Functional Encryption

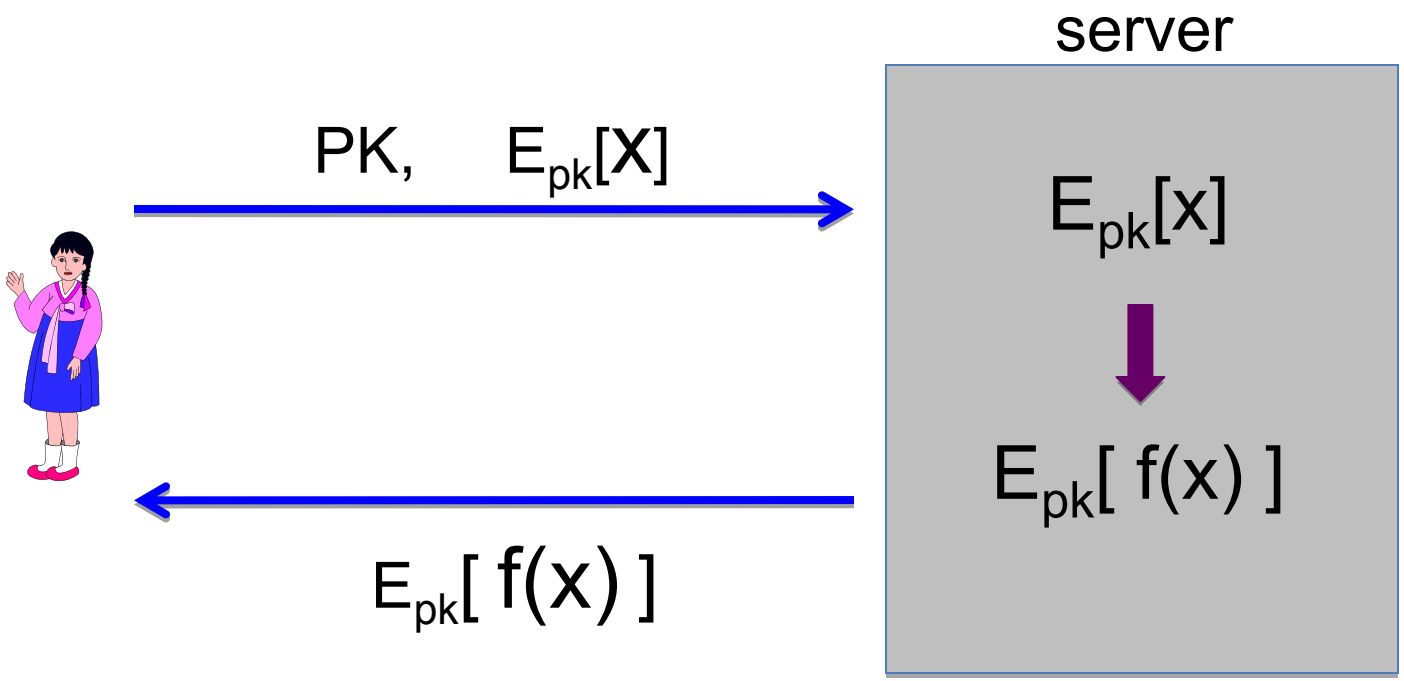
Dan Boneh

Stanford University

Joint work with Amit Sahai and Brent Waters

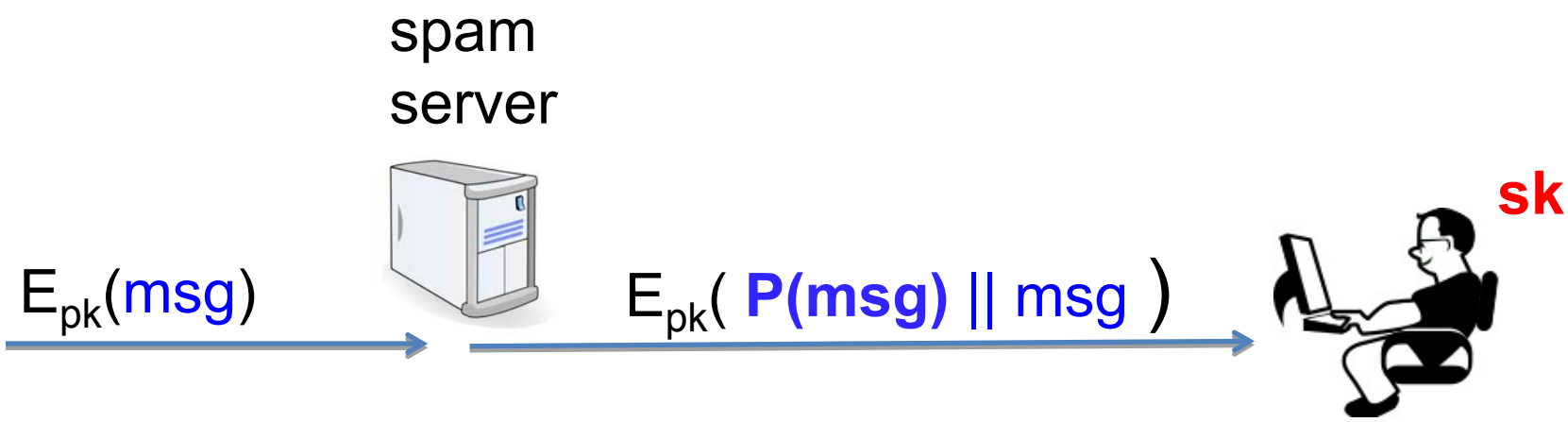
(TCC 2010)

Recall: fully homomorphic encryption



For any function f **[G'09, SV'10, vDGHV'10, …]**

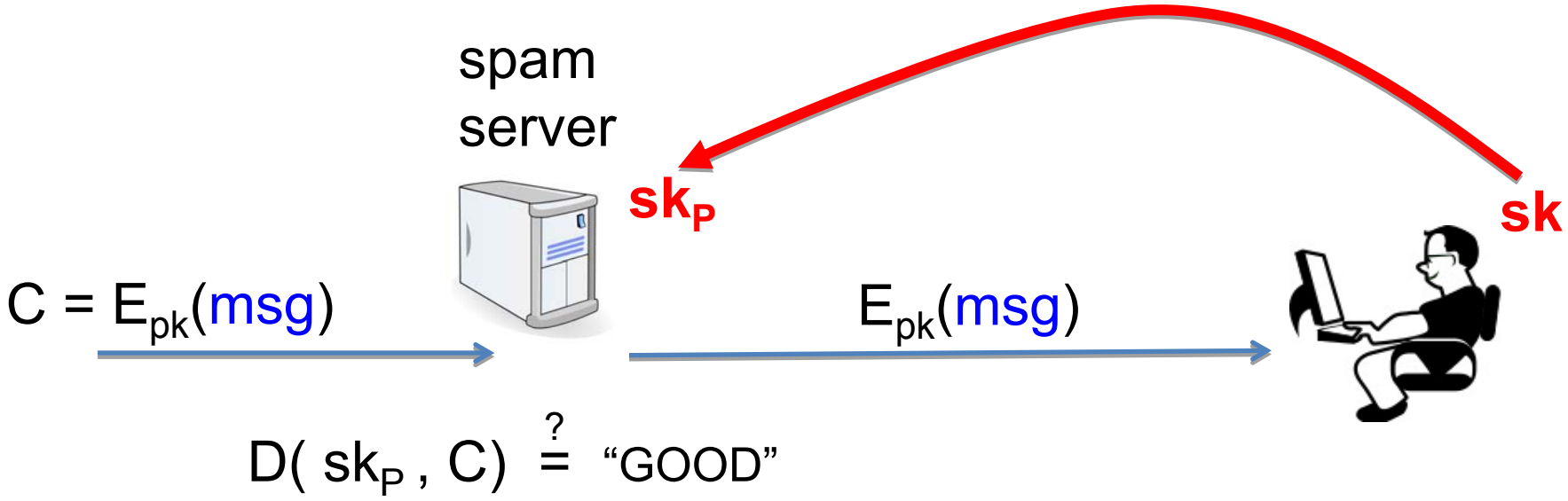
An Application: spam email



Spam predicate: **P(msg)** = { "SPAM" if spam ; "GOOD" if not

- The problem: can only do spam tagging

A Better Approach



- spam functionality: $D(sk_P, C) = P(msg)$
- sk_P reveals nothing else about msg

More generally: functional encryption

Let $\{ f_k : X \to M \}_{k \in K}$ be a family of functions $(\varepsilon \in K)$

Idea: for every $k \in K$ there is a key sk_k such that:

$$d(sk_k, encrypt(pp, x)) = f_k(x)$$

Plaintext space

Four algorithms:

- $(pp, mk) \leftarrow$ **setup**(λ)
- For $k \in K$: $sk_k \leftarrow$ **KeyGen**(mk, k)
- $c \leftarrow$ **enc**(pp, x)
- **dec**(sk_k, c) outputs $\mathbf{f_k(m)}$

The empty function:

$\mathbf{f_\varepsilon(x)}$: outputs info leaked by c, e.g. len(x)

$\mathbf{sk_\varepsilon = \varepsilon}$

A few warm-up examples

Standard public-key encryption: $K = \{ 1, \varepsilon \}$

$\forall x \in X$: $f_1(x) = x$

$dec(sk_1 , enc(pp, x)) = x$

Identity Based Encryption: $K = \{ id_1, \ldots, id_{2^n}, \varepsilon \}$

$X = \{ (id, m) \}$; $f_k((id,m)) = \begin{cases} m & \text{if } k=id \\ \perp & \text{otherwise} \end{cases}$

Plaintext space

$dec(\ldots (id,m)) = (m \text{ iff } k=id)$

A few warm-up examples

Attribute Based Encryption (ciphertext policy): $K = \{0,1\}^n \cup \{\varepsilon\}$

$X = \{ (P, m) \}$; $f_k((P,m)) = \begin{cases} m & \text{if } P(k)=1 \\ \square & \text{otherwise} \end{cases}$

Plaintexts: Predicate on n variables

$\text{dec}(sk_k , \underbrace{\text{enc}(pp, (P,m))}_{\text{leaks P}}) = (m \text{ iff } P(k)=1)$

$f_\varepsilon(P,m) = (P, \text{len}(m))$

General searching on enc. Data (predicate encryption)

Goal: Test if ciphertext **enc(pp, x)** satisfies **P(x)=1**

Key space: **K = { P: X □{0,1} } ∪ {ε}**

$$\forall x \in X: \quad f_P(x) = \begin{cases} 1 & \text{if } P(x)=1 \\ \square & \text{otherwise} \end{cases}$$

dec(**sk_P** , enc(pp, **X**)) = (**1 iff P(x)=1**)

does not leak x

Functional encryption: security

Goal: attacker has keys $sk_1, \ldots, sk_q$ and ciphertext $enc(pp, x)$.

should learn nothing more than $f_1(x), \ldots, f_q(x)$, $f_\varepsilon(x)$

Game based definition:

collusion attacks complicate things

- Attacker adaptively requests keys $sk_1, \ldots, sk_q$
- Cannot distinguish $enc(pp, x_1)$ from $enc(pp, x_2)$

where $f_k(x_1) = f_k(x_2)$ for all k in attacker's possession

(in particular $f_\varepsilon(x_1) = f_\varepsilon(x_2)$)

Known functional constructions

Public policy: [$f_\varepsilon(\text{policy}, m) = (\text{policy}, \text{len}(m))$]

- IBE, ABE (CP & KP) where predicate is a monotone formula

(circuits is still open)

Searching on enc. data: (a.k.a predicate encryption)

- Equality predicates: $P_k(x) = (1 \text{ iff } x=k)$ (anonymous IBE)
- Inner products: $P_u(v) = (1 \text{ iff } u \square v)$
- Anything beyond inner products is still open

Future applications

Main open problem:

- Functional encryption for all (or more) functionalities
- Harder than homomorphic encryption

Applications:

- Facial recognition on enc. images:

  sk_{face} : locate images that contain a specific face

- Cloud-based data mining on encrypted data:

  Run approved data-mining algorithm on enc. data

- Connections to program obfuscation (but not sufficient)

Restricted Homomorphic Encryption

Joint work with Gil Segev and Brent Waters

Restricted Homomorphic Encryption

Back in 2008: best homomorphic systems --

linear or **quadratic** operations

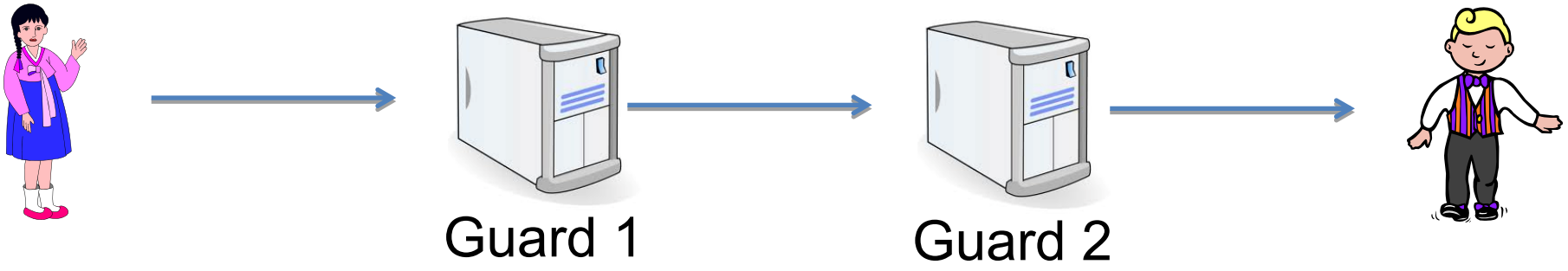
Prabhakaran and Rosulek [PR'08] :

- Built systems that **provably** support only linear operations.

More generally: can we build systems that support a restricted set of homomorphisms F ?

Applications [BSW'11]

Network guards on encrypted traffic:

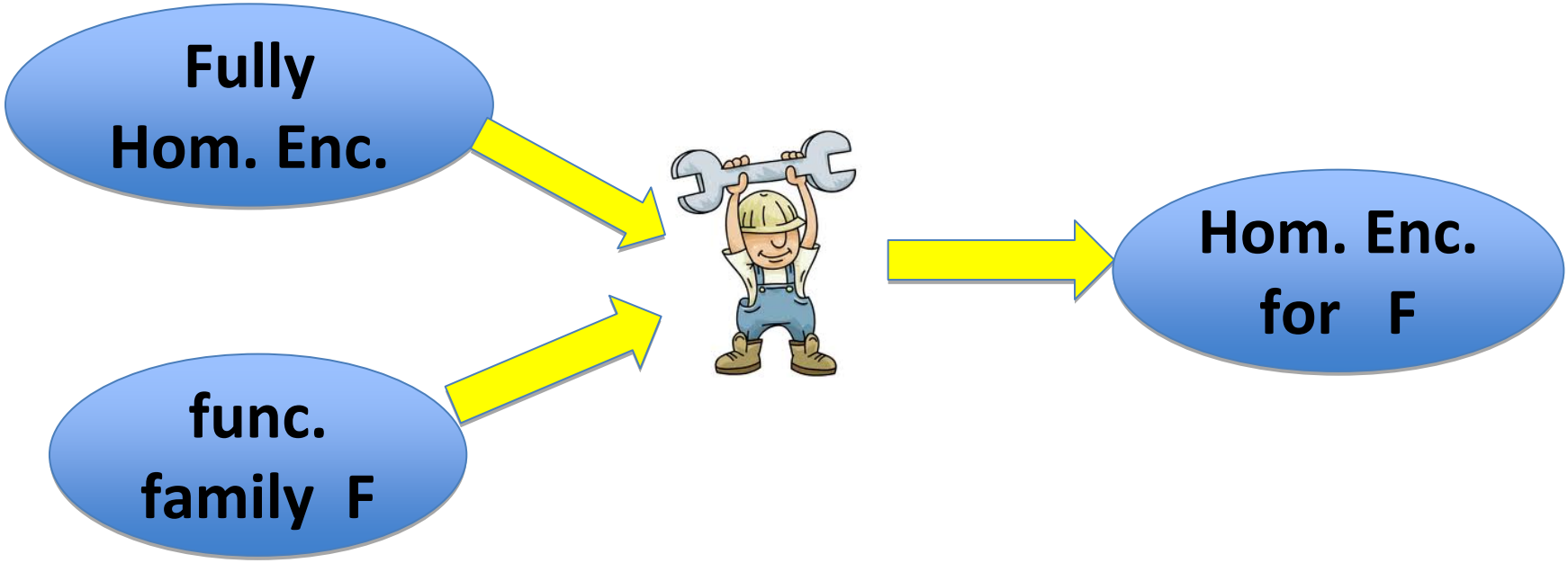


With restricted FHE:

guard can implement policy, but nothing else

Goal: restricted FHE that keeps ciphertext size short

A New Construction [BSW'11]



- Properties: no ciphertext expansion under constant iteration

THE END