Asiacrypt 2016

# Partitioning via Non-Linear Polynomial Functions: More Compact IBEs from Ideal Lattices and Bilinear Maps

Shuichi Katsumata (The University of Tokyo)

Shota Yamada (AIST)

THE UNIVERSITY OF TOKYO

AIST

# ASIACRYPT Born in 1991 (Japan)
# Me Born in 1991 (Japan)

WIKIPEDIA
The Free Encyclopedia

## Asiacrypt

From Wikipedia, the free encyclopedia

**Asiacrypt** (also **ASIACRYPT**) is an important international conference for cryptography research. The full name of the conference is currently **International Conference on the Theory and Application of Cryptology and Information Security**, though this has varied over time. Asiacrypt is a conference sponsored by the International Association for Cryptologic Research (IACR) since 2000, and is one of its three flagship conferences. Asiacrypt is now held annually in November or December at various locations throughout Asia and Australia.

Initially, the Asiacrypt conferences were called **AUSCRYPT**, as the first one was held in Sydney, Australia in 1990, and only later did the community decide that the conference should be held in locations throughout Asia. The first conference to be called "Asiacrypt" was held in 1991 in Fujiyoshida, Japan.

### Conference and proceedings information by year [ edit ]

- 1990: January 8–11, Sydney, Australia, Jennifer Seberry and Josef Pieprzyk, eds. (called AUSCRYPT 1990; ISBN 3-540-53000-2)
- 1991: November 11–14, Fujiyoshida, Japan, Hideki Imai, Ronald Rivest, Tsutomu Matsumoto, eds. (ISBN 3-540-57332-1)
- 1992: December 13–16, Gold Coast, Queensland, Australia, Jennifer Seberry and Yuliang Zheng, eds. (called AUSCRYPT 1992; ISBN 3-540-57220-1)

# Background

Adaptively secure identity-based encryption

- From Lattices

  Adaptively secure lattice IBE requires long public parameters compared to selectively secure ones.

- From Bilinear Maps

  Adaptively secure bilinear map-based IBE under search problems require long public parameters.

Topic of This Talk

⇒ Can we achieve more compact IBEs??

# Our Results:
# New Adaptively Secure IBEs

- Both based on partitioning technique with non-linear functions
- New IBE from ideal lattices:
  - Improve currently best scheme of [Yam16]: super-poly modulus → poly modulus RLWE
  - Use commutativity of Ring in an essential way
- New IBE from bilinear maps:
  - First scheme with sub-linear-size mpk from search problem rather than decisional problem
  - Boneh-Boyen technique in the construction rather than in the security proof

# Agenda

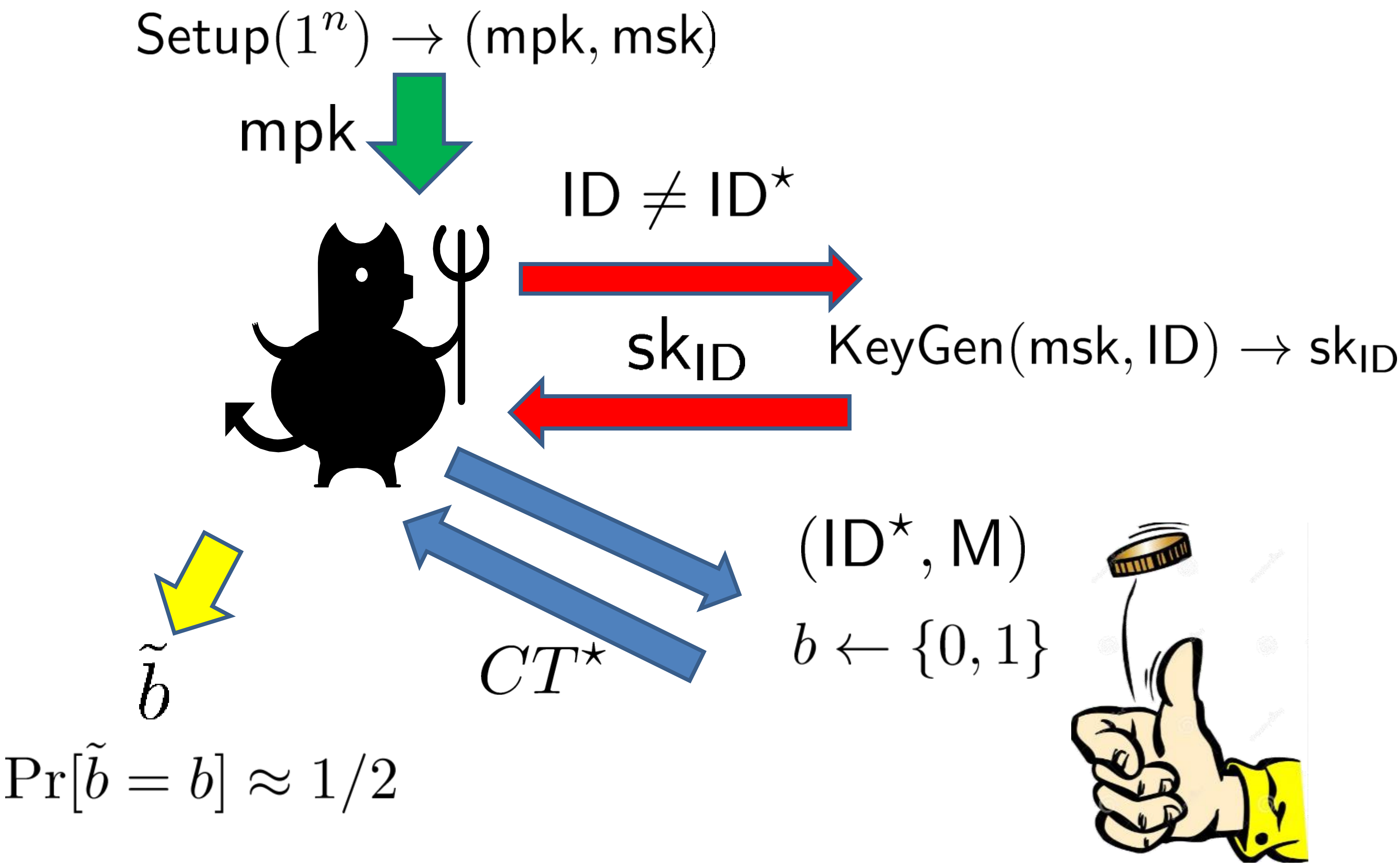
I. Preliminaries

II. Lattice Section
   - ✓ Previous Works
   - ✓ Our Work

III. Bilinear Map Section
   - ✓ Previous Works
   - ✓ Our Work

IV. Summary

# Adaptive Security for IBE



# Agenda

I. Preliminaries

II. Lattice Section
- ✓ Previous Works
- ✓ Our Work

III. Bilinear Map Section
- ✓ Previous Works
- ✓ Our Work

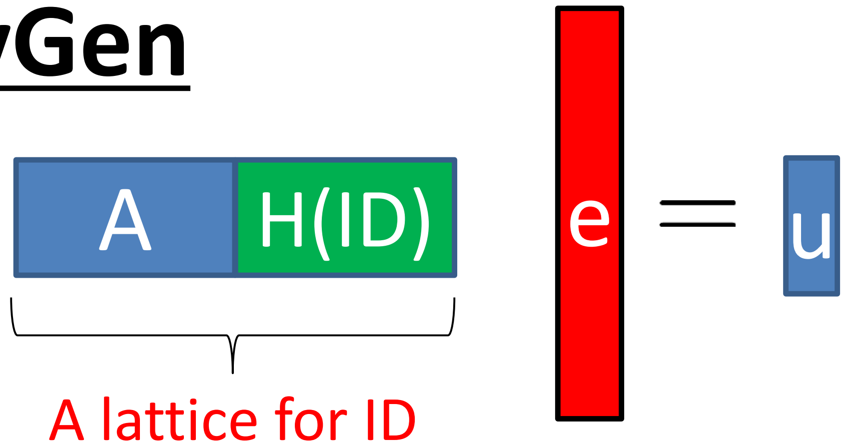
IV. Summary

# Template Construction (1)

$$\mathsf{mpk} = \{ A, u, \ldots \}$$

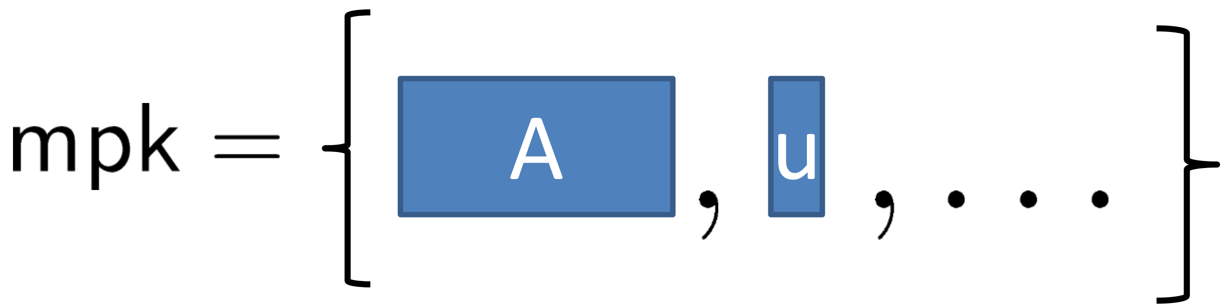
---

**KeyGen**

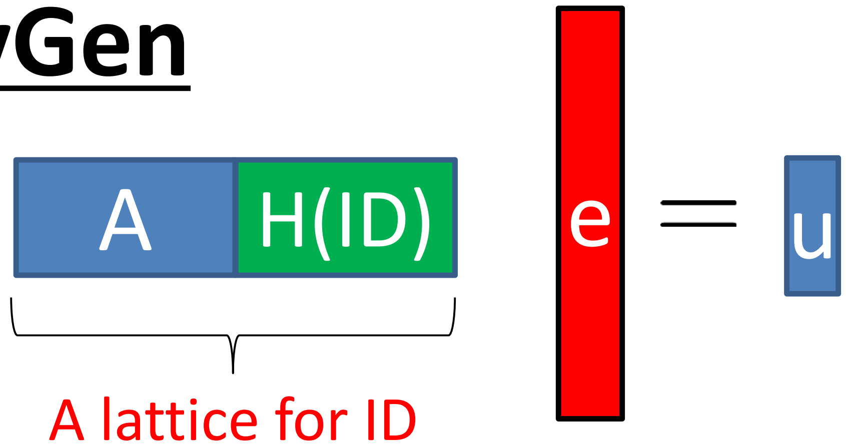



Secret key for ID:
short vector e

# Template Construction



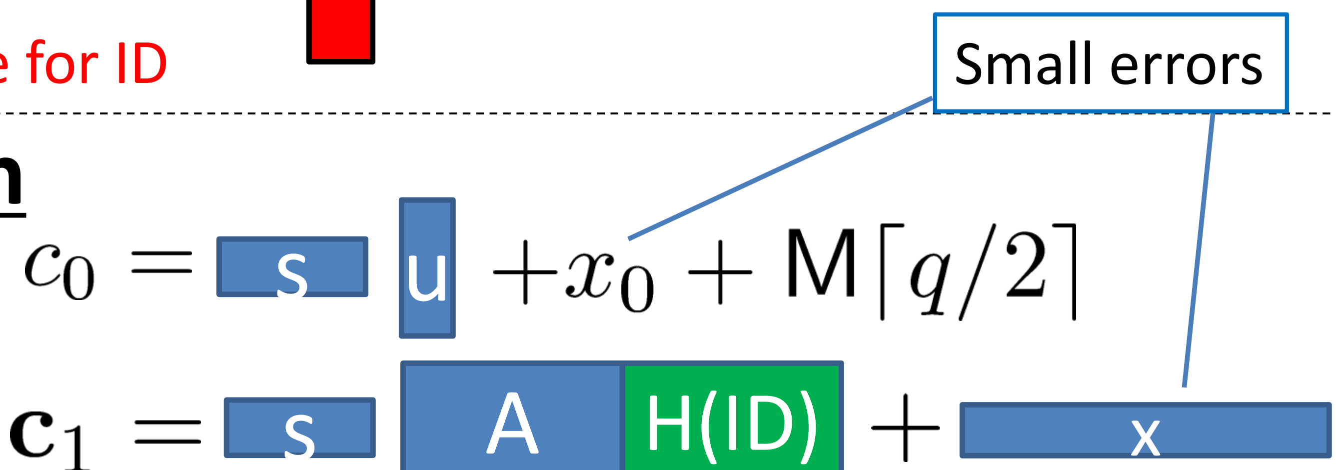

## KeyGen




Secret key for ID: short vector e

## Encryption

$\mathsf{M} \in \{0, 1\}$



# Template for Security Proof

**Partitioning Technique**

We embed the problem instance into the public parameters so that

Publicly Computable

$$H(ID) = A \, R_{ID} + F(ID) \, G$$

In the simulation, We hope

$$F(ID_i) \neq 0 \text{ for queried } ID_i$$
$$F(ID^\star) = 0 \text{ for challenge } ID^\star$$

# Template for Security Proof

## Partitioning Technique

We embed the problem instance into the public parameters so that

$$H(ID) = A \, R_{ID} + F(ID) \, G$$

- $H(ID)$: Publicly Computable
- $R_{ID}$: Simulator's Trapdoor (Needs to be **"small"**)
- $G$: Gadget matrix
- Only Known to Simulator

In the simulation, We hope

$$F(ID_i) \neq 0 \text{ for queried } ID_i$$

$$F(ID^\star) = 0 \text{ for challenge } ID^\star$$

# Hashing the Identities

**Ex. [ABB10]+[Boy10]**

$$\mathsf{mpk} = (\mathbf{A}, \mathbf{u}, \boxed{\mathbf{B}_0, \mathbf{B}_1, \ldots, \mathbf{B}_\kappa})$$

$\kappa$: ID Length

$$H(ID) = B_0 + \sum_{i \in S(ID)} B_i$$

Example) ID Length $\kappa$ = 6

| 0 | 1 | 0 | 0 | 1 | 1 |
|---|---|---|---|---|---|
| $B_1$ | $B_2$ | $B_3$ | $B_4$ | $B_5$ | $B_6$ |

ID=010011

S(ID)={2, 5, 6}

# Hashing the Identities

**Ex. [ABB10]+[Boy10]**

$$\text{mpk} = (\mathbf{A}, \mathbf{u}, \mathbf{B}_0, \mathbf{B}_1, \ldots, \mathbf{B}_\kappa)$$

$\kappa$: ID Length

$$H(ID) = B_0 + \sum_{i \in S(ID)} B_i$$

In Simulation

Set

$$B_i = A \, R_i + y_i \, G$$

Then

$$H(ID) = A \, R_{ID} + \Big( y_0 + \sum_{i \in S(ID)} y_i \Big) G$$

$y_0 + \sum_{i \in S(ID)} y_i \rightarrow F(ID)$

# Hashing the Identities

**Ex. [ABB10]+[Boy10]**

$$\text{mpk} = (\mathbf{A}, \mathbf{u}, \boxed{\mathbf{B}_0, \mathbf{B}_1, \ldots, \mathbf{B}_\kappa})$$

$\kappa$: ID Length

H(ID) = B

**Long public key!**

#matrices **linear** in ID length

In Simulation

Set

$$B_i = A \cdot R_i + y_i \, G$$

Then

**F(ID): Linear Function**

$$H(ID) = A \cdot R_{ID} + \left(y_0 + \sum_{i \in S(ID)} y_i\right) G$$

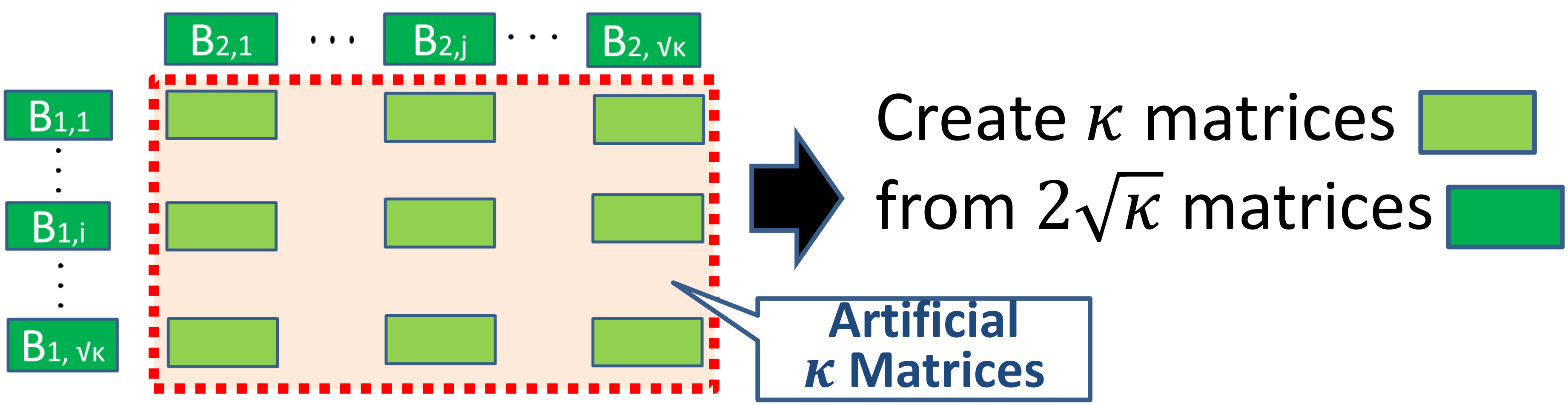
F(ID)

# Hashing the Identities

**Ex. [Yam16]** (Currently, the most (asymptotically) compact lattice-based IBE)

$$\mathsf{mpk} = \left(\mathbf{A}, \mathbf{u}, \boxed{\mathbf{B}_0 \quad \begin{matrix}\mathbf{B}_{1,1}, \cdots, \mathbf{B}_{1,\sqrt{\kappa}} \\ \mathbf{B}_{2,1}, \cdots, \mathbf{B}_{2,\sqrt{\kappa}}\end{matrix}}\right)$$

$$\mathrm{H(ID)} = \mathrm{B}_0 + \sum_{(i,j)\in S(ID)} \mathrm{B}_{1,i}\, \mathrm{G}^{-1}(\mathrm{B}_{2,j})$$



# Hashing the Identities

**Ex. [Yam16]** (Currently, the most (asymptotically) compact lattice-based IBE)

$$\mathsf{mpk} = \left(\mathbf{A}, \mathbf{u}, \boxed{\mathbf{B}_0 \quad \begin{matrix} \mathbf{B}_{1,1}, \cdots, \mathbf{B}_{1,\sqrt{\kappa}} \\ \mathbf{B}_{2,1}, \cdots, \mathbf{B}_{2,\sqrt{\kappa}} \end{matrix}}\right)$$

$$\mathrm{H(ID)} = \mathrm{B_0} + \sum_{(i,j)\in S(ID)} \mathrm{B}_{1,i}\, \mathrm{G}^{-1}\left(\mathrm{B}_{2,j}\right)$$

In Simulation

Set

$$\mathrm{B}_{i,j} = \mathrm{A}\ \mathrm{R}_{i,j} + y_{i,j}\ \mathrm{G}$$

Then

$$\mathrm{H(ID)} = \mathrm{A}\ \mathrm{R}_{\mathrm{ID}} + \underbrace{\left(y_0 + \sum_{i\in S(ID)} y_{1,i} y_{2,j}\right)}_{\mathrm{F(ID)}} \mathrm{G}$$

# Hashing the Identities

**Ex. [Yam16]** (Currently, the most (asymptotically) compact lattice-based IBE)

$$\mathsf{mpk} = \left(\mathbf{A}, \mathbf{u}, \mathbf{B}_0 \quad \begin{matrix} \mathbf{B}_{1,1}, \cdots, \mathbf{B}_{1,\sqrt{\kappa}} \\ \mathbf{B}_{2,1}, \cdots, \mathbf{B}_{2,\sqrt{\kappa}} \end{matrix} \right)$$

H(ID) $=$ B0

**Shorter public key!**

#matrices **sqrt** in ID length

In Simulation

Set

$+ y_{i,j}$ G

F(ID): **Non-Linear Function**

Th

F(ID)

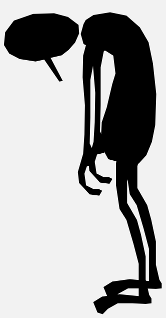
H(ID) $=$ A RID $+ \; y_0 + \sum_{i \in S(ID)} y_{1,i} y_{2,j}$ G

# Hashing the Identities

**Ex. [Yam16]** (Currently, the most (asymptotically) compact lattice-based IBE)

## ⚠ Downside

**For the scheme to be secure, the modulus size $q$ must be super-poly**



$$\mathrm{H(ID)} = \mathbf{A} \quad \mathbf{R}_{\mathrm{ID}} + y_0 + \sum_{i \in S(ID)} y_{1,i} y_{2,j} \quad \mathbf{G}$$

# Agenda

I. Preliminaries

II. Lattice Section
  - ✓ Previous Works
  - ✓ Our Work

III. Bilinear Map Section
  - ✓ Previous Works
  - ✓ Our Work

IV. Summary

# A Closer Look at [Yam16]

In Simulation

$$\mathbf{B}_0 = \mathbf{A}\mathbf{R}_0 + y_0\mathbf{G}\ , \ \ \mathbf{B}_{i,j} = \mathbf{A}\mathbf{R}_{i,j} + y_{i,j}\mathbf{G}$$

$$\mathsf{H(ID)} = \mathsf{A}\ \mathsf{R_{ID}} + \mathsf{F(ID)}\ \mathsf{G}$$

$$\Big(\mathbf{R}_0 + \sum_{(i,j)\in S(\mathsf{ID})} \mathbf{R}_{1,i}\mathbf{G}^{-1}(\mathbf{B}_{2,j}) + y_{1,i}\mathbf{R}_{2,j}\Big)$$

$$y_0 + \sum_{(i,j)\in S(\mathsf{ID})} y_{1,i}y_{2,j}$$

Several conditions on $\mathbf{R}_{\mathrm{ID}}$ and $y_{i,j}$'s must hold for the security proof to hold.

# Main Obstacle of [Yam16]

$$\mathrm{F(ID)} = y_0 + \sum y_{1,i} y_{2,j}$$

$$\mathbf{R}_{\mathrm{ID}} = (\mathbf{R}_0 + \sum \mathbf{R}_{1,i}\mathbf{G}^{-1}(\mathbf{B}_{2,j}) + y_{1,i}\mathbf{R}_{2,j})$$

- For the simulation to succeed $y_{1,j}$ must grow proportionally with Q (#query).

# Main Obstacle of [Yam16]

$$\mathrm{F(ID)} = \boxed{y_0} + \sum \boxed{y_{1,i} y_{2,j}}$$

$$\mathbf{R}_{\mathrm{ID}} = (\mathbf{R}_0 + \sum \mathbf{R}_{1,i}\mathbf{G}^{-1}(\mathbf{B}_{2,j}) + \boxed{y_{1,i}\mathbf{R}_{2,j}})$$

Simulator's "small" Trapdoor

- For the simulation to succeed $y_{1,j}$ must grow proportionally with Q (#query).
- For the trapdoor $\mathbf{R}_{\mathrm{ID}}$ to work, $y_{1,i}$ must be small compared with q (modulus size).

# Main Obstacle of [Yam16]

$$\mathsf{F(ID)} = \boxed{y_0} + \sum \boxed{y_{1,i} y_{2,j}}$$

$$\mathsf{R_{ID}} = (\mathbf{R}_0 + \sum \mathbf{R}_{1,i}\mathbf{G}^{-1}(\mathbf{B}_{2,j}) + \boxed{y_{1,i}\mathbf{R}_{2,j}})$$

- For the simulation to succeed $y_{1,j}$ must grow proportionally with Q (#query).
- For the trapdoor $\mathbf{R}_{\mathrm{ID}}$ to work, $y_{1,i}$ must be small compared with q (modulus size).

**∀Q :poly(n) < y < q ⇒ q needs to be super-poly(n)!!**

# Initial Idea (that doesn't quite work)

Extend the definition of $y_{i,j} \in \mathbb{Z}_q$ to $\mathbf{Y}_{1,j} \in \mathbb{Z}_q^{n\times n}$

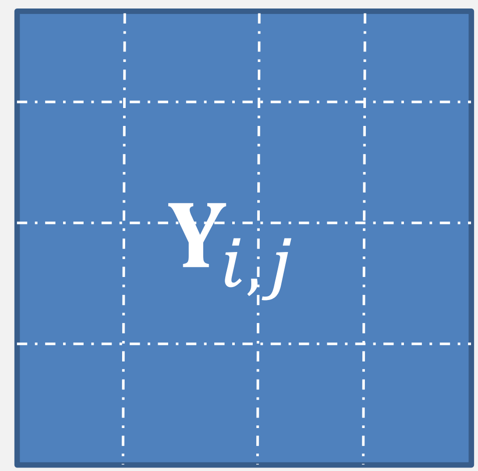
$$\mathbf{B}_{i,j} = \mathbf{A}\mathbf{R}_{i,j} + \underline{y_{i,j}\mathbf{G}} \Rightarrow \mathbf{B}_{i,j} = \mathbf{A}\mathbf{R}_{i,j} + \underline{\mathbf{Y}_{i,j}\mathbf{G}}$$

Before


"pack" Q in one entry

- ➢ $y_{i,j}$ needs to be big.
  => Big modulus q

After




"pack" Q in $n^2$ entries

- ➢ Each entry of $\mathbf{Y}_{i,j}$ can be small. => Small modulus q

# Why it doesn't work

We can't compute the hash homomorphically!!

Since we loose commutativity of $\mathbf{A}$ and $\mathbf{Y}_{i,j}$ .

Let $\mathbf{B} = \mathbf{AR} + \mathbf{YG}, \quad \mathbf{B}' = \mathbf{AR}' + \mathbf{Y}'\mathbf{G}$

# Why it doesn't work

We can't compute the hash homomorphically!!

Since we loose commutativity of $\mathbf{A}$ and $\mathbf{Y}_{i,j}$ .

Let $\mathbf{B} = \mathbf{AR} + \mathbf{YG}, \quad \mathbf{B}' = \mathbf{AR}' + \mathbf{Y}'\mathbf{G}$

$$\mathbf{B} \cdot \mathbf{G}^{-1}(\mathbf{B}') = (\mathbf{AR} + \mathbf{YG}) \cdot \mathbf{G}^{-1}(\mathbf{B}')$$

# Why it doesn't work

We can't compute the hash homomorphically!!

Since <u>we loose commutativity of $\mathbf{A}$ and $\mathbf{Y}_{i,j}$ .</u>

Let $\mathbf{B} = \mathbf{AR} + \mathbf{YG}, \quad \mathbf{B}' = \mathbf{AR}' + \mathbf{Y}'\mathbf{G}$

$$\begin{aligned} \mathbf{B} \cdot \mathbf{G}^{-1}(\mathbf{B}') &= (\mathbf{AR} + \mathbf{YG}) \cdot \mathbf{G}^{-1}(\mathbf{B}') \\ &= \mathbf{AR} \cdot \mathbf{G}^{-1}(\mathbf{B}') + \mathbf{Y}(\mathbf{AR}' + \mathbf{Y}'\mathbf{G}) \end{aligned}$$

# Why it doesn't work

We can't compute the hash homomorphically!!

Since we loose commutativity of $\mathbf{A}$ and $\mathbf{Y}_{i,j}$ .

Let $\mathbf{B} = \mathbf{AR} + \mathbf{YG}, \quad \mathbf{B}' = \mathbf{AR}' + \mathbf{Y}'\mathbf{G}$

$$\begin{aligned}\mathbf{B} \cdot \mathbf{G}^{-1}(\mathbf{B}') &= (\mathbf{AR} + \mathbf{YG}) \cdot \mathbf{G}^{-1}(\mathbf{B}') \\ &= \mathbf{AR} \cdot \mathbf{G}^{-1}(\mathbf{B}') + \mathbf{Y}(\mathbf{AR}' + \mathbf{Y}'\mathbf{G}) \\ &= \underbrace{\mathbf{AR} \cdot \mathbf{G}^{-1}(\mathbf{B}')}_{\text{GOOD!!}} + \underbrace{\mathbf{YAR}'}_{\text{BAD!!}} + \underbrace{\mathbf{YY}'\mathbf{G}}_{\text{GOOD!!}}\end{aligned}$$

❌ Can't obtain H(ID) = $\mathbf{AR}_{\mathrm{ID}} + \mathrm{F}(\mathrm{ID})\mathbf{G}$

In general, $\mathbf{YAR}' \neq \mathbf{AYR}'$

# Idea (that works)

Move to the **polynomial ring** setting.

View elements of $\mathbb{Z}_q^n$ (or a subring of $\mathbb{Z}_q^{n\times n}$) as the polynomial ring $R_q = \mathbb{Z}_q[X]/(X^n + 1)$.

$$\mathbb{Z}_q^n \ni \begin{bmatrix} a_0 \\ \vdots \\ a_{n-1} \end{bmatrix} \Longleftrightarrow \sum_{i=0}^{n-1} a_i X^i \in R_q$$

# Idea (that works)

Move to the **polynomial ring** setting.

View elements of $\mathbb{Z}_q^n$ (or a subring of $\mathbb{Z}_q^{n\times n}$) as the polynomial ring $R_q = \mathbb{Z}_q[X]/(X^n + 1)$.

$$\mathbb{Z}_q^n \ni \begin{bmatrix} a_0 \\ \vdots \\ a_{n-1} \end{bmatrix} \Longleftrightarrow \sum_{i=0}^{n-1} a_i X^i \in R_q$$

Then,

$$\mathbf{B} = \mathbf{AR} + \mathrm{y}\mathbf{G} \quad \Longrightarrow \quad \boldsymbol{b} = \boldsymbol{aR} + y\boldsymbol{g}, \text{ where}$$

$\mathrm{y} \in \mathbb{Z}_q$

$\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{g} \in R_q^k, \boldsymbol{R} \in R_q^{k\times k}$,

$y \in R_q$

# Why it works

$$\boldsymbol{b} = \boldsymbol{a}\boldsymbol{R} + y\boldsymbol{g}$$

※$\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{g} \in R_q^k$,
$\boldsymbol{R} \in R_q^{k\times k}$, $y \in R_q$

- When $y_{i,j} \in R_q$, we get commutativity with $\boldsymbol{a} \in R_q^k$ for free.
- Since $y_{i,j} \in R_q$ can be viewed as vectors in $\mathbb{Z}_q^n$, we can "pack" Q in n entries, which allows us to use poly-sized modulus q.

# Some Ignored Problems

- $R_q$ is no longer a field, so even when $\boldsymbol{a}\boldsymbol{R}_{ID} + \mathrm{F}_y(\mathrm{ID})\boldsymbol{g}$ for $\mathrm{F}_y(\mathrm{ID}) \neq 0$, the trapdoor may not be useful in case $R_q$ is not invertible.
- In Yam16, the “smudging” technique was used to create the challenge ciphertext, however, this necessarily leads to super-poly modulus q.

# Agenda

I. Preliminaries

II. Lattice Section
- ✓ Previous Works
- ✓ Our Work

III. Bilinear Map Section
- ✓ Previous Works
- ✓ Our Work

IV. Summary

# IBE from Search Problems on Bilinear Maps

- Dual system encryption methodology inherently requires decisional problem. (SXDH, DLIN, Matrix-DDH,…)

# IBE from Search Problems on Bilinear Maps

- Dual system encryption methodology inherently requires decisional problem. (SXDH, DLIN, Matrix-DDH,…)
- Known Solutions:

Waters IBE
Boneh-Boyen IBE
+ Hardcore function

# IBE from Search Problems on Bilinear Maps

- Dual system encryption methodology inherently requires decisional problem. (SXDH, DLIN, Matrix-DDH,…)
- Known Solutions:

Waters IBE
Boneh-Boyen IBE + Hardcore function

- Secure Under the Computational BDH assumption
- Short Ciphertexts (Waters).
- Long public parameters.

# Waters IBE + Hardcore-bit Function

$$\mathsf{mpk} = (\, GL,\ g^{w_1}, g^{w_2}, \ldots, e(g,g)^{\alpha})$$

# Waters IBE + Hardcore-bit Function

$$\text{mpk} = (\,GL,\ g^{w_1}, g^{w_2}, \ldots, e(g,g)^{\alpha})$$

GL: Goldreich-Levin hardcore bit function

H(ID): To be determined

# Waters IBE + Hardcore-bit Function

$$\mathsf{mpk} = (\, GL, \; g^{w_1}, g^{w_2}, \ldots, e(g,g)^{\alpha})$$

GL: Goldreich-Levin hardcore bit function

H(ID): To be determined

$$SK_{\mathsf{ID}} = (g^{\alpha} g^{r\mathsf{H}(\mathsf{ID})}, g^{-r})$$

# Waters IBE + Hardcore-bit Function

$$\mathsf{mpk} = (\, GL,\ g^{w_1}, g^{w_2}, \ldots, e(g,g)^{\alpha})$$

GL: Goldreich-Levin hardcore bit function

H(ID): To be determined

$$SK_{\mathsf{ID}} = (g^{\alpha} g^{r\mathsf{H}(\mathsf{ID})}, g^{-r})$$

$$CT_{\mathsf{ID}} = (GL(e(g,g)^{s\alpha}) \oplus M, g^{s}, g^{s\mathsf{H}(\mathsf{ID})})$$

# Waters IBE + Hardcore-bit Function

$$\mathsf{mpk} = (\, GL,\ g^{w_1}, g^{w_2}, \ldots, e(g,g)^{\alpha})$$

GL: Goldreich-Levin hardcore bit function

H(ID): To be determined

$$SK_{\mathsf{ID}} = (g^{\alpha} g^{r\mathsf{H}(\mathsf{ID})}, g^{-r})$$

$$CT_{\mathsf{ID}} = (GL(e(g,g)^{s\alpha}) \oplus M, g^{s}, g^{s\mathsf{H}(\mathsf{ID})})$$

Decryption

$$e(g^{s}, g^{\alpha} g^{r\mathsf{H}(\mathsf{ID})}) \cdot e(g^{-r}, g^{s\mathsf{H}(\mathsf{ID})}) = e(g,g)^{s\alpha}$$

# Hashing the Identities

$$\mathsf{mpk} = \left( \begin{matrix} GL, \\ e(g,g)^{\alpha} \end{matrix} \quad \boxed{g^{w_0}, g^{w_1}, \ldots, g^{w_\kappa}} \right)$$

Waters' hash [Wat05]

$$\mathsf{H}(\mathsf{ID}) = w_0 + \sum_{i \in \mathsf{S}(\mathsf{ID})} w_i$$

# Hashing the Identities

$$\mathsf{mpk} = \begin{pmatrix} GL, \\ e(g,g)^{\alpha} \end{pmatrix} \quad g^{w_0}, g^{w_1}, \ldots, g^{w_\kappa}$$

**Long public key!**

#group elements **linear** in ID length

Waters' h

$$\mathsf{H}(\mathsf{ID}) = w_0 + \sum_{i \in \mathsf{S}(\mathsf{ID})} w_i$$

**Linear Function**

# Initial Idea to Reduce the Key Size (that doesn't quite work)

$$\mathsf{mpk} = \left( \begin{matrix} GL, \\ e(g,g)^\alpha \end{matrix} \;\; \begin{matrix} g^{w_{1,1}}, \ldots, g^{w_{1,\sqrt{\kappa}}} \\ g^{w_{2,1}}, \ldots, g^{w_{2,\sqrt{\kappa}}} \end{matrix} \right)$$

$$\mathsf{H}(\mathsf{ID}) = w_0 + \sum_{(i,j)\in \mathsf{S}(\mathsf{ID})} w_{1,i} w_{2,j}$$

# Initial Idea to Reduce the Key Size (that doesn't quite work)

$$\mathsf{mpk} = \left( \begin{matrix} GL, \\ e(g,g)^{\alpha} \end{matrix} \boxed{\begin{matrix} g^{w_{1,1}}, \ldots, g^{w_{1,\sqrt{\kappa}}} \\ g^{w_{2,1}}, \ldots, g^{w_{2,\sqrt{\kappa}}} \end{matrix}} \right)$$

$$\boxed{\mathsf{H}(\mathsf{ID}) = w_0 + \sum_{(i,j) \in \mathsf{S}(\mathsf{ID})} w_{1,i} w_{2,j}}$$

$$g^{\mathsf{H}(\mathsf{ID})} = g^{w_0} \cdot \prod_{i,j \in \mathsf{S}(\mathsf{ID})} g^{w_{1,i} w_{2,j}}$$

# Initial Idea to Reduce the Key Size (that doesn't quite work)

$$\mathsf{mpk} = \left( \begin{matrix} GL, \\ e(g,g)^{\alpha} \end{matrix} \;\; \begin{matrix} g^{w_{1,1}}, \ldots, g^{w_{1,\sqrt{\kappa}}} \\ g^{w_{2,1}}, \ldots, g^{w_{2,\sqrt{\kappa}}} \end{matrix} \right)$$

H(I… $(i,j)\in\mathsf{S}(\mathsf{ID})$

Non-linear terms cannot be efficiently computed from mpk!!

$$g^{\mathsf{H}(\mathsf{ID})} = g^{w_0} \cdot \prod_{i,j\in\mathsf{S}(\mathsf{ID})} g^{w_{1,i} w_{2,j}}$$

# Initial Idea to Reduce the Key Size (that doesn't quite work)

$$\mathsf{mpk} = \left( \begin{matrix} GL, \\ e(g,g)^{\alpha} \end{matrix} \;\; \begin{matrix} g^{w_{1,1}}, \ldots, g^{w_{1,\sqrt{\kappa}}} \\ g^{w_{2,1}}, \ldots, g^{w_{2,\sqrt{\kappa}}} \end{matrix} \right)$$

H(I

$(i,j) \in \mathsf{S}(\mathsf{ID})$

Non-linear terms cannot be efficiently computed from mpk!!

$$g^{\mathsf{H}(\mathsf{ID})} = g^{w_0} \cdot \prod_{i,j \in \mathsf{S}(\mathsf{ID})} g^{w_{1,i} w_{2,j}}$$

How should we compute this publicly??

# Idea (that works)

Use **Boneh-Boyen technique**:

Some Random Element

$$g^{w_{1,i} w_{2,j}} \Rightarrow \left( g^{w_{1,i} w_{2,j}} g^{w_{2,j} t_{i,j}}, \; g^{t_{i,j}} \right)$$

# Idea (that works)

Use **Boneh-Boyen technique**:

Some Random Element

$$g^{w_{1,i} w_{2,j}} \Rightarrow \left( g^{w_{1,i} w_{2,j}} g^{w_{2,j} t_{i,j}},\ g^{t_{i,j}} \right)$$

Change of Variables:
(Mental Experiment)

$$t_{i,j} = \tilde{t}_{i,j} - w_{1,i}$$

# Idea (that works)

Use **Boneh-Boyen technique**:

**Some Random Element**

$$g^{w_{1,i} w_{2,j}} \Rightarrow \left( g^{w_{1,i} w_{2,j}} g^{w_{2,j} t_{i,j}},\ g^{t_{i,j}} \right)$$

Change of Variables:
(Mental Experiment)

$$t_{i,j} = \tilde{t}_{i,j} - w_{1,i}$$

$$w_{1,i} w_{2,j} + w_{2,j} t_{i,j}$$

$$= \cancel{w_{1,i} w_{2,j}} + w_{2,j} \tilde{t}_{i,j} - \cancel{w_{1,i} w_{2,j}}$$

$$= w_{2,j} \tilde{t}_{i,j}$$

# Idea (that works)

Use **Boneh-Boyen technique**:

Some Random Element

$$g^{w_{1,i} w_{2,j}} \Rightarrow \left( \underline{g^{w_{1,i} w_{2,j}} g^{w_{2,j} t_{i,j}}}, \ \underline{g^{t_{i,j}}} \right)$$

Change of Variables: $t_{i,j} = \tilde{t}_{i,j} - w_{1,i}$

(Mental Experiment)

$$w_{1,i} w_{2,j} + w_{2,j} t_{i,j}$$

$$= \cancel{w_{1,i} w_{2,j}} + w_{2,j} \tilde{t}_{i,j} - \cancel{w_{1,i} w_{2,j}}$$

$$= w_{2,j} \tilde{t}_{i,j}$$

Linear in $w_{1,i}, w_{2,j}$ ? (= Efficiently computable?)

# Idea (that works)

Use **Boneh-Boyen technique**:

Some Random Element

$$g^{w_{1,i}w_{2,j}} \Rightarrow \left( g^{w_{1,i}w_{2,j}} g^{w_{2,j}t_{i,j}}, \; g^{t_{i,j}} \right)$$

Change of Variables: $t_{i,j} = \tilde{t}_{i,j} - w_{1,i}$ ✓

(Mental Experiment)

$$w_{1,i}w_{2,j} + w_{2,j}t_{i,j}$$

$$= w_{1,i}w_{2,j} + w_{2,j}\tilde{t}_{i,j} - w_{1,i}w_{2,j}$$

$$= w_{2,j}\tilde{t}_{i,j}$$

Linear in $w_{1,i}, w_{2,j}$ ? (= Efficiently computable?)

# Idea (that works)

Use **Boneh-Boyen technique**:

**Some Random Element**

$$g^{w_{1,i}w_{2,j}} \Rightarrow \left( g^{w_{1,i}w_{2,j}} g^{w_{2,j}t_{i,j}},\ g^{t_{i,j}} \right)$$

Change of Variables:
(Mental Experiment)

$$t_{i,j} = \tilde{t}_{i,j} - w_{1,i}$$ ✓

$$w_{1,i}w_{2,j} + w_{2,j}t_{i,j}$$

$$= \cancel{w_{1,i}w_{2,j}} + w_{2,j}\tilde{t}_{i,j} - \cancel{w_{1,i}w_{2,j}}$$

$$= w_{2,j}\tilde{t}_{i,j}$$ ✓

Linear in $w_{1,i}, w_{2,j}$ ? (= Efficiently computable?)

# Idea (that works)

Some Random Element

Use **Boneh-Boyen technique**:

$$g^{w_{1,i}w_{2,j}} \Rightarrow \left( \underline{g^{w_{1,i}w_{2,j}}\, g^{w_{2,j}t_{i,j}}},\ \underline{g^{t_{i,j}}} \right)$$

Change of Variables:
(Mental Experiment)

$$\underline{t_{i,j} = \tilde{t}_{i,j} - w_{1,i}} \checkmark$$

$$w_{1,i}w_{2,j} + w_{2,j}t_{i,j}$$

$$= \cancel{w_{1,i}w_{2,j}} + w_{2,j}\tilde{t}_{i,j} - \cancel{w_1 \ldots}$$

$$\underline{= w_{2,j}\tilde{t}_{i,j}} \checkmark$$

Random Element Chosen by the Encryptor

$$g^{w_{1,i}w_{2,j}} \Rightarrow \left( (g^{w_{2,j}})^{\tilde{t}_{i,j}},\ g^{\tilde{t}_{i,j}} \cdot (g^{w_{1,i}})^{-1} \right)$$

# Resulting Scheme

$$\mathsf{mpk} = \left( \begin{matrix} GL, \\ e(g,g)^{\alpha} \end{matrix} \boxed{\begin{matrix} g^{w_{1,1}}, \ldots, g^{w_{1,\sqrt{\kappa}}} \\ g^{w_{2,1}}, \ldots, g^{w_{2,\sqrt{\kappa}}} \end{matrix}} \right)$$

$$\mathsf{H}(\mathsf{ID}) = w_0 + \sum_{(i,j) \in \mathsf{S}(\mathsf{ID})} w_{1,i} w_{2,j}$$

$$SK_{\mathsf{ID}} = (g^{\alpha} g^{r\mathsf{H}(\mathsf{ID})}, g^{-r}, \boxed{\{g^{r w_{2,j}}\}_{j \in [\sqrt{\kappa}]}})$$

$$CT_{\mathsf{ID}} = \left( \begin{matrix} GL(e(g,g)^{s\alpha}) \oplus M, \\ g^{s}, g^{s\mathsf{H}(\mathsf{ID}) \boxed{+ \sum_{j \in [\sqrt{\kappa}]} t_j w_{2,j}}}, \end{matrix} \boxed{\{g^{t_j}\}_{j \in [\sqrt{\kappa}]}} \right)$$

# Resulting Scheme

$$\mathsf{mpk} = \left( \begin{array}{c} GL, \\ e(g,g)^{\alpha} \end{array} \boxed{\begin{array}{c} g^{w_{1,1}}, \ldots, g^{w_{1,\sqrt{\kappa}}} \\ g^{w_{2,1}}, \ldots, g^{w_{2,\sqrt{\kappa}}} \end{array}} \right)$$

$$\mathsf{H}(\mathsf{ID}) = w_0 + \sum_{(i,j) \in \mathsf{S}(\mathsf{ID})} w_{1,i} w_{2,j}$$

$$SK_{\mathsf{ID}} = \left(g^{\alpha} g^{r\mathsf{H}(\mathsf{ID})}, g^{-r}, \boxed{\{g^{r w_{2,j}}\}_{j \in [\sqrt{\kappa}]}}\right)$$

$$CT_{\mathsf{ID}} = \left( \begin{array}{c} GL(e(g,g)^{s\alpha}) \oplus M, \\ g^{s}, g^{s\mathsf{H}(\mathsf{ID}) \boxed{+ \sum_{j \in [\sqrt{\kappa}]} t_j w_{2,j}}}, \end{array} \boxed{\{g^{t_j}\}_{j \in [\sqrt{\kappa}]}} \right)$$

longer

# Resulting Scheme

$$\mathsf{mpk} = \left( \begin{matrix} GL, \\ e(g,g)^{\alpha} \end{matrix} \begin{matrix} g^{w_{1,1}}, \ldots, g^{w_{1,\sqrt{\kappa}}} \\ g^{w_{2,1}}, \ldots, g^{w_{2,\sqrt{\kappa}}} \end{matrix} \right)$$

Shorter!

$$\mathsf{H}(\mathsf{ID}) = w_0 + \sum_{(i,j)\in \mathsf{S}(\mathsf{ID})} w_{1,i} w_{2,j}$$

$$SK_{\mathsf{ID}} = (g^{\alpha} g^{r\mathsf{H}(\mathsf{ID})}, g^{-r}, \{g^{r w_{2,j}}\}_{j\in[\sqrt{\kappa}]})$$

longer

$$CT_{\mathsf{ID}} = \left( \begin{matrix} GL(e(g,g)^{s\alpha}) \oplus M, \\ g^{s}, g^{s\mathsf{H}(\mathsf{ID}) + \sum_{j\in[\sqrt{\kappa}]} t_j w_{2,j}}, \end{matrix} \{g^{t_j}\}_{j\in[\sqrt{\kappa}]} \right)$$

# Comparison

| | $\lvert mpk \rvert$ | $\lvert CT \rvert$ | $\lvert sk \rvert$ | **Assumption** |
|---|---|---|---|---|
| [Wat05] + hardcore | $O(\kappa)$ | $O(1)$ | $O(1)$ | CBDH assumption |
| Ours | $O(\sqrt{\kappa})$ | $O(\sqrt{\kappa})$ | $O(\sqrt{\kappa})$ | 3CBDHE assumption |

*We count the number of group elements.

3CBDH assumption: $(g^a, g^b, g^c) \not\rightarrow e(g,g)^{abc}$

3CBDHE assumption: $(g^a, g^{a^2}, g^c) \not\rightarrow e(g,g)^{ca^3}$

# Agenda

I. Preliminaries

II. Lattice Section
- ✓ Previous Works
- ✓ Our Work

III. Bilinear Map Section
- ✓ Previous Works
- ✓ Our Work

IV. Summary

# Summary: New Adaptively Secure IBEs

- Both based on partitioning technique with non-linear functions
- New IBE from ideal lattices:
  - Improve currently best scheme of [Yam16]: super-poly modulus → poly modulus RLWE
  - Use commutativity of Ring in an essential way
- New IBE from bilinear maps:
  - First scheme with sub-linear-size mpk from search problem rather than decisional problem
  - Boneh-Boyen technique in the construction rather than in the security proof

# Comparison with (Very) Recent Works

- Comparison of adaptively secure lattice IBEs when instantiated with ideal lattices

| | \|mpk\| | \|CT\| | \|SK_ID\| | Assumption | Property |
|---|---|---|---|---|---|
| [ABB10] +[Boy10] | $\tilde{O}(n\kappa)$ | $\tilde{O}(n)$ | $\tilde{O}(n)$ | Poly RLWE | |
| [Yam16] | $\tilde{O}(n\kappa^{1/d})$ | $\tilde{O}(n)$ | $\tilde{O}(n)$ | Super-poly RLWE | |
| [AFL16] | $\tilde{O}(n)$ | $\tilde{O}(n)$ | $\tilde{O}(n)$ | Poly RLWE | |
| [ZCZ16] | $\tilde{O}(\log Q)$ | $\tilde{O}(n)$ | $\tilde{O}(n)$ | Poly RWE | Q-bounded |
| [BL16] | $\tilde{O}(n\kappa)$ | $\tilde{O}(n)$ | $\tilde{O}(n)$ | Super-poly RLWE | Tightly secure |
| [Ours] | $\tilde{O}(n\kappa^{1/d})$ | $\tilde{O}(n)$ | $\tilde{O}(n)$ | Poly RLWE | |