bmatrix} \text{blue} & \text{blue} \\ \text{blue} & \text{black} \end{bmatrix} \right)$$

New Techniques: “Compress-and-Duplicate”

In a nutshell: **Compress** = “Independent-Block-indexing”

e.g. change bases

By blocks indexed $(2,n)$

U

Encode a vector $\vec{x}$
By $(\text{rand}, \text{rand}, \vec{x}) \cdot \mathbf{B}$

$$= \left[2n\text{rand}, -2n\text{rand} \quad , \vec{x} \cdot \begin{bmatrix} \text{blue} & \text{blue} \\ \text{blue} & \text{black} \end{bmatrix} \right]_1$$

Dually, in $\mathbf{B}^*$ we can
encode $\vec{y}$ by

$$= \left[-2n\widehat{\text{rand}}, 2n\widehat{\text{rand}} \quad , \vec{y} \cdot \begin{bmatrix} \text{blue} & \text{black} \\ \text{blue} & \text{blue} \end{bmatrix} \right]_2$$

$\mathcal{B}$

✓ $\Rightarrow$ DPVS-product
Same indices
Cancels aux coord's

$$\mathbf{B} = \mathcal{B} \cdot \begin{bmatrix} \text{blue} & \text{black} \end{bmatrix} = \text{Diag-block} \left(\begin{bmatrix} 2n & 0 \\ 0 & -2n \end{bmatrix}_1, \begin{bmatrix} \text{blue} & \text{blue} \\ \text{blue} & \text{black} \end{bmatrix} \right)$$

New Techniques: “Compress-and-Duplicate”

In a nutshell: **Compress** = “Independent-Block-indexing”

e.g. change bases

By blocks indexed $(2,n)$

U

Encode a vector $\vec{x}$

By $(\text{rand}, \text{rand}, \vec{x}) \cdot \mathbf{B}$

$$= \left[2n\text{rand}, -2n\text{rand} \quad , \vec{x} \cdot \begin{bmatrix} \text{blue} & \text{blue} \\ \text{blue} & \text{black} \end{bmatrix} \right]_1$$

Dually, in $\mathbf{B}^*$ we can

encode $\vec{y}$ by

$$= \left[-2n\widehat{\text{rand}}, 2n\widehat{\text{rand}} \quad , \vec{y} \cdot \begin{bmatrix} \text{blue} & \text{black} \\ \text{blue} & \text{blue} \end{bmatrix} \right]_2$$

✓ $\Rightarrow$ DPVS-product

Different indices

aux coord's

Do NOT cancel & gives random

$\mathcal{B}$

$$\mathbf{B} = \mathcal{B} \cdot \begin{bmatrix} \text{blue} & \text{black} \end{bmatrix} = \text{Diag-block} \left(\begin{bmatrix} 2n & 0 \\ 0 & -2n \end{bmatrix}_1, \begin{bmatrix} \text{blue} & \text{blue} \\ \text{blue} & \text{black} \end{bmatrix} \right)$$

New Techniques: “Compress-and-Duplicate”

In a nutshell: **Compress** = “Independent-Block-indexing”

e.g. change bases

By blocks indexed $(2,n)$

U

$\mathcal{B}$

$$\mathbf{B} = \mathcal{B} \cdot \begin{array}{|c|} \hline \text{blue} \\ \hline \text{black} \\ \hline \end{array} = \text{Diag-block} \left(\begin{bmatrix} 2n & 0 \\ 0 & -2n \end{bmatrix}^1, \begin{array}{|c|} \hline \text{blue} \\ \hline \text{blue} \\ \hline \end{array} \cdot \begin{array}{|c|} \hline \text{blue} \\ \hline \text{black} \\ \hline \end{array} \right)$$

New Techniques: “Compress-and-Duplicate”

In a nutshell: **Compress** = “Independent-Block-indexing”

e.g. change bases

By blocks indexed $(2, n)$

$$\mathbf{U} = \text{Diag-block} \left(\begin{bmatrix} n, 0 \\ 0, -2 \end{bmatrix}^1, \begin{array}{c} \text{blue} \\ \text{black} \end{array} \right)$$

$$\mathcal{B} = \text{Diag-block} \left(\begin{bmatrix} 2, 0 \\ 0, n \end{bmatrix}, \begin{array}{c} \text{blue} \\ \text{blue} \end{array} \right)$$

$$\mathbf{B} = \mathcal{B} \cdot \begin{array}{c} \text{blue} \\ \text{black} \end{array} = \text{Diag-block} \left(\begin{bmatrix} 2n, 0 \\ 0, -2n \end{bmatrix}^1, \begin{array}{c} \text{blue} \\ \text{blue} \end{array} \cdot \begin{array}{c} \text{blue} \\ \text{black} \end{array} \right)$$

New Techniques: “Compress-and-Duplicate”

In a nutshell: **Compress** = “Independent-Block-indexing”

e.g. change bases

By blocks indexed $(2,n)$

$\mathcal{U}$ } Size of **1** block + $\Theta(1)$
 $\mathcal{B}$ } Instead **n** blocks

$$\mathbf{B} = \mathcal{B} \cdot \begin{array}{|c|} \hline \text{blue} \\ \hline \text{black} \\ \hline \end{array} = \text{Diag-block} \left(\begin{bmatrix} 2n & 0 \\ 0 & -2n \end{bmatrix}^1, \begin{array}{|c|c|} \hline \text{blue} & \text{blue} \\ \hline \text{blue} & \text{black} \\ \hline \end{array} \right)$$

New Techniques: “Compress-and-Duplicate”

In a nutshell: **Compress** = “Independent-Block-indexing”

e.g. change bases

By blocks indexed $(2, n)$

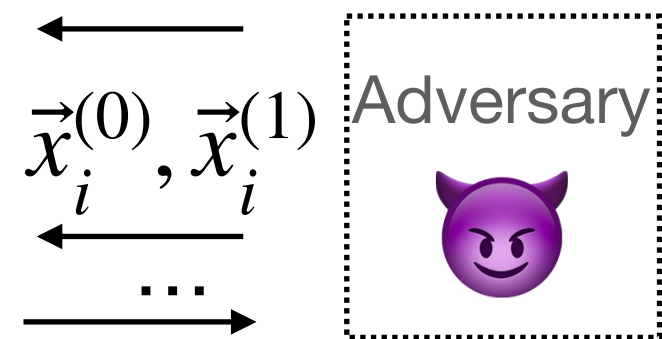
$\mathcal{U}$ } Size of **1** block + $\Theta(1)$
 $\mathcal{B}$ } Instead **n** blocks

New Techniques: DPVS with Complexity Leveraging

In a nutshell: With Formal Changes, **Guesses are Free**
Used in [NPP24, NPS24]

Selective

simulator : guess $(\vec{x}_i^{(0)}, \vec{x}_i^{(1)})$ from 

$$\text{View}_0^{sel} \equiv \text{View}_1^{sel} \equiv \dots \equiv \underline{\text{View}_k^{sel}}$$


Start



Guessing

Challenges $\vec{x}_i^{(0)}, \vec{x}_i^{(1)}$

New Techniques: DPVS with Complexity Leveraging

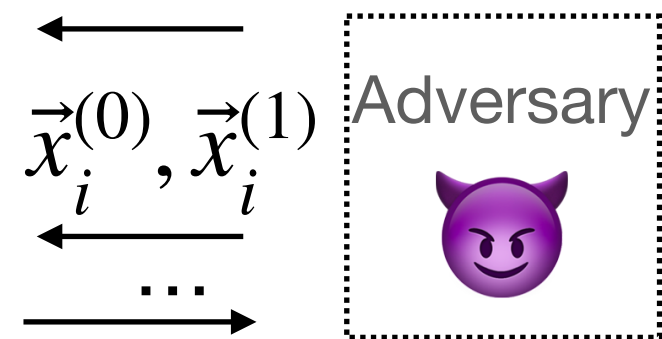
In a nutshell: With Formal Changes, **Guesses are Free**
Used in [NPP24, NPS24]

Selective

simulator : guess $(\vec{x}_i^{(0)}, \vec{x}_i^{(1)})$ from 

$$\text{View}_0^{sel} \equiv \text{View}_1^{sel} \equiv \dots \equiv \underline{\text{View}_k^{sel}}$$

In View_k^{sel} : Some critical step in the proof



Start

$\Rightarrow$

Guessing
Challenges $\vec{x}_i^{(0)}, \vec{x}_i^{(1)}$

New Techniques: DPVS with Complexity Leveraging

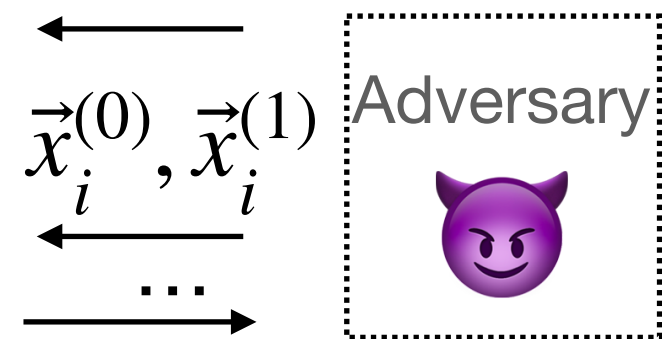
In a nutshell: With Formal Changes, **Guesses are Free**
Used in [NPP24, NPS24]

Selective

simulator : guess $(\vec{x}_i^{(0)}, \vec{x}_i^{(1)})$ from 

$$\text{View}_0^{sel} \equiv \text{View}_1^{sel} \equiv \dots \equiv \underline{\text{View}_k^{sel}}$$

In View_k^{sel} : Some critical step in the proof



New Techniques: DPVS with Complexity Leveraging

In a nutshell: With Formal Changes, **Guesses are Free**

Used in [NPP24, NPS24]

All is Formal basis
Changes DPVS

Selective

simulator : guess $(\vec{x}_i^{(0)}, \vec{x}_i^{(1)})$ from 

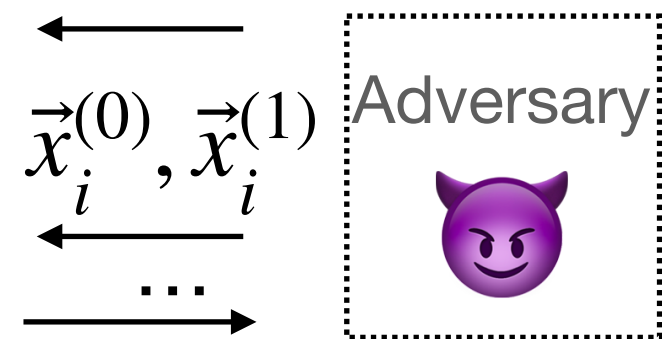
$$\text{View}_0^{sel} \equiv \text{View}_1^{sel} \equiv \dots \equiv \text{View}_k^{sel}$$

Perfectly

Perfectly

Perfectly

In View_k^{sel} : Some critical step in the proof



New Techniques: DPVS with Complexity Leveraging

In a nutshell: With Formal Changes, **Guesses are Free**

Used in [NPP24, NPS24]

All is Formal basis
Changes DPVS

Selective

simulator : guess $(\vec{x}_i^{(0)}, \vec{x}_i^{(1)})$ from 

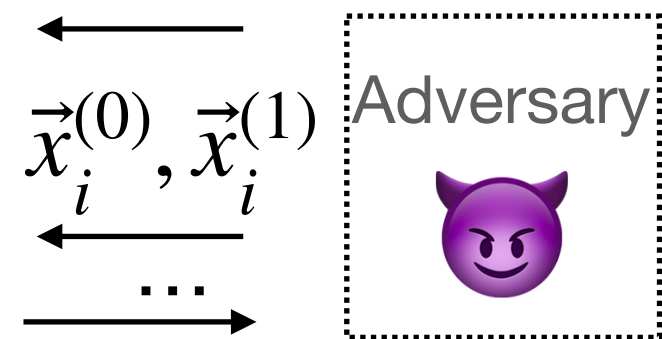
$$\text{View}_0^{sel} \equiv \text{View}_1^{sel} \equiv \dots \equiv \text{View}_k^{sel}$$

Perfectly

Perfectly

Perfectly

In View_k^{sel} : Some critical step in the proof



Start

$\Rightarrow$

Loss factor over choices
of $\vec{x}_i^{(0)}, \vec{x}_i^{(1)}$ in prob of correct

New Techniques: DPVS with Complexity Leveraging

In a nutshell: With Formal Changes, **Guesses are Free**
Used in [NPP24, NPS24]

Selective

simulator : guess $(\vec{x}_i^{(0)}, \vec{x}_i^{(1)})$ from 

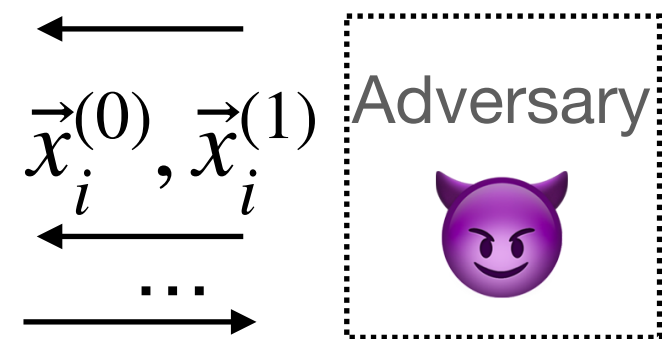
$$\text{View}_0^{sel} \equiv \text{View}_1^{sel} \equiv \dots \equiv \underline{\text{View}_k^{sel}}$$

Perfectly

Perfectly

Perfectly

In View_k^{sel} : Some critical step in the proof



Start

$\Rightarrow$

Loss factor over choices
of $\vec{x}_i^{(0)}, \vec{x}_i^{(1)}$ in prob of correct

New Techniques: DPVS with Complexity Leveraging

In a nutshell: With Formal Changes, **Guesses are Free**
Used in [NPP24, NPS24]

$$\begin{aligned} 0 &\leq \text{diff}(\text{View}_0^{\text{sel}}; \text{View}_k^{\text{sel}}) \\ &\leq \sum_i \text{diff}(\text{View}_i^{\text{sel}}; \text{View}_{i+1}^{\text{sel}}) = 0 \end{aligned}$$

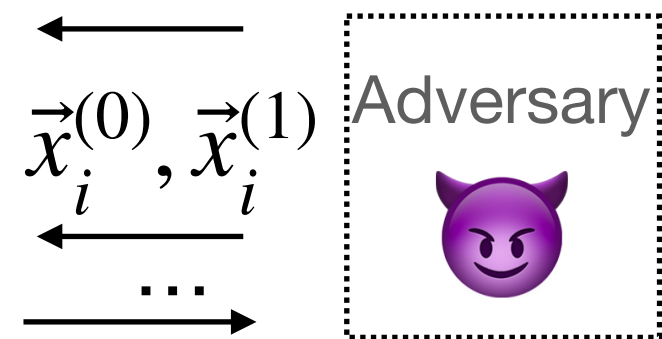
Selective

simulator : guess $(\vec{x}_i^{(0)}, \vec{x}_i^{(1)})$ from 

$$\text{View}_0^{\text{sel}} \equiv \text{View}_1^{\text{sel}} \equiv \dots \equiv \text{View}_k^{\text{sel}}$$

Perfectly Perfectly Perfectly

In $\text{View}_k^{\text{sel}}$: Some critical step in the proof



Start

$\Rightarrow$

Loss factor over choices
of $\vec{x}_i^{(0)}, \vec{x}_i^{(1)}$ in prob of correct

$$\times \text{diff}(\text{View}_0; \text{View}_k) = 0$$

New Techniques: DPVS with Complexity Leveraging

In a nutshell: With Formal Changes, **Guesses are Free**

Used in [NPP24, NPS24]

Selective

simulator : guess $(\vec{x}_i^{(0)}, \vec{x}_i^{(1)})$ from 

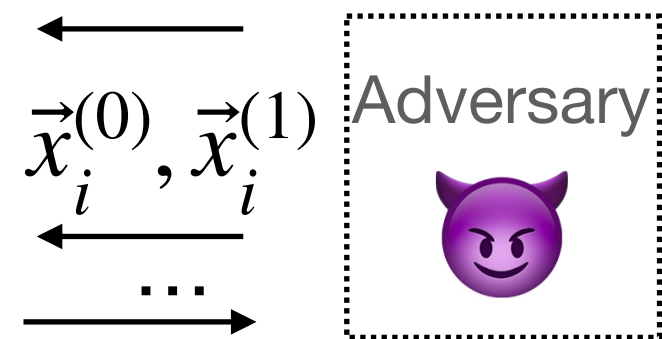
$$\text{View}_0^{sel} \equiv \text{View}_1^{sel} \equiv \dots \equiv \text{View}_k^{sel}$$

Perfectly

Perfectly

Perfectly

In View_k^{sel} : Some critical step in the proof



Start

$\Rightarrow$

Loss factor over choices
of $\vec{x}_i^{(0)}, \vec{x}_i^{(1)}$ in prob of correct

$$\times \text{diff}(\text{View}_0; \text{View}_k) = 0$$

New Techniques: DPVS with Complexity Leveraging

In a nutshell: With Formal Changes, **Guesses are Free**
Used in [NPP24, NPS24]

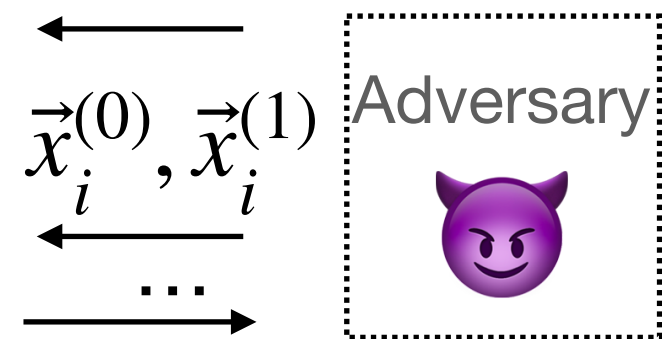
Adaptive

Simulator: no guesses

$$\text{View}_0^{sel} \equiv \text{View}_1^{sel} \equiv \dots \equiv \underline{\text{View}_k^{sel}}$$

Perfectly Perfectly Perfectly

In View_k^{sel} : Some critical step in the proof



Start

$\Rightarrow$

Loss factor over choices
of $\vec{x}_i^{(0)}, \vec{x}_i^{(1)}$ in prob of correct

$$\times \text{diff}(\text{View}_0; \text{View}_k) = 0$$

New Techniques: DPVS with Complexity Leveraging

In a nutshell: With Formal Changes, **Guesses are Free**
Used in [NPP24, NPS24]

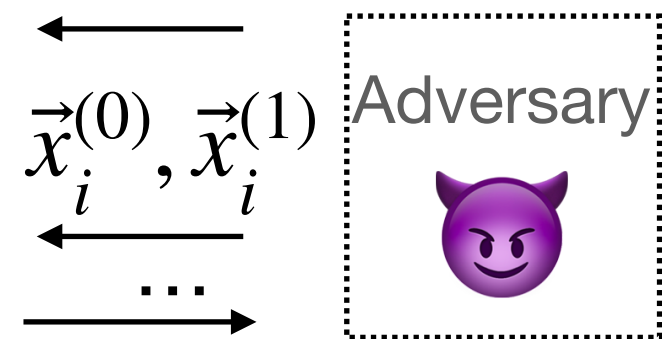
Adaptive

Simulator: no guesses

$$\text{View}_0^{sel} \equiv \text{View}_1^{sel} \equiv \dots \equiv \underline{\text{View}_k^{sel}}$$

Perfectly Perfectly Perfectly

In View_k^{sel} : Some critical step in the proof



New Techniques: DPVS with Complexity Leveraging

In a nutshell: With Formal Changes, **Guesses are Free**
Used in [NPP24, NPS24]

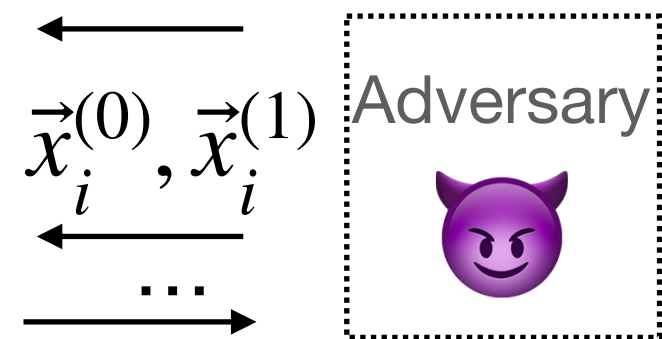
Adaptive

Simulator: no guesses

Perfectly

Perfectly

Perfectly



New Techniques: DPVS with Complexity Leveraging

In a nutshell: With Formal Changes, **Guesses are Free**
Used in [NPP24, NPS24]

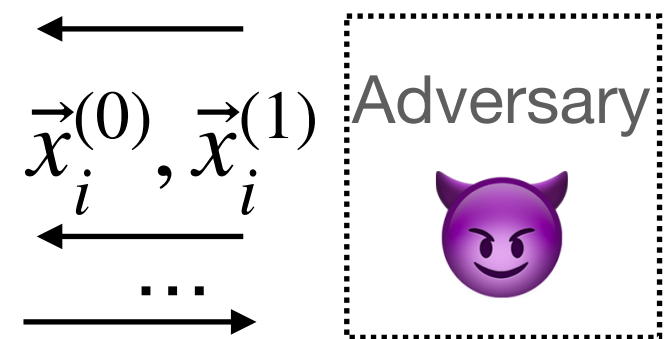
Adaptive

Simulator: no guesses

$\text{View}_0 \equiv \text{View}_1 \equiv \dots \equiv \underline{\text{View}_k}$

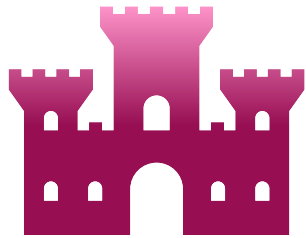
Perfectly Perfectly Perfectly

In View_k : Some critical step in the proof

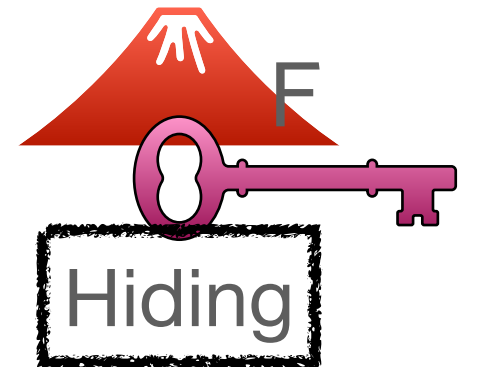
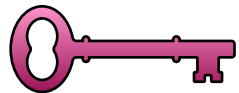


Before moving on - A Recap

Map of projects/Qs ; important tool (DPVS)



Func: G,
Policy: Spade(·)



[CDGPP'18]

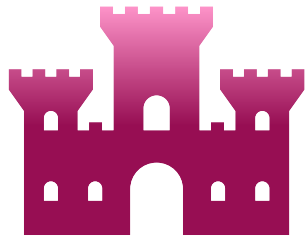
Corrupt(i)

$\Rightarrow x_{0,i} = x_{1,i}$

🤔 Justified?

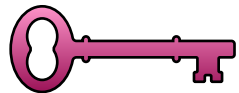
Before moving on - A Recap

Map of projects/Qs ; important tool (DPVS)



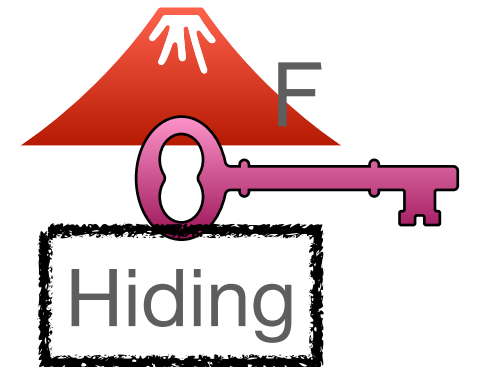
Func: G,

Policy: Spade(·)



(1)

Key-
Control



[CDGPP'18]

Corrupt(i)

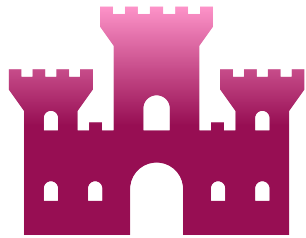
$\Rightarrow x_{0,i} = x_{1,i}$



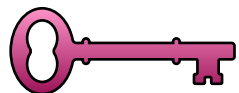
Justified?

Before moving on - A Recap

Map of projects/Qs ; important tool (DPVS)



Func: G ,
Policy: $\text{Spade}(\cdot)$



(1)

Key-
Control

(2)

Optimal
Security



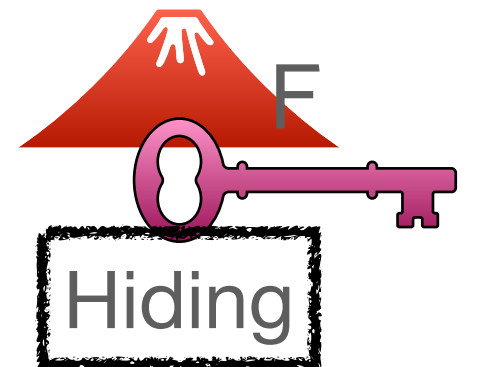
[CDGPP'18]

$\text{Corrupt}(i)$

$\Rightarrow x_{0,i} = x_{1,i}$

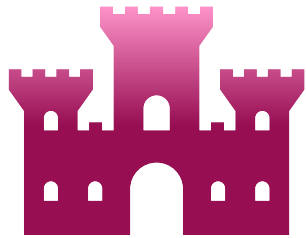


Justified?

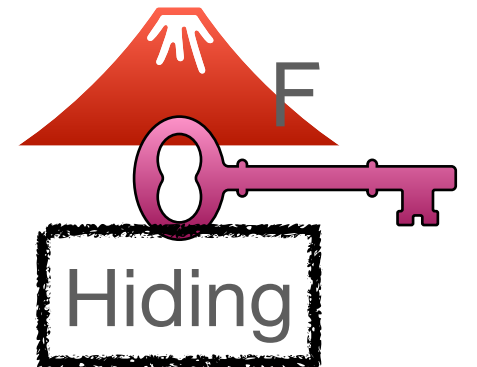
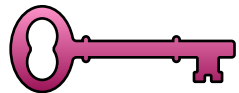


Before moving on - A Recap

Map of projects/Qs ; important tool (DPVS)



Func: G,
Policy: Spade(·)



[CDGPP'18]

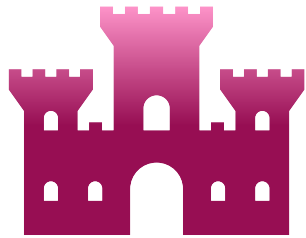
Corrupt(i)

$\Rightarrow x_{0,i} = x_{1,i}$

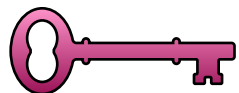
🤔 Justified?

Before moving on - A Recap

Map of projects/Qs ; important tool (DPVS)

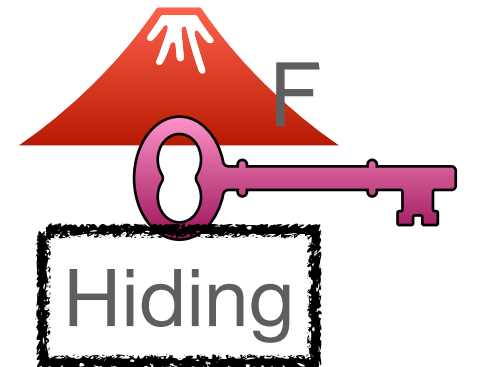


Func: G ,
Policy: $\text{Spade}(\cdot)$



(3)

Control Keys
& strong security



[CDGPP'18]

$\text{Corrupt}(i)$

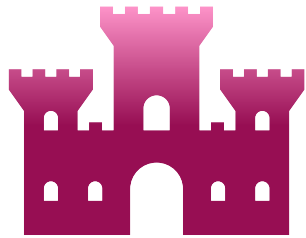
$\Rightarrow x_{0,i} = x_{1,i}$



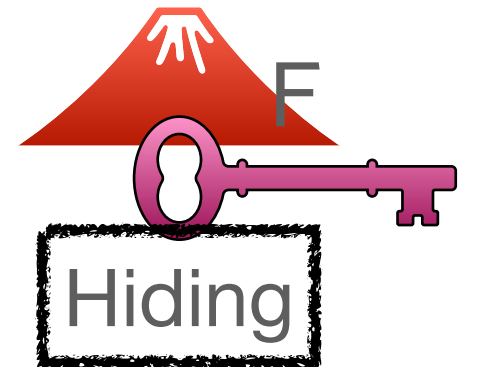
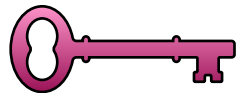
Justified?

Before moving on - A Recap

Map of projects/Qs ; important tool (DPVS)



Func: G,
Policy: Spade(·)



[CDGPP'18]

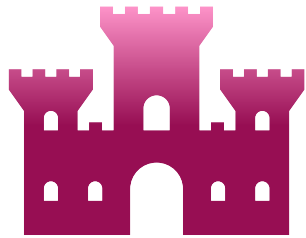
Corrupt(i)

$\Rightarrow x_{0,i} = x_{1,i}$

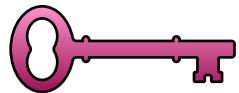
🤔 Justified?

Before moving on - A Recap

Map of projects/Qs ; important tool (DPVS)



Func: G,
Policy: Spade(·)



[CDGPP'18]

Corrupt(i)

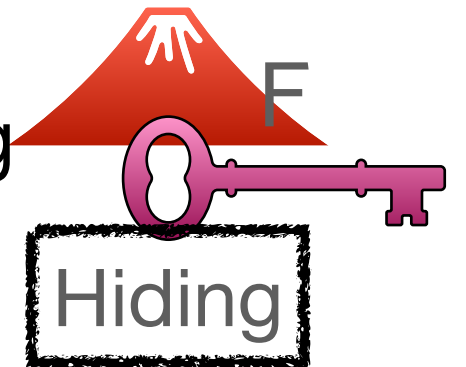
$\Rightarrow x_{0,i} = x_{1,i}$



Justified?

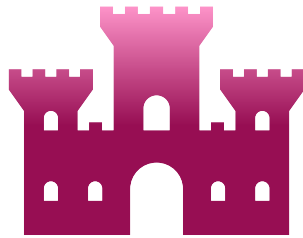
(4)

Function-hiding

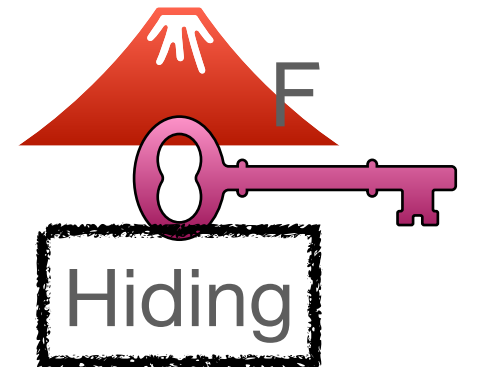
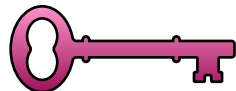


Before moving on - A Recap

Map of projects/Qs ; important tool (DPVS)



Func: G,
Policy: Spade(·)



[CDGPP'18]

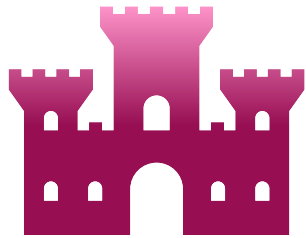
Corrupt(i)

$\Rightarrow x_{0,i} = x_{1,i}$

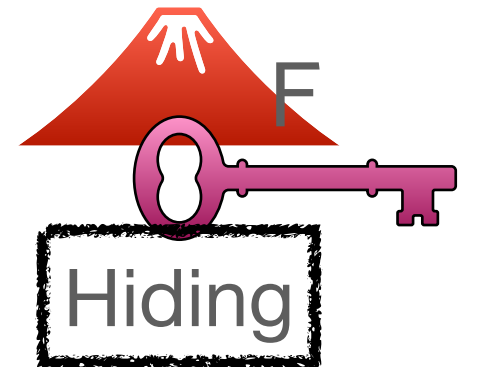
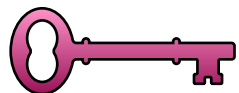
🤔 Justified?

Before moving on - A Recap

Map of projects/Qs ; important tool (DPVS)



Func: G ,
Policy: $\text{Spade}(\cdot)$



[CDGPP'18]

$\text{Corrupt}(i)$

$\Rightarrow x_{0,i} = x_{1,i}$



Justified?

DPVS:

1. Formal



2. Computational



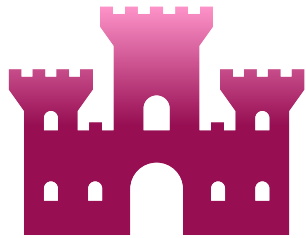
$\Rightarrow$ **New techniques:**

★ “Compress-and-Duplicate”

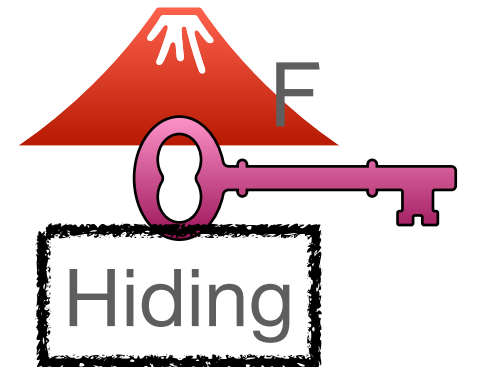
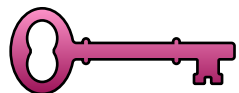
★ DPVS in Complexity Leveraging

Before moving on - A Recap

Map of projects/Qs ; important tool (DPVS)



Func: G ,
Policy: $\text{Spade}(\cdot)$



[CDGPP'18]

$\text{Corrupt}(i)$

$\Rightarrow x_{0,i} = x_{1,i}$

🤔 Justified?

DPVS:

1. Formal



2. Computational



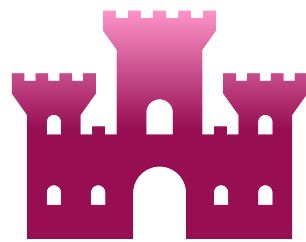
$\Rightarrow$ **New techniques:**

★ “Compress-and-Duplicate”

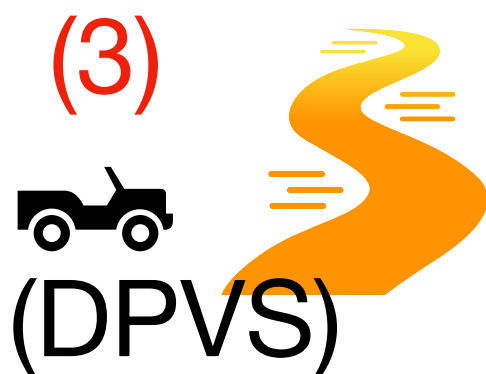
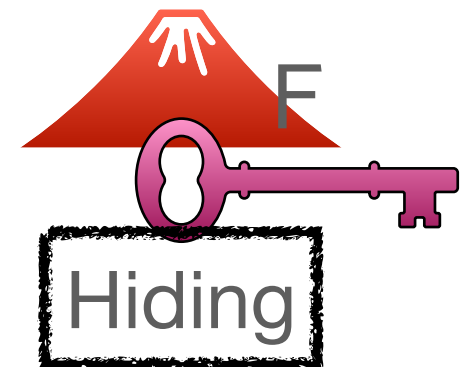
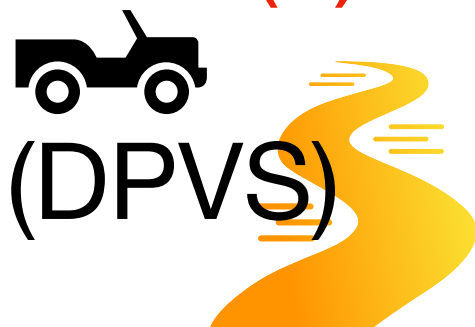
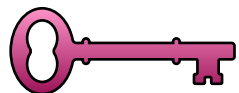
★ DPVS in Complexity Leveraging

Before moving on - A Recap

Map of projects/Qs ; important tool (DPVS)



Func: G,
Policy: Spade(·)



[CDGPP'18]

Corrupt(i)

$$\Rightarrow x_{0,i} = x_{1,i}$$

🤔 Justified?



DPVS:

1. Formal



2. Computational



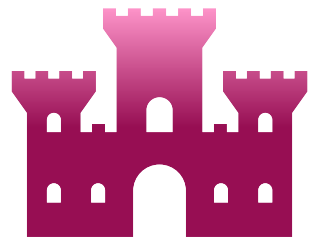
$\Rightarrow$ **New techniques:**

★ “Compress-and-Duplicate”

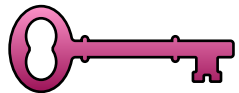
★ DPVS in Complexity Leveraging

Before moving on - A Recap

Map of projects/Qs ; important tool (DPVS)



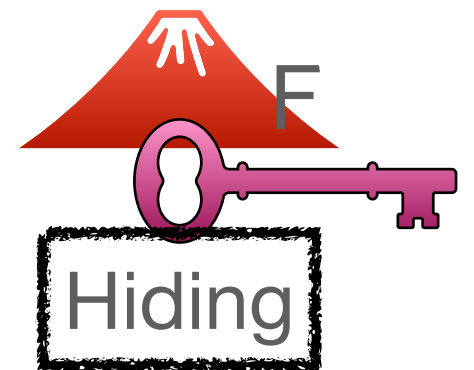
Func: G,
Policy: Spade(·)



(1)



(4)



(2)



(3)



Access control
With Optimality



[CDGPP'18]

Corrupt(i)

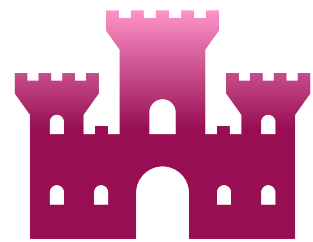
$$\Rightarrow x_{0,i} = x_{1,i}$$



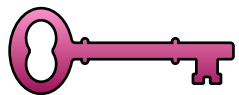
Justified?

Before moving on - A Recap

Map of projects/Qs ; important tool (DPVS)



Func: G,
Policy: Spade(·)



Access control
With Optimality

(1)



(2)



[CDGPP'18]

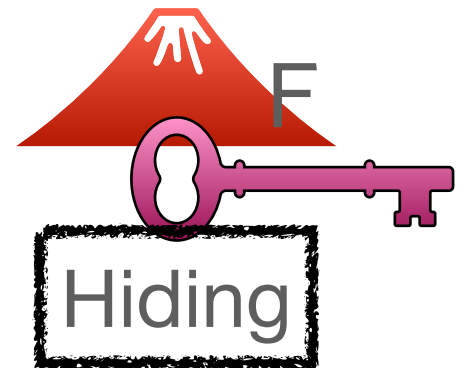
Corrupt(i)

$\Rightarrow x_{0,i} = x_{1,i}$



Justified?

(4)

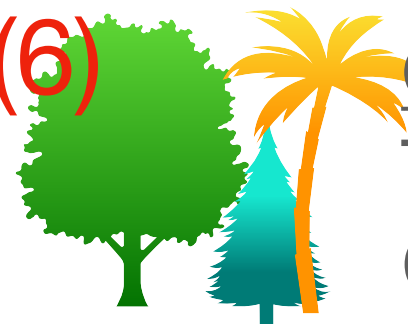


(5)



Other misc:
Blind Sign,
CALBO-TDFs

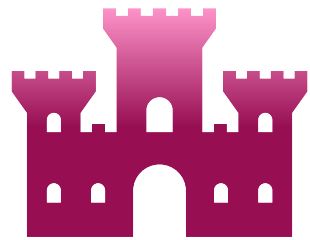
(6)



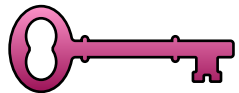
Ongoing:
Traceable FE,
CCA-security

Before moving on - A Recap

Map of projects/Qs ; important tool (DPVS)



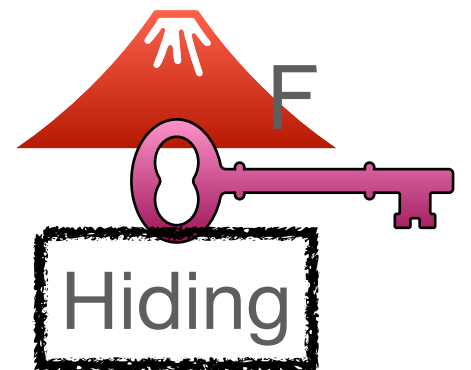
Func: G,
Policy: Spade(·)



(1)



(4)



(2)



(3)



Access control
With Optimality



[CDGPP'18]

Corrupt(i)

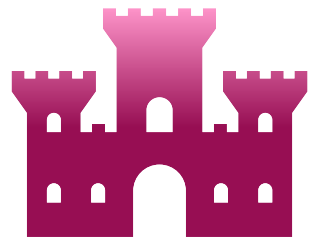
$\Rightarrow x_{0,i} = x_{1,i}$



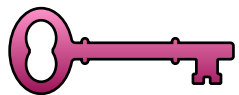
Justified?

Before moving on - A Recap

Map of projects/Qs ; important tool (DPVS)



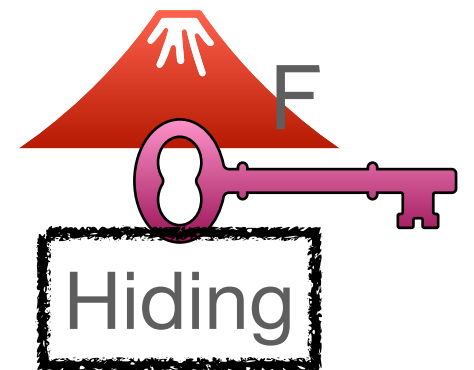
Func: G,
Policy: Spade(·)



(1)



(4)



(2)



(3)



Access control
With Optimality



[CDGPP'18]

Corrupt(i)

$\Rightarrow x_{0,i} = x_{1,i}$



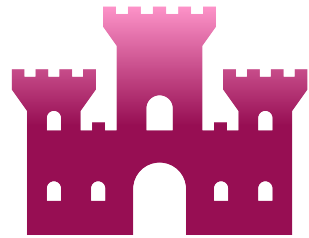
Justified?

This talk:
(1), (2), (3)

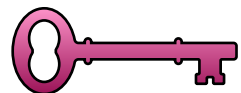
Outline

- Introduction & Main Techniques Overview
- **Controlling** Functional Keys
 - Definitional Framework for Multi-Client (MCFE) Setting
 - Constructions from Dual Pairing Vector Spaces
- Optimal Security Notion
- From Secret-Key to Public-Key Encryption Schemes
- Conclusion and Future Works

This chapter: Controlling functional Keys



Func: G ,
Policy: $\text{Spade}(\cdot)$



(1)
Key-
Control



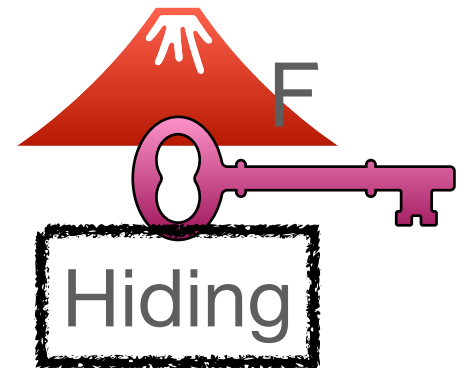
[CDGPP'18]

$\text{Corrupt}(i)$

$\Rightarrow x_{0,i} = x_{1,i}$



Justified?



Other misc:
Blind Sign,
CALBO-TDFs



Ongoing:
Traceable FE,
CCA-security

Related Works and Our Contributions

FE + Access Control in the Multi-User Regime

[ACGU20]

- From pairings and LWE
- $\mathcal{P}$: *Monotone Span Programs*
- F : *Inner Products*
- (Multi-Input) Generic from single-client, adaptive security in standard model
- Quadratic total communication

Related Works and Our Contributions

FE + Access Control in the Multi-User Regime

[ACGU20]

- From pairings and LWE
- $\mathcal{P}$: *Monotone Span Programs*
- F : *Inner Products*
- (Multi-Input) Generic from single-client, adaptive security in standard model
- Quadratic total communication

[NPP22]

- From pairings
- $\mathcal{P}$: *LSSS*
- F : *Inner Products*
- (Multi-Client) Adaptive security in ROM
linear total communication

Related Works and Our Contributions

FE + Access Control in the Multi-User Regime

[ACGU20]

- From pairings and LWE
- $\mathcal{P}$: *Monotone Span Programs*
- F : *Inner Products*
- (Multi-Input) Generic from single-client, adaptive security in standard model
- Quadratic total communication

[NPP22]

- From pairings
- $\mathcal{P}$: *LSSS*
- F : *Inner Products*
- (Multi-Client) Adaptive security in ROM
linear total communication
⇒ (Multi-Input) adaptive security in standard model, linear total comm.

Related Works and Our Contributions

FE + Access Control in the Multi-User Regime

MCFE only for scalars

No security against repeated attributes

In ciphertexts

⇒ Improved in later chapter

[NPP22]

- From pairings
- $\mathcal{P}$: *LSSS*
- F : *Inner Products*
- (Multi-Client) **Adaptive** security in ROM
linear total communication
⇒ (Multi-Input) **adaptive** security in standard model, linear total comm.

Definition

MCFE with Fine-Grained Access Control

$(\mathcal{P} : \text{LSSS})$



$$\text{AC-K} \times \text{AC-Ct}_1 \times \cdots \times \text{AC-Ct}_i \times \cdots \times \text{AC-Ct}_n \rightarrow \{0,1\}$$

Challenger: $b \xleftarrow{\$} \{0,1\}$

Security



Chall

Enc

Adversary



Definition

MCFE with Fine-Grained Access Control

Challenger: $b \xleftarrow{\$} \{0,1\}$



Chall

Security

Extract($\mathcal{P}, \vec{y}$)



Adversary

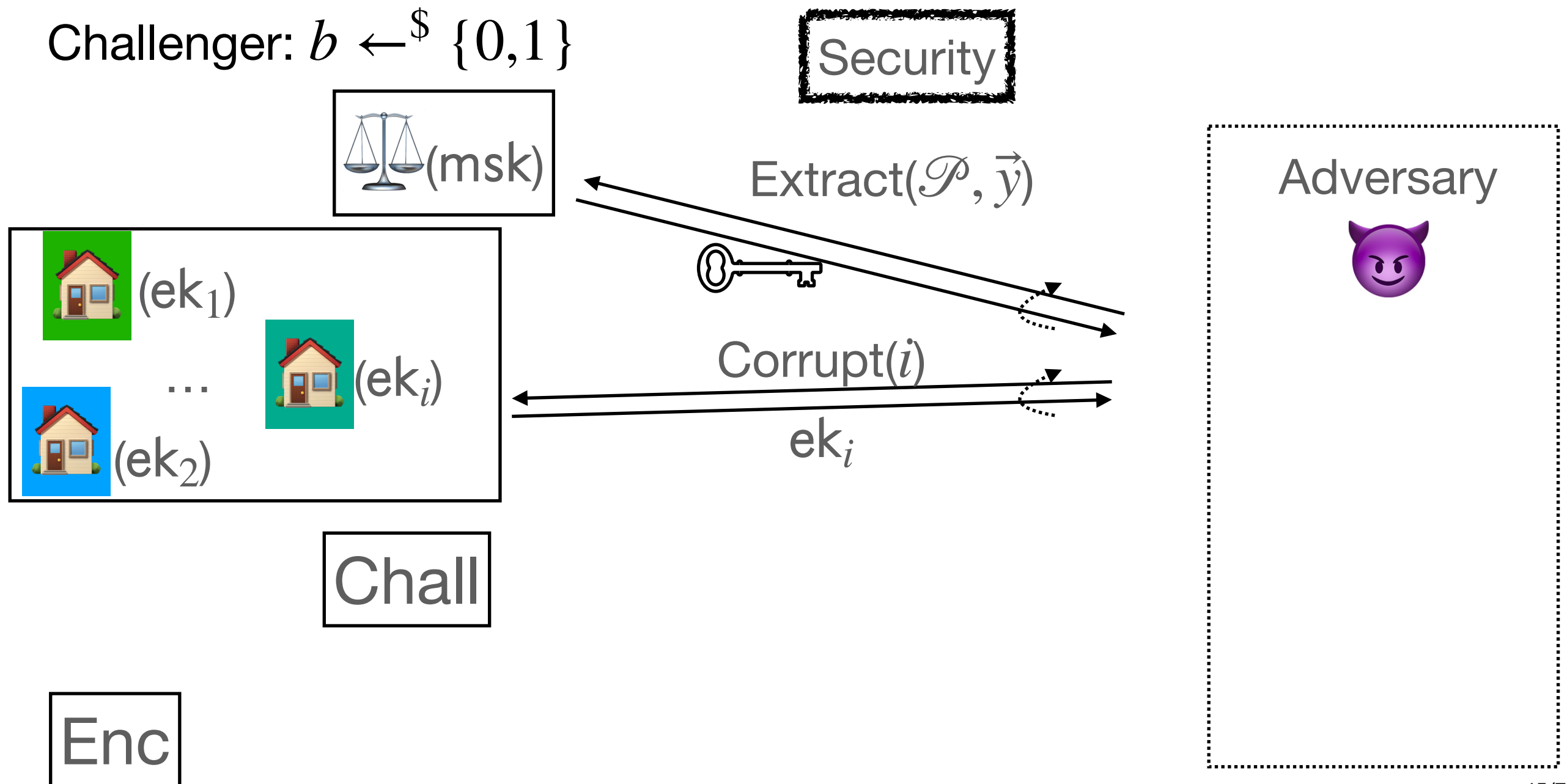


Enc

Definition

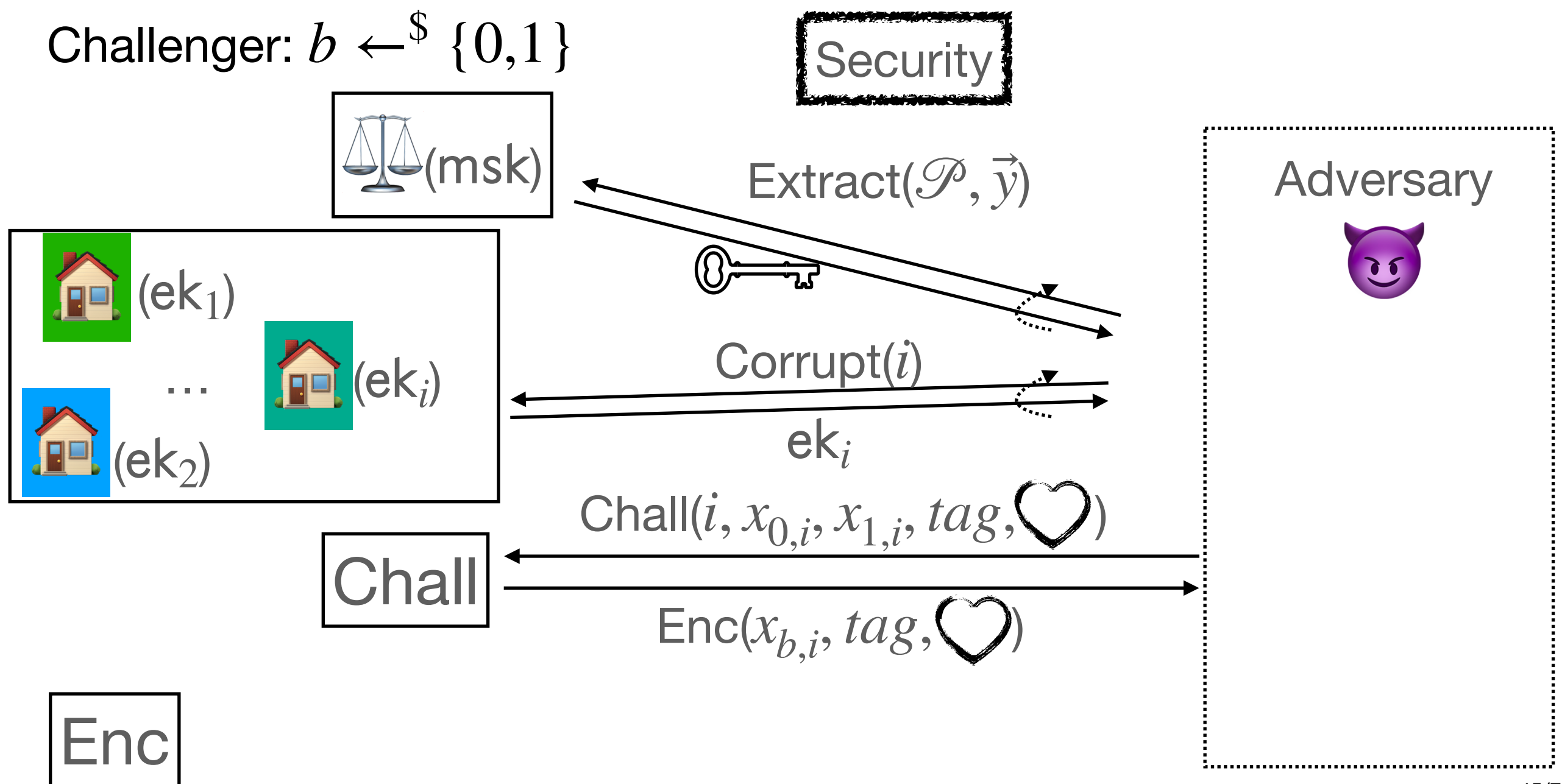
MCFE with Fine-Grained Access Control

Challenger: $b \xleftarrow{\$} \{0,1\}$



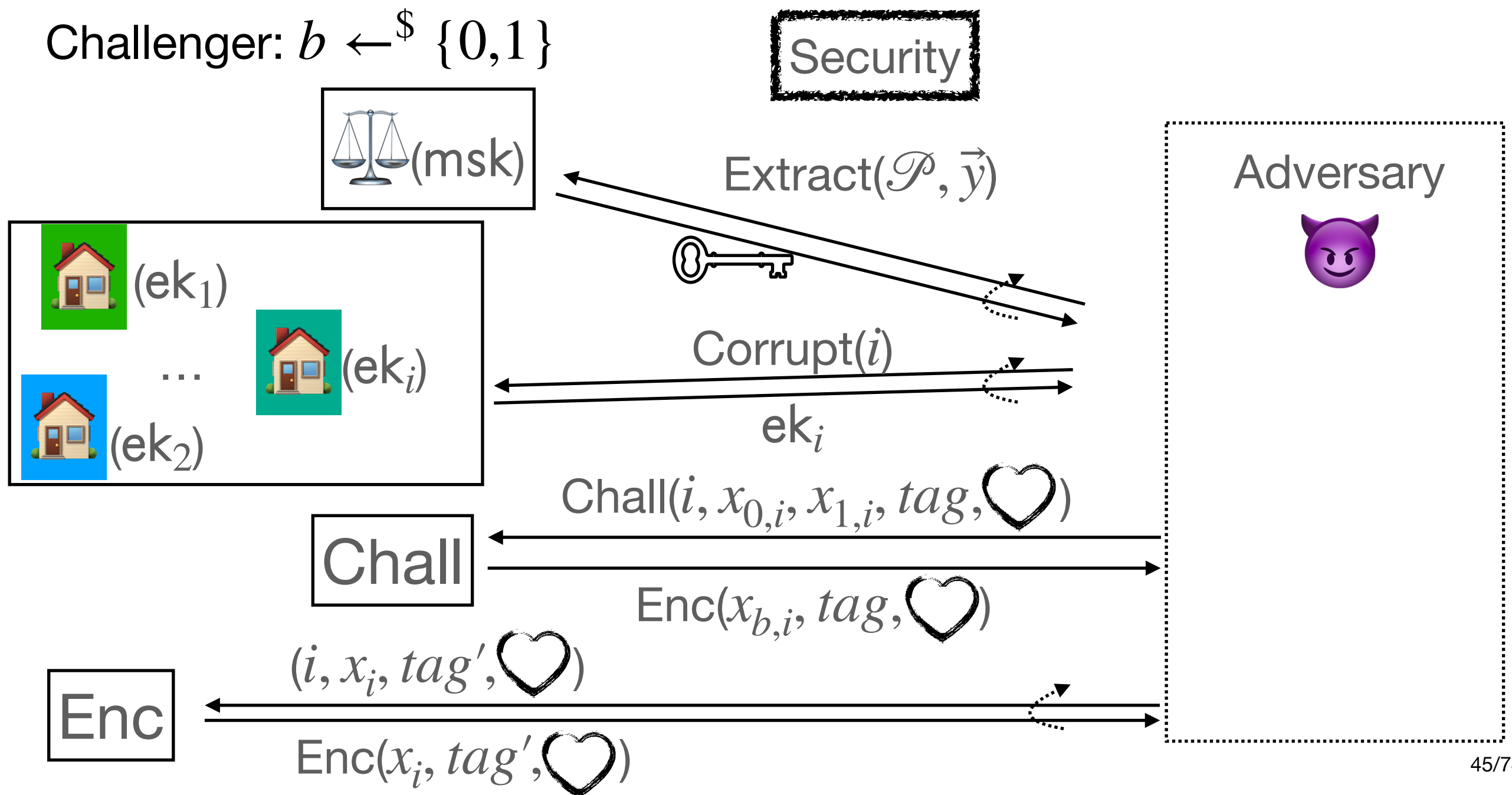
Definition

MCFE with Fine-Grained Access Control



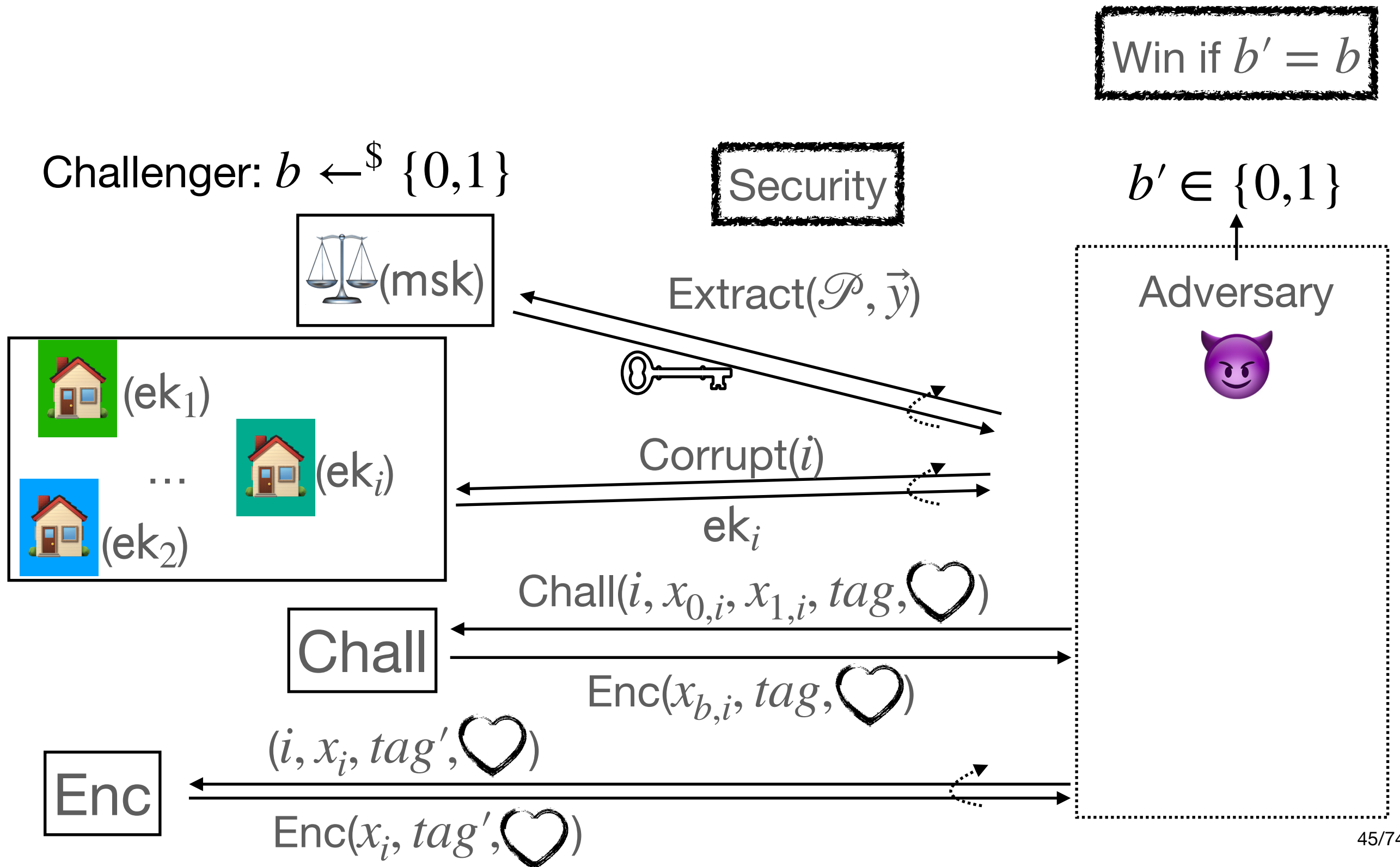
Definition

MCFE with Fine-Grained Access Control



Definition

MCFE with Fine-Grained Access Control



Definition

MCFE with Fine-Grained Access Control

Win if $b' = b$

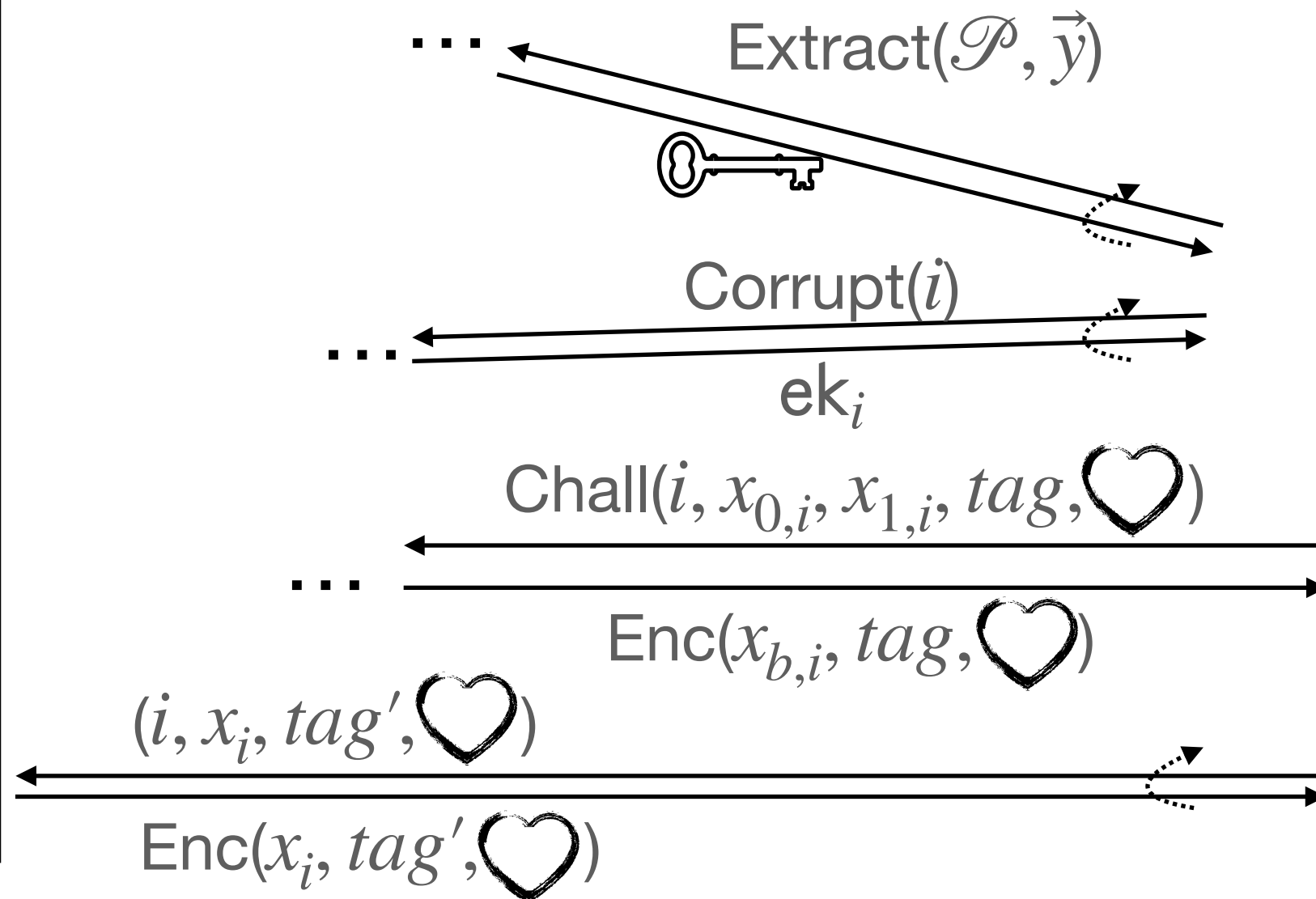
Challenger: $b \leftarrow^{\$} \{0,1\}$

Security

$b' \in \{0,1\}$

Challenger

Adversary



Definition

MCFE with Fine-Grained Access Control

Win if $b' = b$

Challenger: $b \leftarrow^{\$} \{0,1\}$

Security

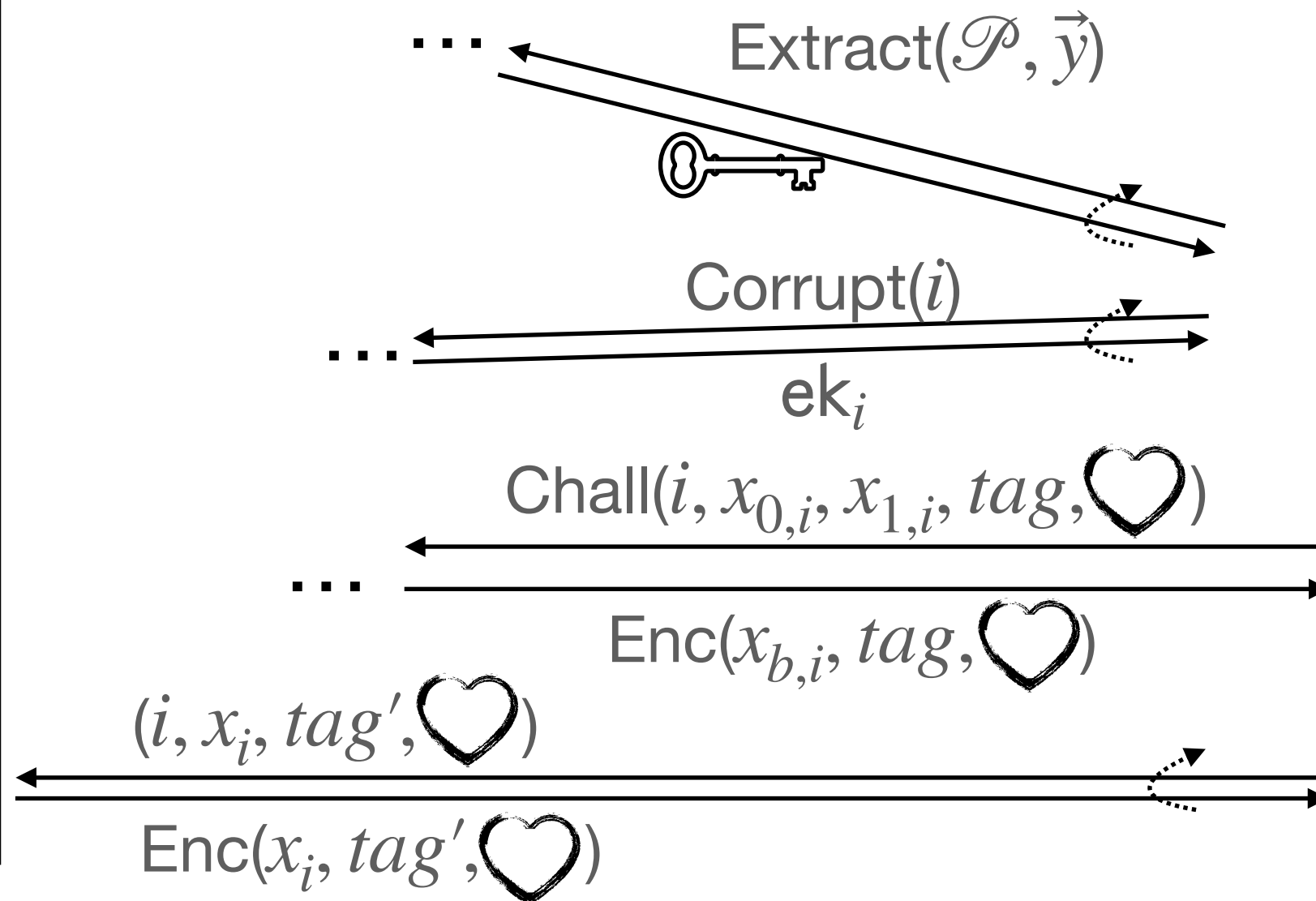
$b' \in \{0,1\}$

Challenger

Adversary



- **AD/SEL** Chall
- **DYN/STAT** Corruption



Definition

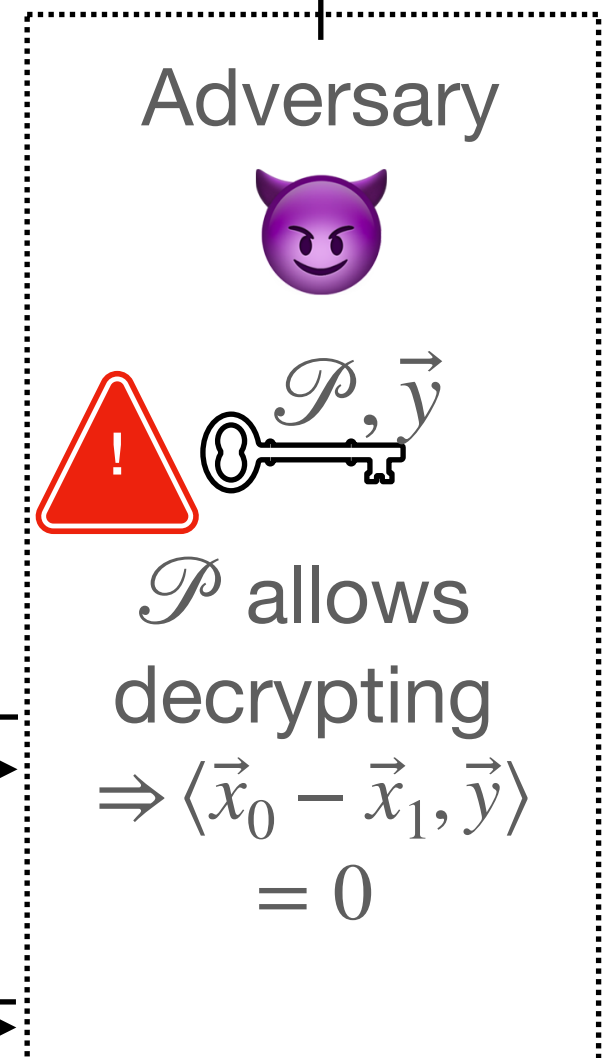
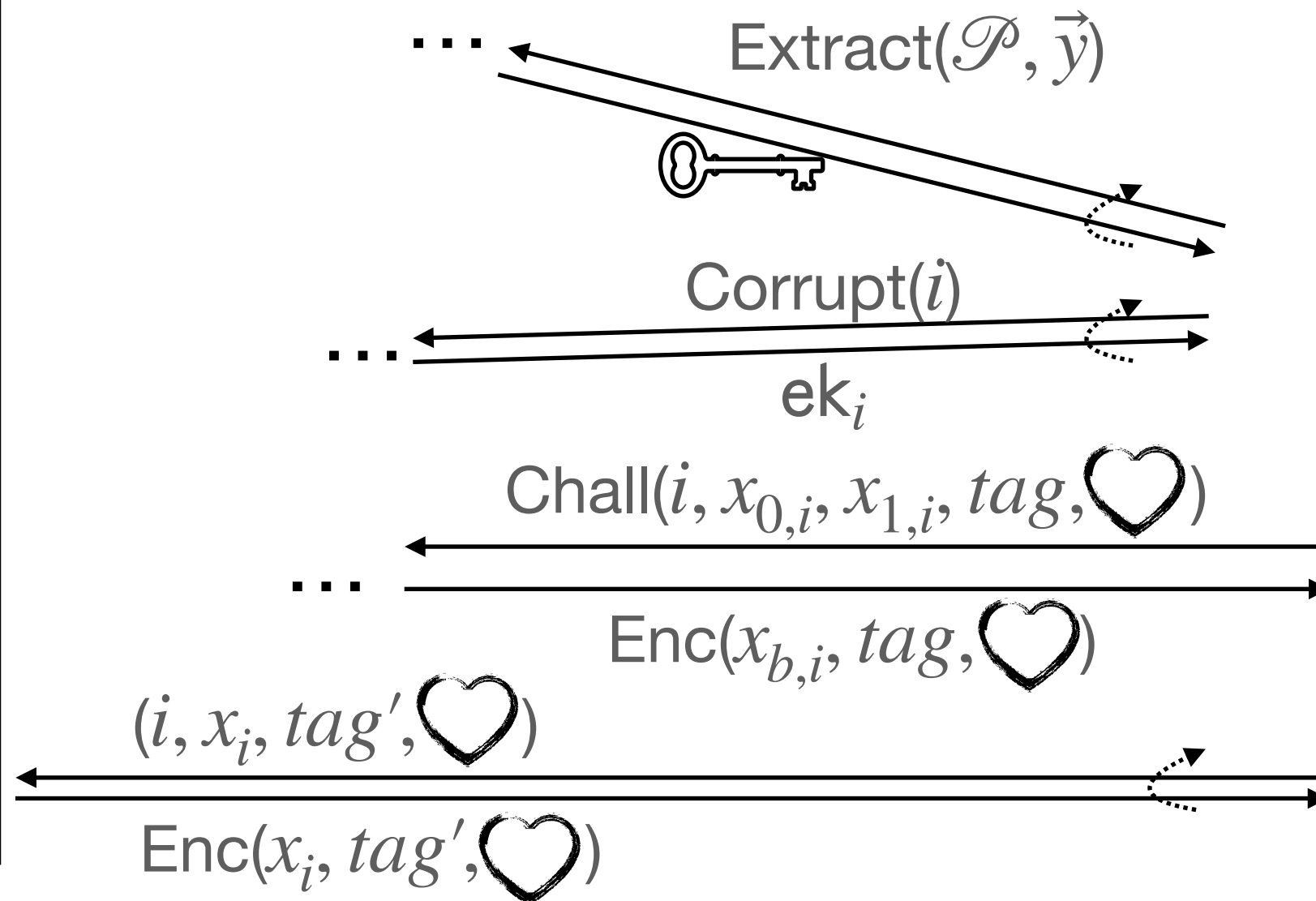
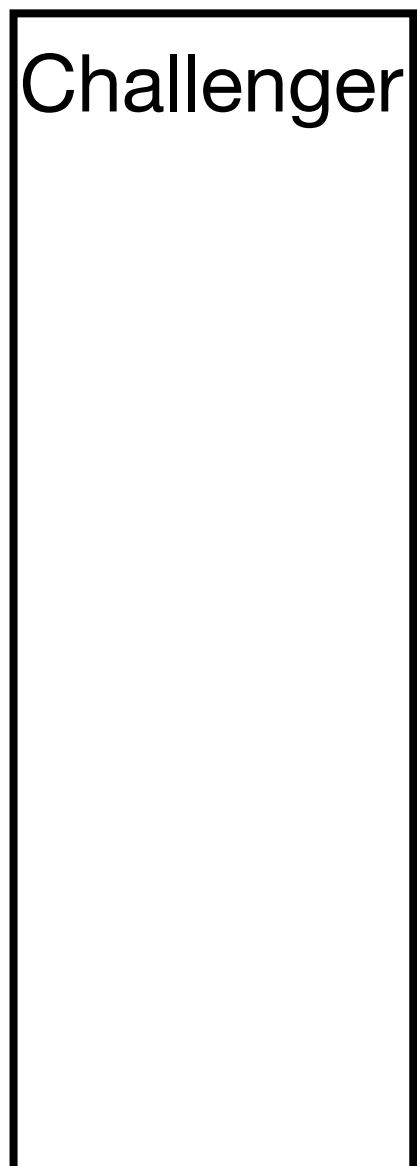
MCFE with Fine-Grained Access Control

Win if $b' = b$

Challenger: $b \leftarrow^{\$} \{0,1\}$

Security

$b' \in \{0,1\}$



Technical Overview - Single-Client

AB-IP^{FE} for LSSS - Adaptive Security, linear ciphertext's size

$$\begin{array}{llll}
 (\mathcal{P} : \text{LSSS}) & & & \\
 \text{AC-K} \times \text{AC-Ct} \rightarrow \{0,1\} & \text{document icon} = (x_1, \dots, x_n) & \text{key icon } \mathcal{P}, \vec{y} & \begin{array}{l} S = (s_1, \dots, s_n) \\ U = (u_1, \dots, u_n) \end{array}
 \end{array}$$

Technical Overview - Single-Client

AB-IP^{FE} for LSSS - Adaptive Security, linear ciphertext's size

$$(\mathcal{P} : \text{LSSS}) \quad \text{AC-K} \times \text{AC-Ct} \rightarrow \{0,1\} \quad \text{Z} = (x_1, \dots, x_n) \quad \text{Key} = \mathcal{P}, \vec{y} \quad S = (s_1, \dots, s_n) \quad U = (u_1, \dots, u_n)$$

$$\text{Ciphertext} \begin{cases} (\llbracket \omega(s_i + \mu u_i) + x_i \rrbracket_1)_{i \in [n]} \\ \vec{c}_{att} = (\dots | \psi | 0 | \dots)_{\mathbf{F}} \end{cases} ; \quad \vec{c}_0 = (\omega | \mu \omega | \psi | 0)_{\mathbf{H}}$$

Technical Overview - Single-Client

AB-IP^{FE} for LSSS - Adaptive Security, linear ciphertext's size

$$(\mathcal{P} : \text{LSSS}) \quad \text{AC-K} \times \text{AC-Ct} \rightarrow \{0,1\} \quad \text{Document} = (x_1, \dots, x_n) \quad \text{Key} = \mathcal{P}, \vec{y} \quad S = (s_1, \dots, s_n) \\ U = (u_1, \dots, u_n)$$

$$\text{Ciphertext} \begin{cases} (\llbracket \omega(s_i + \mu u_i) + x_i \rrbracket_1)_{i \in [n]} \\ \vec{c}_{att} = (\dots | \psi | 0 | \dots)_{\mathbf{F}} \end{cases} ; \quad \vec{c}_0 = (\omega | \mu \omega | \psi | 0)_{\mathbf{H}}$$

$$\text{Key-Policy, } (a_{att})_{att} \leftarrow \text{LSSS-Share}(a_0)$$

$$\vec{k}_{att}^* = (\dots | a_{att} | 0 | \dots)_{\mathbf{F}^*} ; \quad \vec{k}_0^* = (\langle S, \vec{y} \rangle | \langle U, \vec{y} \rangle | a_0 | 0)_{\mathbf{H}^*}$$

Technical Overview - Single-Client

AB-IP^{FE} for LSSS - Adaptive Security, linear ciphertext's size

$$S = (s_1, \dots, s_n)$$

$$U = (u_1, \dots, u_n)$$

Ciphertext $\begin{cases} (\llbracket \omega(s_i + \mu u_i) + x_i \rrbracket_1)_{i \in [n]} \\ \vec{c}_{att} = (\dots | \psi | 0 | \dots)_{\mathbf{F}} \end{cases} ;$

Randomness

$$\vec{c}_0 = (\omega | \mu \omega | \psi | 0)_{\mathbf{H}}$$

Key-Policy, $(a_{att})_{att} \leftarrow \text{LSSS-Share}(a_0)$

$$\vec{k}_{att}^* = (\dots | a_{att} | 0 | \dots)_{\mathbf{F}^*} ; \quad \vec{k}_0^* = (\langle S, \vec{y} \rangle | \langle U, \vec{y} \rangle | a_0 | 0)_{\mathbf{H}^*}$$

Masking key [Agrawal, Libert, Stehlé'16]

Technical Overview - Single-Client

AB-IP^{FE} for LSSS - Adaptive Security, linear ciphertext's size

$$[\omega(s_i + \mu u_i) + x_{b,i}]_1, \Delta x_i = x_{0,i} - x_{b,i}, (a'_{att})_{att} \leftarrow \text{LSSS-Share}(a'_0)$$

Changes using
Basis changes
In DPVS

$$\vec{c}_{att} = (\cdots | \psi | \tau \Delta x_1 z_{att} | \cdots | \tau \Delta x_n z_{att})_{\mathbf{F}}$$

$$\vec{k}_{att}^* = (\cdots | a_{att} | a'_{att} y_1 / z_{att} | \cdots | a'_{att} y_n / z_{att})_{\mathbf{F}^*}$$

$$\vec{c}_0 = (\omega | \mu \omega | \psi | \tau \Delta x_1 | \cdots | \tau \Delta x_n)_{\mathbf{H}}$$

$$\vec{k}_0^* = (\cdots | a_0 | a'_0 y_1 | \cdots | a'_0 y_n)_{\mathbf{H}^*}$$

Technical Overview - Single-Client

AB-IP^{FE} for LSSS - Adaptive Security, linear ciphertext's size

$$[\omega(s_i + \mu u_i) + x_{b,i}]_1, \Delta x_i = x_{0,i} - x_{b,i}, (a'_{att})_{att} \leftarrow \text{LSSS-Share}(a'_0)$$

Changes using
Basis changes
In DPVS

$$\vec{c}_{att} = (\cdots | \psi | \tau \Delta x_1 z_{att} | \cdots | \tau \Delta x_n z_{att})_{\mathbf{F}}$$

$$\vec{k}_{att}^* = (\cdots | a_{att} | a'_{att} y_1 / z_{att} | \cdots | a'_{att} y_n / z_{att})_{\mathbf{F}^*}$$

$$\vec{c}_0 = (\omega | \mu \omega | \psi | \tau \Delta x_1 | \cdots | \tau \Delta x_n)_{\mathbf{H}}$$

$$\vec{k}_0^* = (\cdots | a_0 | a'_0 y_1 | \cdots | a'_0 y_n)_{\mathbf{H}^*}$$

Random z_{att}

Technical Overview - Single-Client

AB-IP^{FE} for LSSS - Adaptive Security, linear ciphertext's size

$$[\omega(s_i + \mu u_i) + x_{b,i}]_1, \Delta x_i = x_{0,i} - x_{b,i}, (a'_{att})_{att} \leftarrow \text{LSSS-Share}(a'_0)$$

Changes using
Basis changes
In DPVS

$$\vec{c}_{att} = (\cdots | \psi | \tau \Delta x_1 z_{att} | \cdots | \tau \Delta x_n z_{att})_{\mathbf{F}}$$

$$\vec{k}_{att}^* = (\cdots | a_{att} | a'_{att} y_1 / z_{att} | \cdots | a'_{att} y_n / z_{att})_{\mathbf{F}^*}$$

$$\vec{c}_0 = (\omega | \mu \omega | \psi | \tau \Delta x_1 | \cdots | \tau \Delta x_n)_{\mathbf{H}}$$

$$\vec{k}_0^* = (\cdots | a_0 | a'_0 y_1 | \cdots | a'_0 y_n)_{\mathbf{H}^*}$$

Random z_{att}



$(\vec{k}_{att}^*, \vec{k}_0)$ can't decrypt: ABE technique

Technical Overview - Single-Client

AB-IP^{FE} for LSSS - Adaptive Security, linear ciphertext's size

$$[\omega(s_i + \mu u_i) + x_{b,i}]_1, \Delta x_i = x_{0,i} - x_{b,i}, (a'_{att})_{att} \leftarrow \text{LSSS-Share}(a'_0)$$

Changes using
Basis changes
In DPVS

$$\vec{c}_{att} = (\cdots | \psi | \tau \Delta x_1 z_{att} | \cdots | \tau \Delta x_n z_{att})_{\mathbf{F}}$$

$$\vec{k}_{att}^* = (\cdots | a_{att} | a'_{att} y_1 / z_{att} | \cdots | a'_{att} y_n / z_{att})_{\mathbf{F}^*}$$

$$\vec{c}_0 = (\omega | \mu \omega | \psi | \tau \Delta x_1 | \cdots | \tau \Delta x_n)_{\mathbf{H}}$$

$$\vec{k}_0^* = (\cdots | a_0 | a'_0 y_1 | \cdots | a'_0 y_n)_{\mathbf{H}^*}$$

Random z_{att}



$(\vec{k}_{att}^*, \vec{k}_0)$ can't decrypt: ABE technique



$(\vec{k}_{att}^*, \vec{k}_0)$ can decrypt $\Rightarrow \langle \Delta x, \vec{y} \rangle = 0$

Technical Overview - Single-Client

AB-IP^{FE} for LSSS - Adaptive Security, linear ciphertext's size

$$[\omega(s_i + \mu u_i) + x_{b,i}]_1, \Delta x_i = x_{0,i} - x_{b,i}, (a'_{att})_{att} \leftarrow \text{LSSS-Share}(a'_0)$$

Changes using
Basis changes
In DPVS

$$\vec{c}_{att} = (\cdots | \psi | \tau \Delta x_1 z_{att} | \cdots | \tau \Delta x_n z_{att})_{\mathbf{F}}$$

$$\vec{k}_{att}^* = (\cdots | a_{att} | a'_{att} y_1 / z_{att} | \cdots | a'_{att} y_n / z_{att})_{\mathbf{F}^*}$$

$$\vec{c}_0 = (\omega | \mu \omega | \psi | \tau \Delta x_1 | \cdots | \tau \Delta x_n)_{\mathbf{H}}$$

$$\vec{k}_0^* = (\cdots | a_0 | a'_0 y_1 | \cdots | a'_0 y_n)_{\mathbf{H}^*}$$

Random z_{att}



$(\vec{k}_{att}^*, \vec{k}_0)$ can't decrypt: ABE technique



$(\vec{k}_{att}^*, \vec{k}_0)$ can decrypt $\Rightarrow \langle \Delta x, \vec{y} \rangle = 0$

Adaptive Security 🤔

Technical Overview - Single-Client

AB-IP^{FE} for LSSS - Adaptive Security, linear ciphertext's size

$$[\omega(s_i + \mu u_i) + x_{b,i}]_1, \Delta x_i = x_{0,i} - x_{b,i}, (a'_{att})_{att} \leftarrow \text{LSSS-Share}(a'_0)$$

Changes using
Basis changes
In DPVS

$$\vec{c}_{att} = (\cdots | \psi | \tau \Delta x_1 z_{att} | \cdots | \tau \Delta x_n z_{att})_{\mathbf{F}}$$

$$\vec{k}_{att}^* = (\cdots | a_{att} | a'_{att} y_1 / z_{att} | \cdots | a'_{att} y_n / z_{att})_{\mathbf{F}^*}$$

$$\vec{c}_0 = (\omega | \mu \omega | \psi | \tau \Delta x_1 | \cdots | \tau \Delta x_n)_{\mathbf{H}}$$

$$\vec{k}_0^* = (\cdots | a_0 | a'_0 y_1 | \cdots | a'_0 y_n)_{\mathbf{H}^*}$$

Random z_{att}



$(\vec{k}_{att}^*, \vec{k}_0)$ can't decrypt: ABE technique



$(\vec{k}_{att}^*, \vec{k}_0)$ can decrypt $\Rightarrow \langle \Delta x, \vec{y} \rangle = 0$

Adaptive Security 🤔

Technical Overview - Single-Client

AB-IPFE for LSSS - Adaptive Security, linear ciphertext's size

$$[\omega(s'_i + \mu u'_i) + x_{0,i}]_1, \Delta x_i = x_{0,i} - x_{b,i}, (a'_{att})_{att} \leftarrow \text{LSSS-Share}(a'_0)$$

Changes using
Basis changes
In DPVS

$$\vec{c}_{att} = (\cdots | \psi | \tau \Delta x_1 z_{att} | \cdots | \tau \Delta x_n z_{att})_{\mathbf{F}}$$

$$\vec{k}_{att}^* = (\cdots | a_{att} | a'_{att} y_1 / z_{att} | \cdots | a'_{att} y_n / z_{att})_{\mathbf{F}^*}$$

$$\vec{k}_0^* = (\cdots | a_0 | r'_0 y_1 | \cdots | r'_0 y_n)_{\mathbf{H}^*}$$

$$\vec{c}_0 = (\omega | \mu \omega | \psi | \tau \Delta x_1 | \cdots | \tau \Delta x_n)_{\mathbf{H}}$$

Technical Overview - Single-Client

AB-IPFE for LSSS - Adaptive Security, linear ciphertext's size

$$[\omega(s'_i + \mu u'_i) + x_{0,i}]_1, \Delta x_i = x_{0,i} - x_{b,i}, (a'_{att})_{att} \leftarrow \text{LSSS-Share}(a'_0)$$

Changes using
Basis changes
In DPVS

$$\vec{c}_{att} = (\cdots | \psi | \tau \Delta x_1 z_{att} | \cdots | \tau \Delta x_n z_{att})_{\mathbf{F}}$$

$$\vec{k}_{att}^* = (\cdots | a_{att} | a'_{att} y_1 / z_{att} | \cdots | a'_{att} y_n / z_{att})_{\mathbf{F}^*}$$



$$\vec{k}_0^* = (\cdots | a_0 | r'_0 y_1 | \cdots | r'_0 y_n)_{\mathbf{H}^*}$$

$$\vec{c}_0 = (\omega | \mu \omega | \psi | \tau \Delta x_1 | \cdots | \tau \Delta x_n)_{\mathbf{H}}$$



n coordinates

🤔 i.e. size of DPVS

To “pay” for this masking

Technical Overview - Single-Client

AB-IPFE for LSSS - Adaptive Security, linear ciphertext's size

$$[\omega(s'_i + \mu u'_i) + x_{0,i}]_1, \Delta x_i = x_{0,i} - x_{b,i}, (a'_{att})_{att} \leftarrow \text{LSSS-Share}(a'_0)$$

Changes using
Basis changes
In DPVS

$$\vec{c}_{att} = (\cdots | \psi | \tau \Delta x_1 z_{att} | \cdots | \tau \Delta x_n z_{att})_{\mathbf{F}}$$

$$\vec{k}_{att}^* = (\cdots | a_{att} | a'_{att} y_1 / z_{att} | \cdots | a'_{att} y_n / z_{att})_{\mathbf{F}^*}$$

$$\vec{k}_0^* = (\cdots | a_0 | r'_0 y_1 | \cdots | r'_0 y_n)_{\mathbf{H}^*}$$

$$\vec{c}_0 = (\omega | \mu \omega | \psi | \tau \Delta x_1 | \cdots | \tau \Delta x_n)_{\mathbf{H}}$$

Towards Multi-Client - “Compress-and-Duplicate”

MC-AB-IPFE for LSSS - Perform masking steps
in linear total communication

Extract (key)

Masking in **FE**

$$\vec{k}_{att}^* = (\cdots | a_{att} | a'_{att}y_1/z_{att} | \cdots | a'_{att}y_n/z_{att})_{\mathbf{F}^*}$$

$$\vec{k}_0^* = (\cdots | a_0 | r'_0y_1 | \cdots | r'_0y_n)_{\mathbf{H}^*}$$

Enc (ciph)

$$\vec{c}_{att} = (\cdots | \psi | \tau\Delta x_1z_{att} | \cdots | \tau\Delta x_nz_{att})_{\mathbf{F}}$$

$$\vec{c}_0 = (\omega | \mu\omega | \psi | \tau\Delta x_1 | \cdots | \tau\Delta x_n)_{\mathbf{H}}$$

Towards Multi-Client - “Compress-and-Duplicate”

MC-AB-IPFE for LSSS - Perform masking steps in linear total communication

Extract (key)

Masking in FE

$$\vec{k}_{att}^* = (\cdots | a_{att} | \underbrace{a'_{att}y_1/z_{att} | \cdots | a'_{att}y_n/z_{att}}_{n \text{ coordinates}})_{\mathbf{F}^*}$$

$$\vec{k}_0^* = (\cdots | a_0 | \underbrace{r'_0y_1 | \cdots | r'_0y_n}_{n \text{ coordinates}})_{\mathbf{H}^*}$$

Enc (ciph)

$$\vec{c}_{att} = (\cdots | \psi | \underbrace{\tau\Delta x_1 z_{att} | \cdots | \tau\Delta x_n z_{att}}_{n \text{ coordinates}})_{\mathbf{F}}$$

$$\vec{c}_0 = (\omega | \mu\omega | \psi | \tau\Delta x_1 | \cdots | \tau\Delta x_n)_{\mathbf{H}}$$

Towards Multi-Client - “Compress-and-Duplicate”

MC-AB-IPFE for LSSS - Perform masking steps in linear total communication

Extract (key)

Masking in FE

🤔 Naively used
For each client i
⇒ **quadratic** total
Communication

Enc (ciph)

$$\vec{k}_{att}^* = (\cdots | a_{att} | \underbrace{a'_{att}y_1/z_{att} | \cdots | a'_{att}y_n/z_{att}}_{n \text{ coordinates}})_{\mathbf{F}^*}$$

$$\vec{k}_0^* = (\cdots | a_0 | \underbrace{r'_0y_1 | \cdots | r'_0y_n}_{n \text{ coordinates}})_{\mathbf{H}^*}$$

$$\vec{c}_{att} = (\cdots | \psi | \underbrace{\tau\Delta x_1 z_{att} | \cdots | \tau\Delta x_n z_{att}}_{n \text{ coordinates}})_{\mathbf{F}}$$

$$\vec{c}_0 = (\omega | \mu\omega | \psi | \tau\Delta x_1 | \cdots | \tau\Delta x_n)_{\mathbf{H}}$$

Towards Multi-Client - “Compress-and-Duplicate”

MC-AB-IPFE for LSSS - Perform masking steps
in linear total communication

Extract (key)

Enc (ciph)

Towards Multi-Client - “Compress-and-Duplicate”

MC-AB-IPFE for LSSS - Perform masking steps in linear total communication

Extract (key)

Applying our new
“Compress-and-Duplicate”

- Basis changes have
Independent indexable
blocks

Enc (ciph)

- “Compress” over $\vec{k}_{i,att}^*$, $\vec{c}_{i,att}$ (Key control)
- “Duplicate” over $\vec{k}_{i,0}^*$, $\vec{c}_{i,0}$ (Computing $\langle \cdot, \cdot \rangle$)

Towards Multi-Client - “Compress-and-Duplicate”

MC-AB-IPFE for LSSS - Perform masking steps
in linear total communication

Extract (key)

“Compress” in $(\mathbf{F}, \mathbf{F}^*)$

$$\vec{k}_{i,att}^* = (\cdots \mid a_{att} \mid \overbrace{a'_{att} y_i / z_{att}}^{\Theta(1) \text{ coordinates}} \mid 0, 0, 0, 0)_{\mathbf{F}^*}$$

Enc (ciph)

Towards Multi-Client - “Compress-and-Duplicate”

MC-AB-IPFE for LSSS - Perform masking steps
in linear total communication

Extract (key)

“Compress” in $(\mathbf{F}, \mathbf{F}^*)$

$$\vec{k}_{i,att}^* = (\cdots | a_{att} | \overbrace{a'_{att} y_i / z_{att}}^{\Theta(1) \text{ coordinates}} | 0, 0, 0, 0)_{\mathbf{F}^*}$$

“Duplicate” $(\mathbf{H}_i, \mathbf{H}_i^*)$

$$\vec{k}_{i,0}^* = (\cdots | a_0 | r'_0 y_i | 0)_{\mathbf{H}_i^*}$$

Enc (ciph)

Towards Multi-Client - “Compress-and-Duplicate”

MC-AB-IPFE for LSSS - Perform masking steps
in linear total communication

Extract (key)

“Compress” in $(\mathbf{F}, \mathbf{F}^*)$

$$\vec{k}_{i,att}^* = (\cdots | a_{att} | a'_{att} y_i / z_{att} | 0,0,0,0)_{\mathbf{F}^*}$$

$\Theta(1)$ coordinates
↙

“Duplicate” $(\mathbf{H}_i, \mathbf{H}_i^*)$

$$\vec{k}_{i,0}^* = (\cdots | a_0 | r'_0 y_i | 0)_{\mathbf{H}_i^*}$$

Enc (ciph)

“Compress” in $(\mathbf{F}, \mathbf{F}^*)$

$$\vec{c}_{i,att} = (\cdots | \psi_i | \tau \Delta x_i z_{att} | 0,0,0,0)_{\mathbf{F}}$$

Towards Multi-Client - “Compress-and-Duplicate”

MC-AB-IPFE for LSSS - Perform masking steps
in linear total communication

Extract (key)

“Compress” in $(\mathbf{F}, \mathbf{F}^*)$

$$\vec{k}_{i,att}^* = (\cdots | a_{att} | a'_{att} y_i / z_{att} | 0,0,0,0)_{\mathbf{F}^*}$$

$\Theta(1)$ coordinates
↙

“Duplicate” $(\mathbf{H}_i, \mathbf{H}_i^*)$

$$\vec{k}_{i,0}^* = (\cdots | a_0 | r'_0 y_i | 0)_{\mathbf{H}_i^*}$$

Enc (ciph)

“Compress” in $(\mathbf{F}, \mathbf{F}^*)$

$$\vec{c}_{i,att} = (\cdots | \psi_i | \tau \Delta x_i z_{att} | 0,0,0,0)_{\mathbf{F}}$$

“Duplicate” $(\mathbf{H}_i, \mathbf{H}_i^*)$

$$\vec{c}_{i,0} = (\omega | \omega' | \psi_i | \tau \Delta x_i | 0)_{\mathbf{H}_i}$$

↑
Hash(tag) as RO

Towards Multi-Client - “Compress-and-Duplicate”

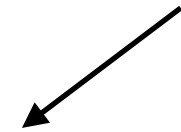
MC-AB-IPFE for LSSS - Perform masking steps
in linear total communication

Extract (key)

“Compress” in $(\mathbf{F}, \mathbf{F}^*)$

$$\vec{k}_{i,att}^* = (\cdots | a_{att} | a'_{att} y_i / z_{att} | 0, 0, 0, 0)_{\mathbf{F}^*}$$

$\Theta(1)$ coordinates



“Duplicate” $(\mathbf{H}_i, \mathbf{H}_i^*)$

$$\vec{k}_{i,0}^* = (\cdots | a_0 | r'_0 y_i | 0)_{\mathbf{H}_i^*}$$

$\theta(1)$ coordinates
Per i

Enc (ciph)

“Compress” in $(\mathbf{F}, \mathbf{F}^*)$

$$\vec{c}_{i,att} = (\cdots | \psi_i | \tau \Delta x_i z_{att} | 0, 0, 0, 0)_{\mathbf{F}}$$



“Duplicate” $(\mathbf{H}_i, \mathbf{H}_i^*)$

$$\vec{c}_{i,0} = (\omega | \omega' | \psi_i | \tau \Delta x_i | 0)_{\mathbf{H}_i}$$

Hash(tag) as RO



Towards Multi-Client - “Compress-and-Duplicate”

MC-AB-IPFE for LSSS - Perform masking steps
in linear total communication

Extract (key)

“Compress” in $(\mathbf{F}, \mathbf{F}^*)$

$$\vec{k}_{i,att}^* = (\cdots | a_{att} | a'_{att} y_i / z_{att} | 0,0,0,0)_{\mathbf{F}^*}$$

$\Theta(1)$ coordinates
↙

“Duplicate” $(\mathbf{H}_i, \mathbf{H}_i^*)$

$$\vec{k}_{i,0}^* = (\cdots | a_0 | r'_0 y_i | 0)_{\mathbf{H}_i^*}$$

Enc (ciph)

“Compress” in $(\mathbf{F}, \mathbf{F}^*)$

$$\vec{c}_{i,att} = (\cdots | \psi_i | \tau \Delta x_i z_{att} | 0,0,0,0)_{\mathbf{F}}$$

“Duplicate” $(\mathbf{H}_i, \mathbf{H}_i^*)$

$$\vec{c}_{i,0} = (\omega | \omega' | \psi_i | \tau \Delta x_i | 0)_{\mathbf{H}_i}$$

↑
Hash(tag) as RO

Towards Multi-Client - “Compress-and-Duplicate”

MC-AB-IPFE for LSSS - Perform masking steps
in linear total communication

Extract (key)

“Compress” in $(\mathbf{F}, \mathbf{F}^*)$

$$\vec{k}_{i,att}^* = (\cdots | a_{att} | a'_{att} y_i / z_{att} | 0,0,0,0)_{\mathbf{F}^*}$$

$\Theta(1)$ coordinates



“Duplicate”($\mathbf{H}_i, \mathbf{H}_i^*$)

$$\vec{k}_{i,0}^* = (\cdots | a_0 | r'_0 y_i | 0)_{\mathbf{H}_i^*}$$

$\Theta(1)$
aux coordinates
per i, att

Enc (ciph)

“Compress” in $(\mathbf{F}, \mathbf{F}^*)$

$$\vec{c}_{i,att} = (\cdots | \psi_i | \tau \Delta x_i z_{att} | 0,0,0,0)_{\mathbf{F}}$$

$$| \text{ciphertexts} | = 8nd + 5n$$

“Duplicate”($\mathbf{H}_i, \mathbf{H}_i^*$)

$$\vec{c}_{i,0} = (\omega | \omega' | \psi_i | \tau \Delta x_i | 0)_{\mathbf{H}_i}$$

↑
Hash(tag) as RO

Towards Multi-Client - “Compress-and-Duplicate”

MC-AB-IPFE for LSSS - Perform masking steps
in linear total communication

Extract (key)

“Compress” in $(\mathbf{F}, \mathbf{F}^*)$

$$\vec{k}_{i,att}^* = (\cdots | a_{att} | a'_{att} y_i / z_{att} | 0,0,0,0)_{\mathbf{F}^*}$$

$\Theta(1)$ coordinates
↙

“Duplicate” $(\mathbf{H}_i, \mathbf{H}_i^*)$

$$\vec{k}_{i,0}^* = (\cdots | a_0 | r'_0 y_i | 0)_{\mathbf{H}_i^*}$$

Enc (ciph)

“Compress” in $(\mathbf{F}, \mathbf{F}^*)$

$$\vec{c}_{i,att} = (\cdots | \psi_i | \tau \Delta x_i z_{att} | 0,0,0,0)_{\mathbf{F}}$$

“Duplicate” $(\mathbf{H}_i, \mathbf{H}_i^*)$

$$\vec{c}_{i,0} = (\omega | \omega' | \psi_i | \tau \Delta x_i | 0)_{\mathbf{H}_i}$$

↑
Hash(tag) as RO

Towards Multi-Client - “Compress-and-Duplicate”

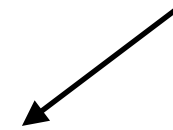
MC-AB-IPFE for LSSS - Perform masking steps
in linear total communication

Extract (key)

“Compress” in $(\mathbf{F}, \mathbf{F}^*)$

$$\vec{k}_{i,att}^* = (\cdots | a_{att} | a'_{att} y_i / z_{att} | 0,0,0,0)_{\mathbf{F}^*}$$

$\Theta(1)$ coordinates



“Duplicate”($\mathbf{H}_i, \mathbf{H}_i^*$)

$$\vec{k}_{i,0}^* = (\cdots | a_0 | r'_0 y_i | 0)_{\mathbf{H}_i^*}$$

Enc (ciph)

“Compress” in $(\mathbf{F}, \mathbf{F}^*)$

$$\vec{c}_{i,att} = (\cdots | \psi_i | \tau \Delta x_i z_{att} | 0,0,0,0)_{\mathbf{F}}$$

“Duplicate”($\mathbf{H}_i, \mathbf{H}_i^*$)

$$\vec{c}_{i,0} = (\omega | \omega' | \psi_i | \tau \Delta x_i | 0)_{\mathbf{H}_i}$$

Hash(tag) as RO

Thanks to
DPVS
technicality:
Independent
Basis changes
On
independent
Blocks
(with indexing)

Research Question (1)

Controlling Functional Keys

- Definitional Framework for Multi-Client (MCFE) Setting
- Constructions from Dual Pairing Vector Spaces

Our Results:

**MCFE Candidates for Inner Products
with LSSS, achieving Adaptive Security
in ROM and Linear Total Communication**

Research Question (1)

Controlling Functional Keys

- Definitional Framework for Multi-Client (MCFE) Setting
- Constructions from Dual Pairing Vector Spaces

Our Results:

**MCFE Candidates for Inner Products
with LSSS, achieving Adaptive Security
in ROM and Linear Total Communication**

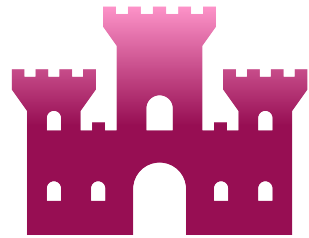
Thanks to
“Compress-and-Duplicate”



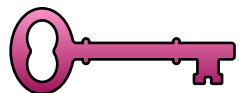
Outline

- Introduction & Main Techniques Overview
- Controlling Functional Keys
- **Optimal** Security Notion
 - Definitional Framework to Refine Decentralized MCFE Security
 - Constructions from DPVS
- From Secret-Key to Public-Key Encryption Schemes
- Conclusion and Future Works

This chapter: Refinements of Security



Func: G ,
Policy: $\text{Spade}(\cdot)$



(2)
Optimal
Security



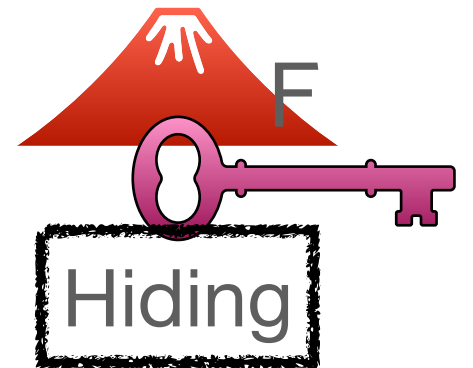
[CDGPP'18]

$\text{Corrupt}(i)$

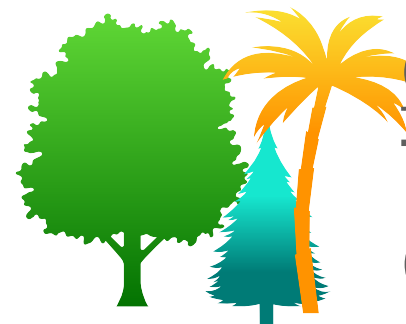
$\Rightarrow x_{0,i} = x_{1,i}$



Justified?



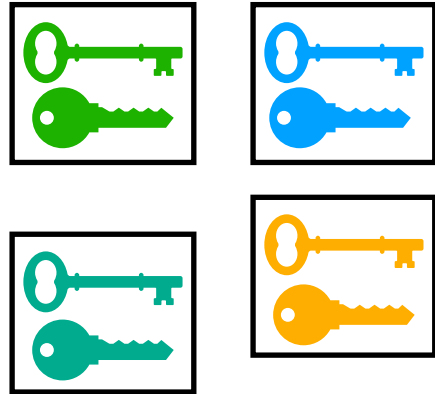
Other misc:
Blind Sign,
CALBO-TDFs



Ongoing:
Traceable FE,
CCA-security

DMCFE: Corruption

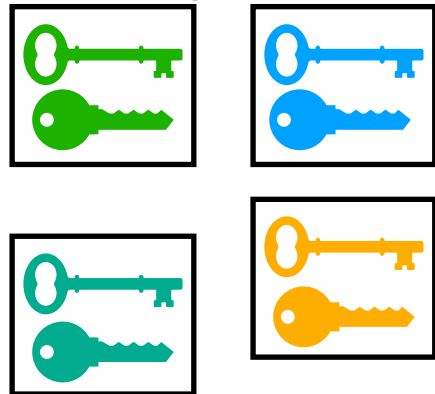
“All-or-Nothing”
Corruptible
by 🍆



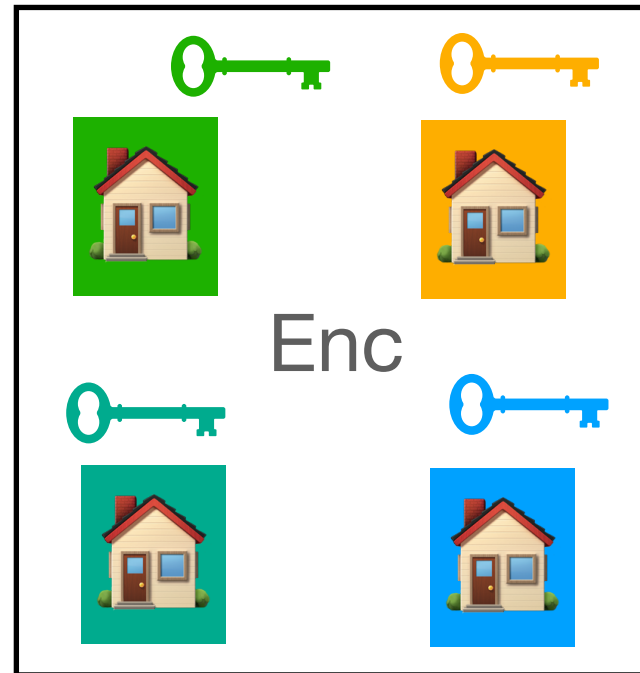
DMCFE: Corruption

“All-or-Nothing”
Corruptible

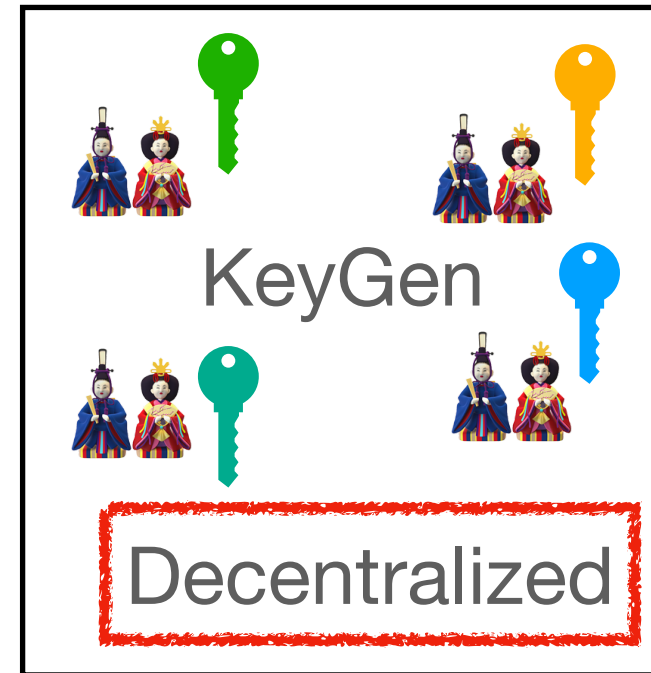
by 



But ...



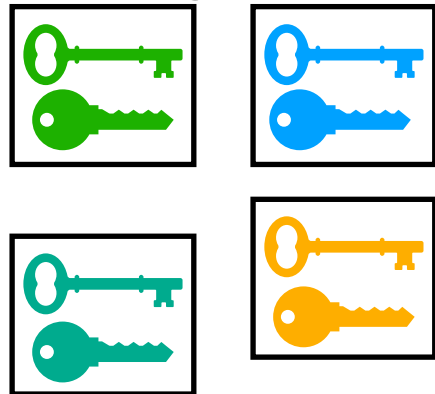
≠



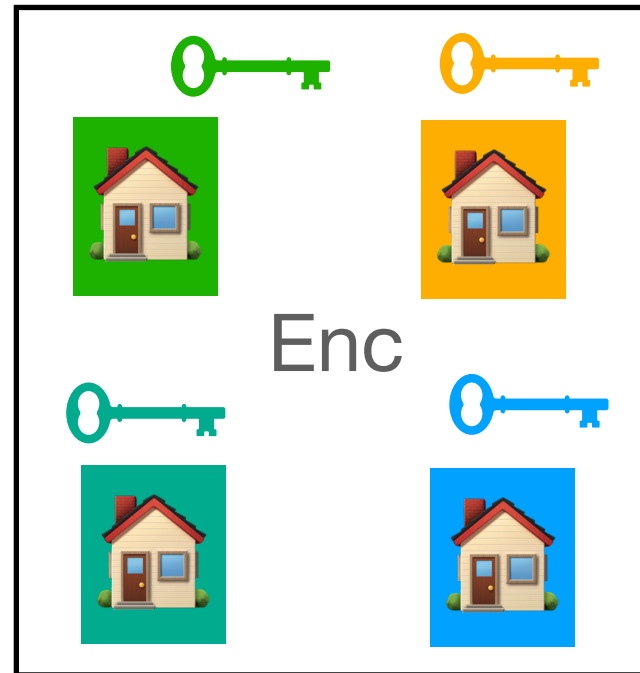
DMCFE: Corruption

“All-or-Nothing”
Corruptible

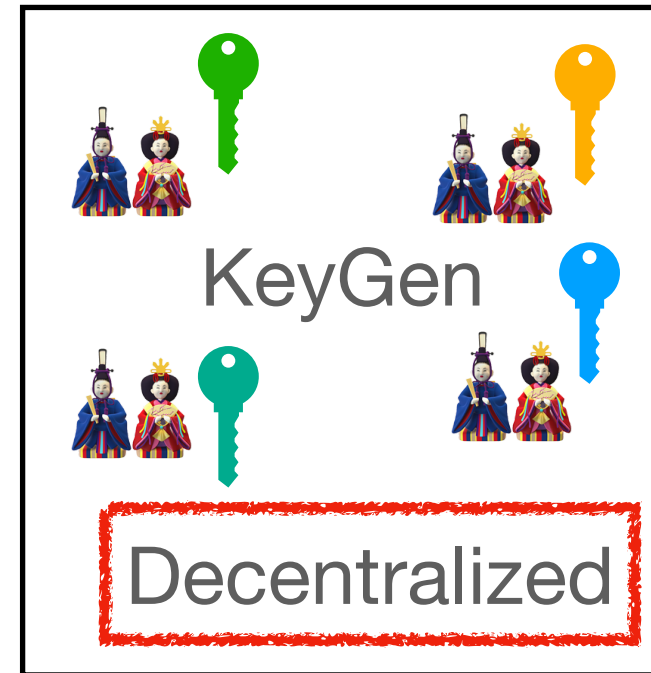
by 



But ...



≠



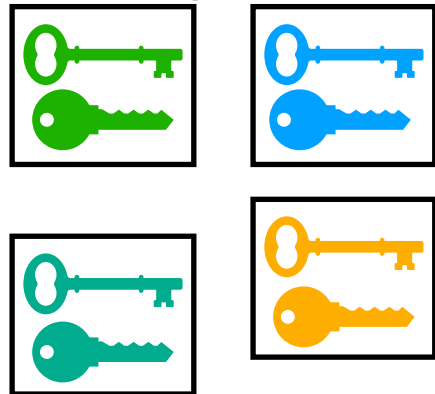
Previous works: $F_{\vec{y}}(\vec{x}) = \langle \vec{x}, \vec{y} \rangle$ only

- [CDGPP'18; LT'19]
- [ABKW'19; ABG'19]

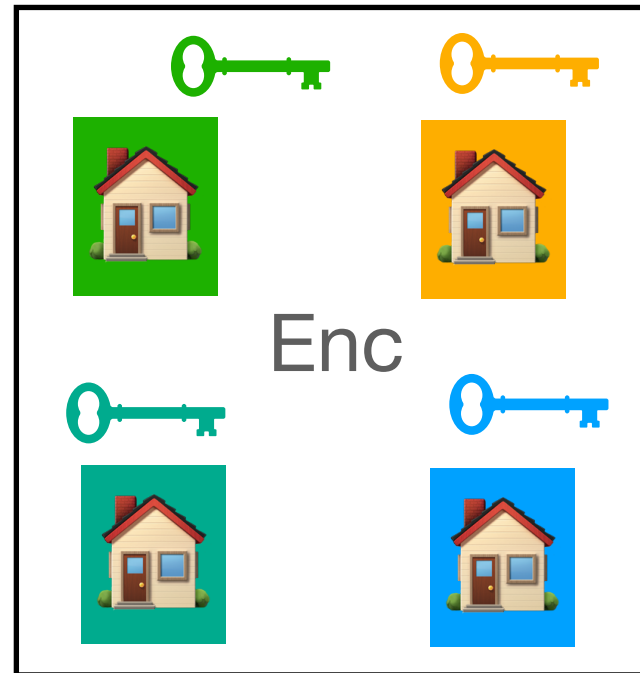
DMCFE: Corruption

“All-or-Nothing”
Corruptible

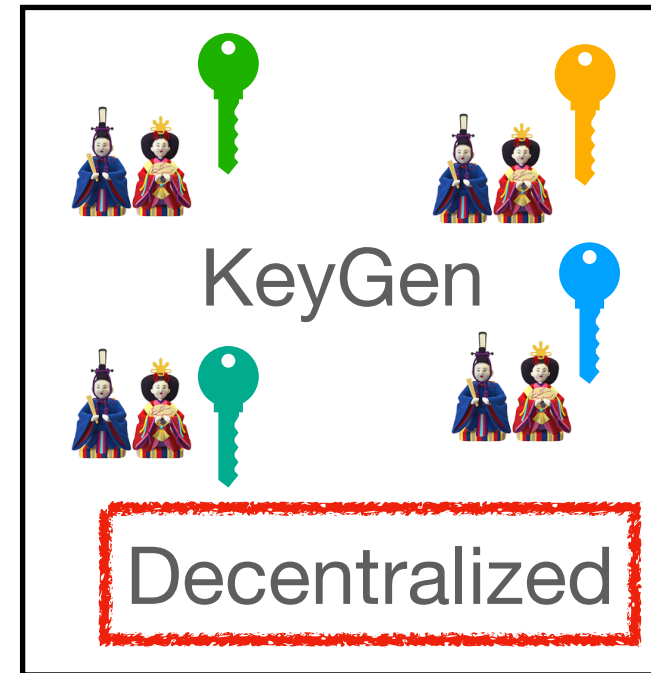
by 



But ...



≠



Previous works: $F_{\vec{y}}(\vec{x}) = \langle \vec{x}, \vec{y} \rangle$ only

- [CDGPP'18; LT'19]

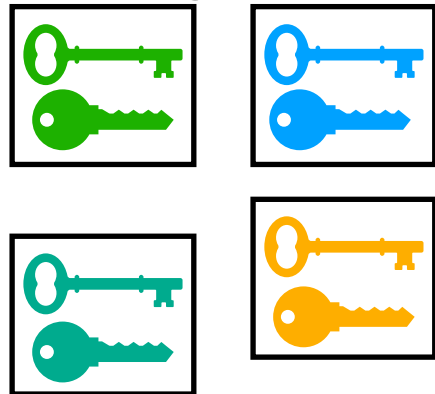


- [ABKW'19; ABG'19]

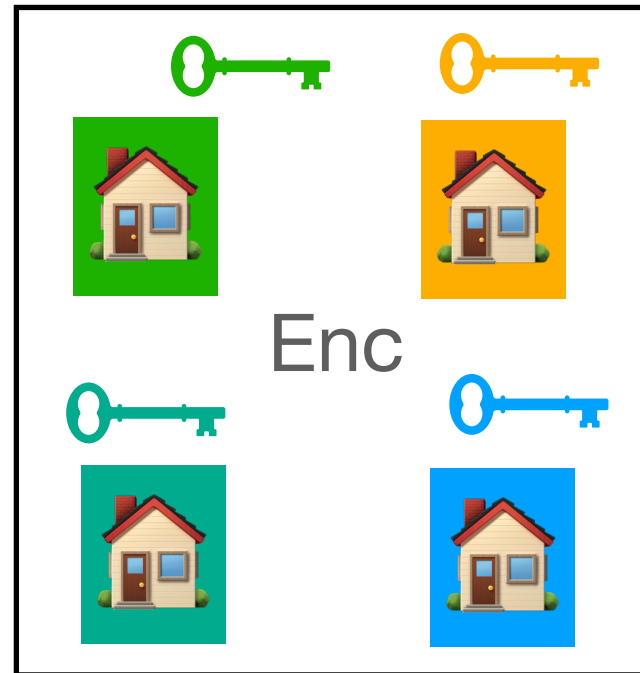
DMCFE: Corruption

“All-or-Nothing”
Corruptible

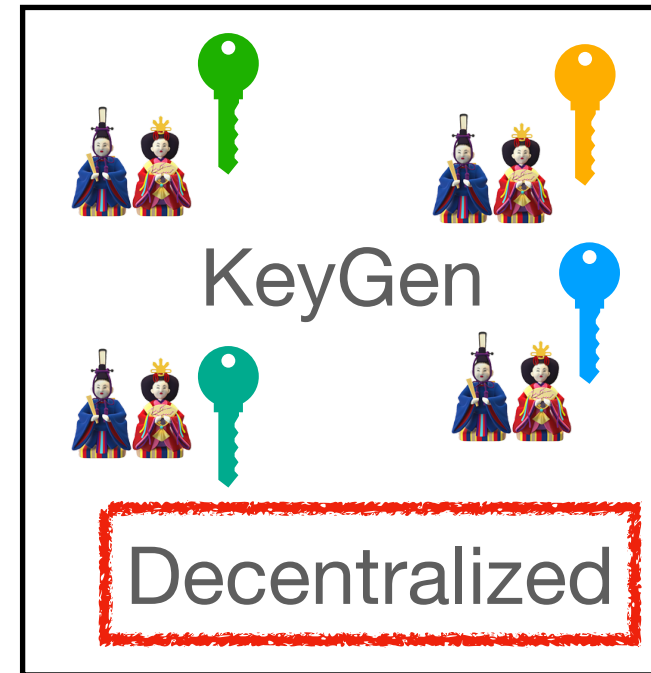
by 



But ...



≠

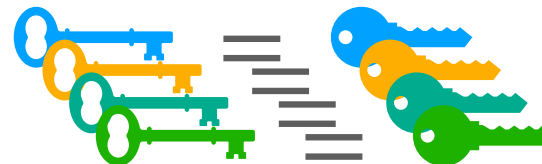


Previous works: $F_{\vec{y}}(\vec{x}) = \langle \vec{x}, \vec{y} \rangle$ only

- [CDGPP'18; LT'19]



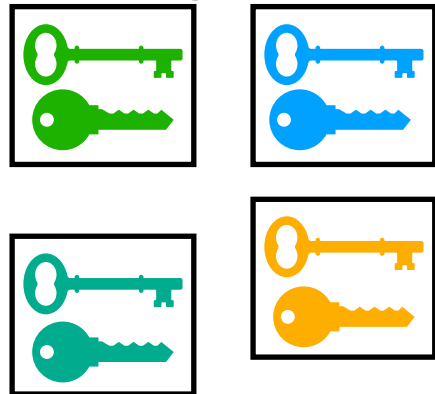
- [ABKW'19; ABG'19]



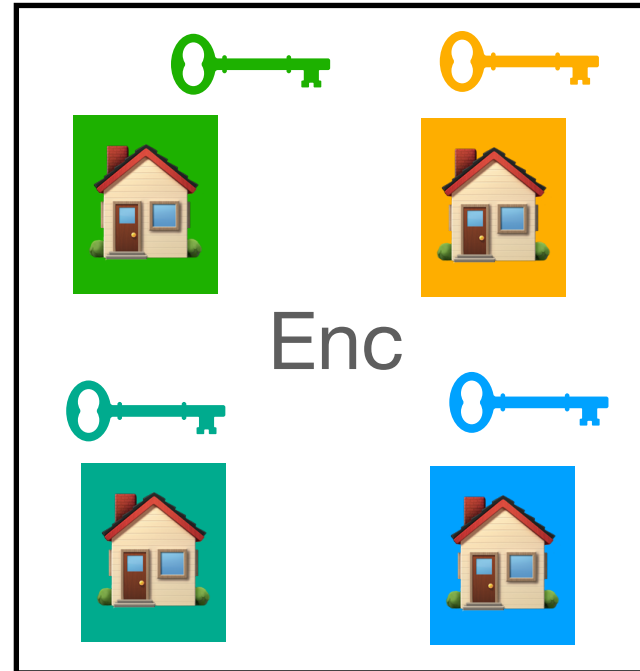
DMCFE: Corruption

“All-or-Nothing”
Corruptible

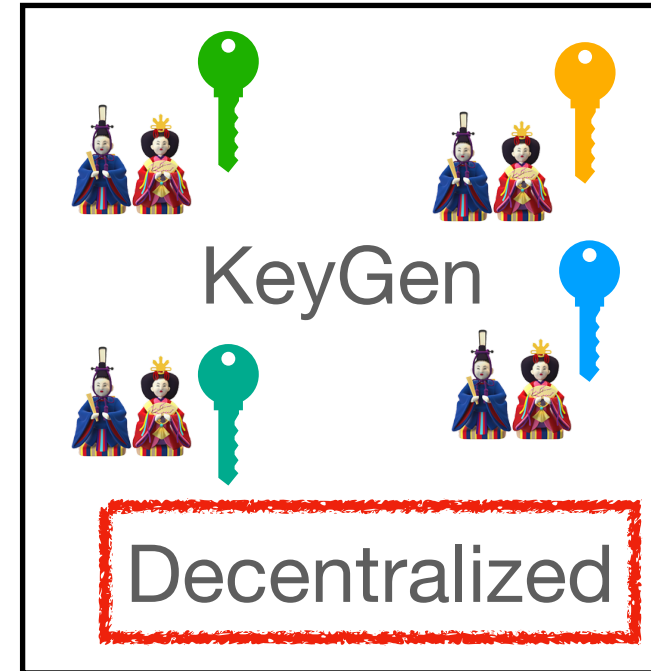
by 



But ...



≠

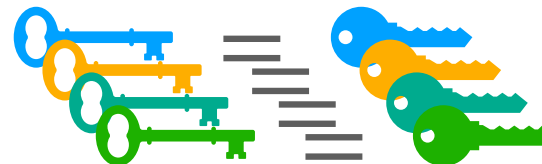


Previous works: $F_{\vec{y}}(\vec{x}) = \langle \vec{x}, \vec{y} \rangle$ only

- [CDGPP'18; LT'19]



- [ABKW'19; ABG'19]



Quadratic?

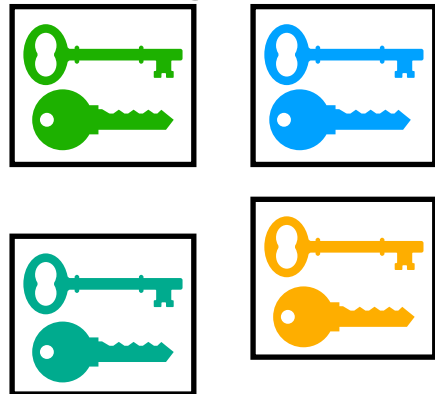
$$F_{\mathbf{A}}(\vec{x}) = \vec{x}^{\top} \mathbf{A} \vec{x}$$

 ≠ # 

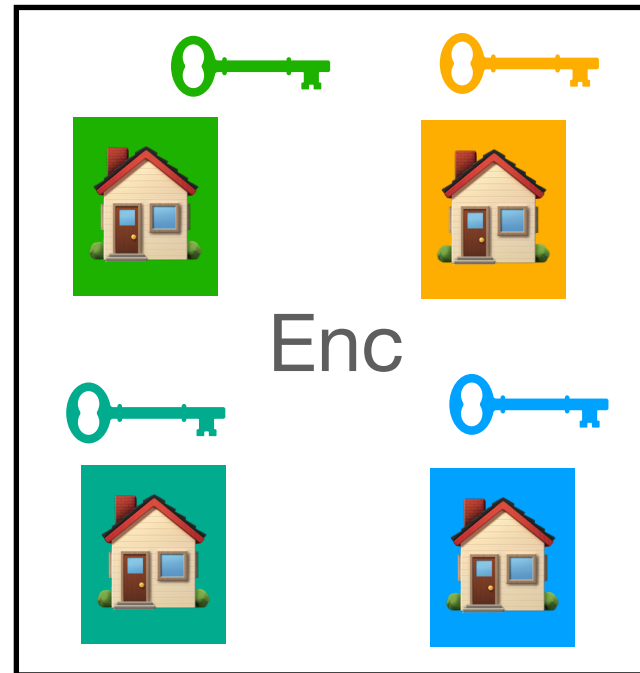
DMCFE: Corruption

“All-or-Nothing”
Corruptible

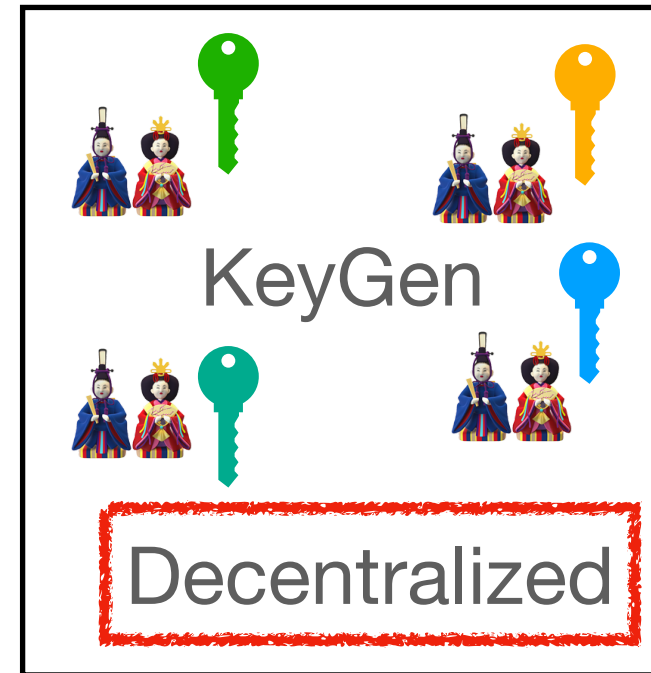
by 



But ...



≠

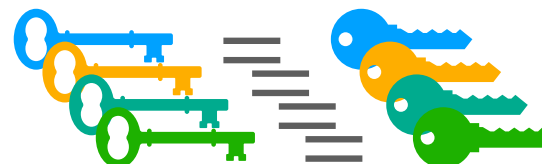


Previous works: $F_{\vec{y}}(\vec{x}) = \langle \vec{x}, \vec{y} \rangle$ only

- [CDGPP'18; LT'19]



- [ABKW'19; ABG'19]



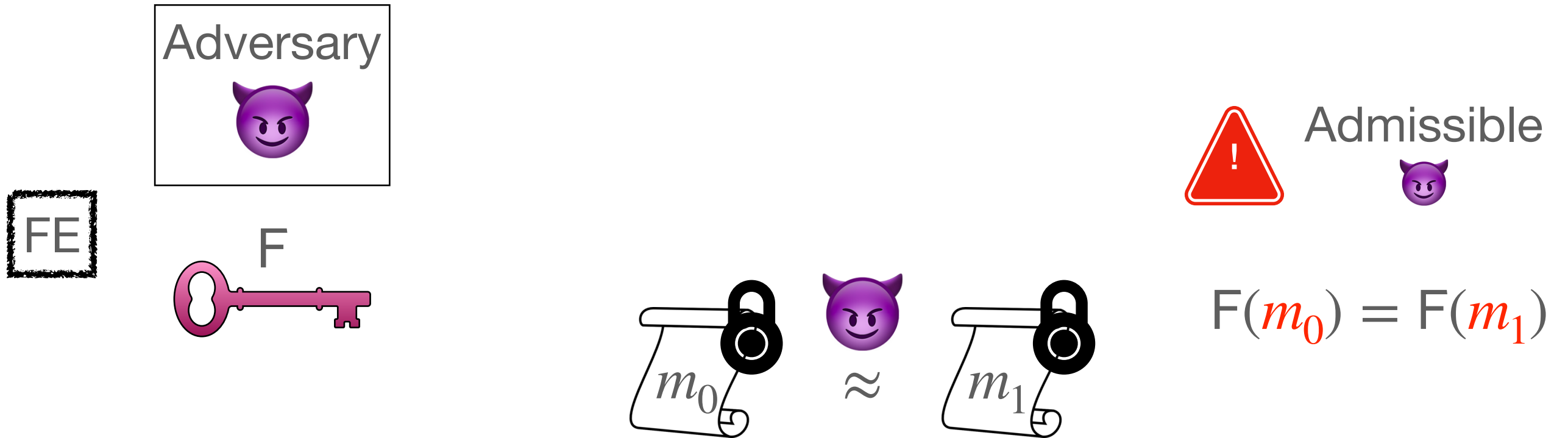
Quadratic?

$$F_{\mathbf{A}}(\vec{x}) = \vec{x}^{\top} \mathbf{A} \vec{x}$$

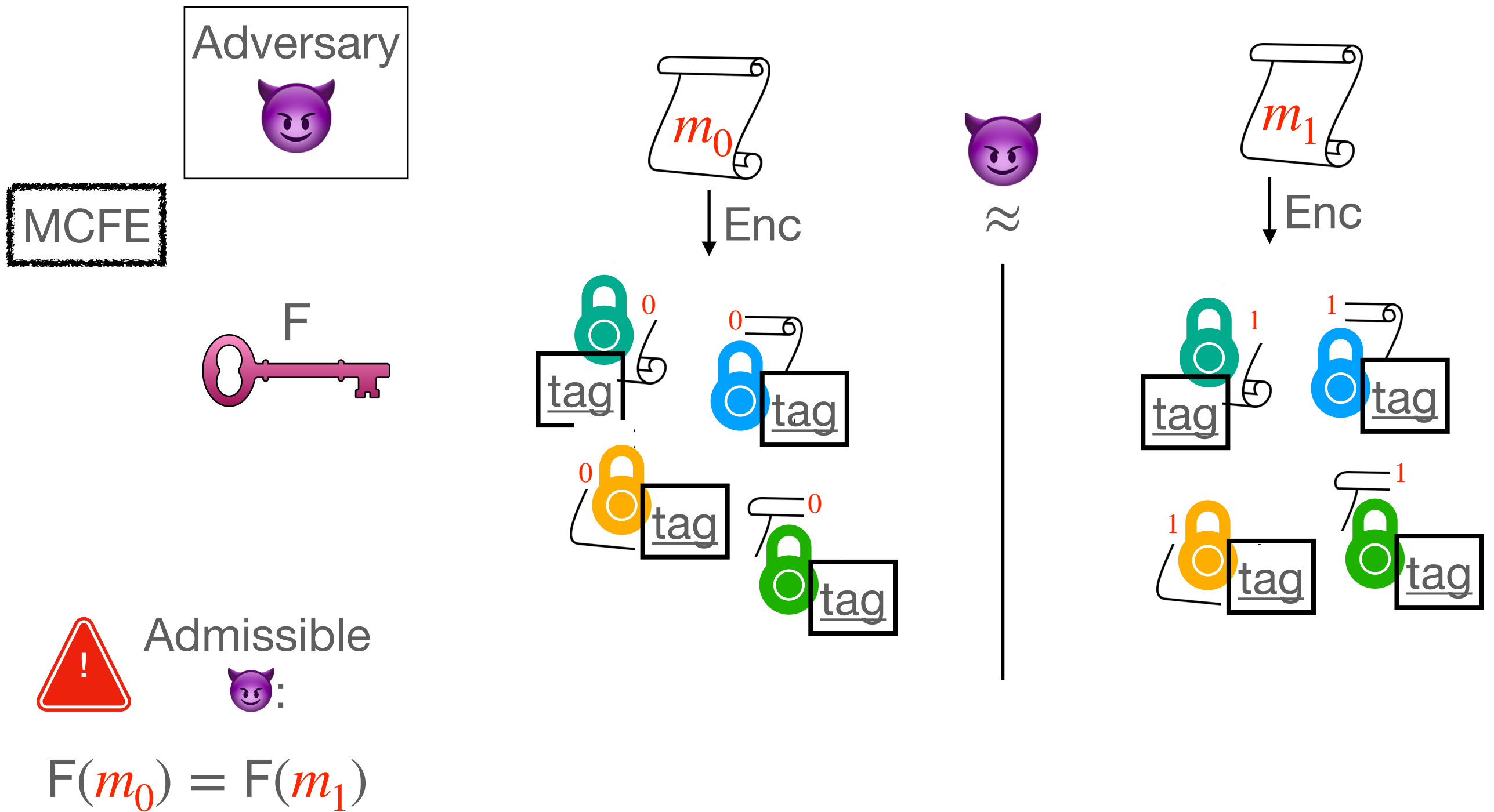
 ≠ # 

⇒ Separated Corruption?

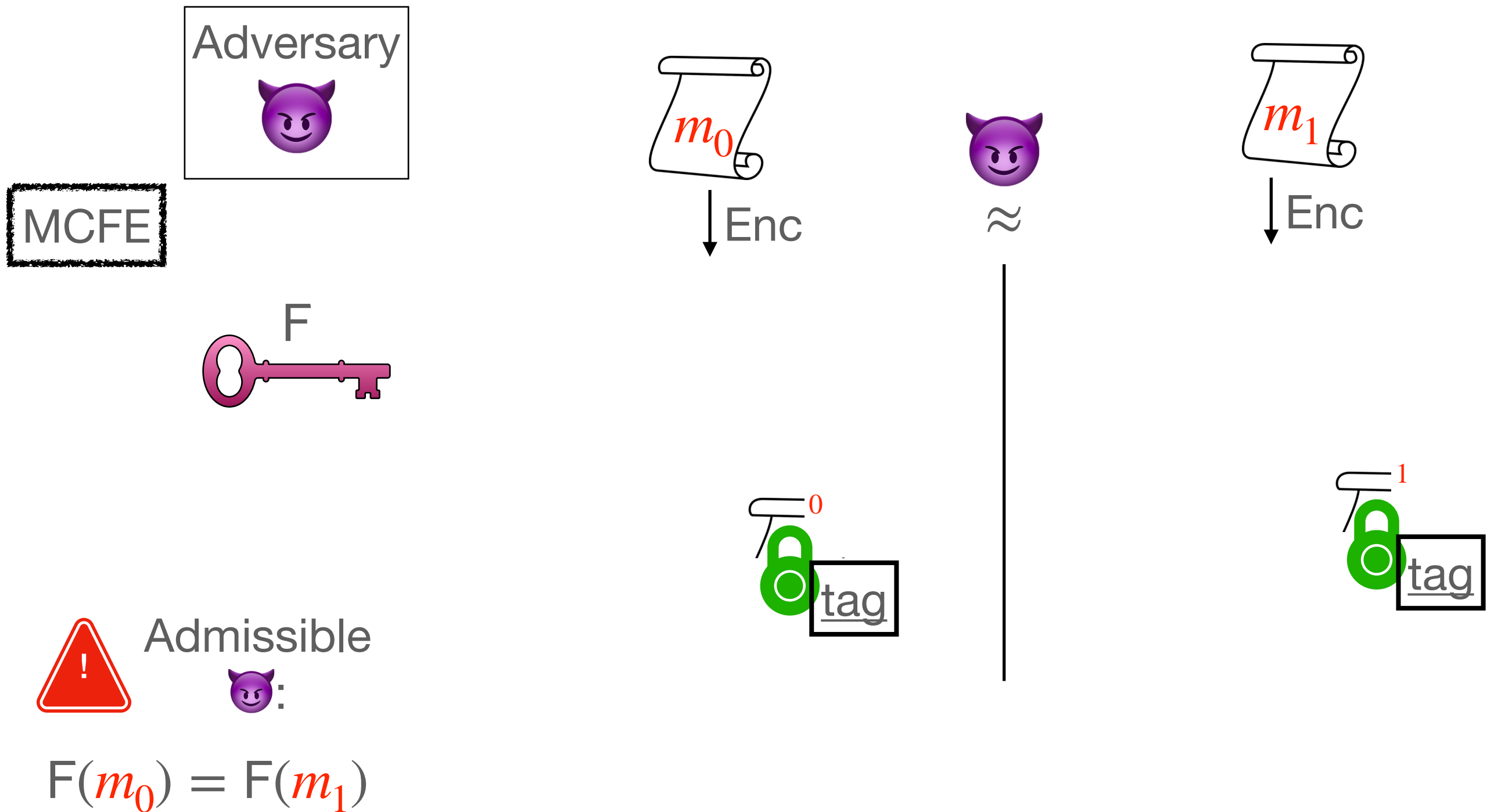
DMCFE: Admissibility



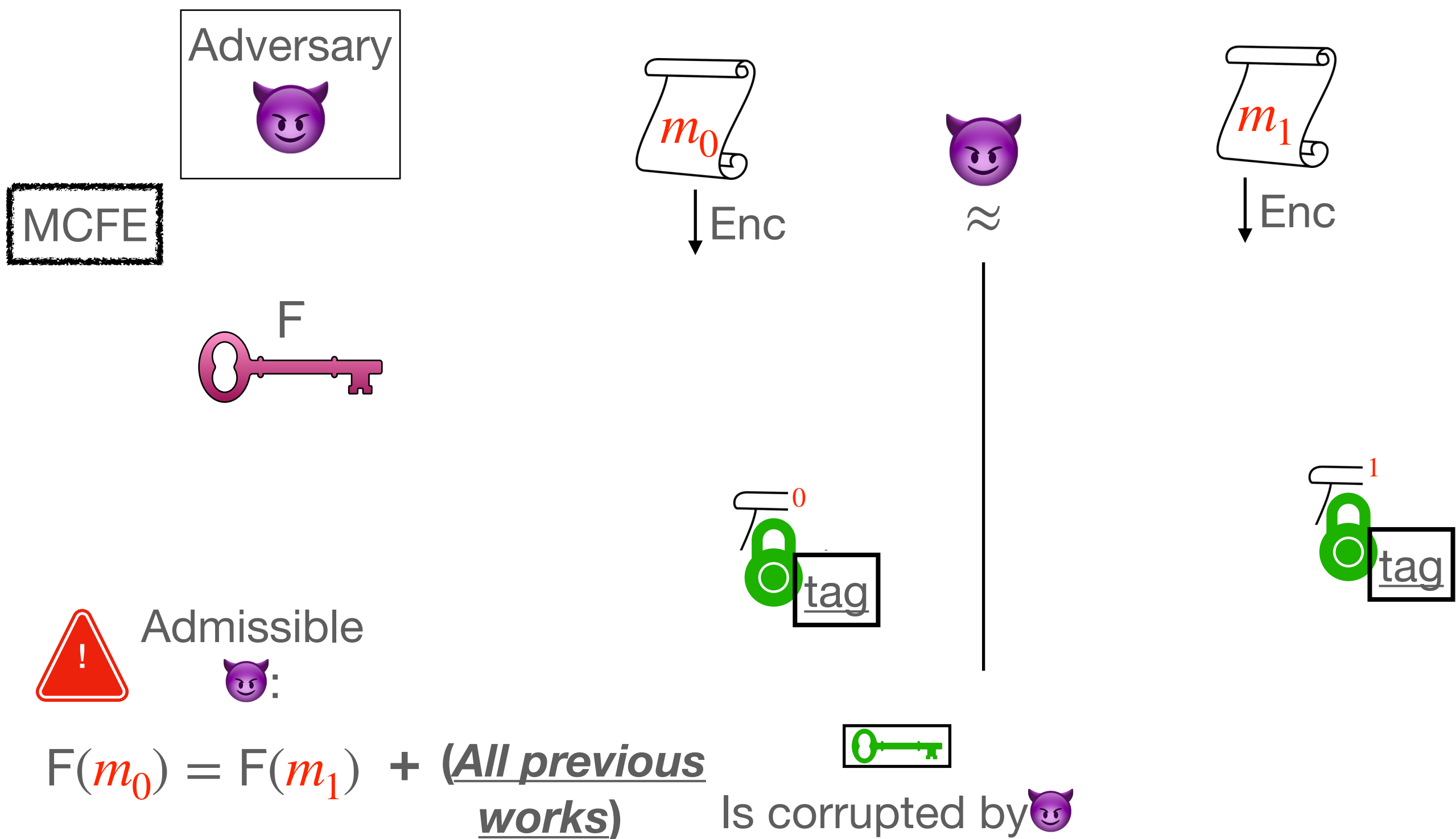
DMCFE: Admissibility



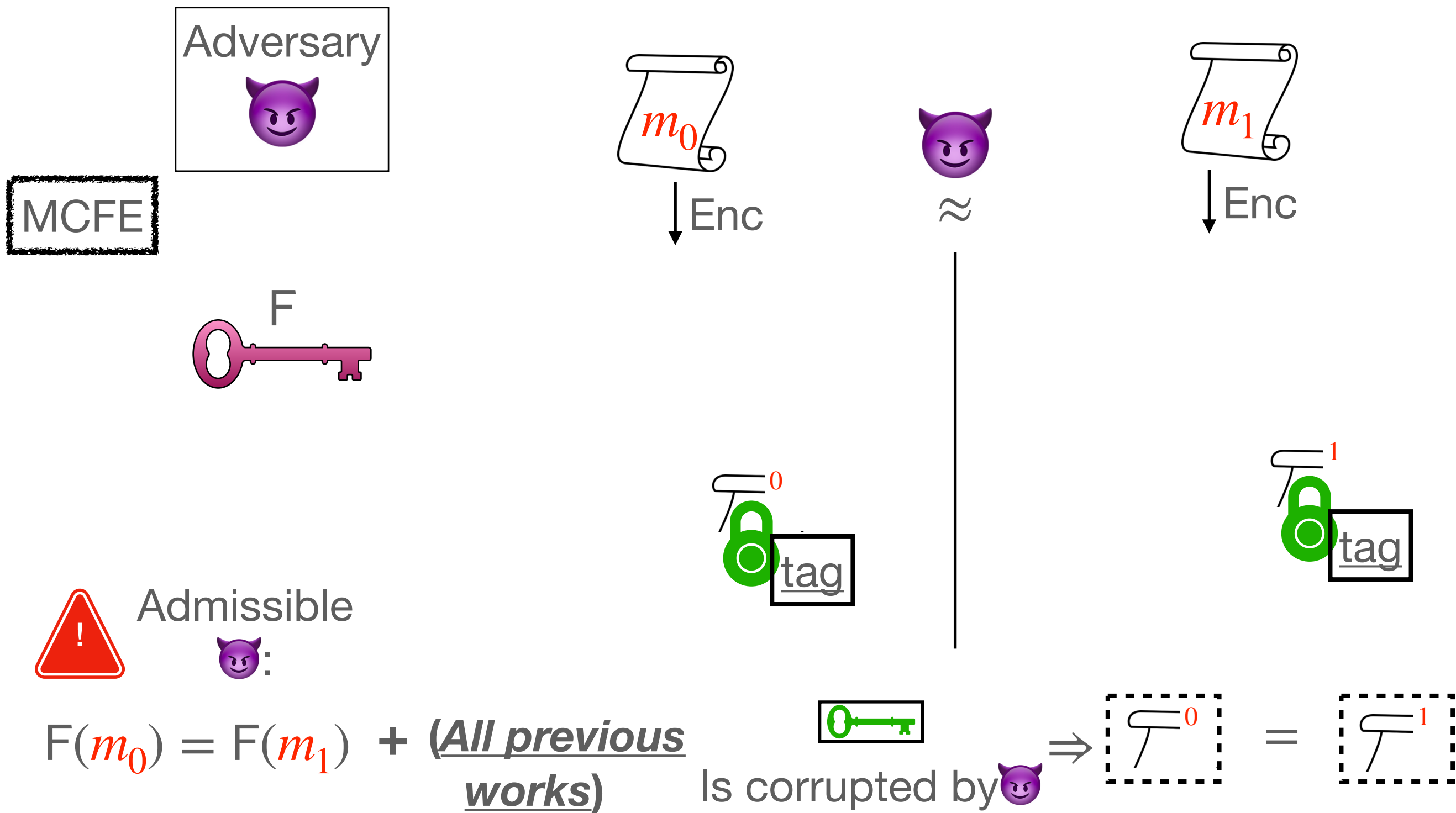
DMCFE: Admissibility



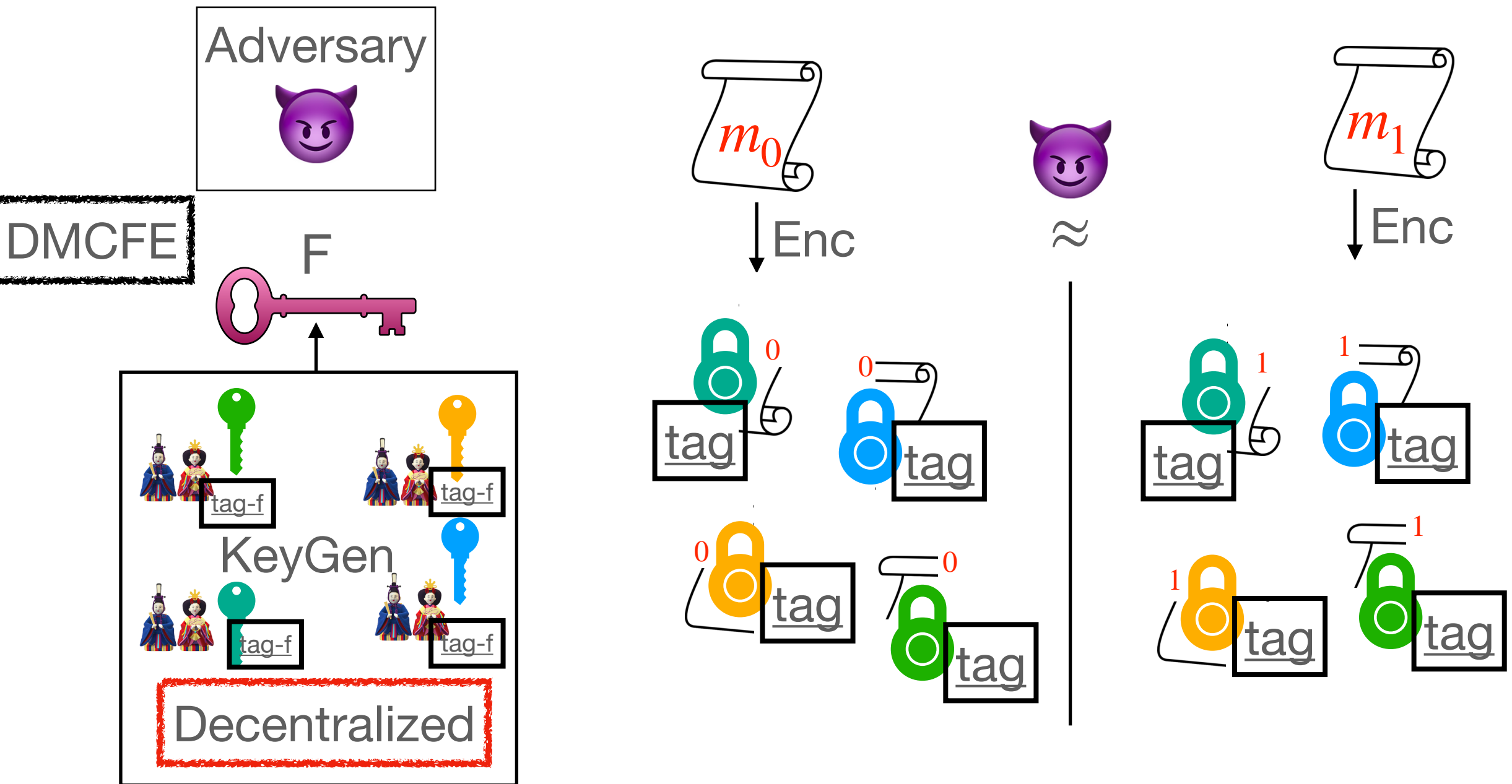
DMCFE: Admissibility



DMCFE: Admissibility



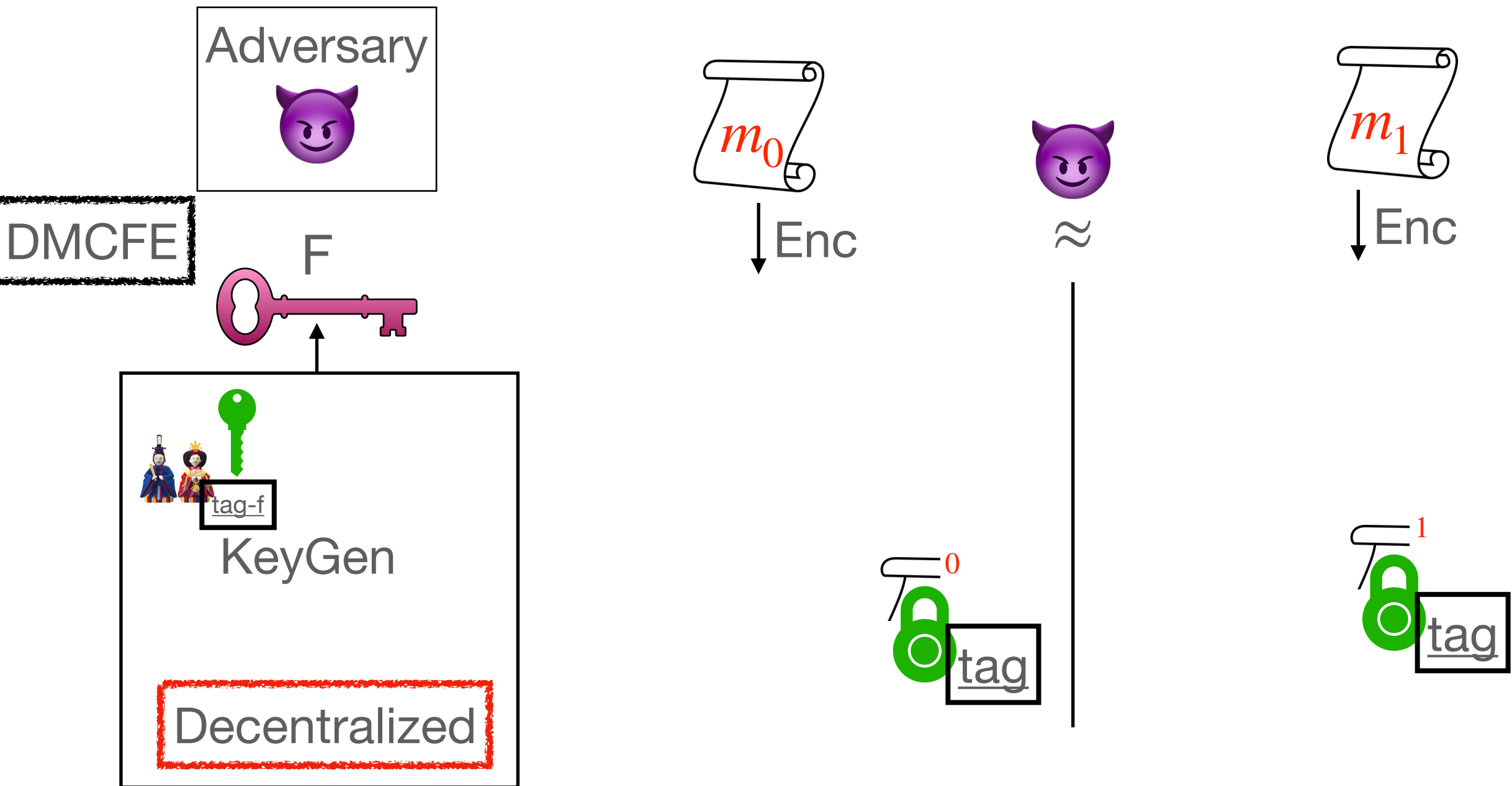
DMCFE: Admissibility



⚠ Admissible
👹:

$$F(m_0) = F(m_1)$$

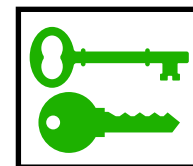
DMCFE: Admissibility



Admissible

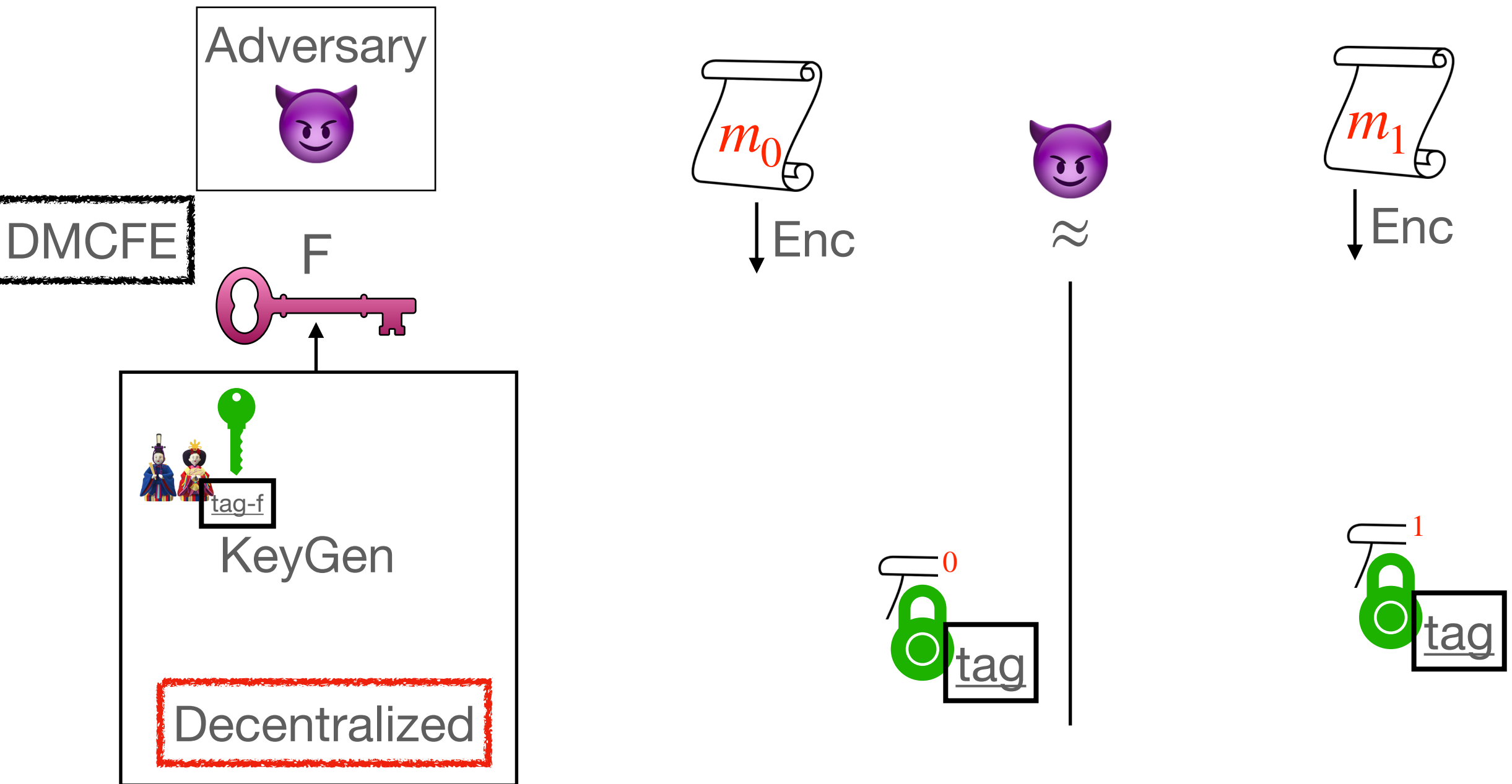


$$F(m_0) = F(m_1) + \text{(All previous works)}$$



Is corrupted by 

DMCFE: Admissibility



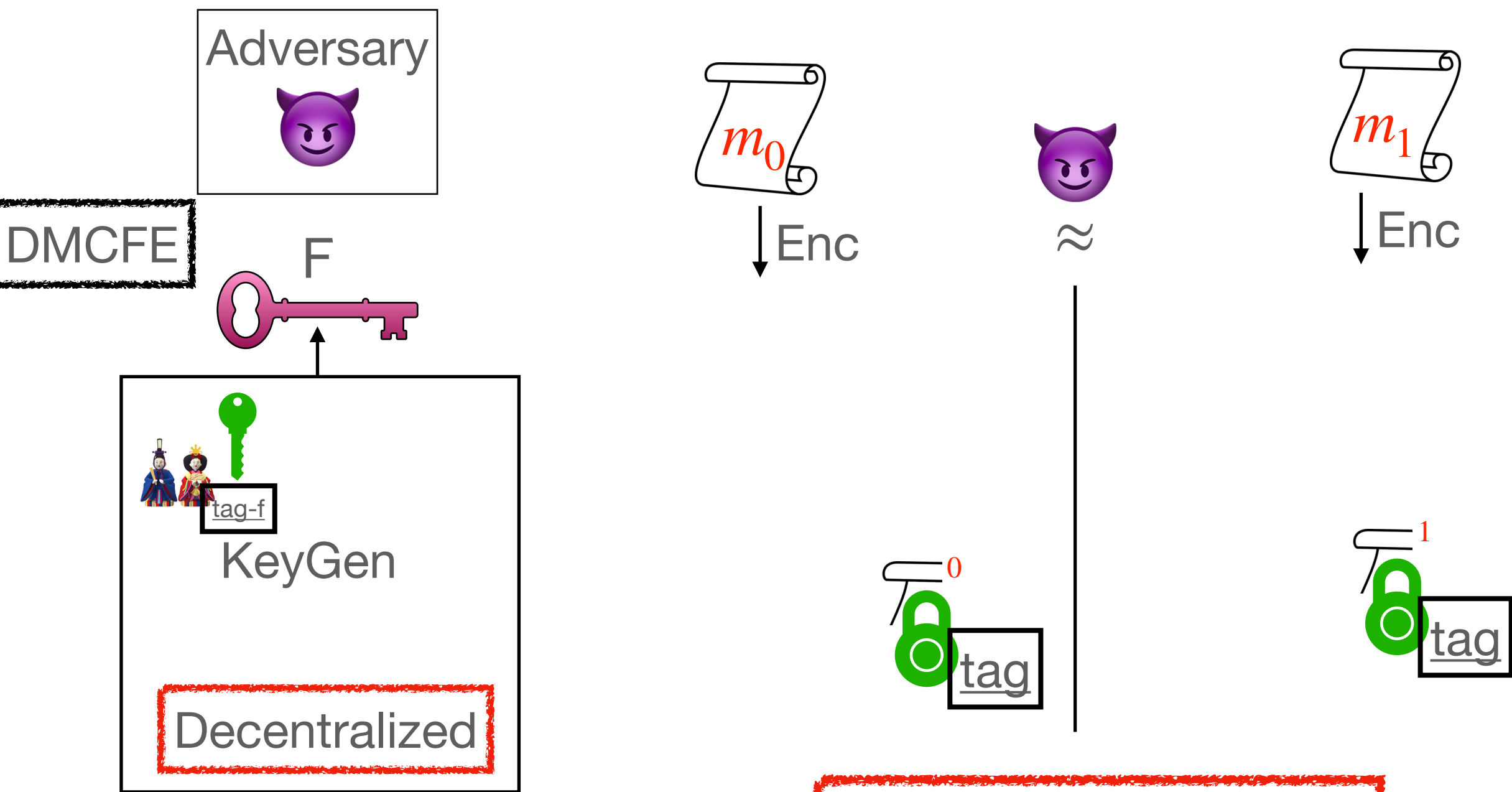
Admissible



$$F(m_0) = F(m_1) + \text{(All previous works)}$$

Is corrupted by  $\Rightarrow$ $\left(\begin{matrix} * \\ \tau^0 \end{matrix} \right) = \left(\begin{matrix} \tau^1 \end{matrix} \right)$

DMCFE: Admissibility



Admissible



$$F(m_0) = F(m_1) + \text{(All previous works)}$$

🤔 $\Rightarrow$ $(*)$ is removable?

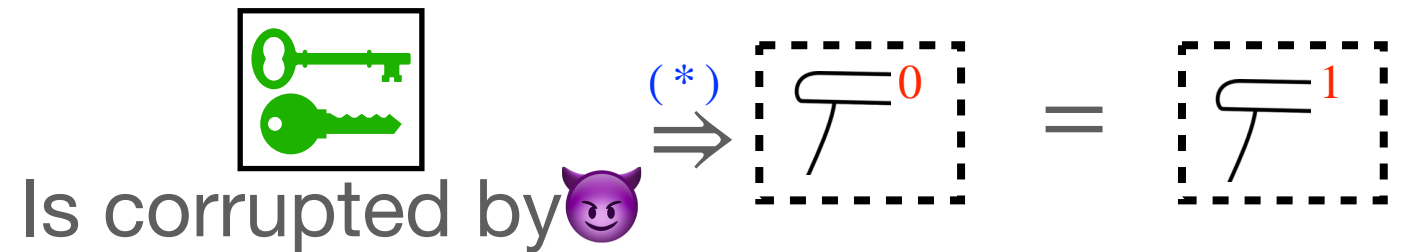
Is corrupted by  $\Rightarrow$ $(*)$ $\Rightarrow$ $\tau^0 = \tau^1$

Our Contributions

An Improved Security Model for DMCFE

Separated Corruption?

(*) is removable?



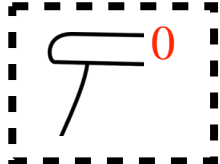
Our Contributions

An Improved Security Model for DMCFE

Separated Corruption?

Also studied
In [AGT21]

(*) is removable?

Is corrupted by  $\xRightarrow{(*)}$  = 

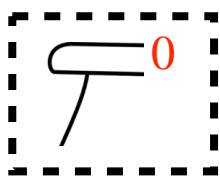
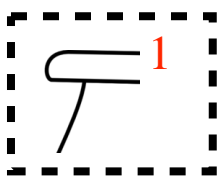
Our Contributions

An Improved Security Model for DMCFE

Separated Corruption?

Also studied
In [AGT21]

(*) is removable?

Is corrupted by  $\Rightarrow$  = 

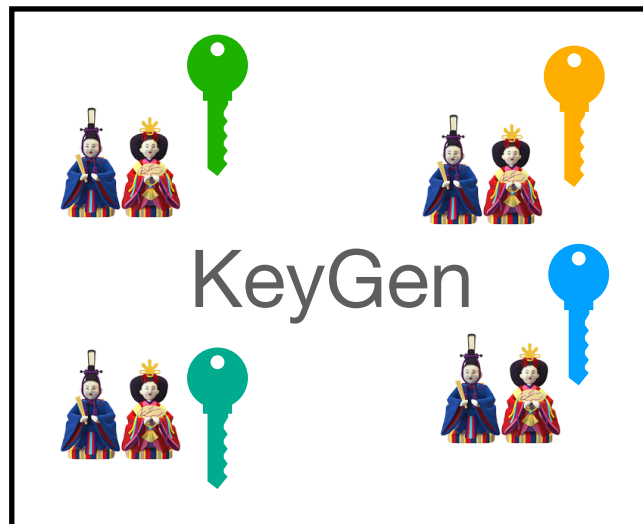
 First model strictly stronger than [CDGPP18]

- A new model with separated corruption of keys and less restrictive admissibility condition.
- A framework to argue the optimality of our new admissibility.
- A feasibility result for inner products.

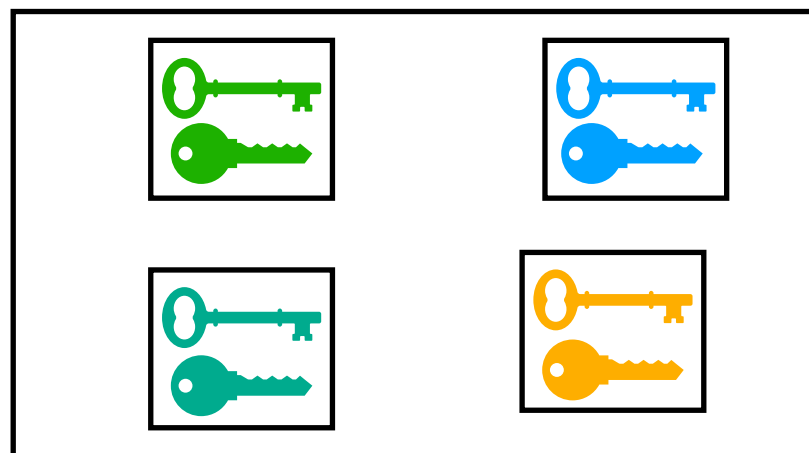
Definition

Security model [CDGPP'18]

Challenger: $b \xleftarrow{\$} \{0,1\}$



$$\vec{y} = (y_1, \dots, y_n)$$



Chall

Enc

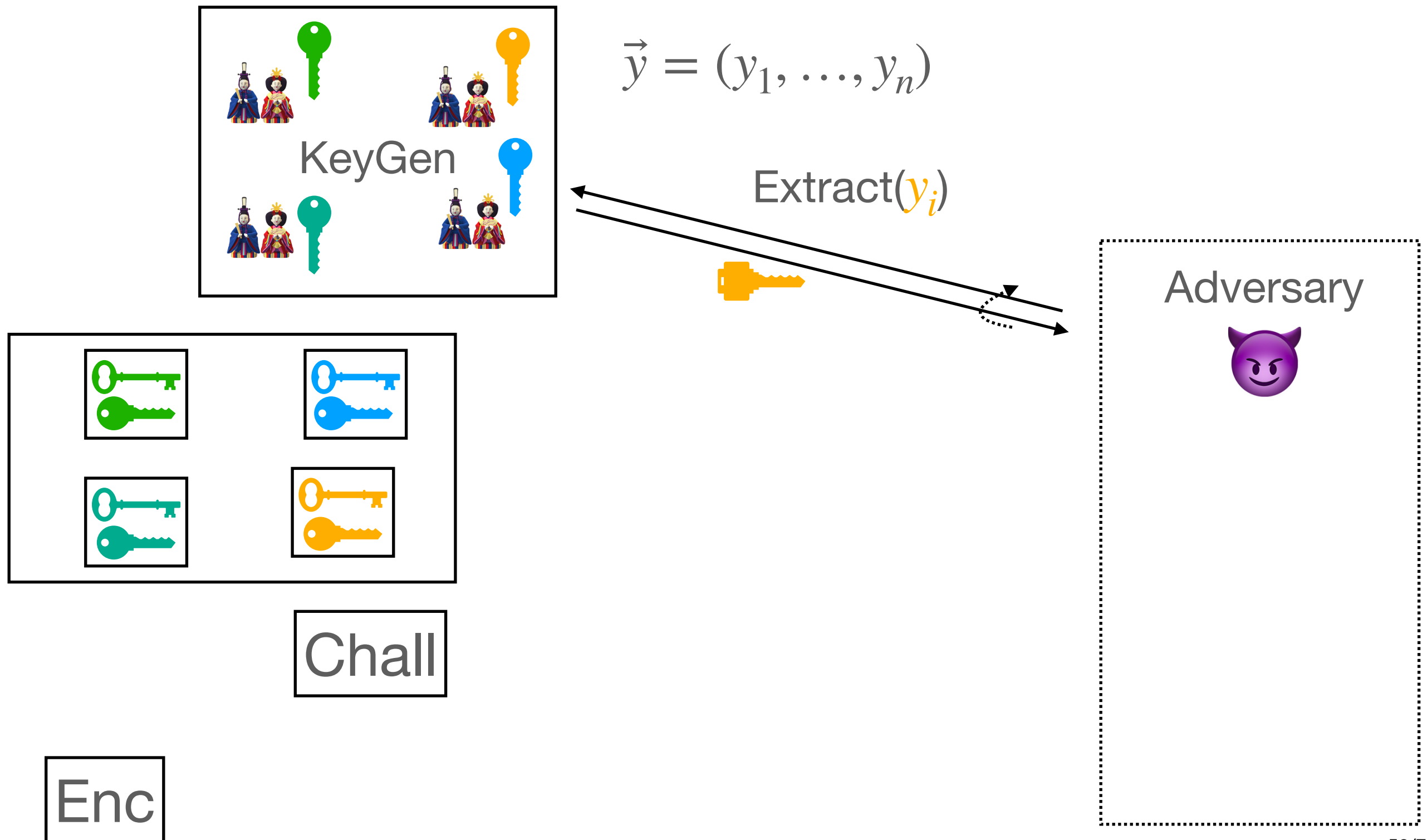
Adversary



Definition

Security model [CDGPP'18]

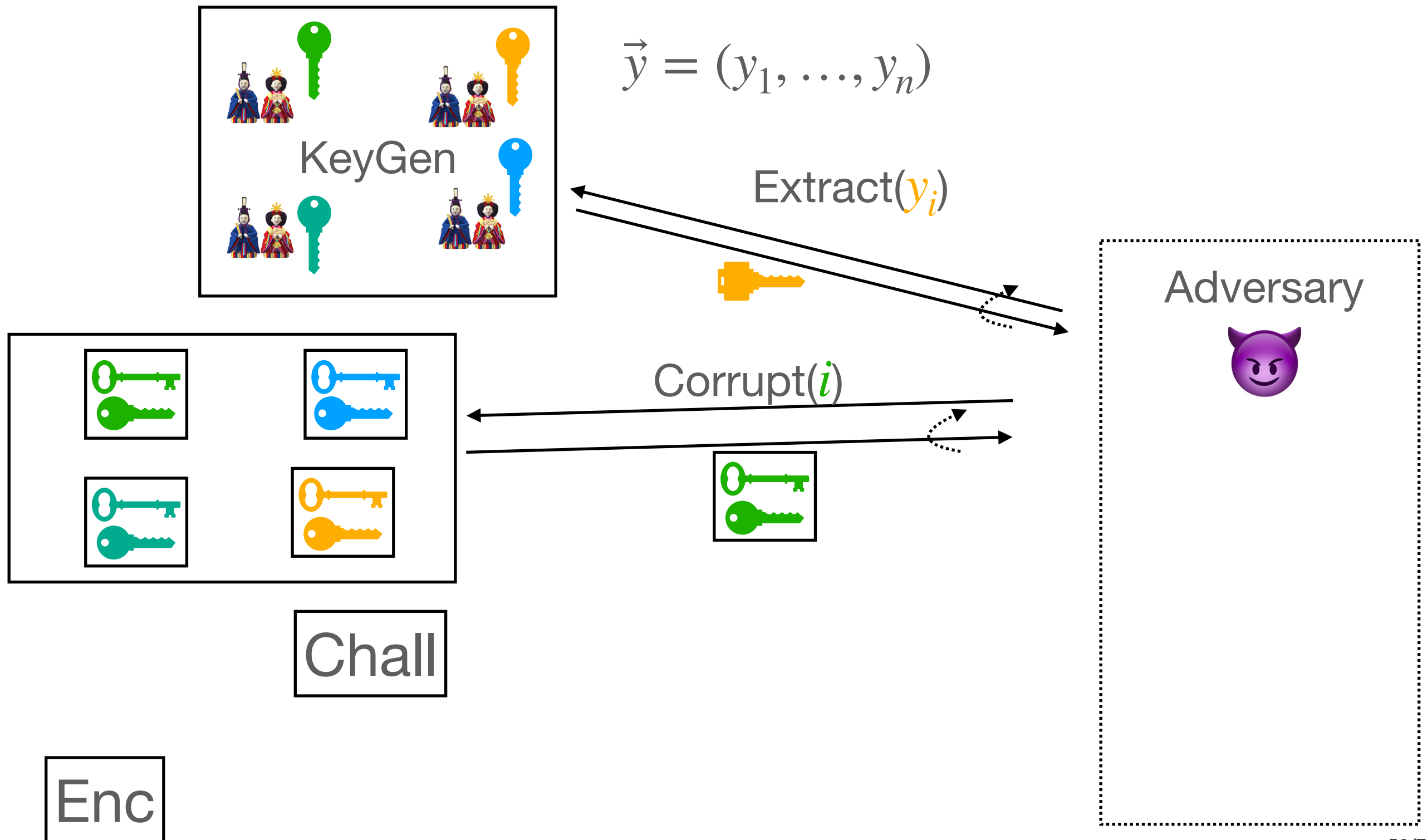
Challenger: $b \leftarrow^{\$} \{0,1\}$



Definition

Security model [CDGPP'18]

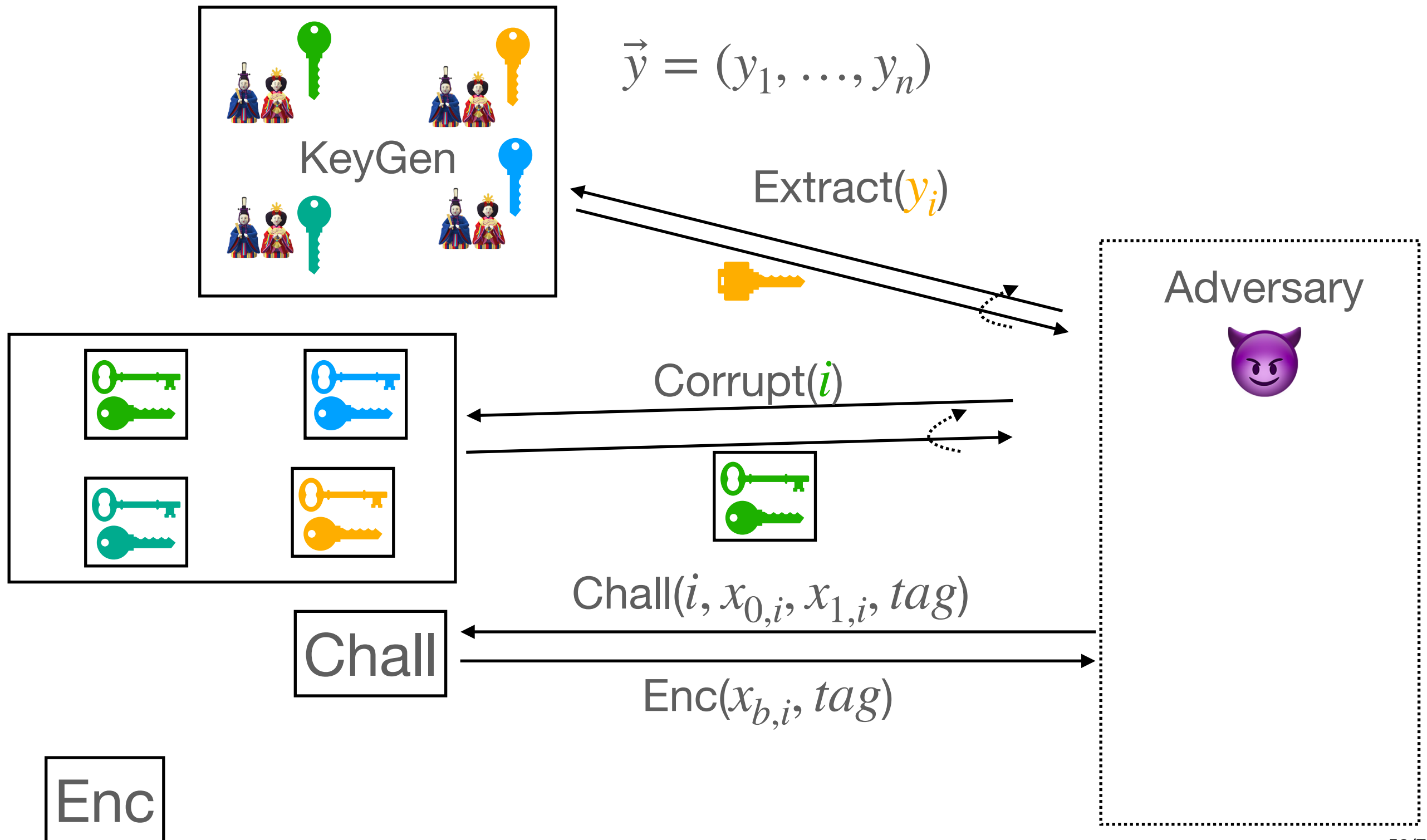
Challenger: $b \leftarrow^{\$} \{0,1\}$



Definition

Security model [CDGPP'18]

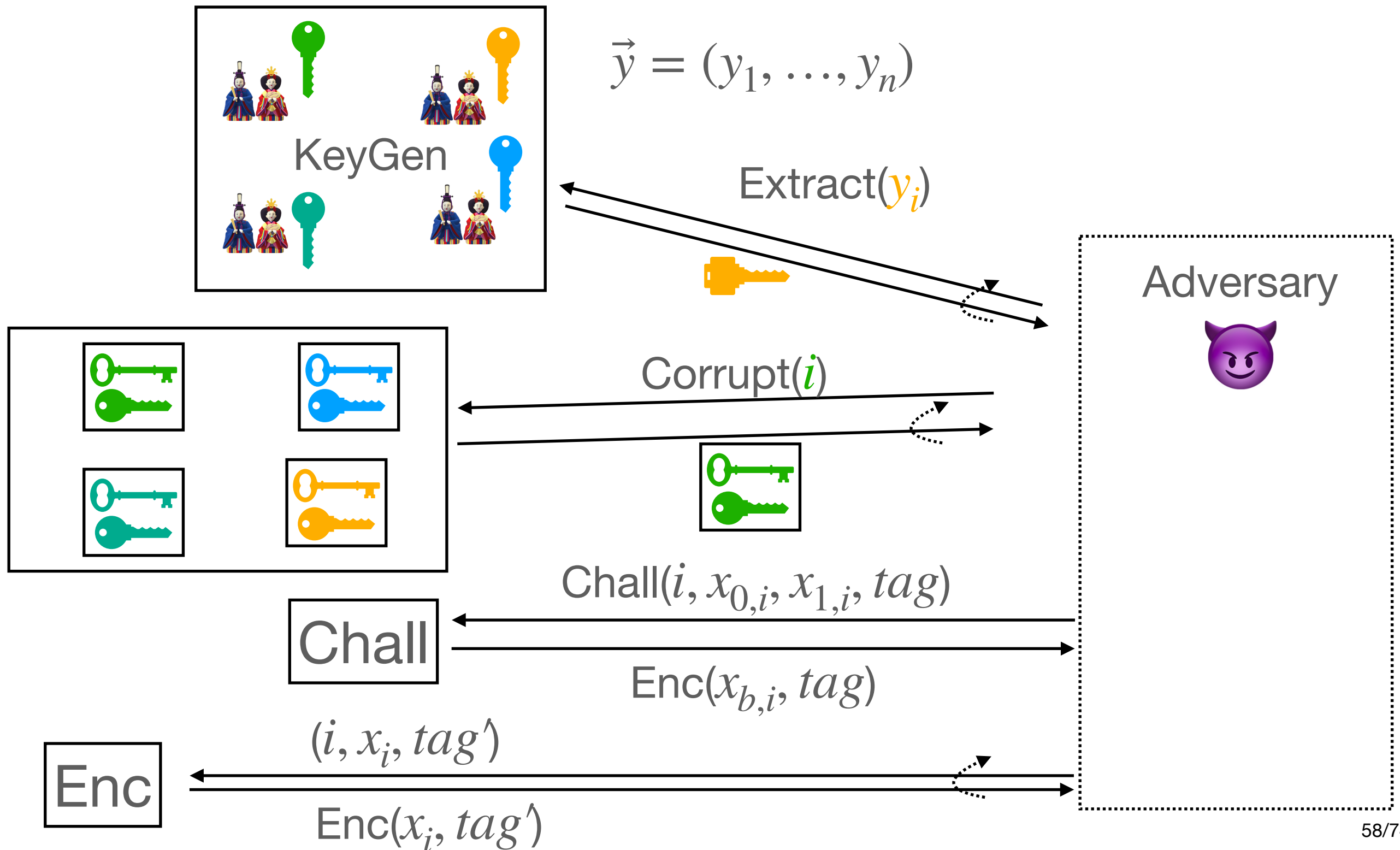
Challenger: $b \leftarrow^{\$} \{0,1\}$



Definition

Security model [CDGPP'18]

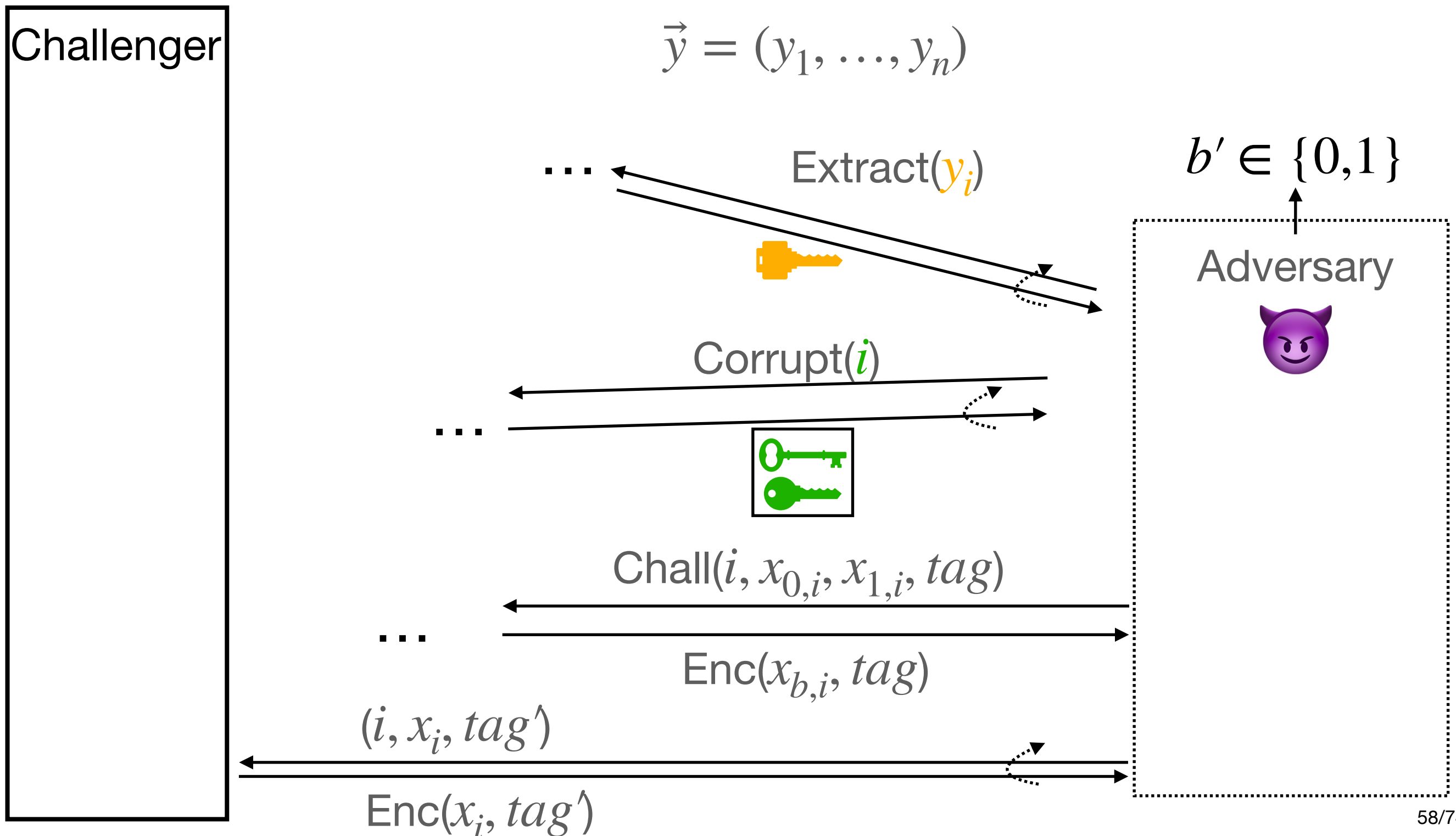
Challenger: $b \leftarrow^{\$} \{0,1\}$



Definition

Security model [CDGPP'18]

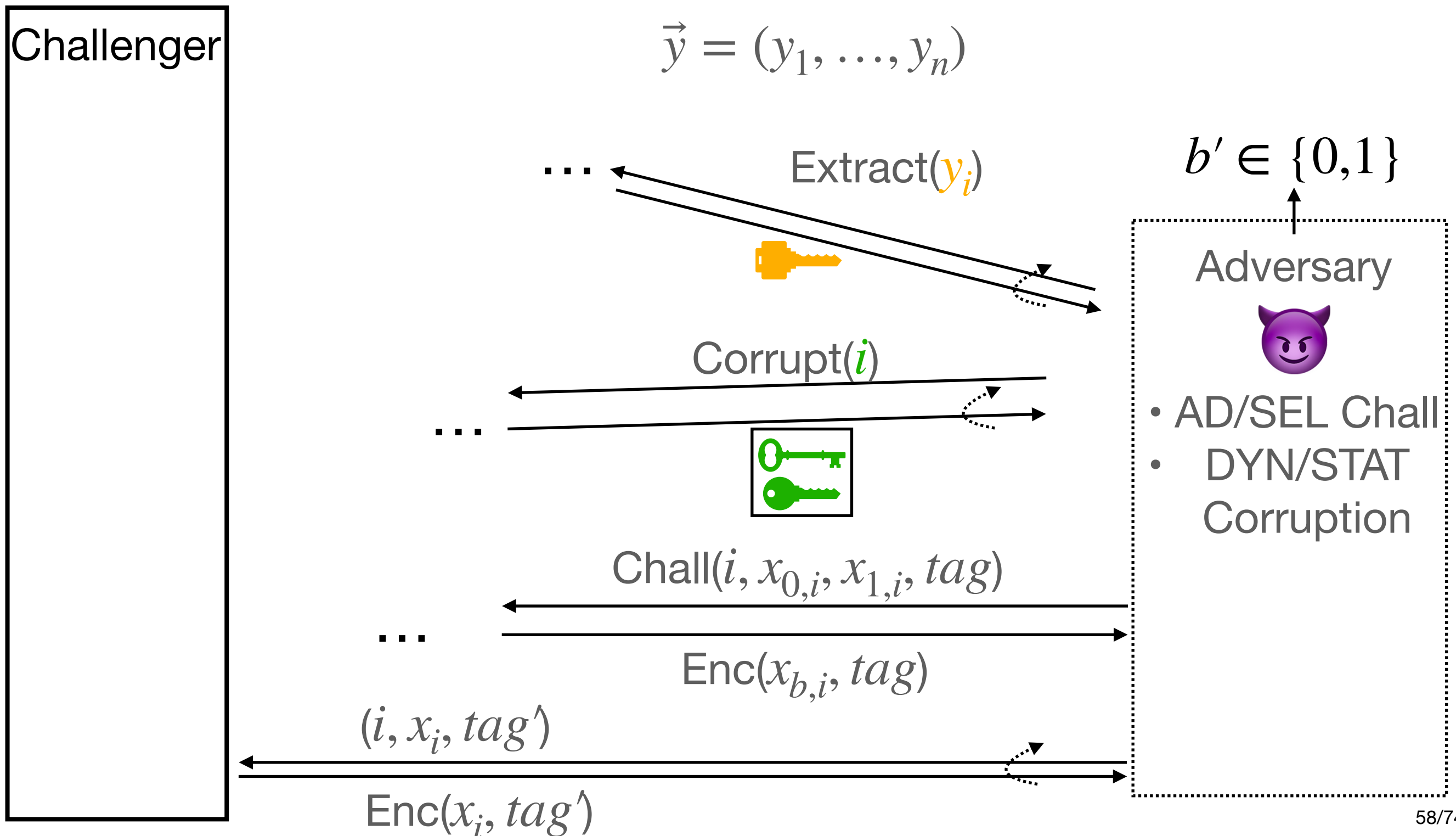
Challenger: $b \leftarrow^{\$} \{0,1\}$



Definition

Security model [CDGPP'18]

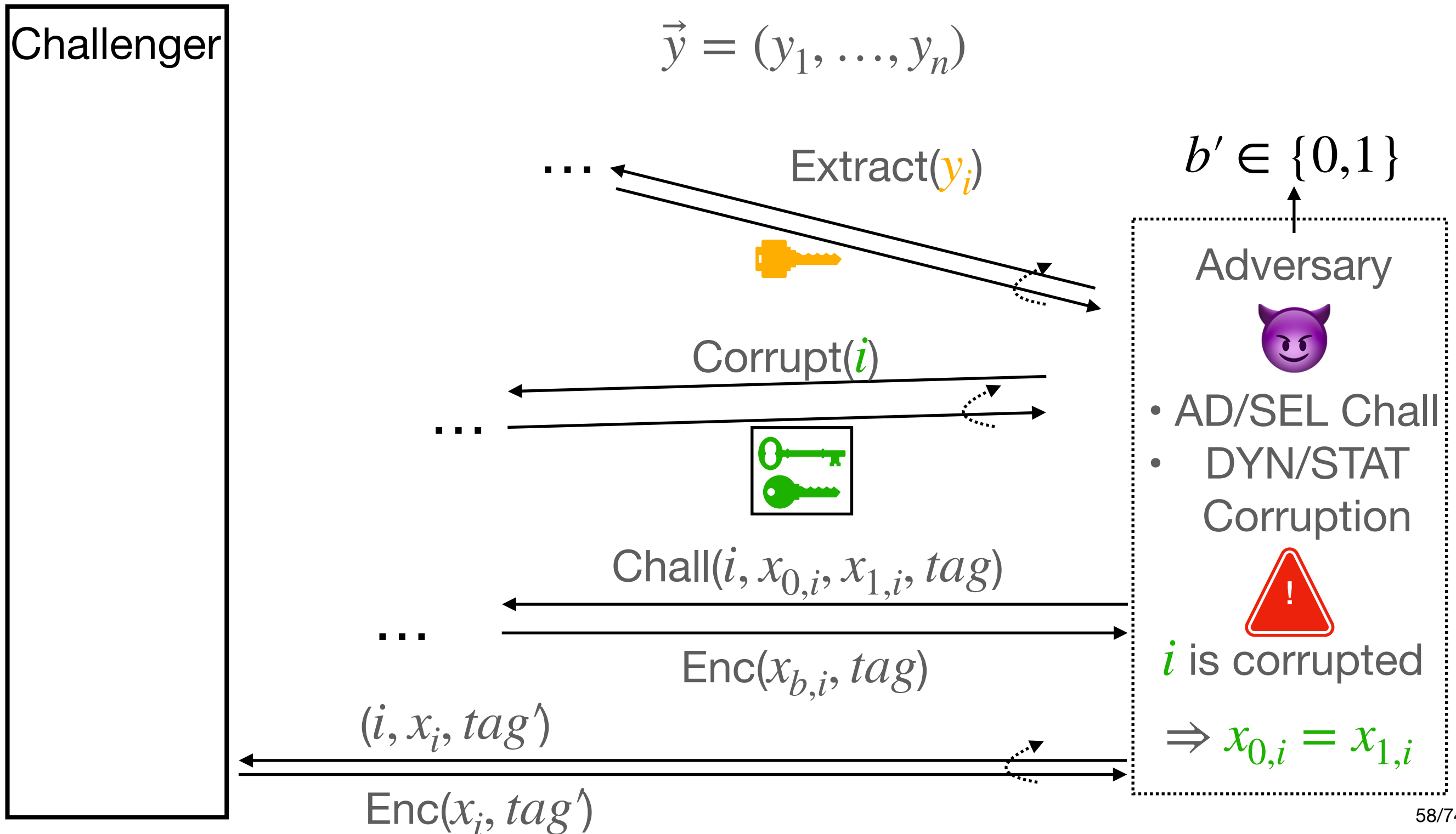
Challenger: $b \leftarrow^{\$} \{0,1\}$



Definition

Security model [CDGPP'18]

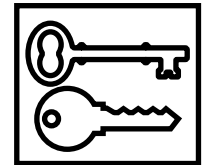
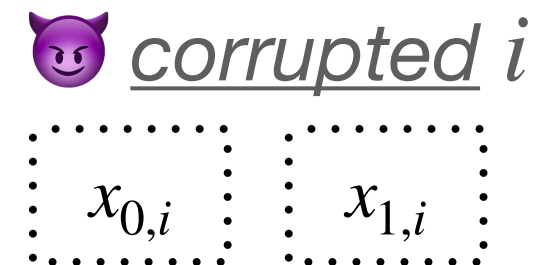
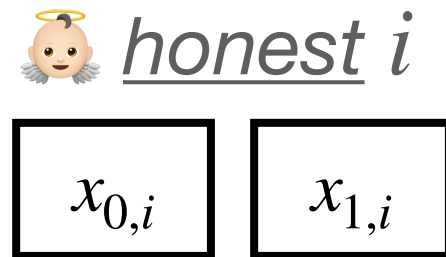
Challenger: $b \leftarrow^{\$} \{0,1\}$



Admissibility

Concrete conditions for IP [CDGPP'18]

For all $(x_{0,i}, x_{1,i})_i$,
for all $\vec{y}$



$$\vec{y} = \left(\begin{array}{c} \boxed{y_i} \end{array} \right)_i$$

Admissibility

Concrete conditions for IP [CDGPP'18]

For all $(x_{0,i}, x_{1,i})_i$,
for all $\vec{y}$

 honest i

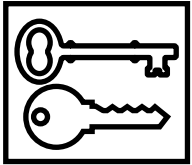
$x_{0,i}$

$x_{1,i}$

 corrupted i

$x_{0,i}$

$x_{1,i}$



$$\vec{y} = \left(\begin{bmatrix} \\ y_i \\ \end{bmatrix} \right)_i$$

(1)

 honest i
 $\sum \boxed{x_{0,i}} \begin{bmatrix} \\ y_i \\ \end{bmatrix}$

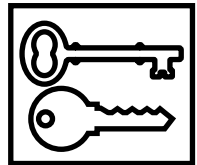
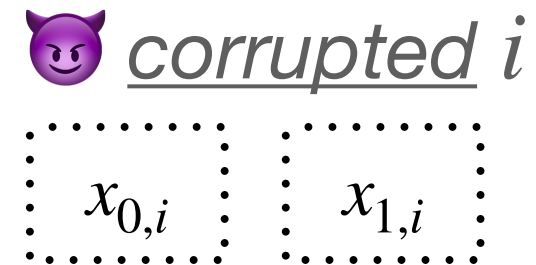
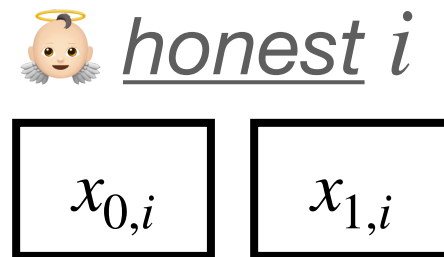
=

 honest i
 $\sum \boxed{x_{1,i}} \begin{bmatrix} \\ y_i \\ \end{bmatrix}$

Admissibility

Concrete conditions for IP [CDGPP'18]

For all $(x_{0,i}, x_{1,i})_i$,
for all $\vec{y}$



$$\vec{y} = \left(\begin{bmatrix} \\ y_i \\ \end{bmatrix} \right)_i$$

$$(1) \quad \sum_{\text{angel } \underline{\text{honest}} \ i} \boxed{x_{0,i}} \begin{bmatrix} \\ y_i \\ \end{bmatrix} = \sum_{\text{angel } \underline{\text{honest}} \ i} \boxed{x_{1,i}} \begin{bmatrix} \\ y_i \\ \end{bmatrix}$$

$$(2) \quad \text{devil } \underline{\text{corrupted}} \ i: \quad \boxed{\phantom{x_{0,i}}} = \boxed{\phantom{x_{1,i}}}$$

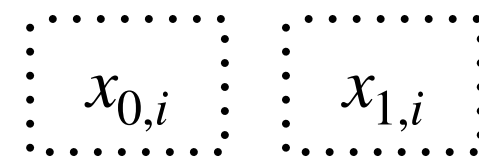
Our **New** Admissibility

Concrete conditions for IP

 honest i



 corrupted i



For all $(x_{0,i}, x_{1,i})_i$,
for all $\vec{y}$

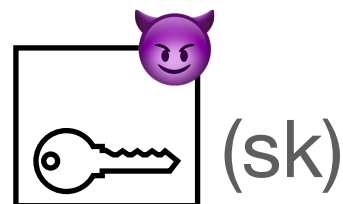
$$\vec{y} = \left(\begin{bmatrix} \\ y_i \\ \end{bmatrix} \right)_i$$

$$(1) \quad \sum \boxed{x_{0,i}} \begin{bmatrix} \\ y_i \\ \end{bmatrix} = \sum \boxed{x_{1,i}} \begin{bmatrix} \\ y_i \\ \end{bmatrix}$$

Separated
Corruption

(2a)

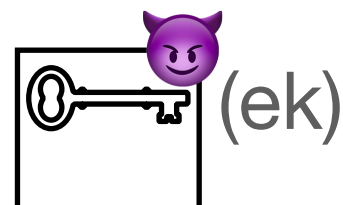
 honest i



 honest i

$$\boxed{x_{0,i}} = \boxed{x_{1,i}}$$

(2b)



$$\boxed{x_{0,i}} \begin{bmatrix} \\ y_i \\ \end{bmatrix} = \boxed{x_{1,i}} \begin{bmatrix} \\ y_i \\ \end{bmatrix}$$

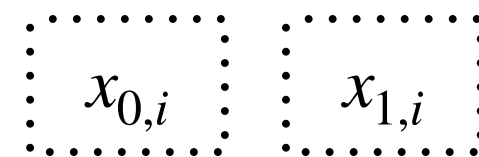
Our **New** Admissibility

Concrete conditions for IP

 honest i



 corrupted i



For all $(x_{0,i}, x_{1,i})_i$,
for all $\vec{y}$

$$\vec{y} = \left(\begin{bmatrix} \\ \\ \end{bmatrix} y_i \right)_i$$

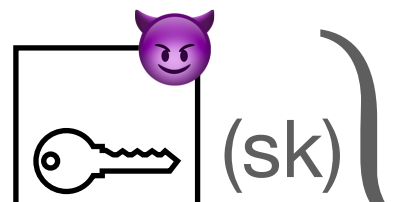
(1)

$$\sum \boxed{x_{0,i}} \begin{bmatrix} \\ \\ \end{bmatrix} y_i = \sum \boxed{x_{1,i}} \begin{bmatrix} \\ \\ \end{bmatrix} y_i$$

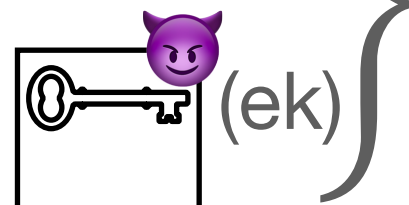
Separated
Corruption

(2a)

 honest i



(2b)

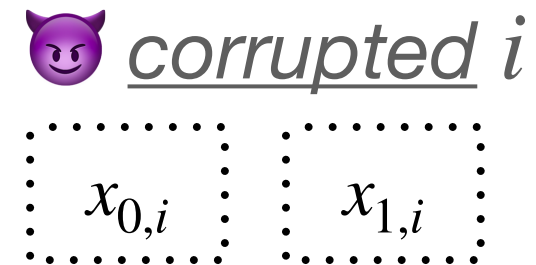
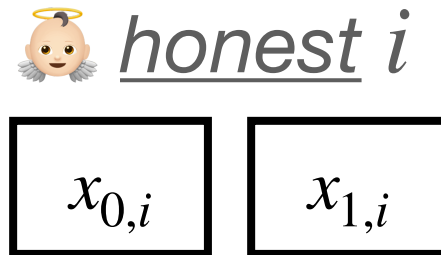


Less
Restrictive

Our **New** Admissibility

Concrete conditions for IP

Section **5.3**
In Chapter 5



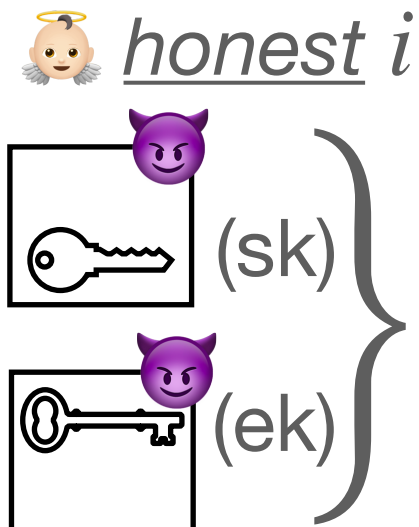
For all $(x_{0,i}, x_{1,i})_i$,
for all $\vec{y}$

$$\vec{y} = \left(\begin{bmatrix} \\ y_i \\ \end{bmatrix} \right)_i$$

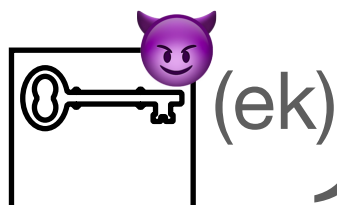
$$(1) \quad \sum \boxed{x_{0,i}} \begin{bmatrix} \\ y_i \\ \end{bmatrix} = \sum \boxed{x_{1,i}} \begin{bmatrix} \\ y_i \\ \end{bmatrix}$$

Separated
Corruption

(2a)



(2b)



Less
Restrictive

Thm. 12

Optimal!

Running Example

DMCFE from [CDGPP'18]

Setup

$$\left(\begin{array}{c} s_i \\ T_i \end{array} \right)_i$$

S.t. $\sum T_i = \mathbf{0}$

Enc



(ek)

x_i, tag

$$\left[\begin{array}{c} \text{H1}(\text{tag}) \\ s_i \end{array} + x_i \right]_1$$

Running Example

DMCFE from [CDGPP'18]

Setup

$$\rightarrow \left(\begin{array}{c} \text{orange box } s_i \\ \text{blue box } T_i \end{array} \right)_i$$

$$\text{S.t. } \sum \text{blue box } T_i = \mathbf{0}$$

Enc



x_i, tag

$$\left[\begin{array}{c} \text{pink box } H1(\text{tag}) \\ \text{orange box } s_i \end{array} + x_i \right]_1$$



Deterministic

Running Example

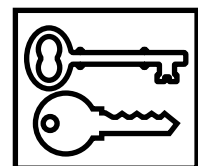
DMCFE from [CDGPP'18]

Setup

$$\rightarrow \left(\begin{array}{c} \boxed{s_i} \\ \boxed{T_i} \end{array} \right)_i$$

$$\text{S.t. } \sum \boxed{T_i} = \mathbf{0}$$

Enc   (ek)
 x_i, tag $\left[\begin{array}{c} \boxed{H1(\text{tag})} \\ \boxed{s_i} \end{array} + x_i \right]_1$



corrupted i :

$$\boxed{x_{0,i}}$$

$\neq$

$$\boxed{x_{1,i}}$$



Deterministic

Running Example

DMCFE from [CDGPP'18]

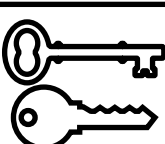
Setup

$$\rightarrow \left(\begin{array}{c} s_i \\ T_i \end{array} \right)_i$$

$$\text{S.t. } \sum T_i = \mathbf{0}$$

Enc  (ek)  x_i, tag

$$\left[\begin{array}{c} \text{H1}(\text{tag}) \\ s_i \end{array} + x_i \right]_1$$

  corrupted i : $x_{0,i} \neq x_{1,i}$

 Deterministic

$$\Rightarrow \left[\begin{array}{c} \text{H1}(\text{tag}) \\ s_i \end{array} + x_{0,i} \right]_1 \neq \left[\begin{array}{c} \text{H1}(\text{tag}) \\ s_i \end{array} + x_{1,i} \right]_1$$

Running Example

DMCFE from [CDGPP'18]

Setup

$$\longrightarrow \left(\begin{array}{c} \text{orange box } s_i \\ \text{blue box } T_i \end{array} \right)_i$$

S.t. $\sum \text{blue box } T_i = \mathbf{0}$

Enc

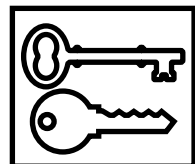


x_i, tag



(ek)

$$\left[\begin{array}{c} \text{pink box } H1(\text{tag}) \\ \text{orange box } s_i \end{array} + x_i \right]_1$$



corrupted i :

$$\boxed{x_{0,i}}$$

$\neq$

$$\boxed{x_{1,i}}$$

$$\Rightarrow \left[\begin{array}{c} \text{pink box } H1(\text{tag}) \\ \text{orange box } s_i \end{array} + \boxed{x_{0,i}} \right]_1 \neq \left[\begin{array}{c} \text{pink box } H1(\text{tag}) \\ \text{orange box } s_i \end{array} + \boxed{x_{1,i}} \right]_1$$



Even LWE-based
With slightly rand.

Trivially distinguishable
From deterministic ciphertexts

Running Example

DMCFE from [CDGPP'18]

Setup

$$\rightarrow \left(\begin{array}{c} \text{orange box } s_i \\ \text{blue box } T_i \end{array} \right)_i$$

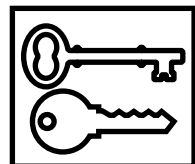
$$\text{S.t. } \sum \text{blue box } T_i = \mathbf{0}$$

Enc



x_i, tag

$$\left[\begin{array}{c} \text{pink box } H1(\text{tag}) \\ \text{orange box } s_i \end{array} + x_i \right]_1$$



corrupted i :

$$\boxed{x_{0,i}} \neq \boxed{x_{1,i}}$$

$$\Rightarrow \left[\begin{array}{c} \text{pink box } H1(\text{tag}) \\ \text{orange box } s_i \end{array} + \boxed{x_{0,i}} \right]_1 \neq \left[\begin{array}{c} \text{pink box } H1(\text{tag}) \\ \text{orange box } s_i \end{array} + \boxed{x_{1,i}} \right]_1$$



Probabilistic
Enc?

New Technique: “Revoking” a Random Mask

Towards DMCFE with New Admissibility

Setup

$$\rightarrow \left(\begin{array}{c} \boxed{s_i} , \boxed{T_i} , \boxed{t_i, v_i} \end{array} \right)_i$$

r_i is randomness

$$\text{S.t. } \sum \boxed{T_i} = \mathbf{0}$$

Enc

$$\left[\begin{array}{c} \boxed{H1(\text{tag})} \quad \boxed{s_i} \quad +x_i \\ \boxed{-r_i v_i} \end{array} \right]_1 , \left[\boxed{r_i t_i} \right]_1$$

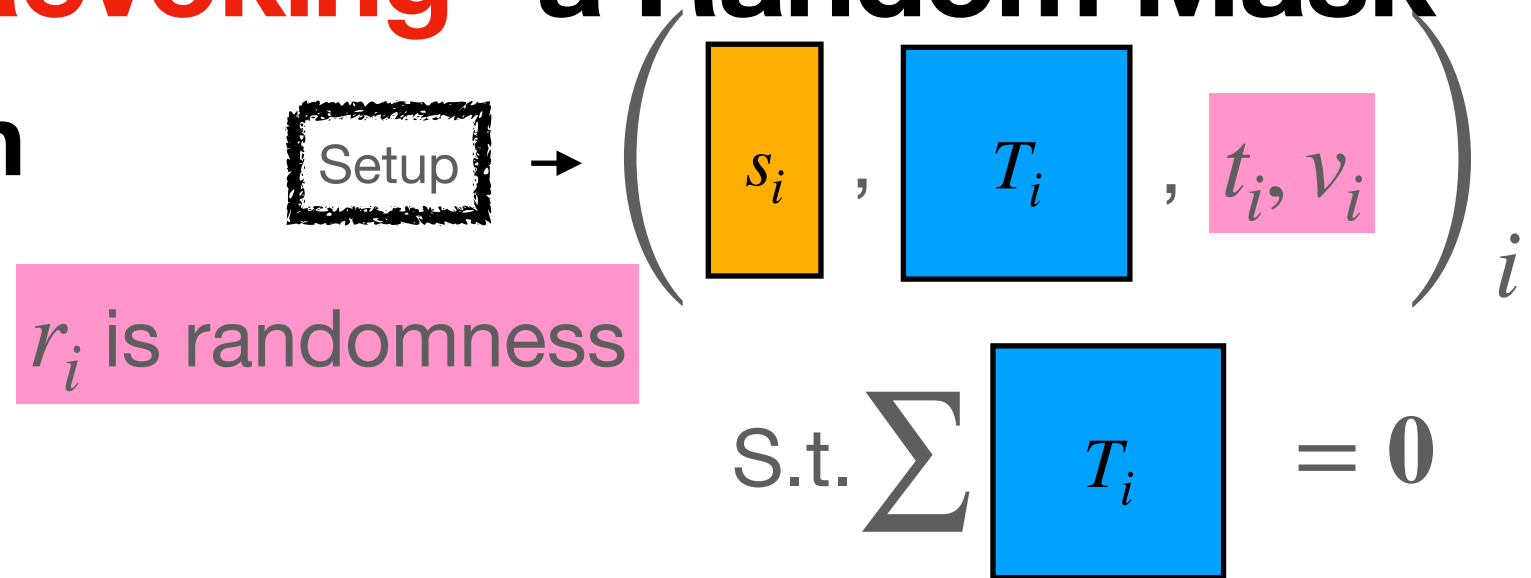
🎉 Probabilistic
Enc

DKeyGen

$$\left[\begin{array}{c} \boxed{T_i} \quad \boxed{H2(\text{tag-f})} \quad +y_i \quad \boxed{s_i} \end{array} \right]_2 , \left[\boxed{y_i v_i / t_i} \right]_2$$

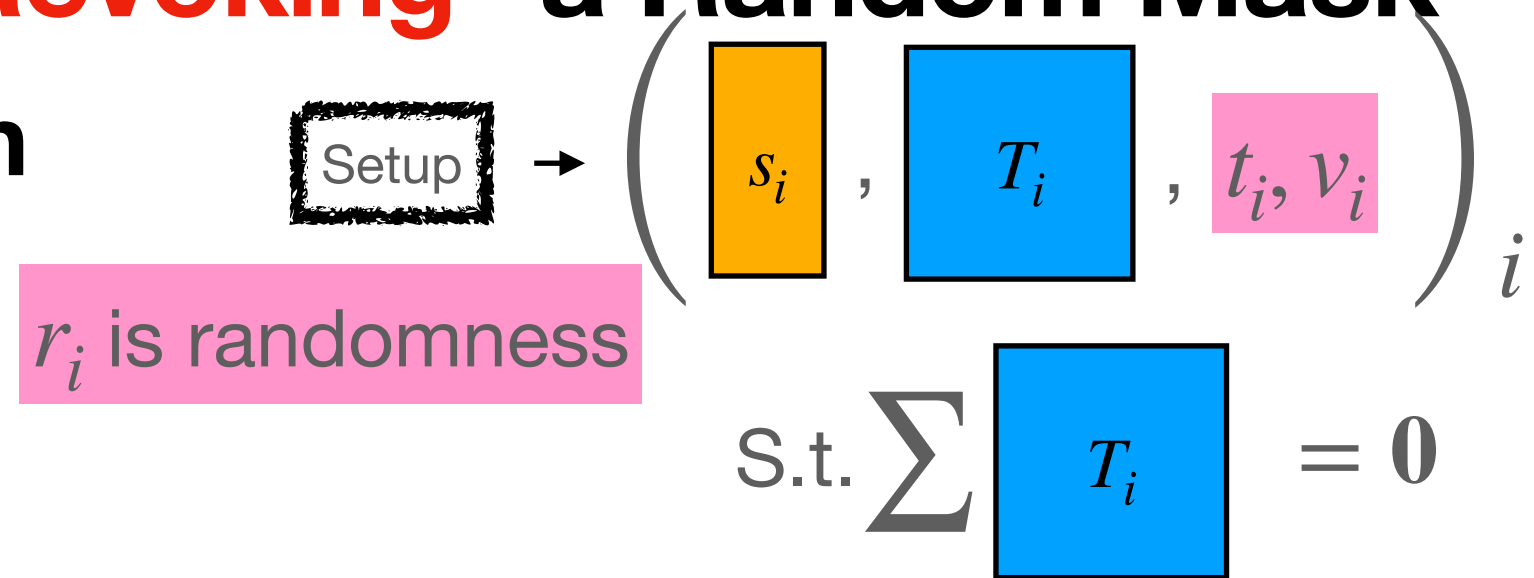
New Technique: “**Revoking**” a Random Mask

Towards DMCFE with
New Admissibility



New Technique: “Revoking” a Random Mask

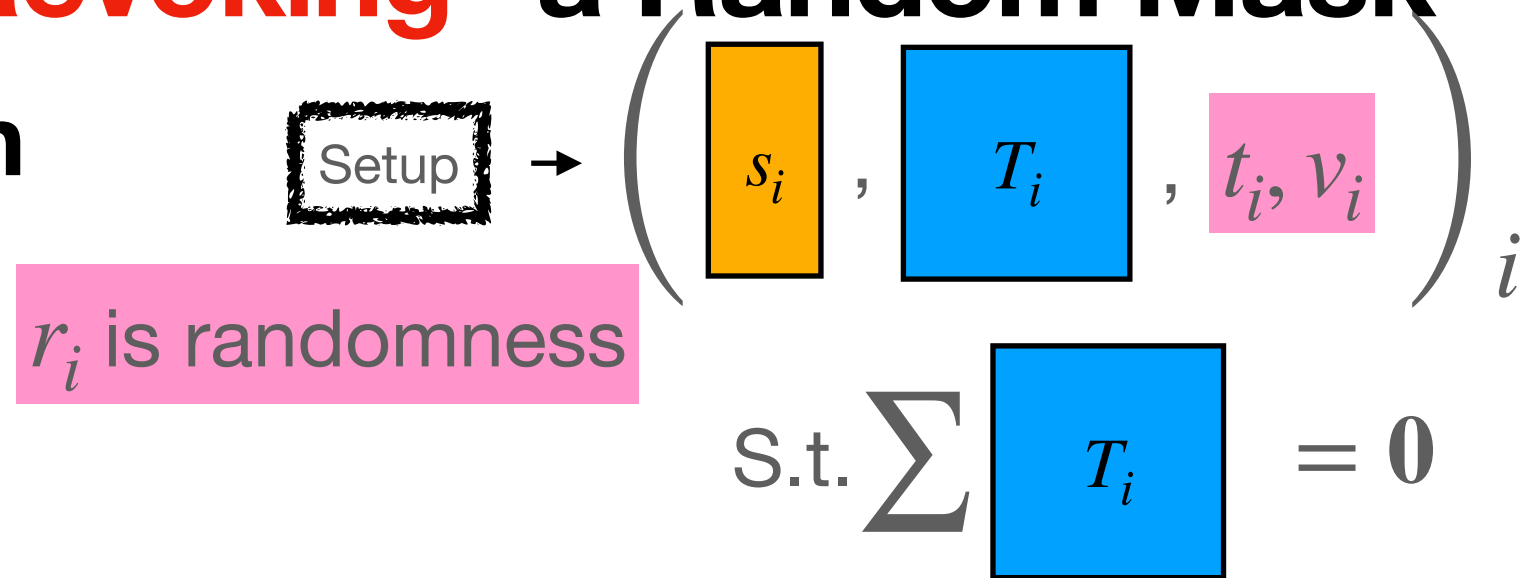
Towards DMCFE with New Admissibility



- ElGamal mask by r_i
- Bind unmasking factor to y_i + usage of pairings
- View as a “revoking” problem

New Technique: “Revoking” a Random Mask

Towards DMCFE with New Admissibility

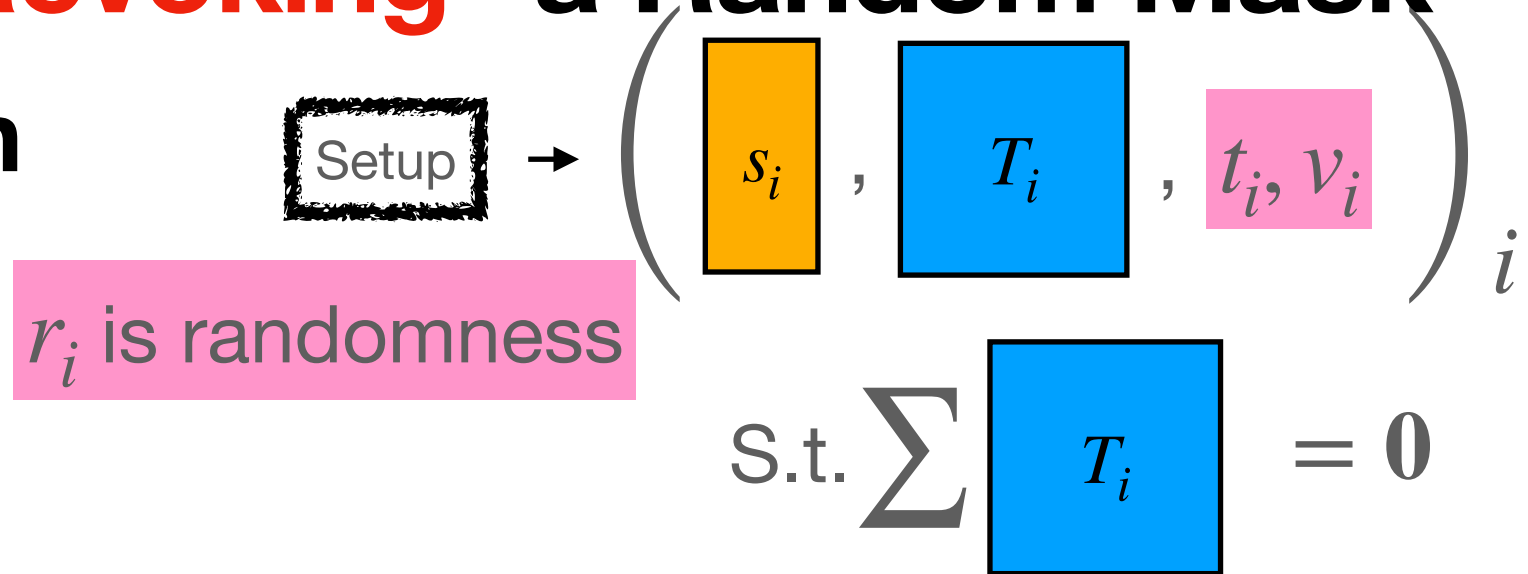


- ElGamal mask by r_i
- Bind unmasking factor to y_i + usage of pairings
- View as a “revoking” problem

(ek) $\left[\begin{array}{c} \cdots x_{0,i} \cdots \\ \hline y_i \end{array} \right] = \left[\begin{array}{c} \cdots x_{1,i} \cdots \\ \hline y_i \end{array} \right]$

New Technique: “Revoking” a Random Mask

Towards DMCFE with New Admissibility



- ElGamal mask by r_i
- Bind unmasking factor to y_i + usage of pairings
- View as a “revoking” problem

$$(ek) \boxed{x_{0,i}} = \boxed{x_{1,i}} \text{ OR } \boxed{y_i} = 0 \in \mathbb{Z}/q\mathbb{Z}$$

New Technique: “Revoking” a Random Mask

Towards DMCFE with New Admissibility

$$\text{Setup} \rightarrow \left(\begin{array}{c} \boxed{s_i} \\ \boxed{T_i} \\ \boxed{t_i, v_i} \end{array} \right)_i$$

r_i is randomness

$$\text{S.t. } \sum \boxed{T_i} = \mathbf{0}$$

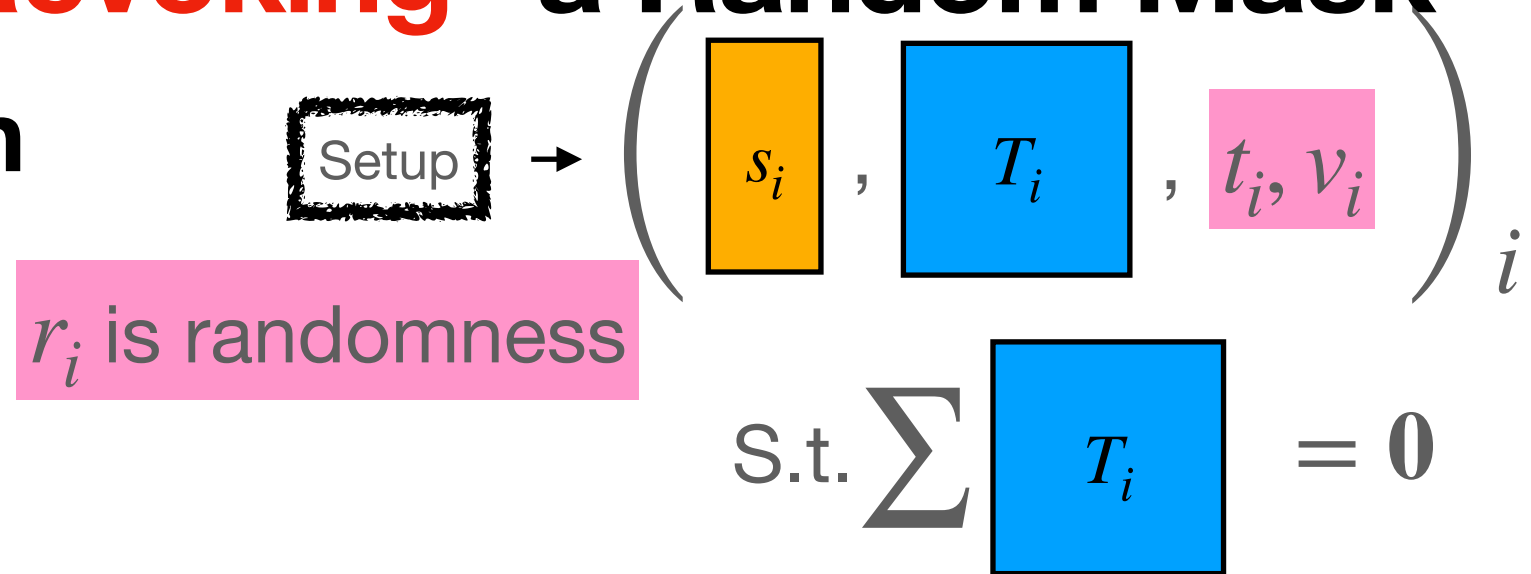
- ElGamal mask by r_i
- Bind unmasking factor to y_i + usage of pairings
- View as a “revoking” problem

$$\boxed{\text{key}}^{\text{evil}}(\text{ek}) \boxed{x_{0,i}} = \boxed{x_{1,i}} \text{ OR } \boxed{y_i} = \mathbf{0} \in \mathbb{Z}/q\mathbb{Z}$$

- Security {
- Adaptive $x_{0,i}, x_{1,i}$
 - Partial static corruption

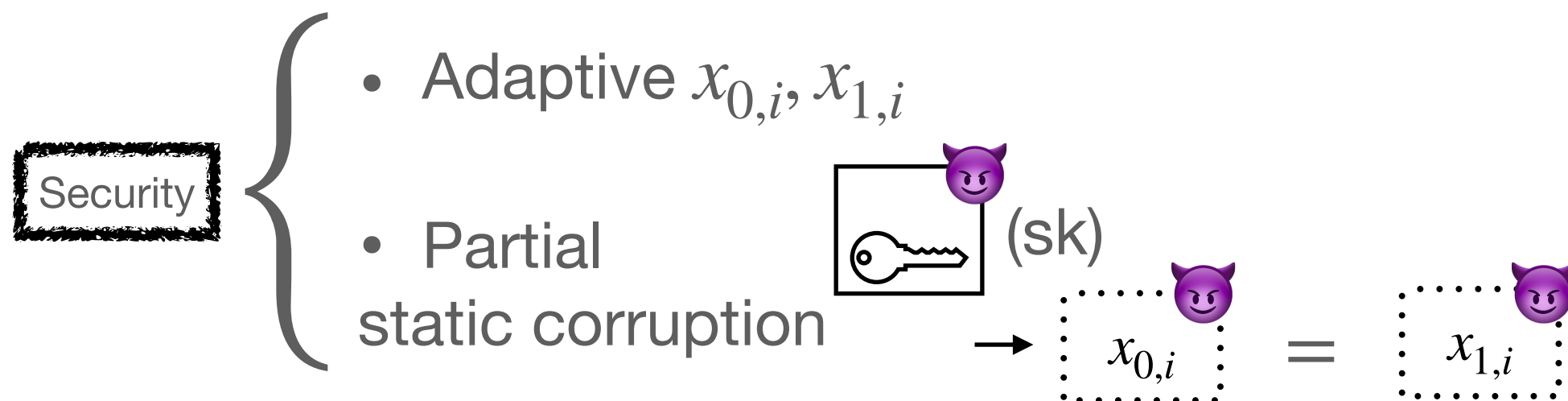
New Technique: “Revoking” a Random Mask

Towards DMCFE with New Admissibility



- ElGamal mask by r_i
- Bind unmasking factor to y_i + usage of pairings
- View as a “revoking” problem

$$\boxed{\text{key}}^{\text{devil}} (ek) \quad x_{0,i} = x_{1,i} \text{ OR } y_i = 0 \in \mathbb{Z}/q\mathbb{Z}$$



New Technique: “Revoking” a Random Mask

Towards DMCFE with New Admissibility

Setup

$$\rightarrow \left(\begin{array}{c} \text{orange box } s_i, \text{ blue box } T_i, \text{ pink box } t_i, v_i \end{array} \right)_i$$

r_i is randomness

$$\text{S.t. } \sum \text{blue box } T_i = \mathbf{0}$$

- ElGamal mask by r_i
- Bind unmasking factor to y_i + usage of pairings
- View as a “revoking” problem

$$\boxed{\text{key icon}}^{\text{devil}}(\text{ek}) \cdot \boxed{x_{0,i}} = \boxed{x_{1,i}} \text{ OR } \boxed{y_i} = \mathbf{0} \in \mathbb{Z}/q\mathbb{Z}$$

Security

- Adaptive $x_{0,i}, x_{1,i}$

- Partial static corruption

$\boxed{\text{key icon}}^{\text{devil}}(\text{sk})$

New Technique: “Revoking” a Random Mask

Towards DMCFE with New Admissibility

Setup

$$\rightarrow \left(\begin{array}{c} \text{orange box } s_i \\ \text{blue box } T_i \\ \text{pink box } t_i, v_i \end{array} \right)_i$$

r_i is randomness

$$\text{S.t. } \sum \text{blue box } T_i = \mathbf{0}$$

- ElGamal mask by r_i
- Bind unmasking factor to y_i + usage of pairings
- View as a “revoking” problem

$$\boxed{\text{key icon}}^{\text{devil}}(\text{ek}) \quad \boxed{x_{0,i}} = \boxed{x_{1,i}} \text{ OR } \boxed{y_i} = 0 \in \mathbb{Z}/q\mathbb{Z}$$

Security

- Adaptive $x_{0,i}, x_{1,i}$

- Partial static corruption

$\boxed{\text{key icon}}^{\text{devil}}(\text{sk})$

Section 5.4
In Chapter 5

Research Question (2) - Our Result

Optimal Security Notion

- Definitional Framework to Refine Decentralized MCFE Security
- Constructions from DPVS

Our Results:

- Improved security models for *secret-key* (D)MCFE
- Optimal proof for specific functionalities
- Feasibility result for IP-DMCFE, *scalar per client*

Research Question (2) - Our Result

Optimal Security Notion

- Definitional Framework to Refine Decentralized MCFE Security
- Constructions from DPVS

Our Results:

- Improved security models for *secret-key* (D)MCFE
- Optimal proof for specific functionalities
- Feasibility result for IP-DMCFE, *scalar per client*

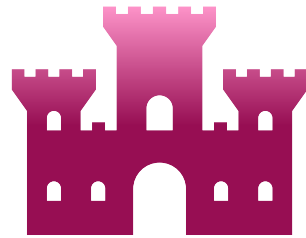
Improved to *vector per client* in later chapter
& new implication to *public-key* setting

Outline

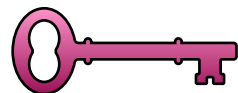
- Introduction & Main Techniques Overview
- Controlling Functional Keys
- Optimal Security Notion
- **From Secret-Key to Public-Key** Encryption Schemes
 - Definitional Framework with Public Inputs
 - Constructions from DPVS
- Conclusion and Future Works

This chapter: Access Control & Strong Security

Connection to Public-Key Primitives



Func: G ,
Policy: $\text{Spade}(\cdot)$



(3)

Control Keys
& strong security



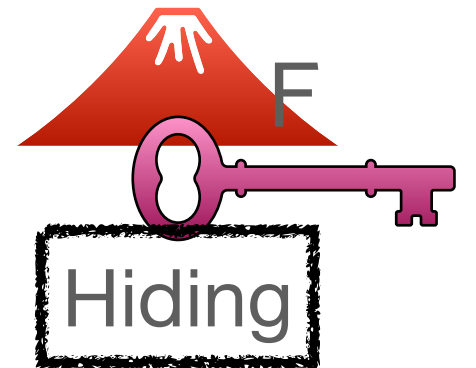
[CDGPP'18]

$\text{Corrupt}(i)$

$\Rightarrow x_{0,i} = x_{1,i}$



Justified?



Other misc:
Blind Sign,
CALBO-TDFs



Ongoing:
Traceable FE,
CCA-security

Improvements on Our Previous Results

FE + Access Control in the Multi-User Regime

Improvements on Our Previous Results

FE + Access Control in the Multi-User Regime

This work
[NPP24]
+ [NPS24]

MCFE for sub-vector per client

- Strong admissibility as per [NPP23]
- With access control as per [NPS24]
- Repetitions of sub-vectors in ciphertexts

⇒ Keeping linear total communication,
Improved over [NPP22] and others

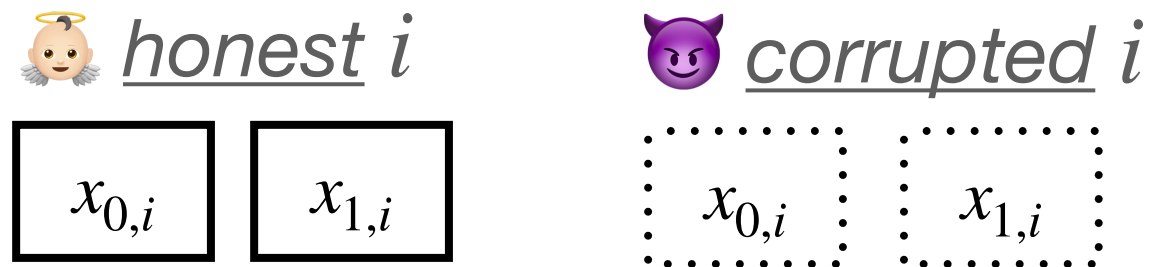
[NPP22]

- From pairings
- $\mathcal{P}$: LSSS
- F: Inner Products, *scalar per client*
- (Multi-Client) Adaptive security in ROM
linear total communication
- *Weaker security* as per [CDGPP18]

Recall: Our New Admissibility

Concrete conditions for IP, DMCFE, **one scalar/client**

For all $(x_{0,i}, x_{1,i})_i$,
for all $\vec{y}$



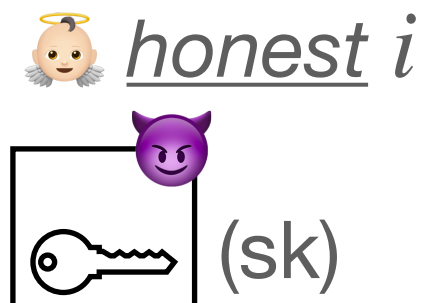
$$\vec{y} = \left(\begin{bmatrix} \\ y_i \\ \end{bmatrix} \right)_i$$

(1)

$$\sum \boxed{x_{0,i}} \begin{bmatrix} \\ y_i \\ \end{bmatrix} = \sum \boxed{x_{1,i}} \begin{bmatrix} \\ y_i \\ \end{bmatrix}$$

Separated
Corruption

(2a)



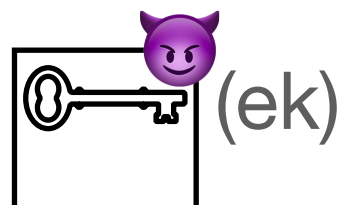
 honest i

$x_{0,i}$

 $=$

$x_{1,i}$

(2b)



$x_{0,i}$

 $\begin{bmatrix} \\ y_i \\ \end{bmatrix} = \begin{bmatrix} \\ y_i \\ \end{bmatrix}$

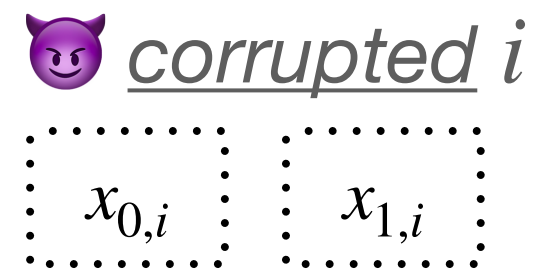
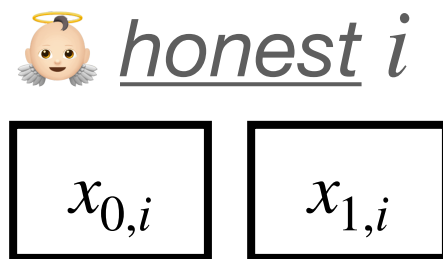
$x_{1,i}$

 $\begin{bmatrix} \\ y_i \\ \end{bmatrix}$

Recall: Our New Admissibility

Concrete conditions for IP, MCFE, **one subvector/client**

For all $(x_{0,i}, x_{1,i})_i$,
for all $\vec{y}$



$$\vec{y} = \left(\begin{bmatrix} y_i \end{bmatrix} \right)_i$$

$$(1) \quad \sum_{\text{angel } \text{honest } i} \left\langle \boxed{x_{0,i}}, \begin{bmatrix} y_i \end{bmatrix} \right\rangle = \sum_{\text{angel } \text{honest } i} \left\langle \boxed{x_{1,i}}, \begin{bmatrix} y_i \end{bmatrix} \right\rangle$$

MCFE

$$(2) \quad \boxed{\text{key}}^{\text{devil}}_{(ek)} \left\langle \boxed{x_{0,i}}, \begin{bmatrix} y_i \end{bmatrix} \right\rangle = \left\langle \boxed{x_{1,i}}, \begin{bmatrix} y_i \end{bmatrix} \right\rangle$$

On Security: Admissibility All Over Again

Implications of Pub-Inputs & Strong Admissibility



Our
Work!

MCFE: Pub-Inputs,
Allowing repetitions $\vec{x}_i$

On Security: Admissibility All Over Again

Implications of Pub-Inputs & Strong Admissibility

🥳 Our
Work!

MCFE: Pub-Inputs,
Allowing repetitions $\vec{x}_i$

#clients
 $n = 1$

On Security: Admissibility All Over Again

Implications of Pub-Inputs & Strong Admissibility

Our
Work!

MCFE: Pub-Inputs,
Allowing repetitions $\vec{x}_i$

#clients
 $n = 1$

Strong Admissibility:

$$\boxed{\text{key}}^{\text{devil}}_{\text{ek}} \left\langle \boxed{\vec{x}_0}, \boxed{\vec{y}} \right\rangle = \left\langle \boxed{\vec{x}_1}, \boxed{\vec{y}} \right\rangle$$

On Security: Admissibility All Over Again

Implications of Pub-Inputs & Strong Admissibility

Our
Work!

MCFE: Pub-Inputs,
Allowing repetitions $\vec{x}_i$

#clients
 $n = 1$

Strong Admissibility:

$$\boxed{\text{key}}^{\text{devil}}_{\text{ek}} \left\langle \boxed{\vec{x}_0}, \boxed{\vec{y}} \right\rangle = \left\langle \boxed{\vec{x}_1}, \boxed{\vec{y}} \right\rangle$$

+ Static Corruption $\boxed{\text{key}}^{\text{devil}}_{\text{ek}}$

On Security: Admissibility All Over Again

Implications of Pub-Inputs & Strong Admissibility

Our
Work!

MCFE: Pub-Inputs,
Allowing repetitions $\vec{x}_i$

#clients
 $n = 1$

Strong Admissibility:

Admissibility in
Public-key FE !!

$$\boxed{\text{key}}^{\text{devil}}(\text{ek}) \left\langle \boxed{\vec{x}_0}, \boxed{\vec{y}} \right\rangle = \left\langle \boxed{\vec{x}_1}, \boxed{\vec{y}} \right\rangle$$

+ Static Corruption $\boxed{\text{key}}^{\text{devil}}(\text{ek})$

Becomes pk !!

On Security: Admissibility All Over Again

Implications of Pub-Inputs & Strong Admissibility

🥳 Our
Work!

MCFE: Pub-Inputs,
Allowing repetitions $\vec{x}_i$



Public-key FE
With Pub-Inputs

On Security: Admissibility All Over Again

Implications of Pub-Inputs & Strong Admissibility

🥳 Our
Work!

MCFE: Pub-Inputs,
Allowing repetitions $\vec{x}_i$



Public-key FE
With Pub-Inputs

Access control in
Public inputs

Covering syntactically
Non-pol/att-hiding
ABE !!

On Security: Admissibility All Over Again

Implications of Pub-Inputs & Strong Admissibility

🥳 Our
Work!

MCFE: Pub-Inputs,
Allowing repetitions $\vec{x}_i$



Public-key FE
With Pub-Inputs



Public-key ABE
With or w/out
Hiding properties

On Security: Admissibility All Over Again

Implications of Pub-Inputs & Strong Admissibility

🥳 Our
Work!

MCFE: Pub-Inputs,
Allowing repetitions $\vec{x}_i$



Public-key FE
With Pub-Inputs



Fix one tag for all
Enc

Public-key ABE
With or w/out
Hiding properties

Repetitions on $\vec{x}_i$
Covers basic MIFE

On Security: Admissibility All Over Again

Implications of Pub-Inputs & Strong Admissibility

🥳 Our
Work!

MCFE: Pub-Inputs,
Allowing repetitions $\vec{x}_i$



Public-key FE
With Pub-Inputs



MIFE: “vanilla”
Same functions



Public-key ABE
With or w/out
Hiding properties

On Security: Admissibility All Over Again

Implications of Pub-Inputs & Strong Admissibility

🥳 Our Work!

MCFE: Pub-Inputs,
Allowing repetitions $\vec{x}_i$



Public-key FE
With Pub-Inputs



MIFE: “vanilla”
Same functions



Public-key ABE
With or w/out
Hiding properties

Towards AB-MIFE:
Repetitions on pub-inputs
Follows [NPS24]

New **MC**-AB-IP**FE** for LSSS

**Towards Strong Security -Security - New
technique: DPVS with Complexity Leveraging**

Selective

simulator : guess $(\vec{x}_i^{(0)}, \vec{x}_i^{(1)})$ from 😈

New **MC**-AB-IP**FE** for LSSS

Towards Strong Security -Security - New
technique: DPVS with Complexity Leveraging

Selective

simulator : guess $(\vec{x}_i^{(0)}, \vec{x}_i^{(1)})$ from 

$$\text{View}_0^{sel} \equiv \text{View}_1^{sel} \equiv \dots \equiv \underline{\text{View}_k^{sel}}$$

New **MC-AB-IPFE** for LSSS

Towards Strong Security -Security - New
technique: DPVS with Complexity Leveraging

$$\underline{\text{View}_0^{sel}} \ \& \ \underline{\text{View}_0} : \vec{t}_{i,0} = (\omega \vec{s}_i + \omega' \vec{u}_i + \boxed{\vec{x}_i^{(0)}} | \nu_i | \vec{0})_{B_i}$$

Selective

simulator : guess $(\vec{x}_i^{(0)}, \vec{x}_i^{(1)})$ from 

$$\text{View}_0^{sel} \equiv \text{View}_1^{sel} \equiv \dots \equiv \underline{\text{View}_k^{sel}}$$

Start

$\Rightarrow$

Guessing

Challenges $\vec{x}_i^{(0)}, \vec{x}_i^{(1)}$

New **MC**-AB-IP**FE** for LSSS

Towards Strong Security -Security - New
technique: DPVS with Complexity Leveraging

Selective

simulator : guess $(\vec{x}_i^{(0)}, \vec{x}_i^{(1)})$ from 🐾

$$\text{View}_0^{sel} \equiv \text{View}_1^{sel} \equiv \dots \equiv \underline{\text{View}_k^{sel}}$$

$$\text{In } \underline{\text{View}_k^{sel}}: \vec{t}_{i,0} = (\omega \vec{s}_i + \omega' \vec{u}_i + \vec{x}_i^{(1)} | \nu_i | \vec{0})_{B_i}$$

New **MC**-AB-IP**FE** for LSSS

Towards Strong Security -Security - New
technique: DPVS with Complexity Leveraging

$$\underline{\text{View}_0^{sel}} \ \& \ \underline{\text{View}_0} : \vec{t}_{i,0} = (\omega \vec{s}_i + \omega' \vec{u}_i + \boxed{\vec{x}_i^{(0)}} | \nu_i | \vec{0})_{B_i}$$

Selective

simulator : guess $(\vec{x}_i^{(0)}, \vec{x}_i^{(1)})$ from 

$$\text{View}_0^{sel} \equiv \text{View}_1^{sel} \equiv \dots \equiv \underline{\text{View}_k^{sel}}$$

$$\text{In } \underline{\text{View}_k^{sel}} : \vec{t}_{i,0} = (\omega \vec{s}_i + \omega' \vec{u}_i + \boxed{\vec{x}_i^{(1)}} | \nu_i | \vec{0})_{B_i}$$

New **MC-AB-IPFE** for LSSS

Towards Strong Security -Security - New
technique: DPVS with Complexity Leveraging

$$\underline{\text{View}_0^{sel}} \ \& \ \underline{\text{View}_0} : \vec{t}_{i,0} = (\omega \vec{s}_i + \omega' \vec{u}_i + \boxed{\vec{x}_i^{(0)}} | \nu_i | \vec{0})_{B_i}$$

Selective

simulator : guess $(\vec{x}_i^{(0)}, \vec{x}_i^{(1)})$ from 

$$\text{View}_0^{sel} \equiv \text{View}_1^{sel} \equiv \dots \equiv \underline{\text{View}_k^{sel}}$$

$$\text{In } \underline{\text{View}_k^{sel}} : \vec{t}_{i,0} = (\omega \vec{s}_i + \omega' \vec{u}_i + \boxed{\vec{x}_i^{(1)}} | \nu_i | \vec{0})_{B_i}$$

⇒ Loss factor over choices
of $\vec{x}_i^{(0)}, \vec{x}_i^{(1)}$ in prob of correct

New **MC-AB-IPFE** for LSSS

Towards Strong Security -Security - New
technique: DPVS with Complexity Leveraging

$$\underline{\text{View}_0^{sel}} \ \& \ \underline{\text{View}_0} : \vec{t}_{i,0} = (\omega \vec{s}_i + \omega' \vec{u}_i + \boxed{\vec{x}_i^{(0)}} | \nu_i | \vec{0})_{B_i}$$

Selective

simulator : guess $(\vec{x}_i^{(0)}, \vec{x}_i^{(1)})$ from 

$$\text{View}_0^{sel} \equiv \text{View}_1^{sel} \equiv \dots \equiv \underline{\text{View}_k^{sel}}$$

Perfectly Perfectly Perfectly

$$\text{In } \underline{\text{View}_k^{sel}} : \vec{t}_{i,0} = (\omega \vec{s}_i + \omega' \vec{u}_i + \boxed{\vec{x}_i^{(1)}} | \nu_i | \vec{0})_{B_i}$$

Formal basis
Changes
DPVS

**Guesses
are Free**

⇒ Loss factor over choices
of $\vec{x}_i^{(0)}, \vec{x}_i^{(1)}$ in prob of correct

New **MC-AB-IPFE** for LSSS

Towards Strong Security -Security - New
technique: DPVS with Complexity Leveraging

$$\underline{\text{View}_0^{sel}} \ \& \ \underline{\text{View}_0} : \vec{t}_{i,0} = (\omega \vec{s}_i + \omega' \vec{u}_i + \boxed{\vec{x}_i^{(0)}} | \nu_i | \vec{0})_{B_i}$$

Selective

simulator : guess $(\vec{x}_i^{(0)}, \vec{x}_i^{(1)})$ from 🤖

$$\text{View}_0^{sel} \equiv \text{View}_1^{sel} \equiv \dots \equiv \underline{\text{View}_k^{sel}}$$

Perfectly Perfectly Perfectly

$$\text{In } \underline{\text{View}_k^{sel}} : \vec{t}_{i,0} = (\omega \vec{s}_i + \omega' \vec{u}_i + \boxed{\vec{x}_i^{(1)}} | \nu_i | \vec{0})_{B_i}$$

Formal basis
Changes
DPVS

**Guesses
are Free**

$$\Rightarrow \boxed{\text{Loss factor over choices of } \vec{x}_i^{(0)}, \vec{x}_i^{(1)} \text{ in prob of correct}} \times \text{diff}(\text{View}_0; \text{View}_k) = 0$$

New **MC-AB-IPFE** for LSSS

Towards Strong Security -Security - New
technique: DPVS with Complexity Leveraging

$$\underline{\text{View}_0^{sel}} \ \& \ \underline{\text{View}_0} : \vec{t}_{i,0} = (\omega \vec{s}_i + \omega' \vec{u}_i + \vec{x}_i^{(0)} | \nu_i | \vec{0})_{B_i}$$

Adaptive

Simulator: no guesses

$$\text{View}_0^{sel} \equiv \text{View}_1^{sel} \equiv \dots \equiv \underline{\text{View}_k^{sel}}$$

Perfectly Perfectly Perfectly

$$\text{In } \underline{\text{View}_k^{sel}} : \vec{t}_{i,0} = (\omega \vec{s}_i + \omega' \vec{u}_i + \vec{x}_i^{(1)} | \nu_i | \vec{0})_{B_i}$$

Formal basis
Changes
DPVS

**Guesses
are Free**

$$\Rightarrow \left[\begin{array}{l} \text{Loss factor over choices} \\ \text{of } \vec{x}_i^{(0)}, \vec{x}_i^{(1)} \text{ in prob of correct} \end{array} \right] \times \text{diff}(\text{View}_0; \text{View}_k) = 0$$

New **MC-AB-IPFE** for LSSS

Towards Strong Security -Security - New
technique: DPVS with Complexity Leveraging

$$\underline{\text{View}_0^{sel}} \ \& \ \underline{\text{View}_0} : \vec{t}_{i,0} = (\omega \vec{s}_i + \omega' \vec{u}_i + \vec{x}_i^{(0)} | \nu_i | \vec{0})_{B_i}$$

Adaptive

Simulator: no guesses

$$\text{View}_0 \equiv \text{View}_1 \equiv \dots \equiv \underline{\text{View}_k}$$

Perfectly Perfectly Perfectly

$$\text{In } \underline{\text{View}_k^{sel}} : \vec{t}_{i,0} = (\omega \vec{s}_i + \omega' \vec{u}_i + \vec{x}_i^{(1)} | \nu_i | \vec{0})_{B_i}$$

Formal basis
Changes
DPVS

**Guesses
are Free**

$$\Rightarrow \boxed{\text{Loss factor over choices of } \vec{x}_i^{(0)}, \vec{x}_i^{(1)} \text{ in prob of correct}} \times \text{diff}(\text{View}_0; \text{View}_k) = 0$$

New **MC-AB-IPFE** for LSSS

Towards Strong Security -Security - New
technique: DPVS with Complexity Leveraging

$$\underline{\text{View}_0^{sel}} \ \& \ \underline{\text{View}_0} : \vec{t}_{i,0} = (\omega \vec{s}_i + \omega' \vec{u}_i + \vec{x}_i^{(0)} | \nu_i | \vec{0})_{B_i}$$

Adaptive

Simulator: no guesses

$$\text{View}_0 \equiv \text{View}_1 \equiv \dots \equiv \underline{\text{View}_k}$$

Perfectly Perfectly Perfectly

$$\text{In } \underline{\text{View}_k} : \vec{t}_{i,0} = (\omega \vec{s}_i + \omega' \vec{u}_i + \vec{x}_i^{(1)} | \nu_i | \vec{0})_{B_i}$$

Formal basis
Changes
DPVS

**Guesses
are Free**

$$\Rightarrow \boxed{\text{Loss factor over choices of } \vec{x}_i^{(0)}, \vec{x}_i^{(1)} \text{ in prob of correct}} \times \text{diff}(\text{View}_0; \text{View}_k) = 0$$

Research Question (3) - Our Result

- From Secret-Key to Public-Key Encryption Schemes
 - Definitional Framework with Public Inputs
 - Constructions from DPVS

Our Results:

- Definitional framework of MCFE with public inputs
- Implications from *secret-key* MCFE to *public-key* FE/KP-ABE
- Constructions with **sub-vectors** *per* client and **strong security**

Research Question (3) - Our Result

- From Secret-Key to Public-Key Encryption Schemes
 - Definitional Framework with Public Inputs
 - Constructions from DPVS

Our Results:

- Definitional framework of MCFE with public inputs
- Implications from *secret-key* MCFE to *public-key* FE/KP-ABE
- Constructions with **sub-vectors** *per* client and **strong security**

Improving [NPP22,NPP23]

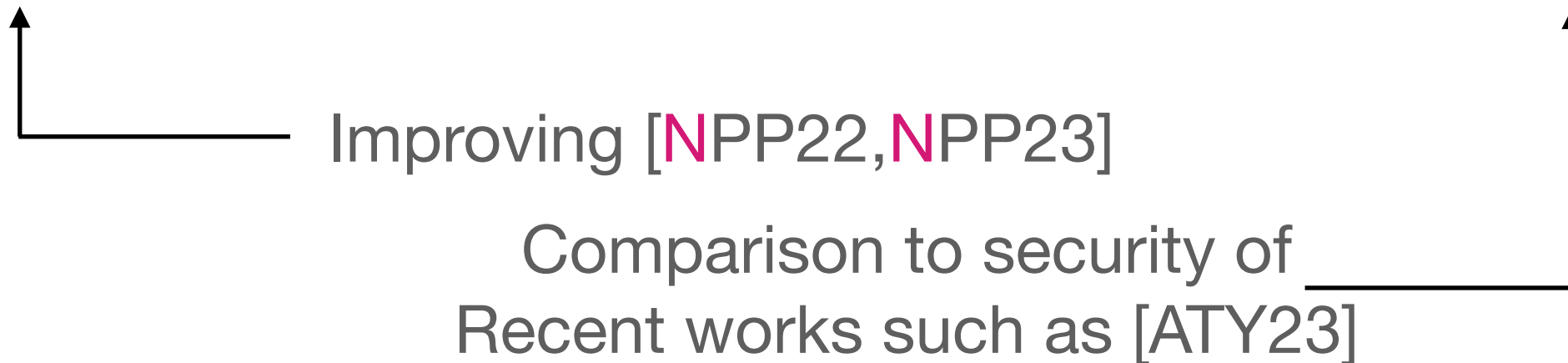


Research Question (3) - Our Result

- **From Secret-Key to Public-Key** Encryption Schemes
 - Definitional Framework with Public Inputs
 - Constructions from DPVS

Our Results:

- Definitional framework of MCFE with public inputs
- Implications from *secret-key* MCFE to *public-key* FE/KP-ABE
- Constructions with **sub-vectors** *per* client and **strong security**

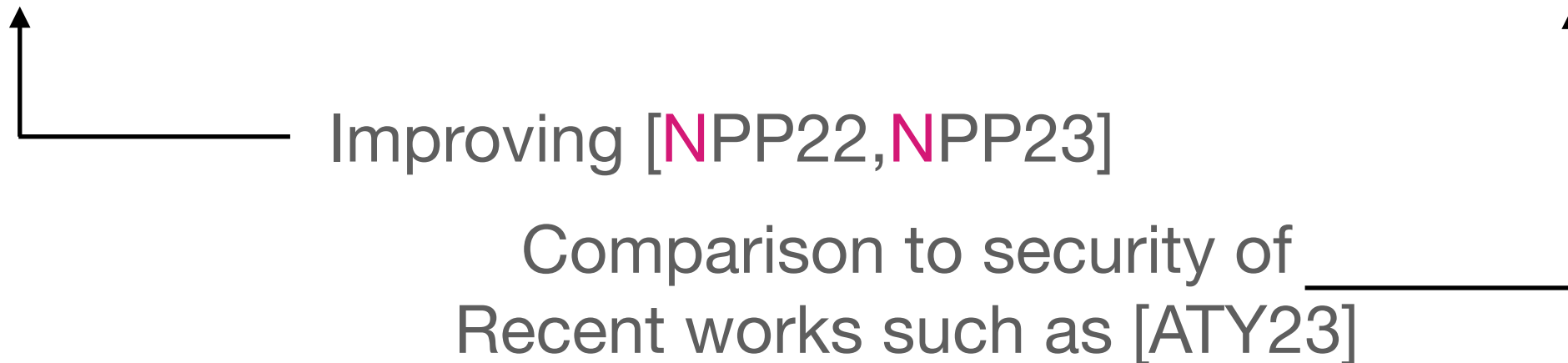


Research Question (3) - Our Result

- From Secret-Key to Public-Key Encryption Schemes
 - Definitional Framework with Public Inputs
 - Constructions from DPVS

Our Results:

- Definitional framework of MCFE with public inputs
- Implications from *secret-key* MCFE to *public-key* FE/KP-ABE
- Constructions with **sub-vectors** *per* client and **strong security**



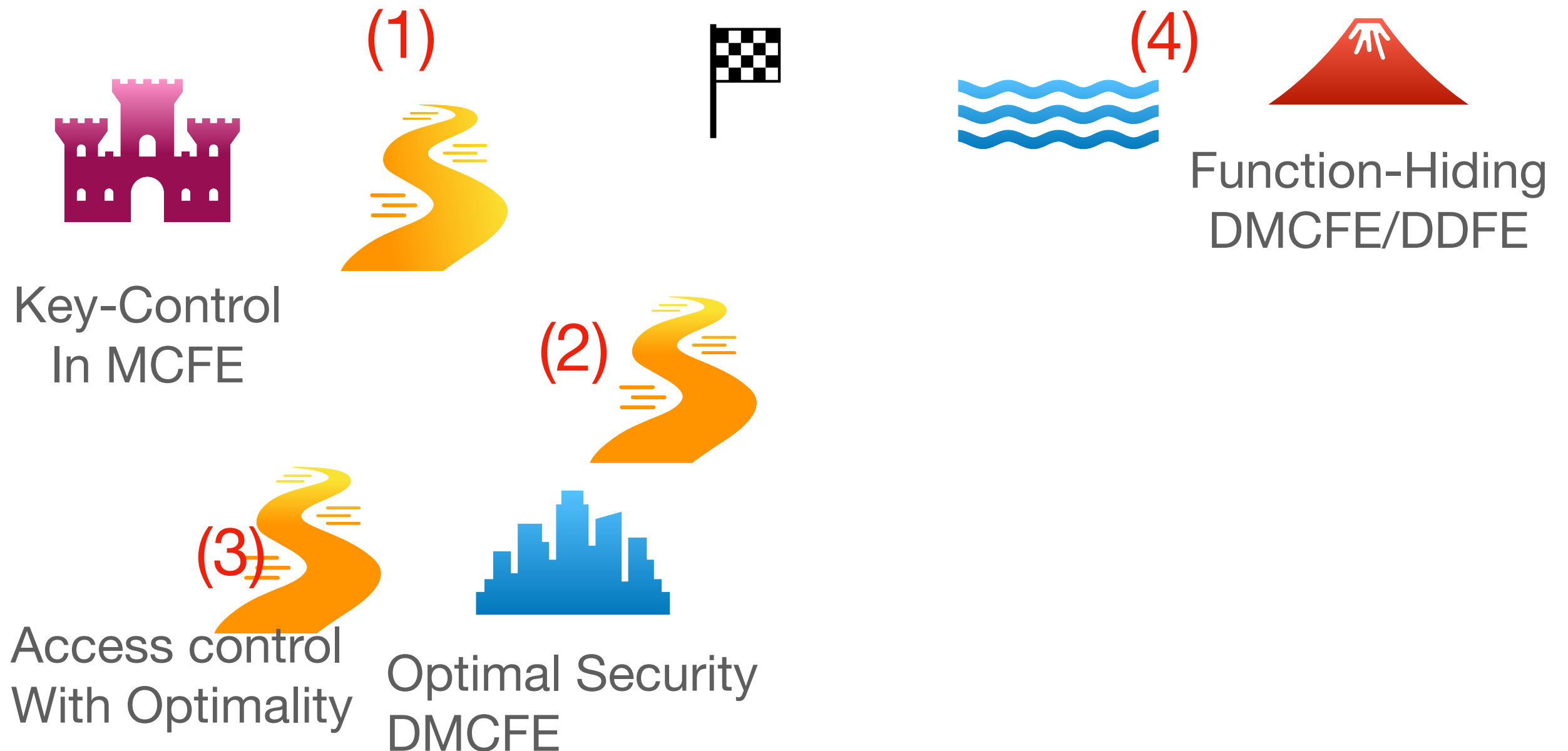
Can be enhanced further with [NPS24],
e.g. repetitions on attributes

Outline

- Introduction & Main Techniques Overview
- Controlling Functional Keys
- Optimal Security Notion
- From Secret-Key to Public-Key Encryption Schemes
- Conclusion and Future Works

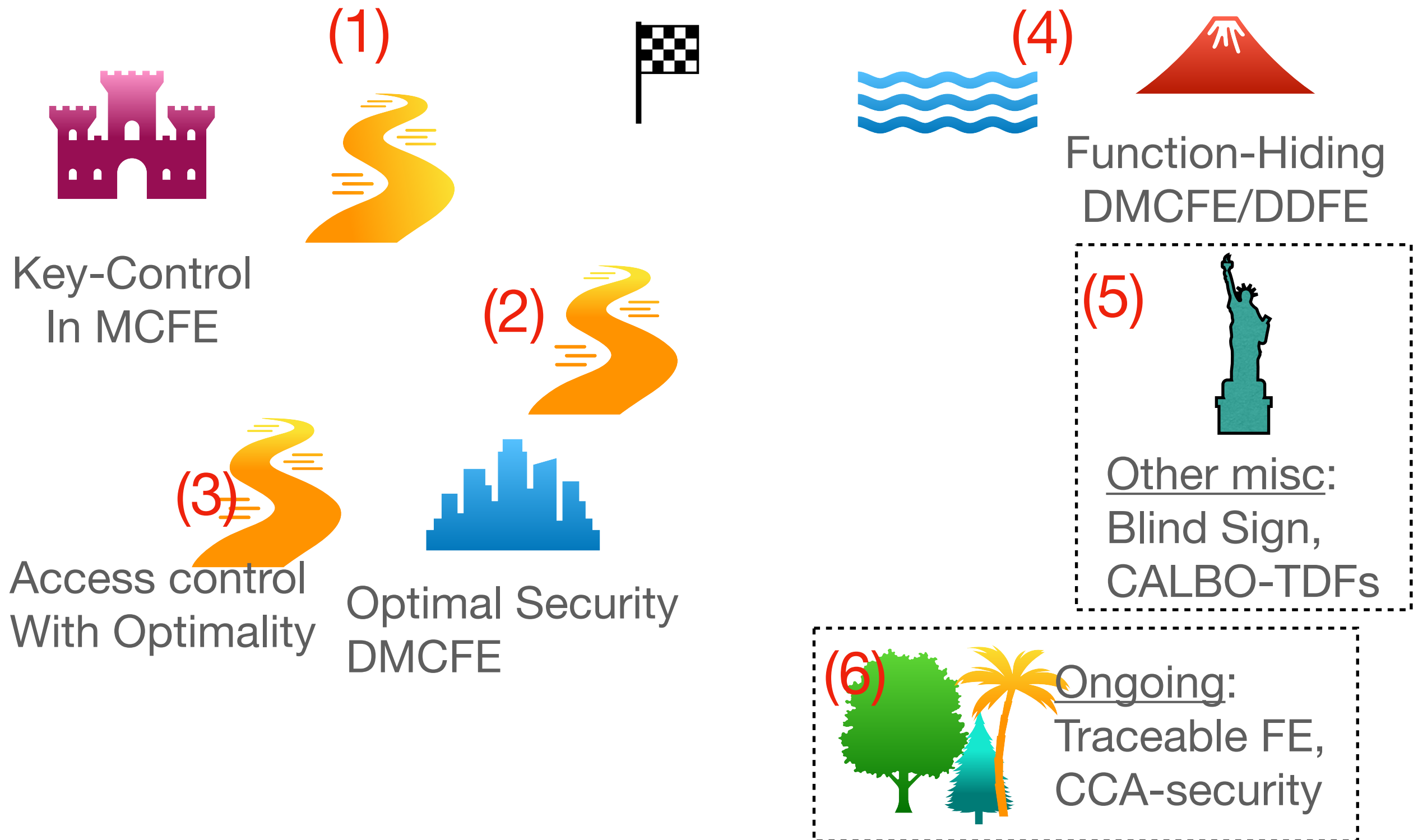
The picture so far

Map of projects/Qs - Future works



The picture so far

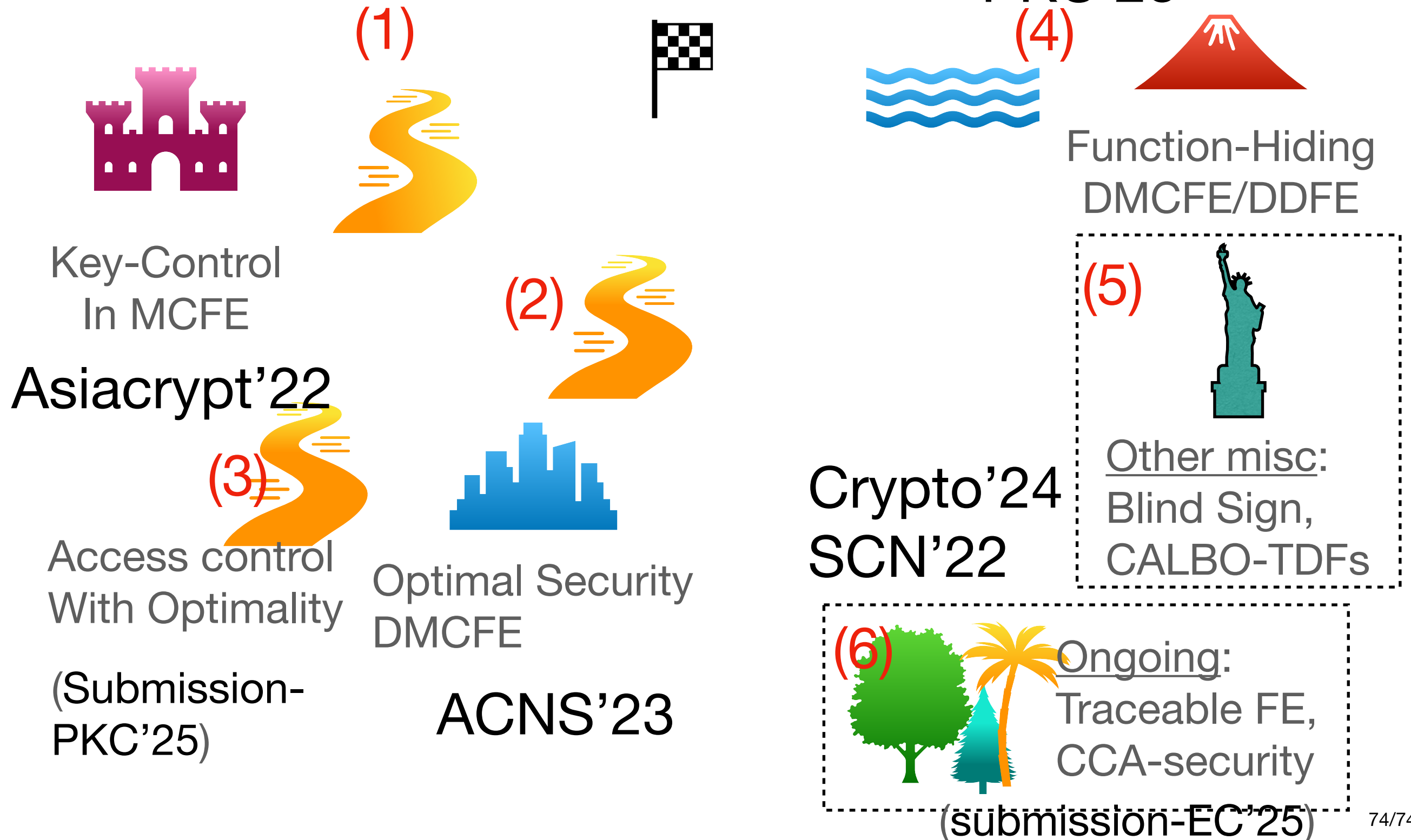
Map of projects/Qs - Future works



The picture so far

Map of projects/Qs - Future works

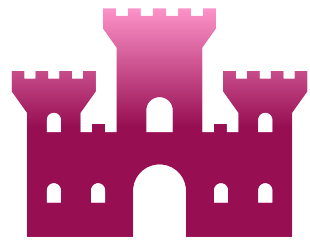
In CiC'24
And submission-
PKC'25



The picture so far

Map of projects/Qs - Future works

In CiC'24
And submission-
PKC'25



Key-Control
In MCFE

(1)



Asiacrypt'22

(3)

Access control
With Optimality

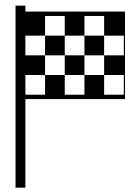
(Submission-
PKC'25)

(2)



Optimal Security
DMCFE

ACNS'23



(4)



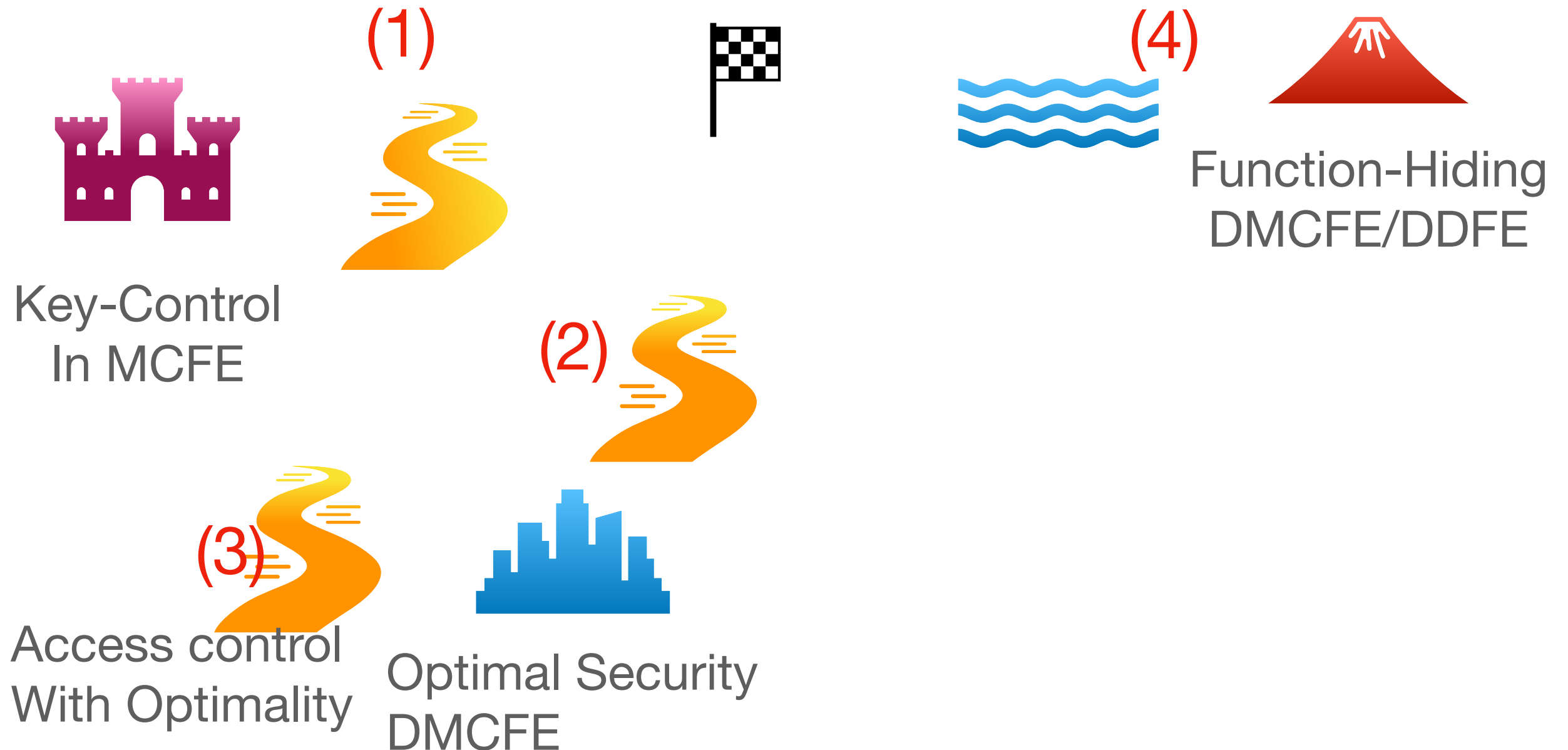
Function-Hiding
DMCFE/DDFE

Crypto'24
SCN'22

(submission-EC'25)

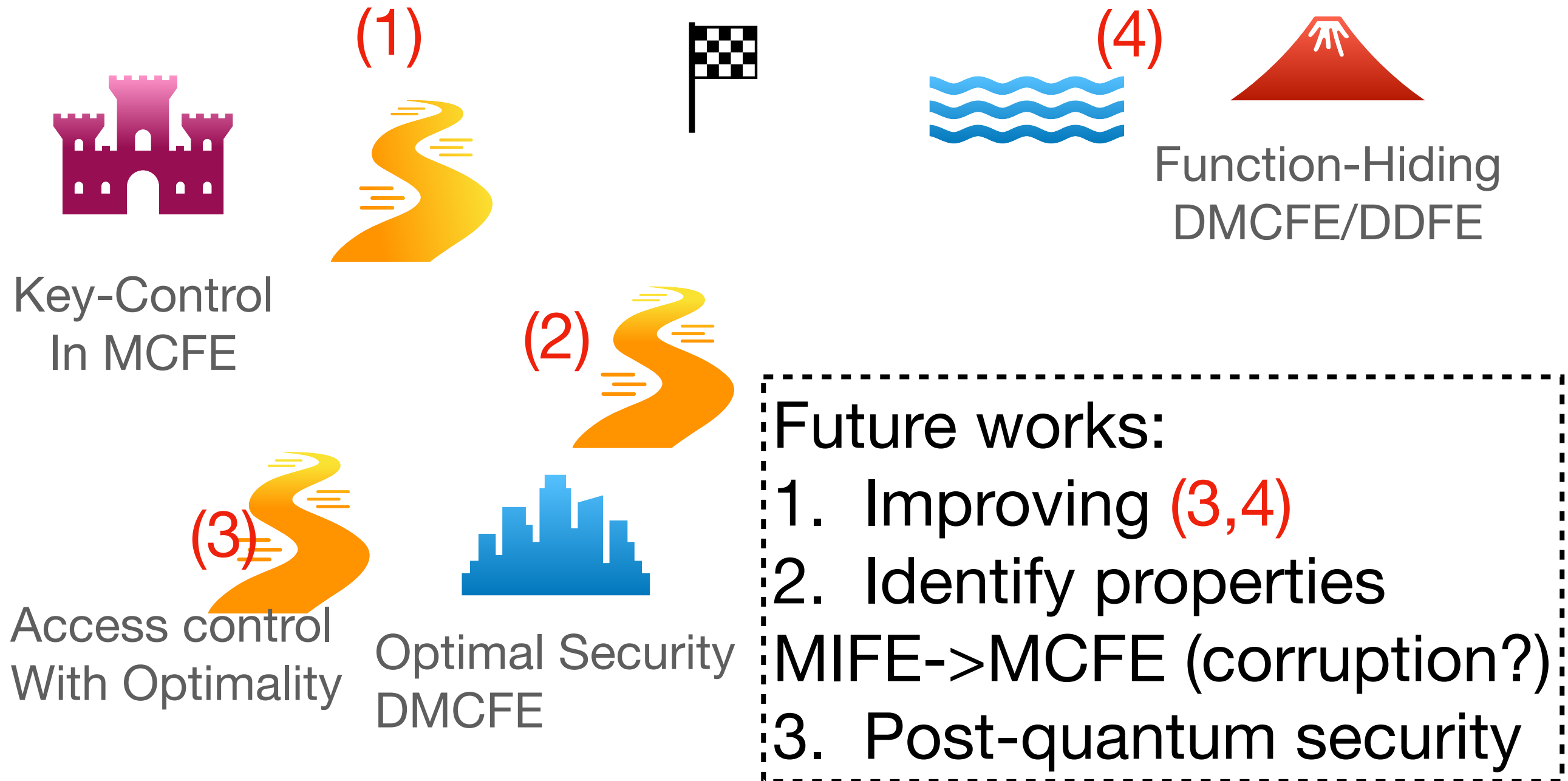
The picture so far

Map of projects/Qs - Future works



The picture so far

Map of projects/Qs - Future works



The picture so far

Map of projects/Qs - Future works

