Functional Encryption

Allison Lewko, Microsoft Research

The Cast of Characters

This talk will feature work by:

Dan Boneh

Jon Katz

Amit Sahai

Brent Waters

Shweta Agrawal

Sergey Gorbunov

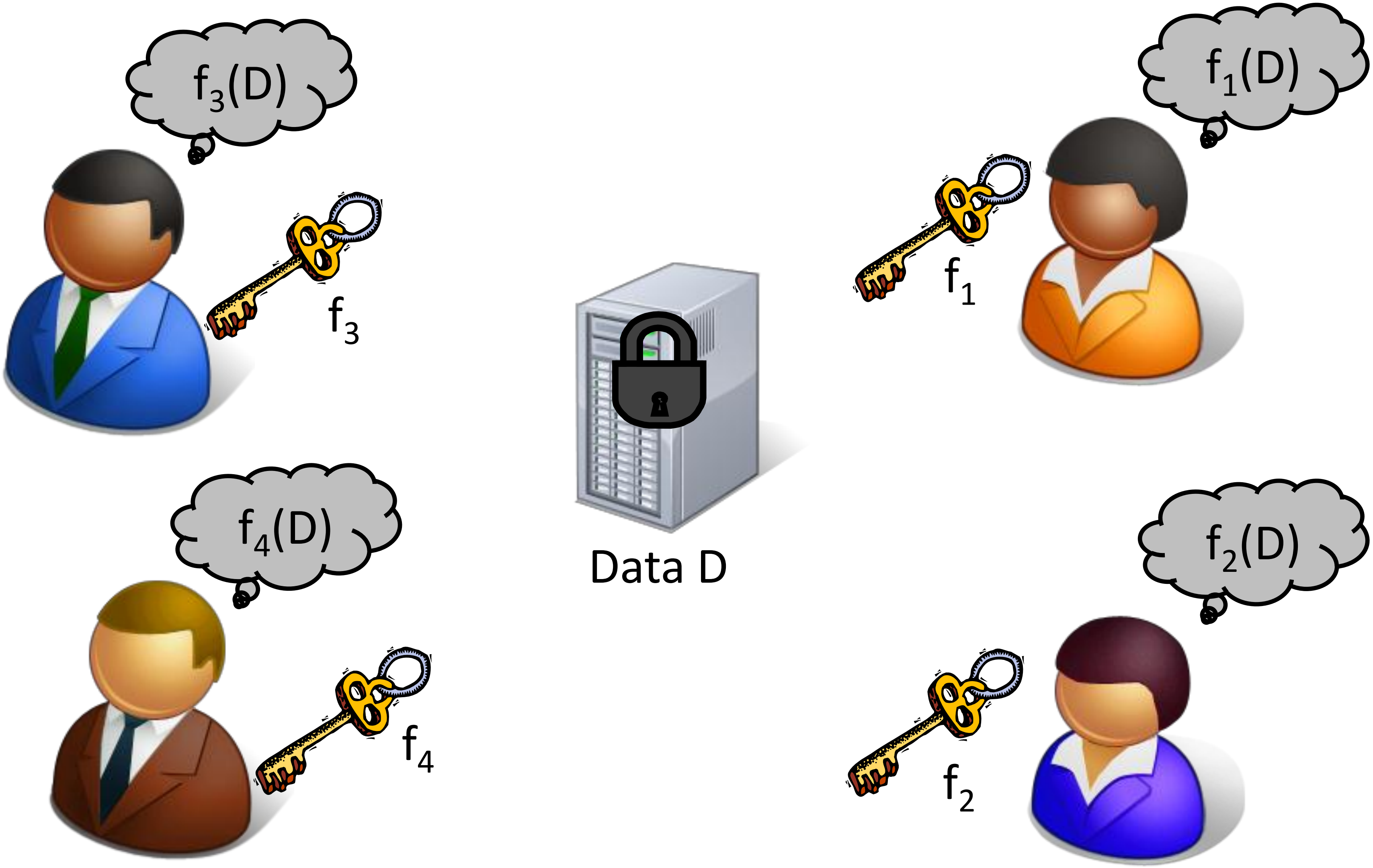
Vinod Vaikuntanathan

Hoeteck Wee

Special guest appearances by:
Adam O'Neill,
Yael Kalai,
Shafi Goldwasser,
Raluca Ada Popa,
Nickolai Zeldovich

Motivation

A Broad Vision



A First Step

How might we hide the access policy itself?

Inner Product Encryption

λ= security parameter

n = vector length

Setup(λ, n):

generate public parameters PP and master key MSK

KeyGen($\vec{v}$, MSK):

generate a user key for a given vector of length n

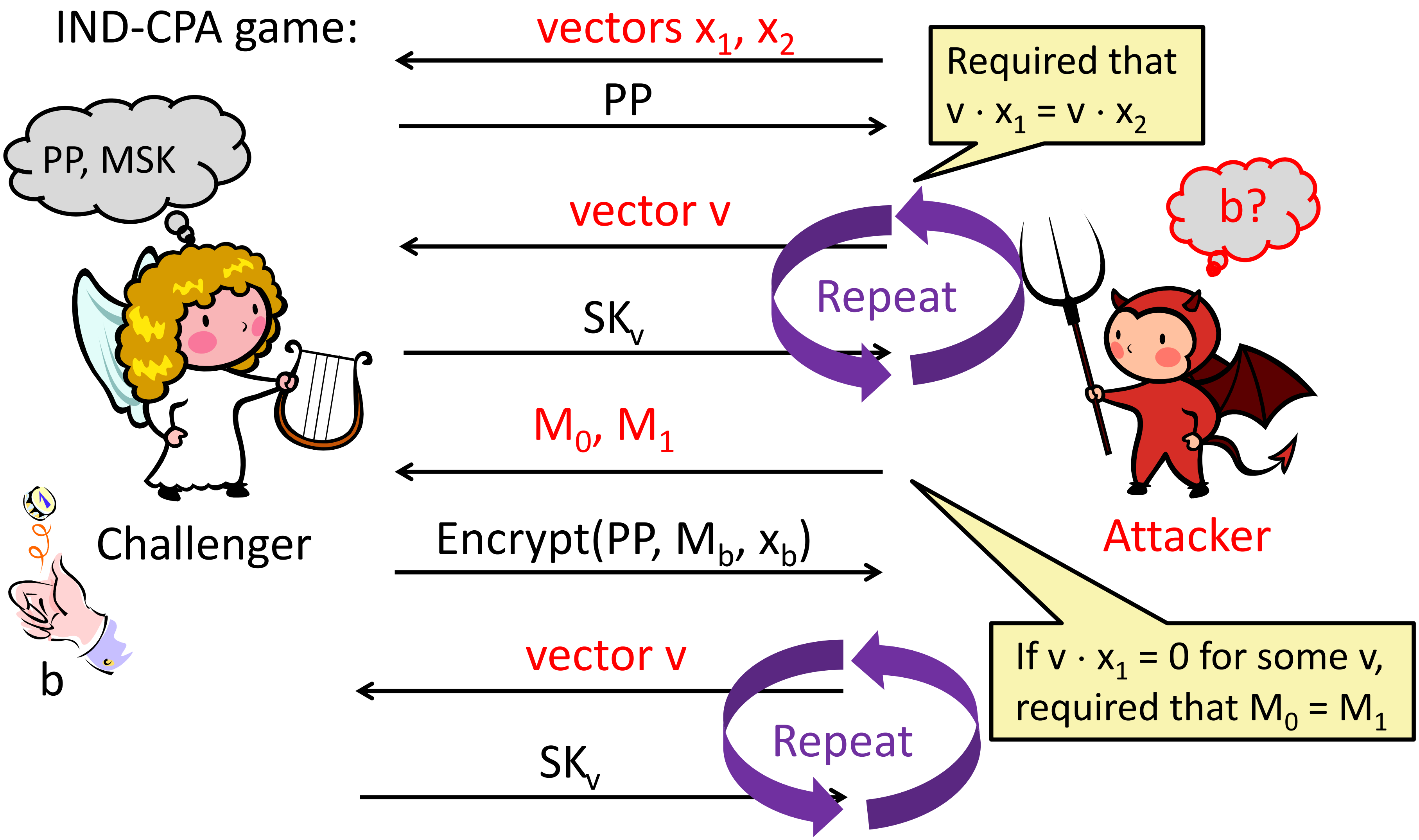
Encrypt(PP, M, $\vec{x}$):

encrypt message M under a vector of length n

Decrypt(CT, SK):

decrypt ciphertext using a key: successful iff $\vec{x} \cdot \vec{v} \equiv 0$

Security Definition for IPE (selective) [KSW08]



Construction Intuition

We want to compute $\vec{x} \cdot \vec{v}$ while hiding $\vec{x}$:

Basic idea: compute $\vec{x} \cdot \vec{v}$ in the exponent

CT: g^{x_1} g^{x_2} g^{x_3} $\ldots$ g^{x_n}

SK: g^{v_1} g^{v_2} g^{v_3} $\ldots$ g^{v_n}

$e(g,g)^{x_1 v_1} e(g,g)^{x_2 v_2} e(g,g)$

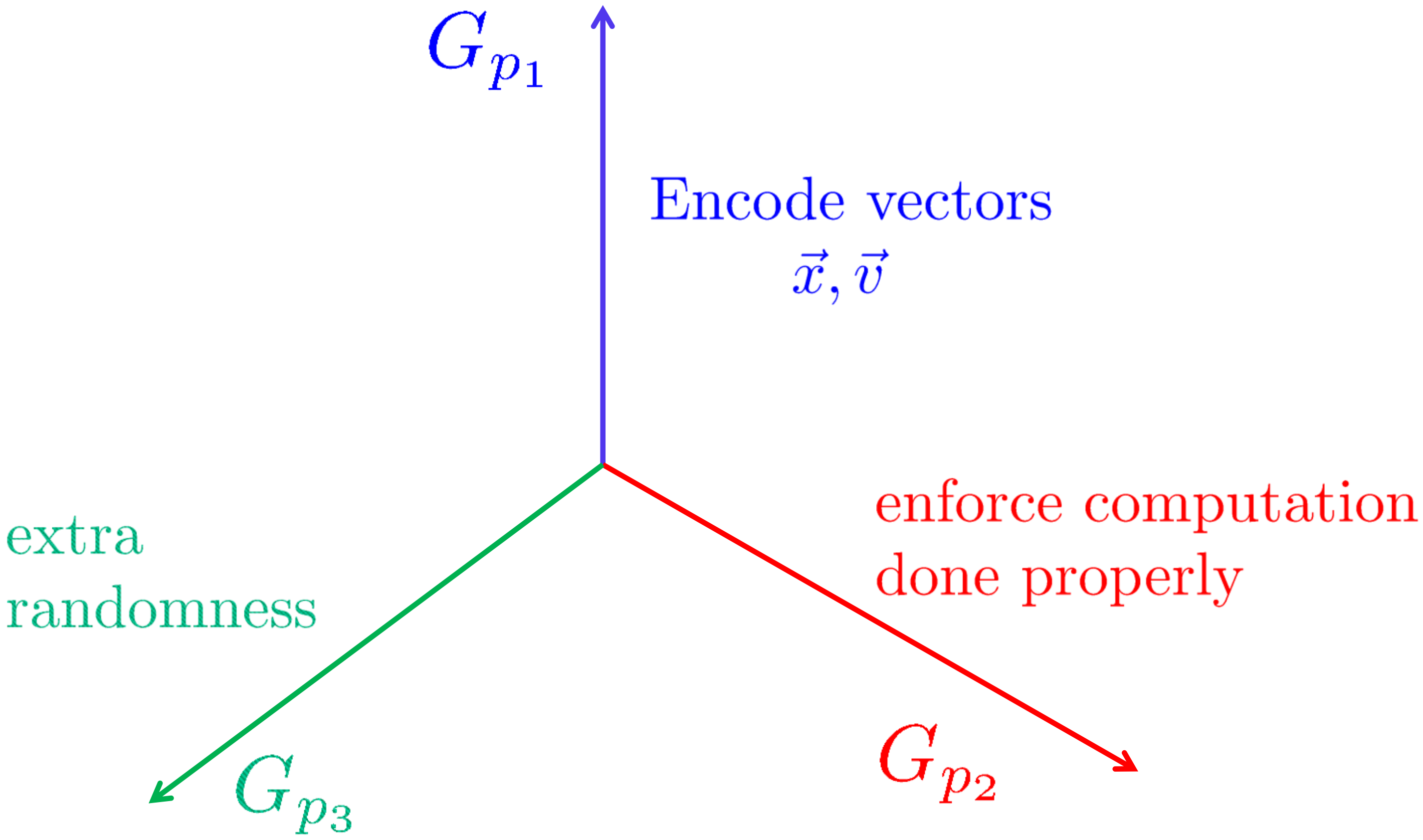
add randomness that cancels only when computation done properly

Some remaining problems:

Can tell if $x_i = 0$ or not

Can permute the computation

Subgroup Roles



IPE Construction (Predicate Only) [KSW08]

Setup (λ, n): generate G of order $N = p_1 p_2 p_3$

$$PP = g,\ g,\ gR,\ \{h_i V_i,\ k_i W_i\}_{i=1}^{n}$$

Encrypt$(\vec{x} = (x_1, x_2, \ldots, x_n))$:

choose $s, \alpha, \beta \in \mathbb{Z}_N$

$$\mathrm{CT} = g^s,\ \{(h_i V_i)^s (gV_i)^{\alpha x_i} U_i,\ (k_i W_i)^s (gV_i)^{\beta x_i} Z_i\}_{i=1}^{n}$$

2 parallel systems, stayed tuned for why we want this

KeyGen$(\vec{v} = (v_1, v_2, \ldots, v_n))$:

$$\mathrm{SK} = AB \prod_{i=1}^{n} h_i^{-r_i} k_i^{-t_i},\ \{g^{r_i} g^{\delta v_i}, g^{t_i} g^{\sigma v_i}\}_{i=1}^{n}$$

Decryption

CT: g^s $(h_i V_i)^s (g V_i)^{\alpha x_i} U_i$ $(k_i W_i)^s (g V_i)^{\beta x_i} Z_i$

Take product for $i = 1$ to n

SK: $AB \prod_{i=1}^{n} h_i^{-r_i} k_i^{-t_i}$ $g^{r_i} g^{\delta v_i}$ $g^{t_i} g^{\sigma v_i}$

$\prod_{i=1}^{n} e(g, h_i)^{-s r_i} e(g, k_i)^{-s t_i}$ $e(g, h_i)^{s r_i} e(g, g)^{\alpha \delta x_i v_i}$ $e(g, k_i)^{s t_i} e(g, g)^{\beta \sigma x_i v_i}$

$e(g, g)^{(\alpha\delta + \beta\sigma)\vec{x}\cdot\vec{v}}$

$= 1$ if and only if $\vec{x} \cdot \vec{v} \equiv 0 \mod p_1$

Proof Intuition

New challenge:

Adversary attempting to distinguish CT under $\vec{x}$ *from CT under* $\vec{y}$

requests key for $\vec{v}$ *such that* $\vec{v} \cdot \vec{x} = \vec{v} \cdot \vec{y} = 0$

Natural approach would be a hybrid
changing $\vec{x}$ to $\vec{y}$ one coordinate at a time:

$$(x_1, \ldots, x_n) \quad \Rightarrow \quad \boxed{(x_1, \ldots, x_i, y_{i+1}, \ldots, y_n)} \quad \Rightarrow \quad (y_1, \ldots, y_n)$$

may not be orthogonal to $\vec{v}$!

Proof Intuition

Idea: use two parallel systems, change one half at a time

Hybrid structure: CT exponent vectors change as

$$(\vec{x}, \vec{x}) \;\Rightarrow\; (\vec{x}, \vec{0}) \;\Rightarrow\; (\vec{x}, \vec{y}) \;\Rightarrow\; (\vec{0}, \vec{y}) \;\Rightarrow\; (\vec{y}, \vec{y})$$

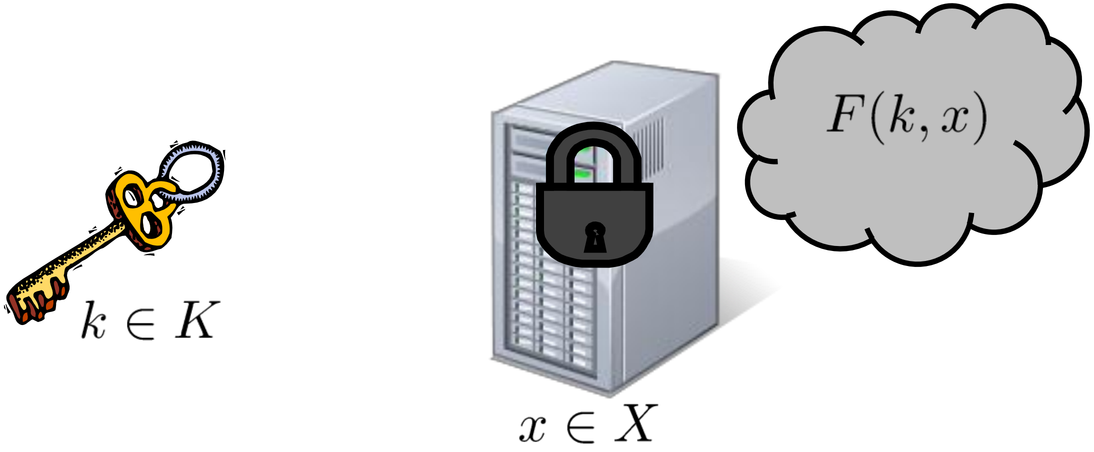
Why go through $\vec{0}$?

$\vec{0}$ *is orthogonal to everything, and we can go from* $\vec{x}$ *to* $\vec{0}$ *with Subgroup Decision Assumptions*

Back to General Functionalities…

A general definition [BSW11]:

Def. 1.
A functionality F is a function $F : K \times X \to \{0,1\}^*$ whose domain is the product of a key space K and a plaintext space X.
It is required that K include an "empty key" ϵ.



Formal Specification

λ= security parameter

Setup(λ):

generate public parameters PP and master key MSK

KeyGen(k, MSK):

generate a user key for a given $k \in K$

Encrypt(PP, $x \in X$):

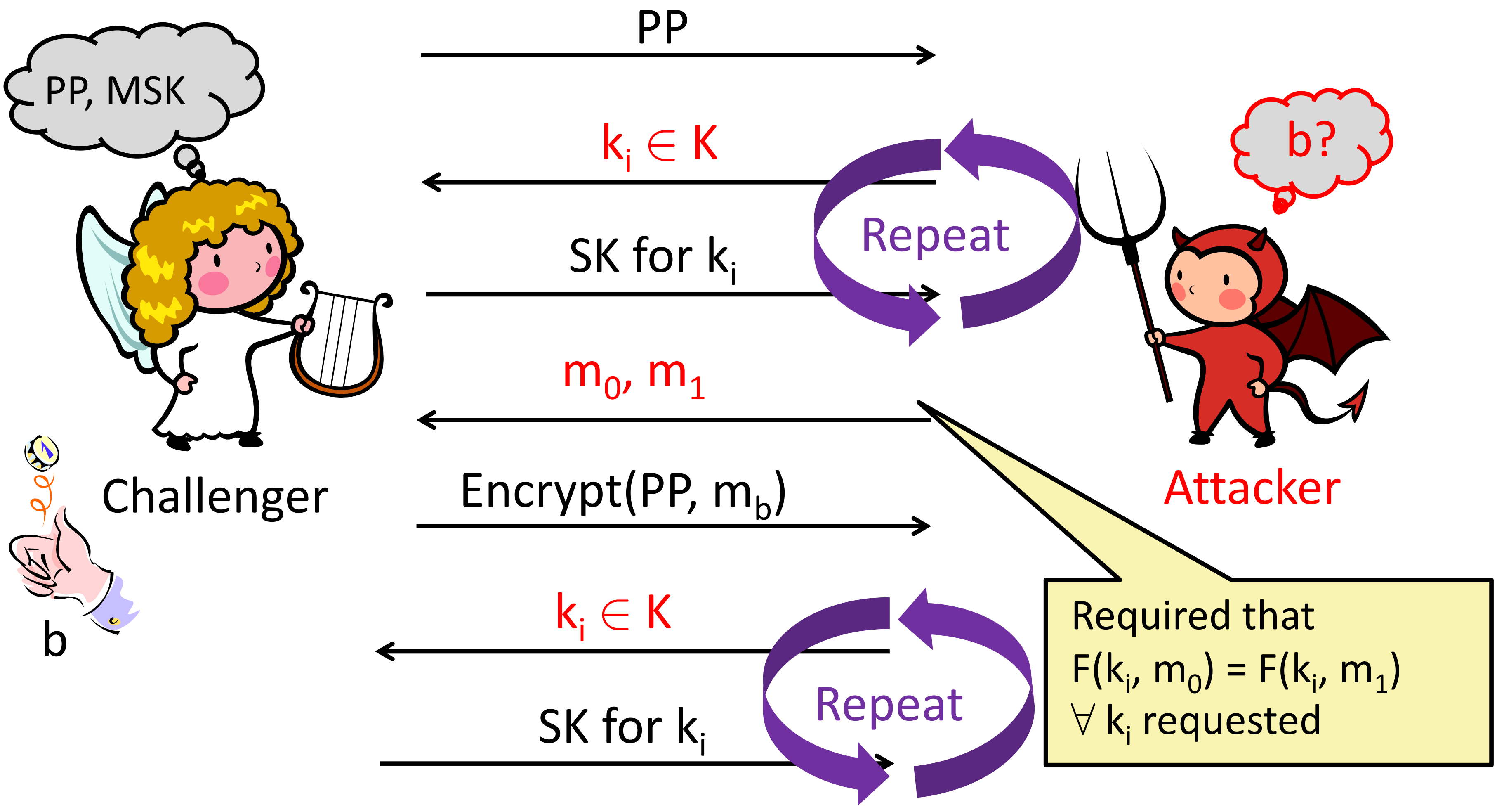
encrypt message x

Decrypt(CT, SK):

decrypt ciphertext using a key to obtain F(k,x)

IND Game-Based Security Definition

IND-CPA game:



When IND Security May Be Insufficient

It does not capture computational properties of the functionality F:

Example:
Consider $K = \{\epsilon\}$, $X = \{0,1\}^n$
$F(\epsilon, x) := \pi(x)$

one-way permutation

Proposed Construction:
$Encrypt(x) = x$

$$F(\epsilon, x_1) = F(\epsilon, x_2) \Leftrightarrow x_1 = x_2$$

So this is "secure" under game-based definition!

A Further Example

from [O10]: Let $\mathcal{F} = \{f_1, \ldots, f_n\}$ be set of functions associated with keys
Let g be a function so that given $f_1(x)||\ldots||f_n(x)$,
it is hard to guess $g(x)$
And $g(x) = g(y) \Leftrightarrow f_1(x)||\ldots||f_n(x) = f_1(y)||\ldots||f_n(y)$

Let (Setup, Encrypt, Decrypt) be a Public Key Encryption scheme

Let (Setup*, KeyGen*, Encrypt*, Decrypt*) be a FE scheme

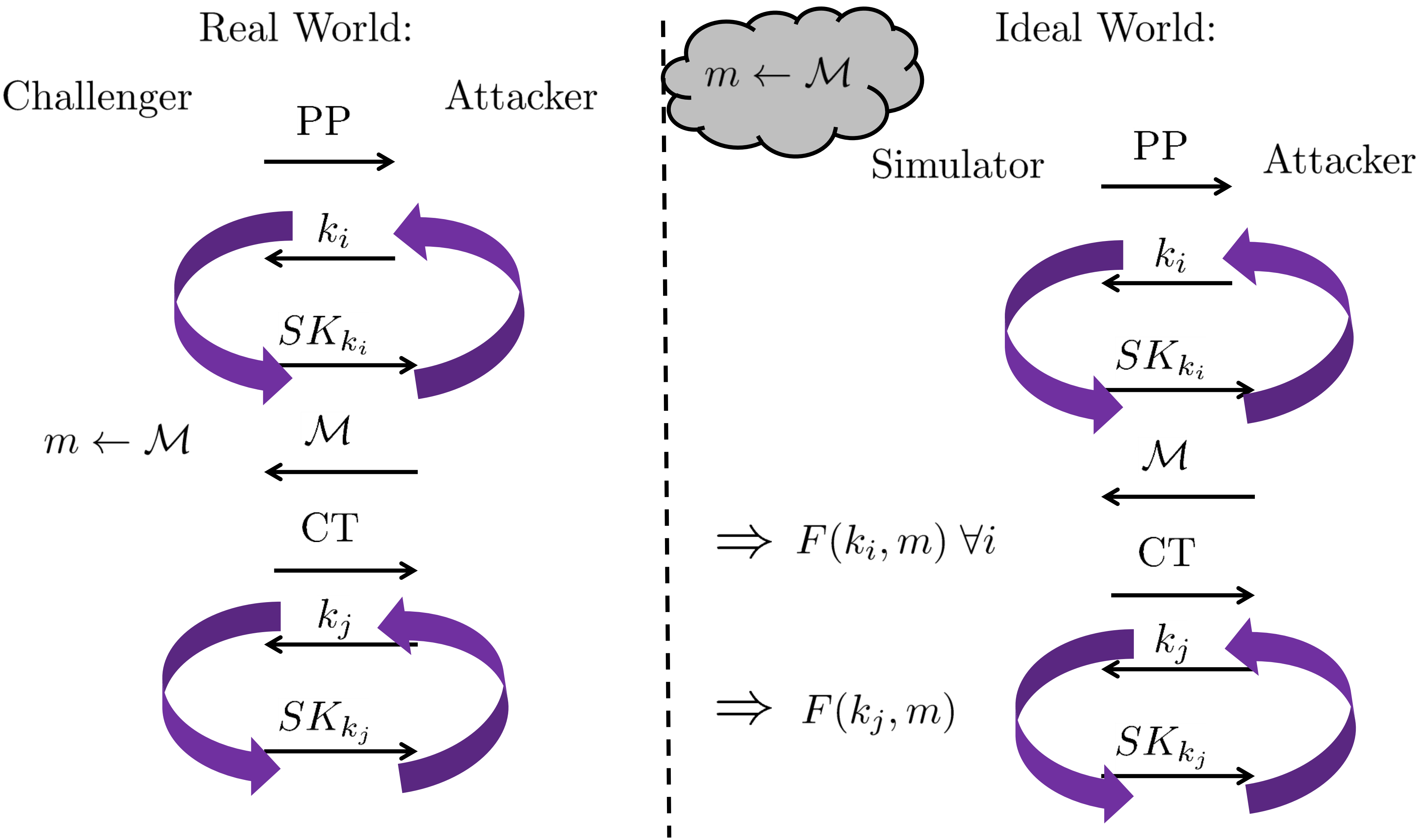
New FE scheme:

Setup: run Setup and Setup* to get $(pk, sk), (pk^*, sk^*)$
secret share sk as $\omega_1, \ldots, \omega_n$
set $pk := pk||pk^*$, $sk := \omega_1||\ldots||\omega_n||sk^*$

KeyGen(f_i): run KeyGen*(f_i) to get sk_i, set $sk_i := \omega_i||sk_i$

Encrypt(m): run Encrypt*(m) and Encrypt($g(m)$), concat results

A Simulation-Based Security Definition



Impossibility Results for Sim-Based Security

$m_1, \ldots, m_N \leftarrow \mathcal{M}$

Ideal World:

[BSW11]

Simulator $\xrightarrow{\text{PP}}$ Attacker

$\xleftarrow{k_i}$

$\xrightarrow{SK_{k_i}}$

$\xleftarrow{\mathcal{M}}$

$\Rightarrow \{F(k_i, m_\ell)\ \forall i, \ell\}$ $\xrightarrow{\text{CT}_1, \ldots, \text{CT}_N}$

$\xleftarrow{k_j}$

$\Rightarrow F(k_j, m_1), \ldots, F(k_j, m_N)$ $\xrightarrow{SK_{k_j}}$

needs to encode too much!

Impossibility Results for Sim-Based Security

$m \leftarrow \mathcal{M}$

Ideal World:

[AGVW12]

$$\text{Simulator} \xrightarrow{\text{PP}} \text{Attacker}$$

$$\xleftarrow{k_i}$$

$$\xrightarrow{SK_{k_i}}$$

$$\xleftarrow{\mathcal{M}}$$

$$\Rightarrow \{F(k_i, m)\ \forall i\} \qquad \xrightarrow{\text{CT}}$$

for some F, needs to encode too much!

**Can avoid this by bounding the queries*

Positive Result for Bounded Collusion [GVW12]

- Impose bound of q on key queries
- Can build a scheme for general circuit functionalities
- Techniques include:
  secret sharing, garbled circuits

Breaking News Update

Result for general functionalities with succinct CT, Bounded collusion [GKPVZ13]:

- draws upon ABE and FHE constructions
- can be instantiated from LWE
- applications to garbled circuits, obfuscation, delegation