Attribute-Based Encryption

Allison Lewko, Microsoft Research

The Cast of Characters

This talk will feature work by:

Brent Waters

Amit Sahai

Vipul Goyal

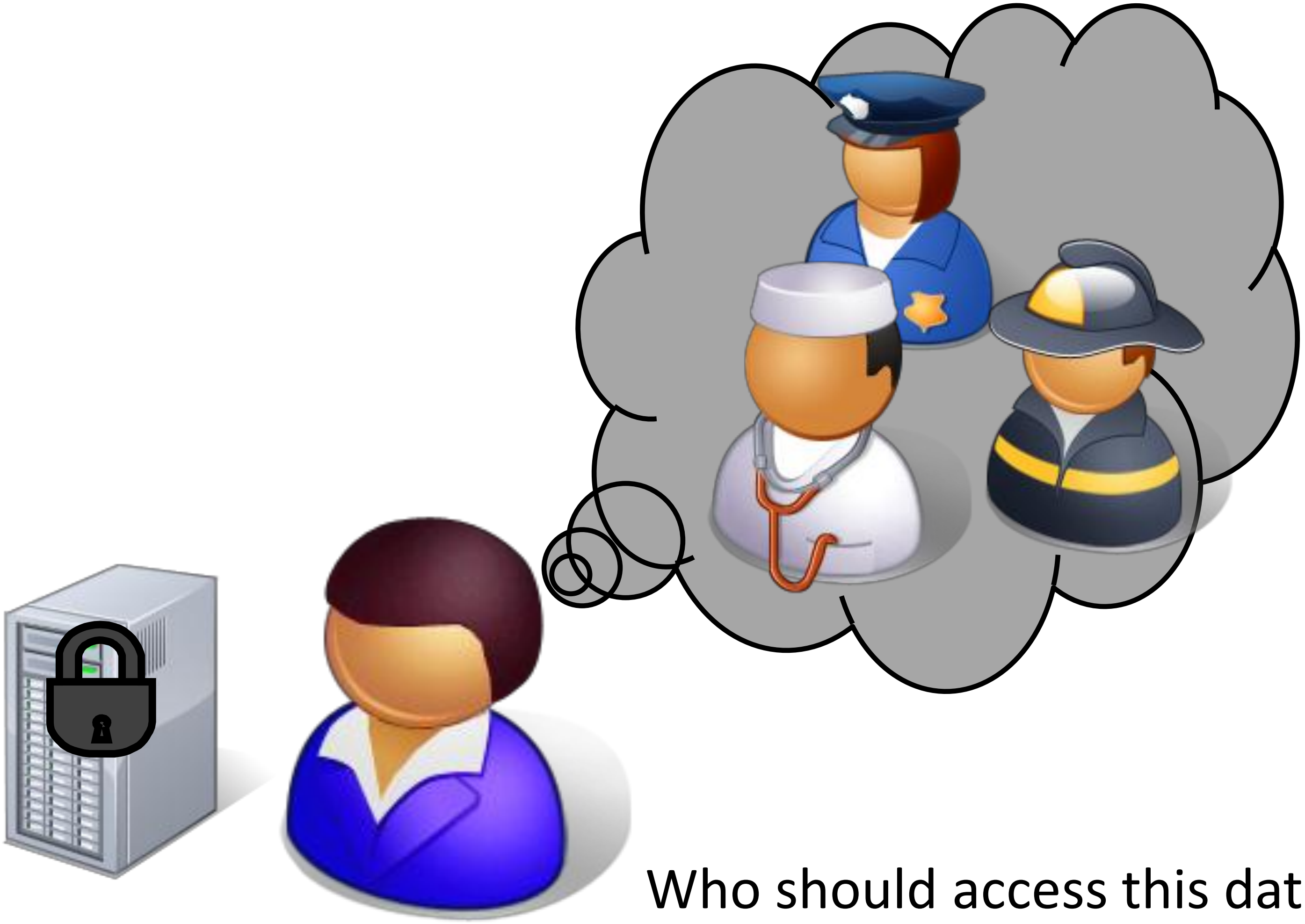
Omkant Pandey

Tatsuaki Okamoto

Katsuyuki Takashima

With special guest appearances by:
Melissa Chase, Dan Boneh,
Matt Franklin, Ran Canetti,
Shai Halevi, Jon Katz, Xavier Boyen,
Sanjam Garg, Craig Gentry,

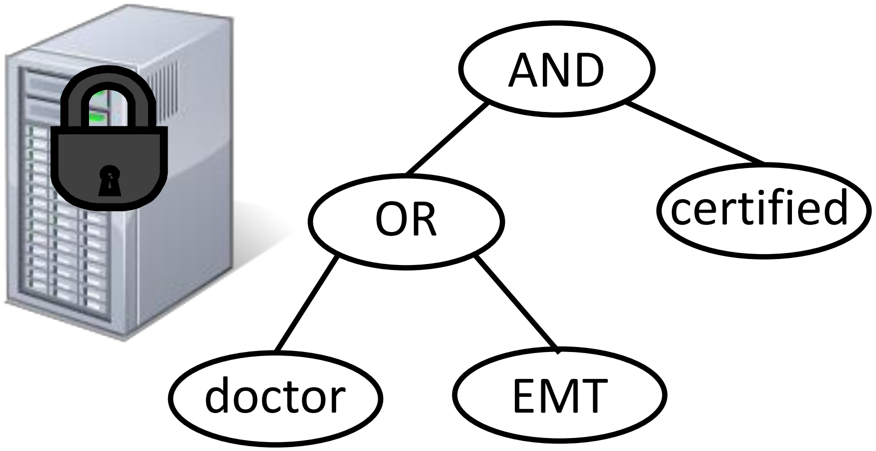
Motivation



Who should access this data?

ABE [SW05, GPSW06]

Describe authorized users with "attributes":



Key-Policy ABE Specification

λ= security parameter

U = attribute universe

Setup(λ, U):

generate public parameters PP and master key MSK

KeyGen(Policy, MSK):

generate a user key for a given policy

Encrypt(PP, M, $S \subseteq U$):

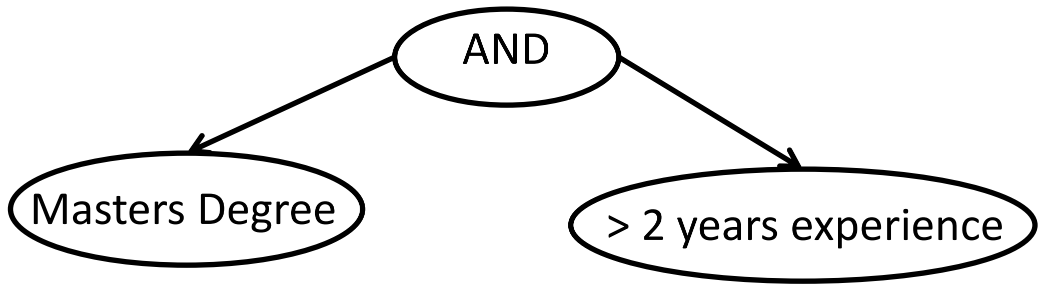
encrypt message M under attribute set S

Decrypt(CT, SK):

decrypt ciphertext using a key

Security Threat: Collusion

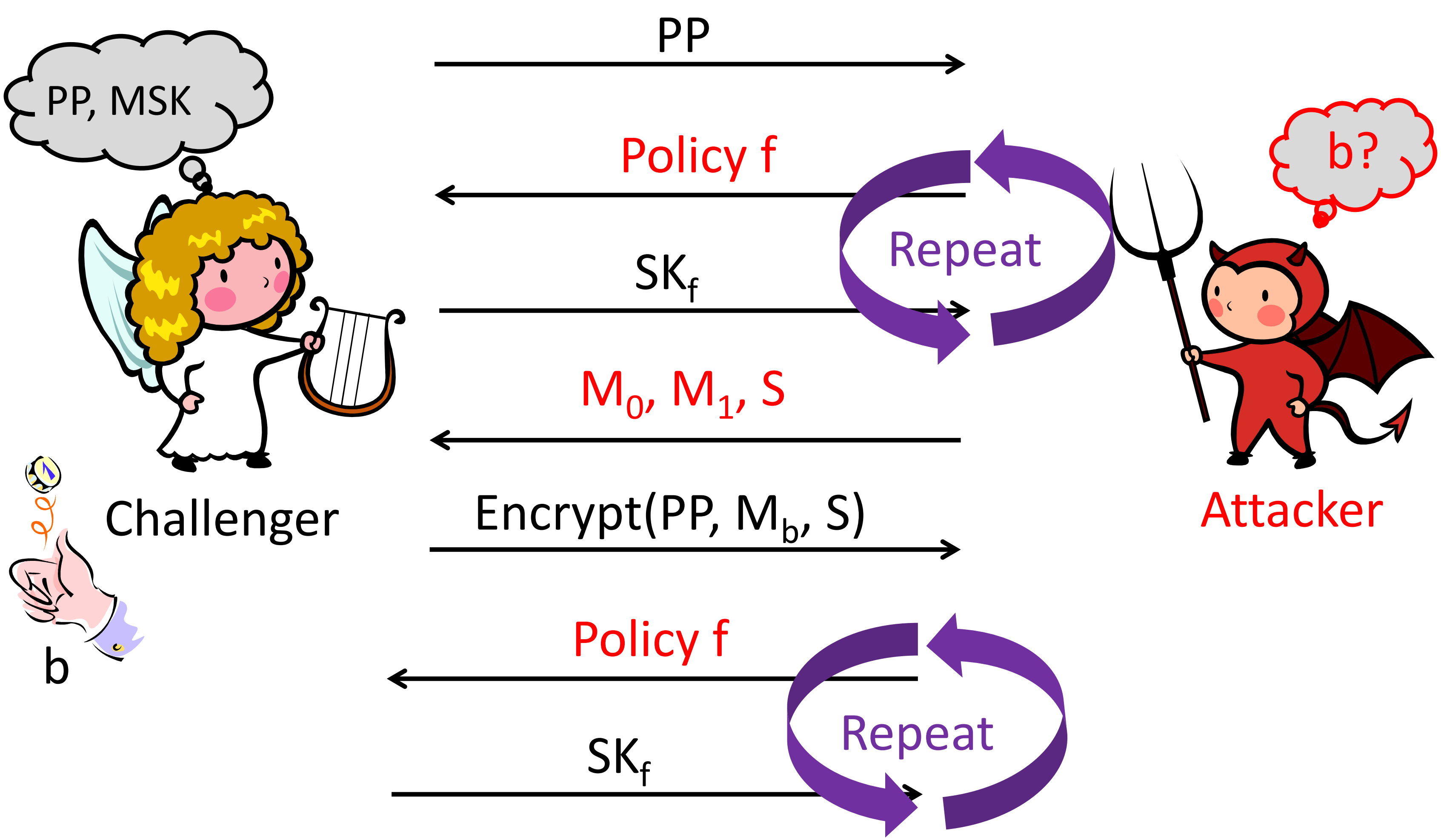
Example: encrypt a job posting:



Security Threat:

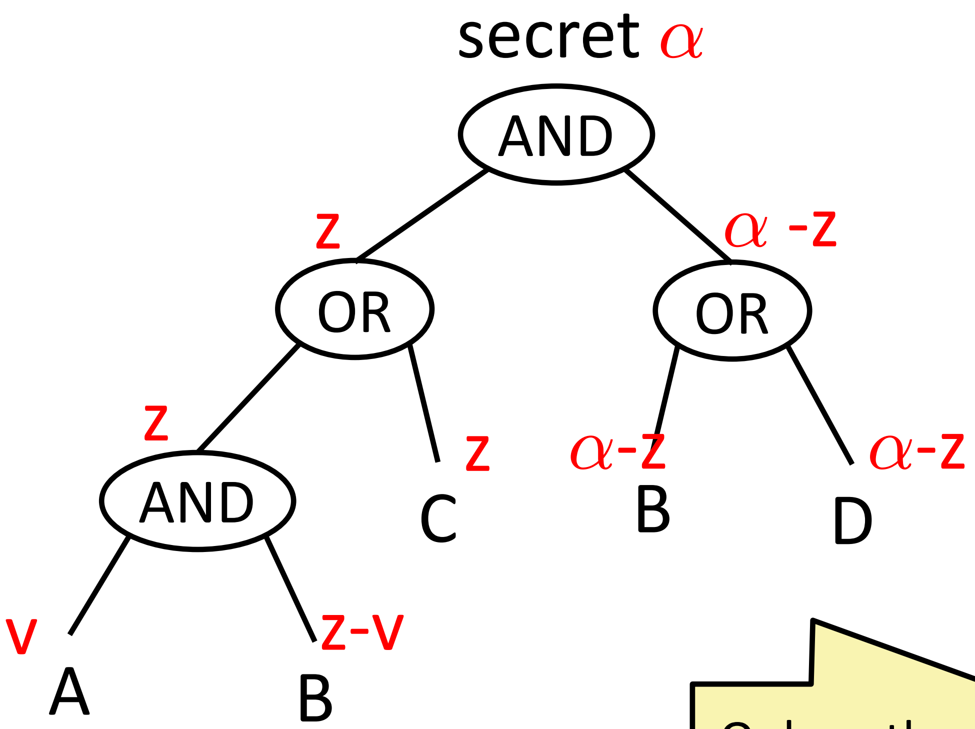
Key-Policy ABE Security Definition

IND-CPA game:



Construction Tools

Linear Secret Sharing:



Only authorized set of shares allows reconstruction of α

Construction Tools

Canceling with independent randomness on two sides:

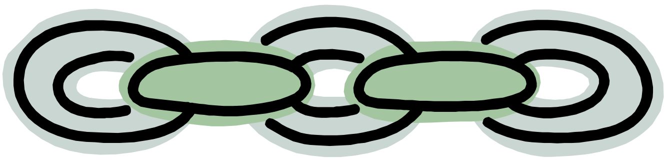
example:

$$K_1 = g^{ar}, \quad K_2 = g^{r}$$

$$C_1 = g^{-s}, \quad C_2 = g^{as}$$

Independent randomness

$$e(C_1, K_1)e(C_2, K_2) = e(g,g)^{-sar}e(g,g)^{sar} = 1$$



links computations together

Basic KP-ABE Construction [GPSW06]

Setup:

bilinear group G of prime order p, generator g
random exponent $\alpha \in \mathbb{Z}_p$,
random elements $H_i \in G$ for each $i \in U$

$$PP := \{g, e(g,g)^{\alpha},\ H_i\ \forall i \in U\} \quad MSK := \alpha$$

KeyGen(f):

split α into shares $\{\lambda_i\}$ following f
choose random $r_i \in \mathbb{Z}_p$

personalized randomness

$$SK = \{g^{\lambda_i} H_i^{r_i},\ \ g^{r_i}\}$$

Encrypt$(M, S \subseteq U)$:

choose random $s \in \mathbb{Z}_p$

$$CT = Me(g,g)^{\alpha s},\ g^s,\ \{H_i^s\}_{i \in S}$$

Stepping through Decryption

Goal: recover M

We have: $Me(g,g)^{\alpha s}$ $\Longrightarrow$ Subgoal: compute $e(g,g)^{\alpha s}$

CT: g^s H_i^s

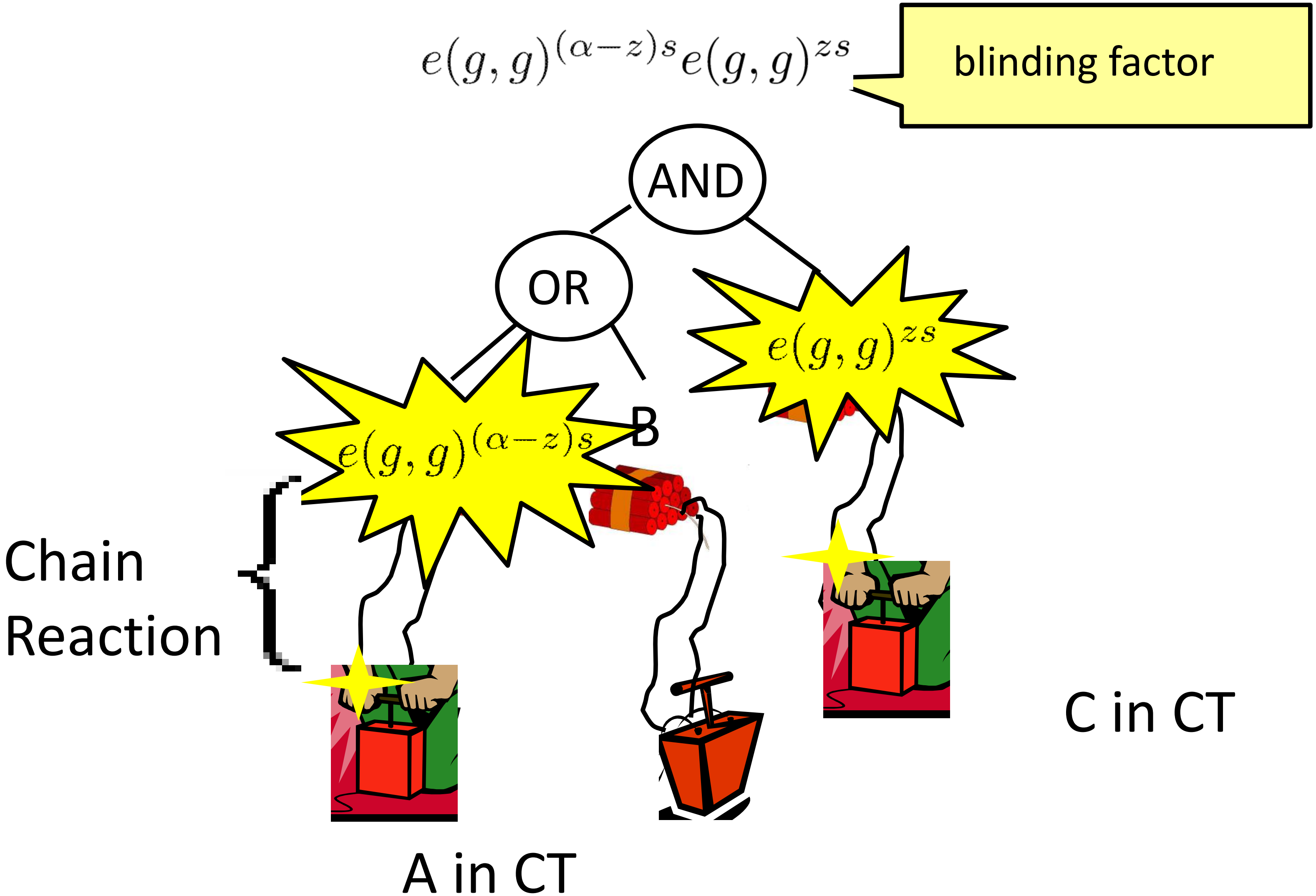
SK: $g^{\lambda_i} H_i^{r_i}$ divide g^{r_i}

$e(g,g)^{\lambda_i s}\boxed{e(g,H_i)^{r_i s}}$ $e(g,H_i)^{r_i s}$

need to cancel

If enough shares,
reconstruct α in the exponent

High Level View of Scheme



Main Features of the Construction

Conceptual Checklist:

1. How are we encoding the access formula?

   with a linear secret sharing scheme

2. How are we tying shares to attributes?

   multiplying by the H_i *elements raised to random exponents*

3. How are we enforcing presence of attributes in the ciphertext?

   the need to cancel the H_i *contributions*

4. How are we preventing collusion?

   injecting personalized randomness during sharing

Proof Challenges

Hard
Problem

Simulator

Attacker

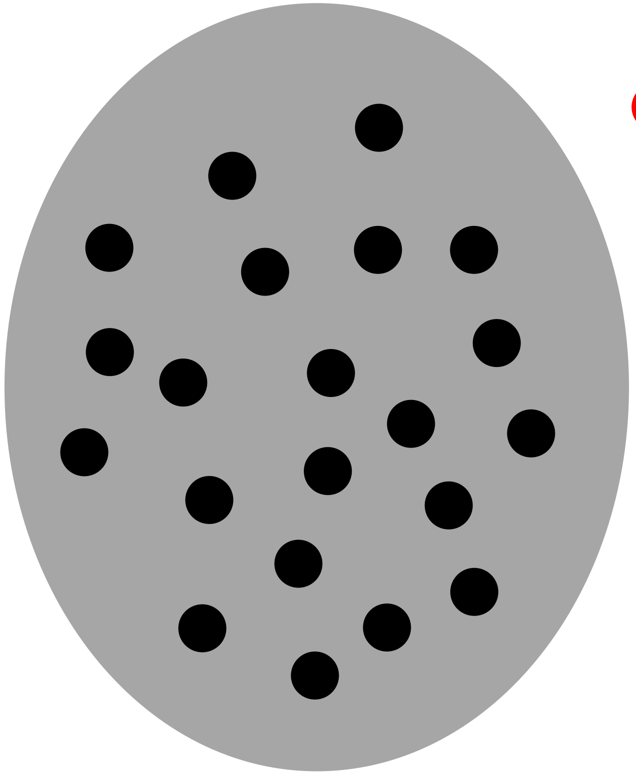
Simulator must balance two competing goals:

answer
attacker
queries

leverage
attacker
success

Partitioning Proofs [BF01,CHK03,BB04,…]

Space of formulas for keys:



= key simulator can make

= key simulator can't make

Problem: Why Should the Attacker Respect the Partition?

Two Approaches:

1. *Make Attacker Commit*
   (weaker) selective security

Too costly for ABE

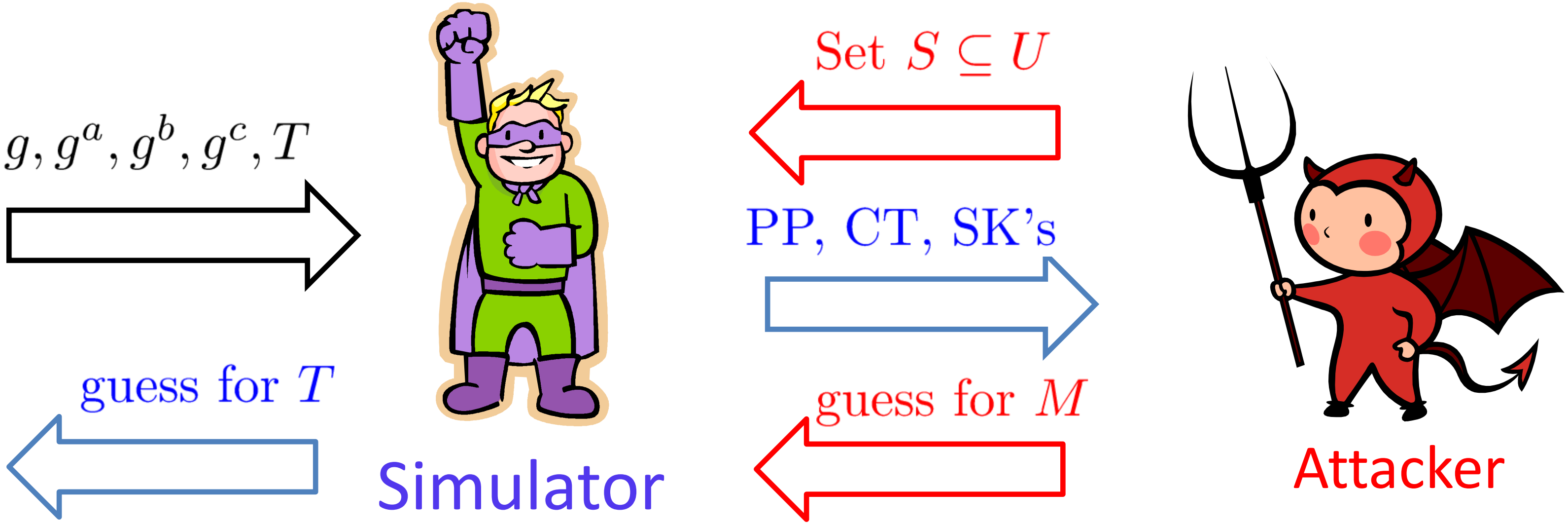
~~2. Guess and quit when wrong~~

Proof of Selective Security [GPSW06]

Assumption (BDH):

Given $g,\ g^a,\ g^b,\ g^c \in G$ and $T \in G_T$,
it is hard to distinguish $T = e(g,g)^{abc}$ from random

Proof Schematic:



How the Simulator Should Work

Input: $g, g^a, g^b, g^c, T = e(g,g)^{abc}?$

Create ciphertext: $\boxed{Me(g,g)^{\alpha s},\ g^s,\ \{H_i^s\}_{i \in S}}$

Start with: MT, ?, ? (requires $\alpha s = abc$)

set $s = c$

need to produce g^s ✓

set $\alpha = ab$

need to produce $e(g,g)^\alpha$ ✓

need to produce g^{λ_i} for keys? ?

Simulating Keys

Goal: simulator must produce *unsatisfied* keys
without knowing g^α

SK components:

shares of α

$$g^{\lambda_i} H_i^{r_i}, \quad g^{r_i}$$

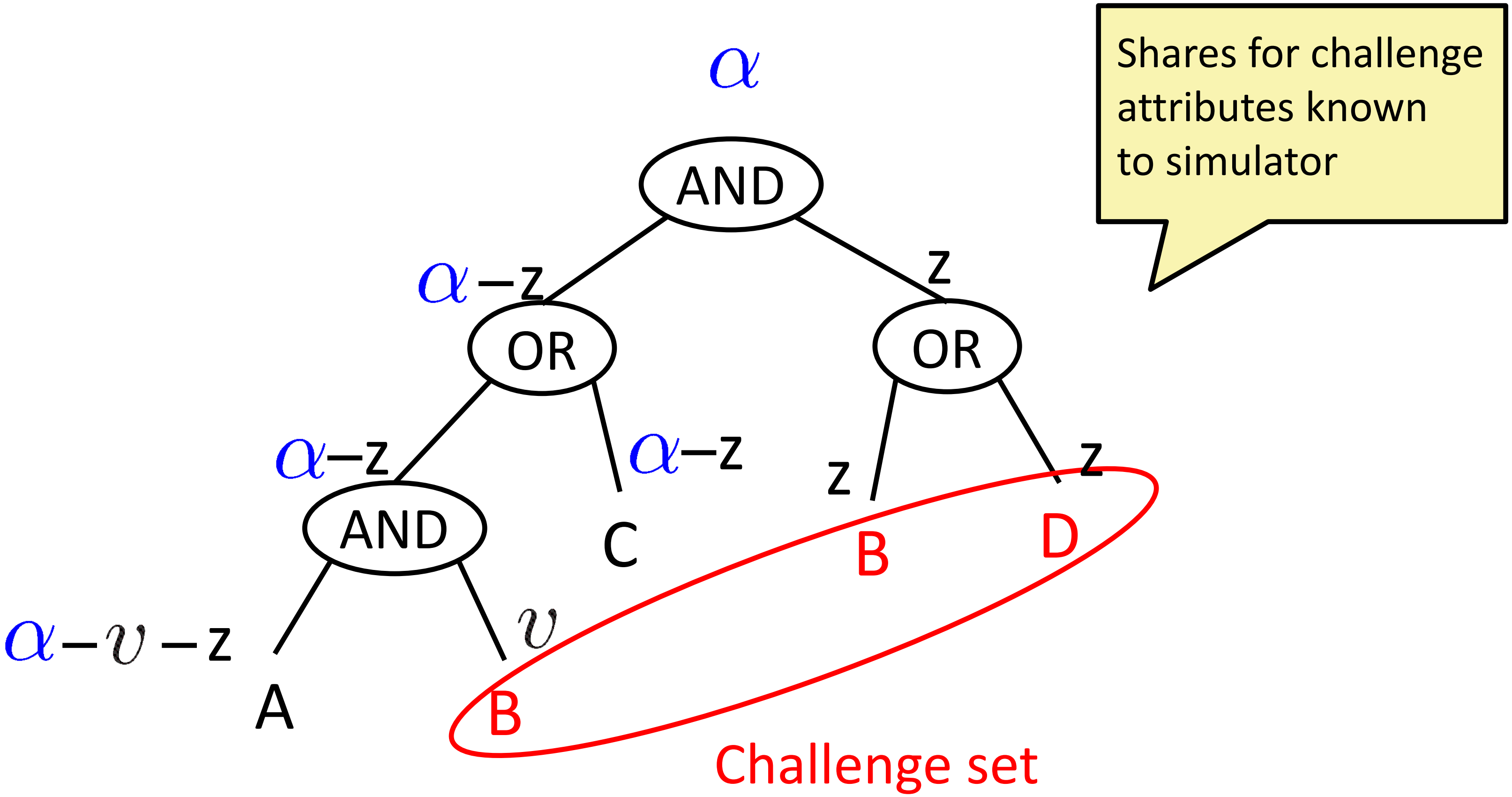
Two observations:

1. Simulator can "place" α in a subset of shares

2. $H_i^{r_i}$ term attached provides cancelation opportunity

Strategic Sharing

Route α "away from" the challenge set:



Cancelations

Recall $\alpha = ab$, suppose $\lambda_i = \alpha - z$

We must produce: $g^{\lambda_i} H_i^{r_i}$, g^{r_i}

Goal: cancel g^{ab} contribution from g^{λ_i}

Consider if $H_i = g^a$, $r_i = -b$

Then $g^{\lambda_i} H_i^{r_i} = g^{ab} g^{-z} g^{-ab} = g^{-z}$!

(Note that we can compute $H_i = g^a$ and $g^{r_i} = (g^b)^{-1}$)

Idea: embed g^a in H_i for all i *not* in the challenge set S
This is enough for canceling α, and we can still produce $H_j^s = H_j^c$ for $j \in S$

Straightline Simulation Recap

Input: $g, g^a, g^b, g^c, T = e(g,g)^{abc}?$

Attacker declares $S \subseteq U$

Simulator sets $\alpha = ab$, picks random $y_i \in \mathbb{Z}_p$ for each $i \in U$

Defines $H_i = g^{y_i}$ for $i \in S$, $H_i = g^{a+y_i}$ for $i \notin S$

$$\mathrm{PP} := \{g,\ e(g,g)^\alpha = e(g^a, g^b),\ H_i = g^{y_i}\ \forall i \in S,\ H_i = g^a g^{y_i}\ \forall i \notin S\}$$

To create ciphertext, simulator sets $s = c$:

$$\mathrm{CT} := MT,\ g^s = g^c,\ \{H_i^s = (g^c)^{y_i}\}_{i \in S}$$

To create key for unsatisfied formula f:

Simulator routes α into shares for $i \notin S$,

uses $r_i = -b + z_i$ to cancel g^{ab} with $H_i^{r_i}$ for these i

possible because f is unsatisfied

shares for $i \in S$ are known

Why did we need selective security?

Simulator should not know g^{α} (if it did, it could decrypt for itself!)
But it needs to make shares of α and cancel out unknown parts

Solution: use some H_i parameters for canceling

But those that are used for canceling can't appear in the ciphertext!
(recall: we set $s = c$ in the ciphertext,
we used g^a inside H_i to cancel, and we do not know g^{ac})

Solution: use some H_i's for canceling, and some for ciphertext

This only works if we know which ones we'll need for the ciphertext,
hence selective security!

Looking Ahead

What we've shown:

- Selectively secure Key-Policy ABE for formulas

What else might we want?

- Full security
- Ciphertext-Policy ABE for formulas
- ABE for circuits
- Hiding policies
- Decentralized authorities

⋮

Ciphertext-Policy ABE Specification

λ= security parameter

U = attribute universe

Setup(λ, U):

generate public parameters PP and master key MSK

KeyGen($S \subseteq U$, MSK):

generate a user key for an attribute set S

Encrypt(PP, M, Policy):

encrypt message M under the given policy

Decrypt(CT, SK):

decrypt ciphertext using a key

Constructing CP-ABE for formulas

Intuition: switch SK and CT from KP-ABE construction

<u>Before:</u>

CT: $Me(g,g)^{\alpha s}, \quad g^s, \quad \{H_i\}^s$

SK: $g^{\lambda_i} H_i^{r_i}, \quad g^{r_i}$

doesn't make sense yet

Adding a Layer of Indirection

PP: $h,\ g,\ e(g,g)^{\alpha},\ \{H_i\}_{i \in U}$

shares of s

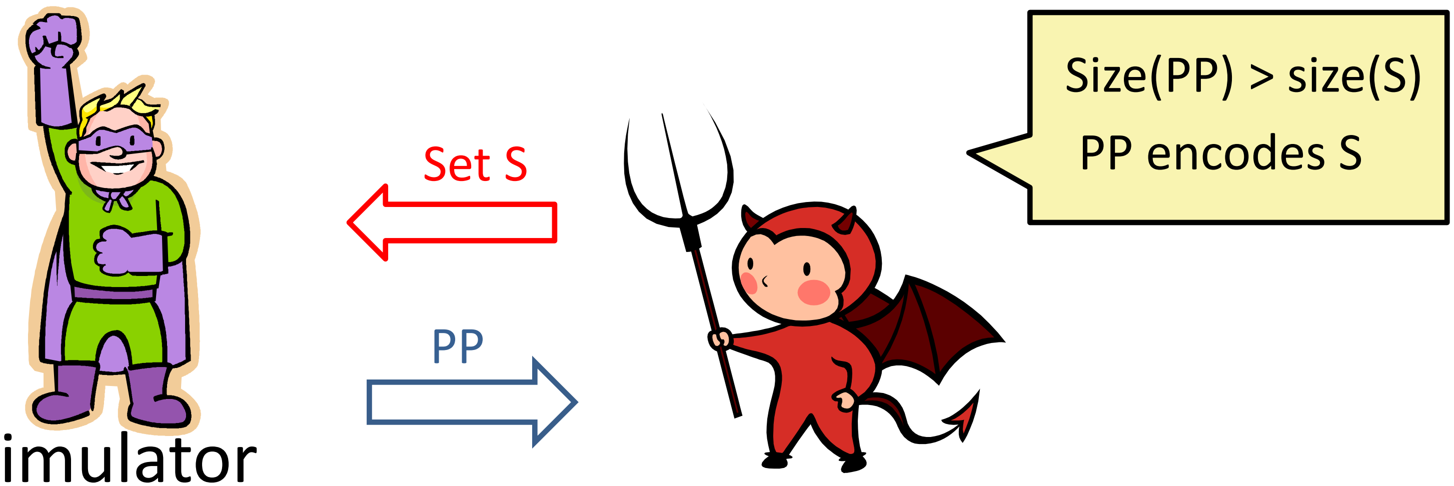
CT: $Me(g,g)^{\alpha s}, \quad g^{s}, \quad h^{\lambda_i} H_i^{r_i}, \quad g^{r_i}$

SK: $g^{\alpha} h^{t}, \quad g^{t}, \quad H_i^{t}$

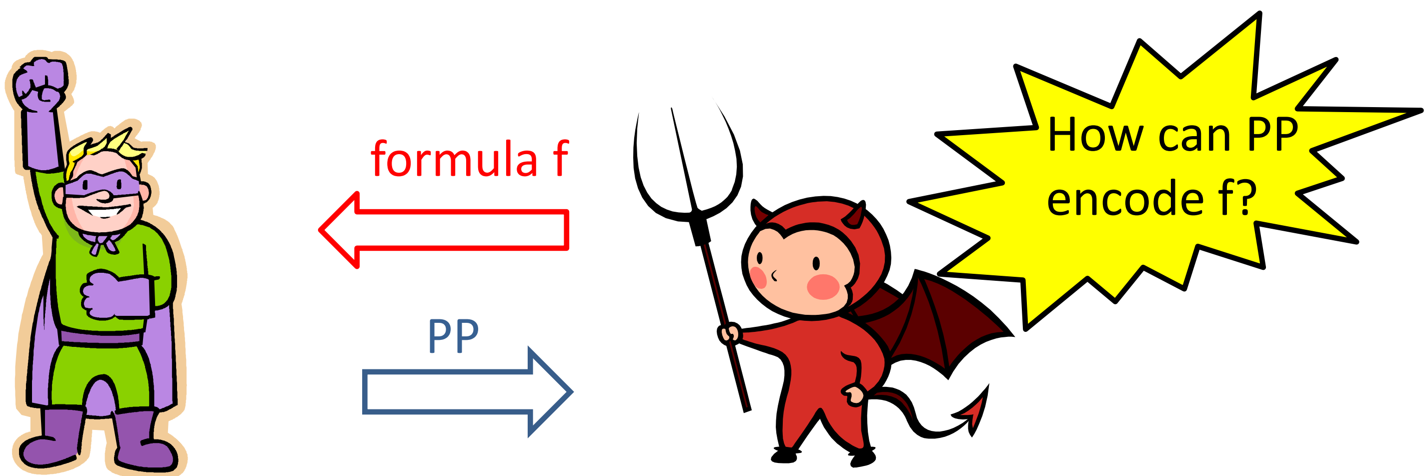
$t =$ personalized randomness

Proving Selective Security?

Selective security for KP-ABE:



Selective security for CP-ABE:



One Approach [e.g. G06, W11]

To fit a big formula into small PP:

Use a big assumption!

Example:
assumption includes terms $g^{a}, g^{a^2}, g^{a^3}, \ldots, g^{a^q}$

single element $g^{\beta_1 a + \beta_2 a^2 + \cdots + \beta_q a^q}$
now encodes a sequence $\beta_1, \beta_2, \ldots, \beta_q$

Moving Beyond Selective Security

Dual System Encryption [W09,LW10]

Main goal:
Simulator prepared to make *any* key, use *any* ciphertext

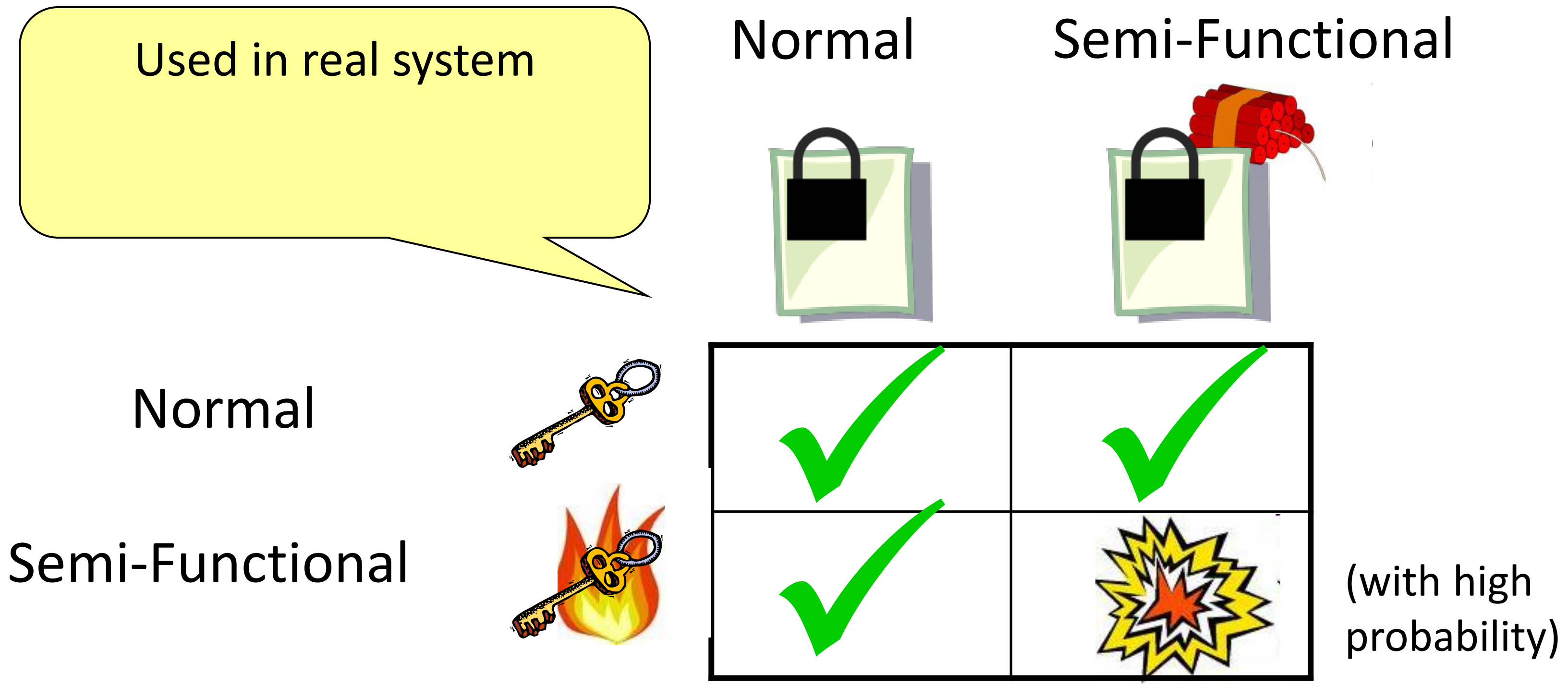
How can this be possible?

Consider **joint** distribution of key and ciphertext

Maybe simulator can sample from alternate distribution and fool attacker on subset of allowable pairs

Dual System Encryption

2 distributions of keys, 2 distributions of ciphertexts:



Overview of a Dual System Encryption Proof

Real Security Game:

Hybrid Argument:

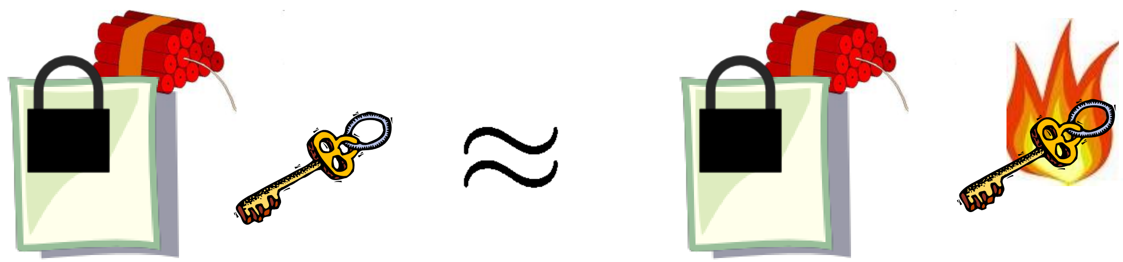
Hardest step!

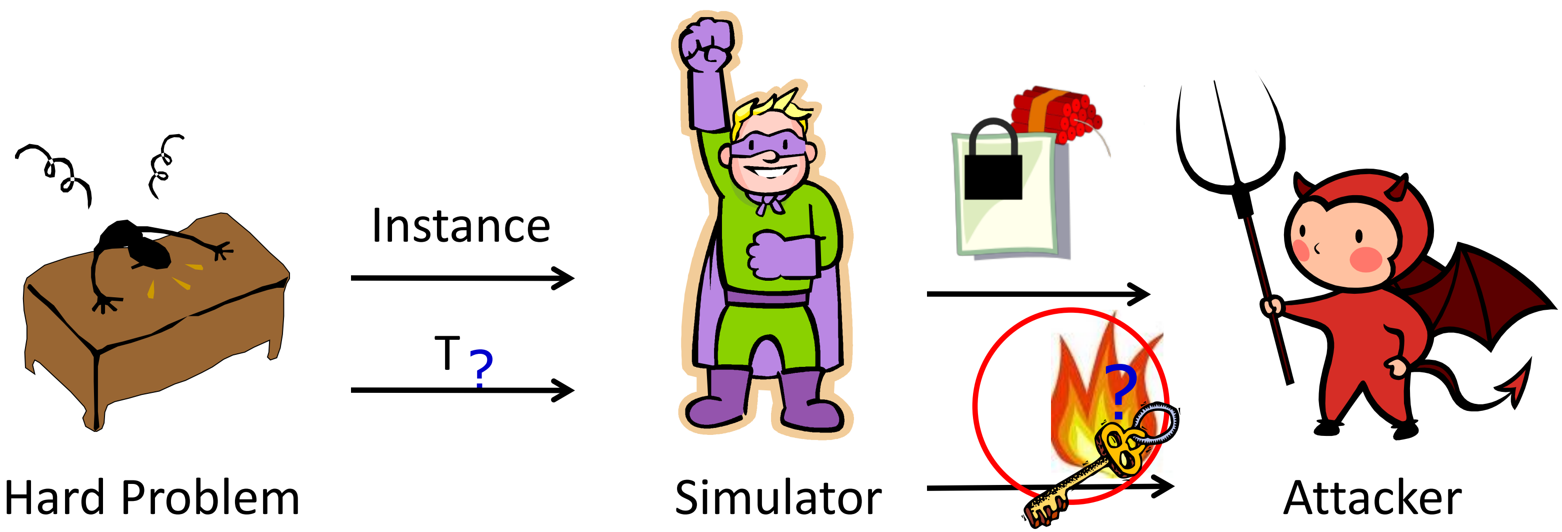
Regardless of Compability!

Incompatiblity of key/CT ⟶ High probability decryption failure

Decryption failure ⟶ Message independent CT

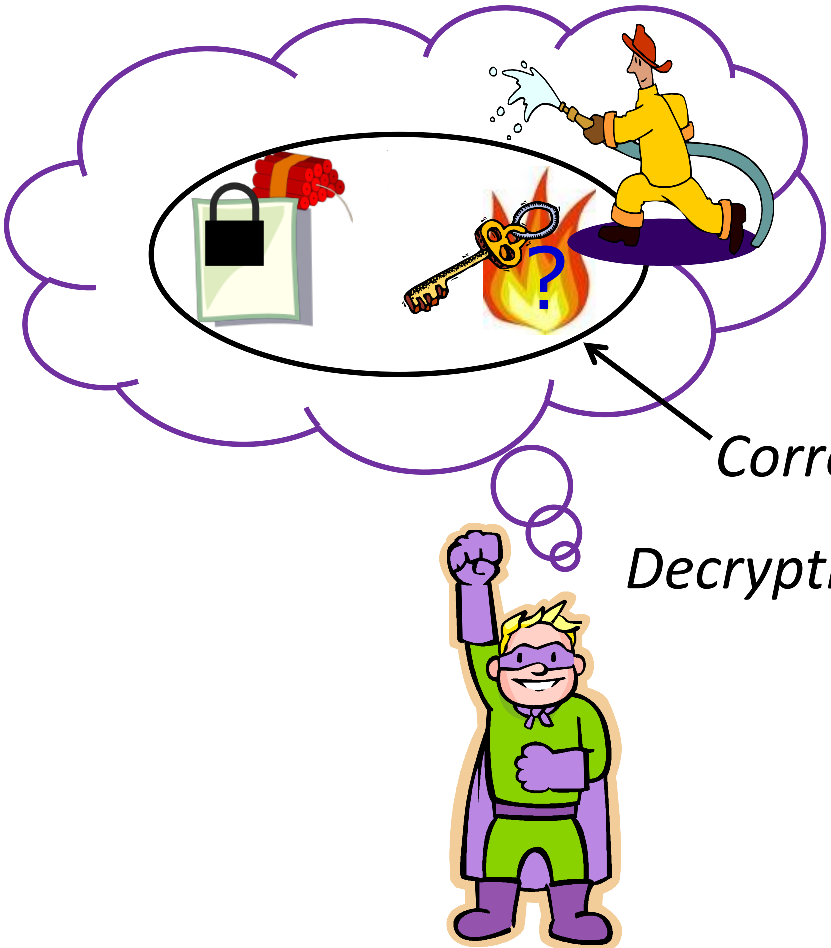
The Critical Step





Simulator cannot know nature of key!

Nominal Semi-Functionality [LW10]



Correlation!

Decryption always succeeds

Simulator

How is Correlation Hidden?

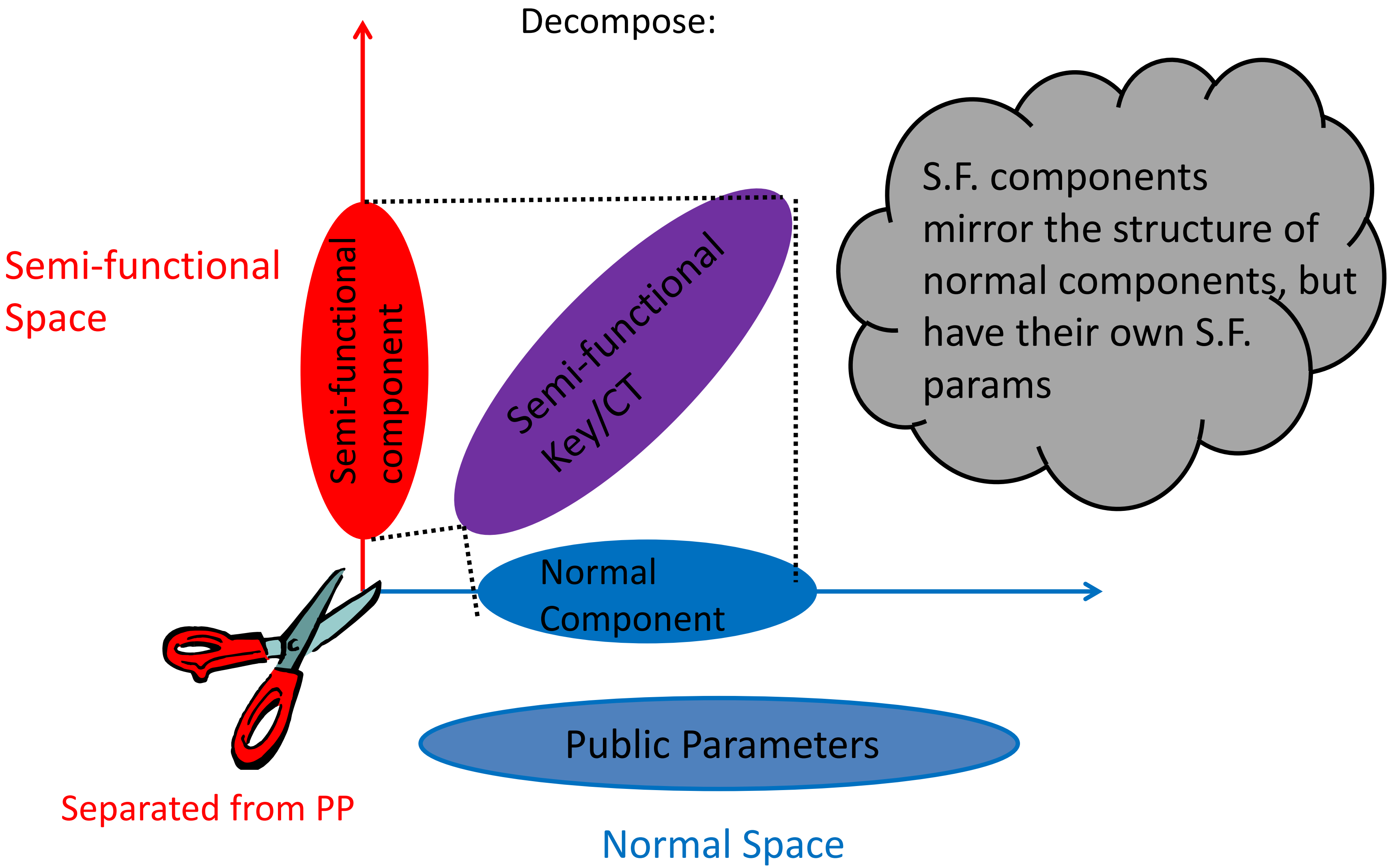
Public Parameters
PP

V|PP - random variable

- has some entropy

Attacker

Conceptualizing Semi-functional Space



Convenient Setting: Composite Order Bilinear Groups

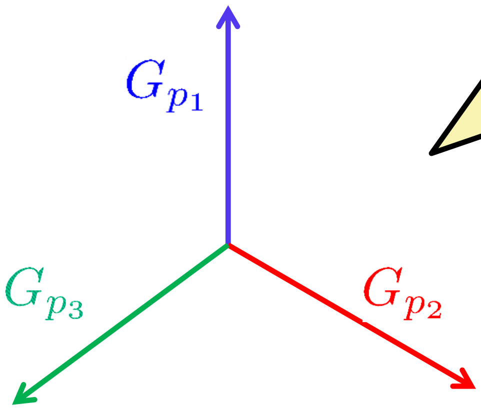
Let G be a bilinear group of order $N = p_1 p_2 p_3$

Let $g \in G_{p_1}$, $g \in G_{p_2}$, $g \in G_{p_3}$ generate its prime order subgroups

An element of G can be written as $g^x g^y g^z$

These subgroups are orthogonal under the bilinear map e:

$$e(g^x g^y g^z, g^u g^v g^w) = e(g,g)^{xu} e(g,g)^{yv} e(g,g)^{zw}$$

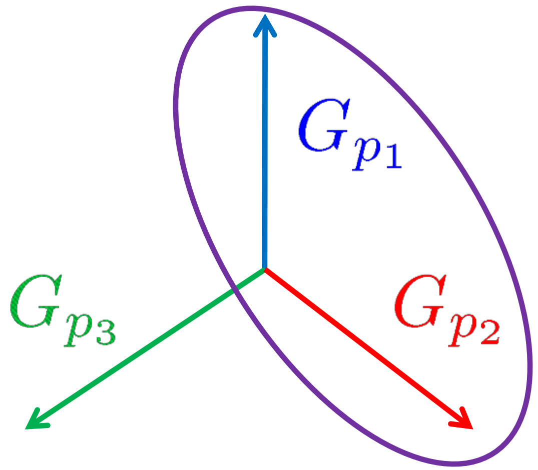


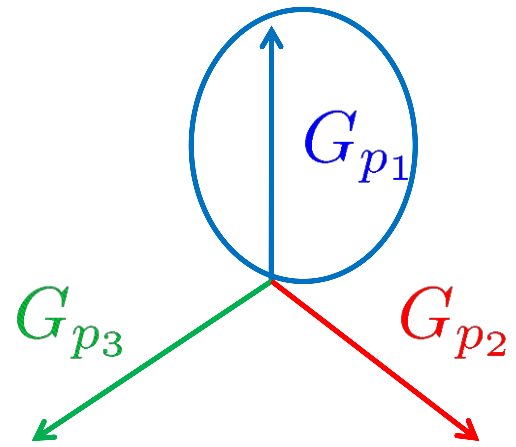
For us,
G_{p_1} = normal space,
G_{p_2} = S.F. space,
G_{p_3} = extra randomness

Computational Assumptions

Subgroup Decision Problems

Example:





$$\text{Hard to distinguish random} \in G_{p_1 p_2} \text{ from random} \in G_{p_1}$$

$$\textit{unless given element} \in G_{p_2}$$

KP-ABE Construction: Dual System Version

(*Slightly simplified)

Bilinear group G of order $N = p_1 p_2 p_3$

prime order orthogonal subgroups $G_{p_1}, G_{p_2}, G_{p_3}$

$$PP = \{g, e(g,g)^\alpha, H_i \ \forall i \in U\}$$

(same as before, just now inside G_{p_1})

Note that $g \neq g$, $H_i \neq H_i$

Normal Ciphertext: choose $s \in \mathbb{Z}_N$

$$Me(g,g)^{\alpha s},\ g^s,\ \{H_i^s\}_{i \in S}$$

S.F. Ciphertext: $Me(g,g)^{\alpha s},\ g^s g^{s'},\ \{H_i^s H_i^{s'}\}$

Normal Key: $\{g^{\lambda_i} H_i^{r_i} V_i,\ g^{r_i} W_i\}$ (random elements in G_{p_3} appended)

S.F. Key: $\{g^{\lambda_i} H_i^{r_i} g^{\omega_i} H_i^{r'_i} V_i,\ g^{r_i} g^{r'_i} W_i\}$

Shares of random

Decrypting S.F. Ciphertext with S.F. Key

CT: $g^{s} g^{s'}$ $\qquad$ $H_i^{s} H_i^{s'}$

divide

SK: $g^{\lambda_i} H_i^{r_i} g^{\omega_i} H_i^{r_i'}$ $\qquad$ $g^{r_i} g^{r_i'}$

$$e(g,g)^{\lambda_i s} e(g,g)^{\lambda_i s} e(g,g)^{\omega_i s'} e(g,H_i)^{r_i' s'} \qquad e(g,H_i)^{r_i s} e(g,H_i)^{r_i' s'}$$

If enough shares,
reconstruct α in the exponent

*random secret in G_{p_2} will ruin result,
unless it happens to be 0!

Executing the Dual System Proof

Step 1: Change CT from normal to S.F.

Subgroup Decision Problem:
Given g, g, distinguish $T \in G_{p_1}$ from $T \in G_{p_1 p_2}$

Simulator chooses $\alpha \in \mathbb{Z}_N$, chooses $y_i \in \mathbb{Z}_N \; \forall i \in U$,

sets $PP = \{g, e(g,g)^{\alpha}, H_i = g^{y_i} \; \forall i \in U\}$

Simulator can easily produce normal keys:
$\{g^{\lambda_i} H_i^{r_i} V_i, g^{r_i} W_i\}$

For CT, simulator uses T for role of g^s:

$$CT = \{Me(g,T)^{\alpha},\; T,\; \{T^{y_i}\}$$

If $T = g^s$, this is $Me(g,g)^{\alpha s}, g^s, \{H_i^s\}$ (normal CT)

If $T = g^s$, this is $Me(g,g)^{\alpha s}, g^s g^{s'}, \{H_i^s H_i^{s'}\}$ (S.F. CT)

Executing the Dual System Proof

Step 2: Change a key from normal to S.F.

Subgroup Decision Problem:
Given g, g, $g^x g^y$, $g^v g^w$, used to make S.F. keys
distinguish $T \in G_{p_1 p_3}$ from $T \in G_{p_1 p_2 p_3}$

Simulator chooses $\alpha \in \mathbb{Z}_N$, chooses $y_i \in \mathbb{Z}_N \ \forall i \in U$,

sets $PP = \{g, e(g,g)^\alpha, H_i = g^{y_i} \ \forall i \in U\}$

To make S.F. ciphertext, simulator uses $s = x$:

$$\mathrm{CT} = Me(g, g^x g^y)^\alpha, \ g^x g^y, \ (g^x g^y)^{y_i} \ \forall i \in S$$

well-distributed b/c
$y_i \ mod \ p_1$ and $y_i \ mod \ p_2$
are uncorrelated

To make key of uncertain type:

ω_i = shares of 0, λ_i = shares of α

$$\mathrm{SK} = \{g^{\lambda_i} T^{\omega_i} T^{y_i r_i} V_i, \ T^{r_i} W_i\}$$

(*note that $\lambda_i + \beta\omega_i$ = shares of α)

The Bait and Switch

But Wait!!!

We were supposed to be sharing something random in S.F. space, not 0!

Suppose T does have g^τ component:

S.F. exponents on key appear as

$$\tau\omega_i + \tau y_i r_i, \; \tau r_i$$

If attribute doesn't appear elsewhere,
this is random modulo p_2!

If enough shares are hidden, the shared value is hidden

Finishing the Dual System Proof

Step 3: Change S.F. CT to encrypt random message

Subgroup Decision Problem:
Given g, g, g, $g^{\alpha}g^{y}$, $g^{s}g^{v}$,
distinguish $T = e(g,g)^{\alpha s}$ from random in G_T

Simulator chooses $y_i \in \mathbb{Z}_N \ \forall i \in U$

$$\mathrm{PP} = \{g, e(g,g)^{\alpha} = e(g, g^{\alpha}g^{y}),\ H_i = g^{y_i}\ \forall i\}$$

To make S.F. keys:

make α shares in exponent with $g^{\alpha}g^{y}$,
fully randomize with g, g raised to random exponents

To make S.F. ciphertext:

$$MT,\ g^{s}g^{v},\ \{(g^{s}g^{v})^{y_i}\}$$

Many Keys and Reusing Parameters

How do we make sure the shared value in the S.F. space is hidden?

- We rely on attributes *not* appearing in the CT
- We use the third subgroup for a hybrid over keys (skipping details)
- We impose limit on reuses of attributes per key

Summary of Dual System Proof

- Semi-functional space “shadows” normal space
- Entropy from unpublished S.F. parameters can hide correlation
- Subgroup decision assumptions can move objects in and out of S.F. space

Dual System in Prime Order Groups [e.g. OT10, L12, LW12]

Use vectors in the exponent:

$$g \in G, \quad \vec{v} \in Z_p^d$$

$$g^{\vec{v}} := (g^{v_1}, g^{v_2}, \ldots, g^{v_d})$$

$$e(g^{\vec{v}}, g^{\vec{w}}) := \prod_{i=1}^{d} e(g^{v_i}, g^{w_i}) = e(g,g)^{\vec{v}\cdot\vec{w}}$$

orthogonality:

$$\vec{v} \cdot \vec{w} \equiv 0 \text{ modulo } p \implies e(g^{\vec{v}}, g^{\vec{w}}) = 1$$

Other Improvements and Extensions

- Multiple authorities [C07, LW11a, OT12a]

  Enables decentralizing functionality and trust

- Large universe constructions [GPSW06, LW11b,OT12b]

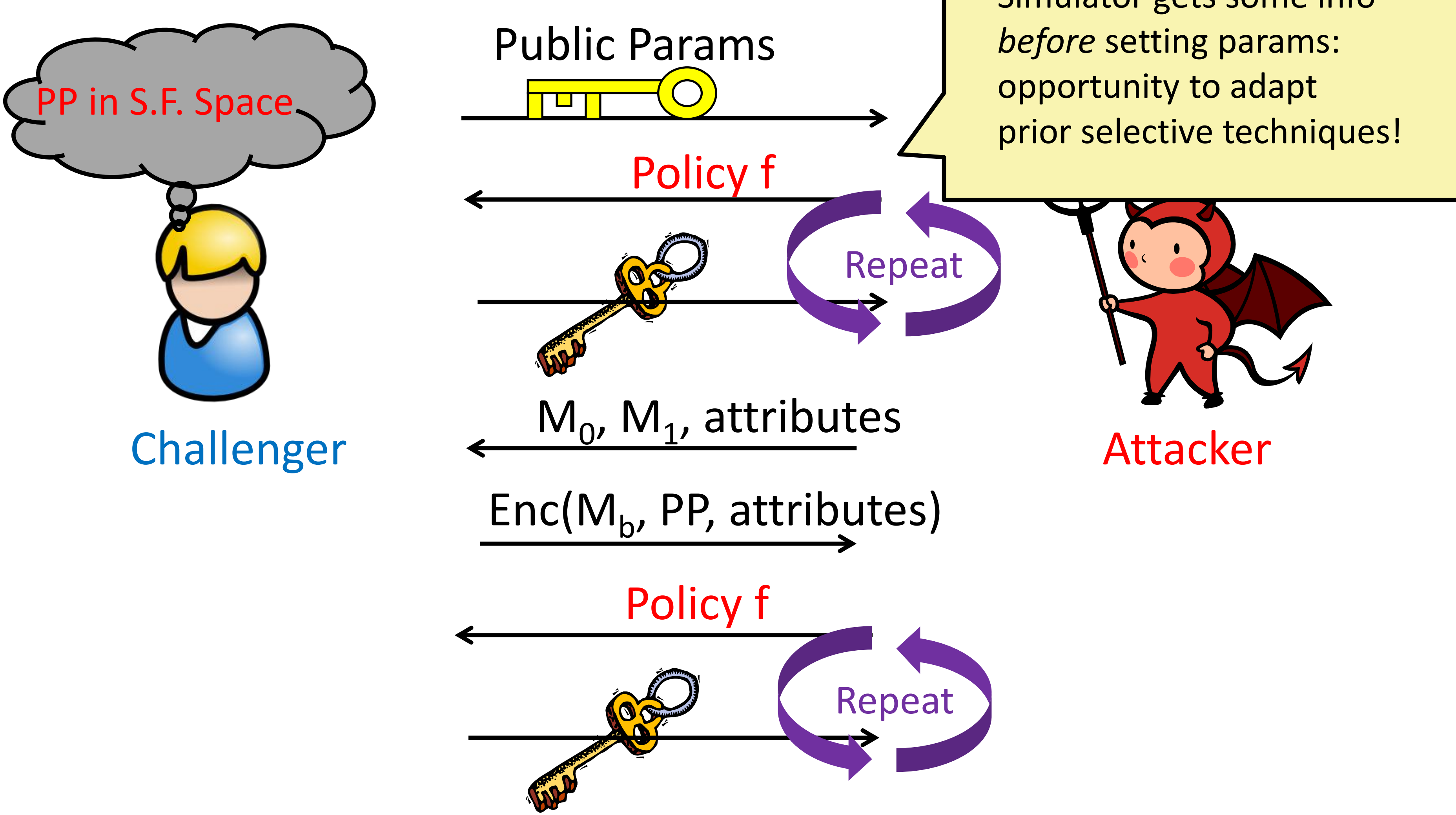
  Attribute universe size is exponential

- Full security without restricting reuse of parameters [LW12]

  New understanding of relationship between selective security techniques and dual system

Alternate View of Dual System Encryption [LW12]

Viewing the security game in S.F. space:



Breaking News Updates

Brought to you by lattice-based cryptographers:

1. Lattice-based KP-ABE scheme for formulas from LWE [B13]

2. Lattice-based scheme for circuits from LWE [GVW13]

3. Multilinear maps [GGH12]

4. ABE for circuits from multi-linear maps [SW12]

ABE for Circuits from Multi-Linear Maps [SW12]

Main idea:

Consider groups $G_1, \ldots, G_k$ and maps
$e_{i,j} : G_i \times G_j \rightarrow G_{i+j}$ for $1 \leq i, j,\ i+j \leq k$

We fix generators $g_1, g_2, \ldots, g_k$

We require:
$e_{i,j}(g_i^a, g_j^b) = g_{i+j}^{ab}$

Key fact: map can only move forward!

intuition: use forward map to move up through circuit one gate a time