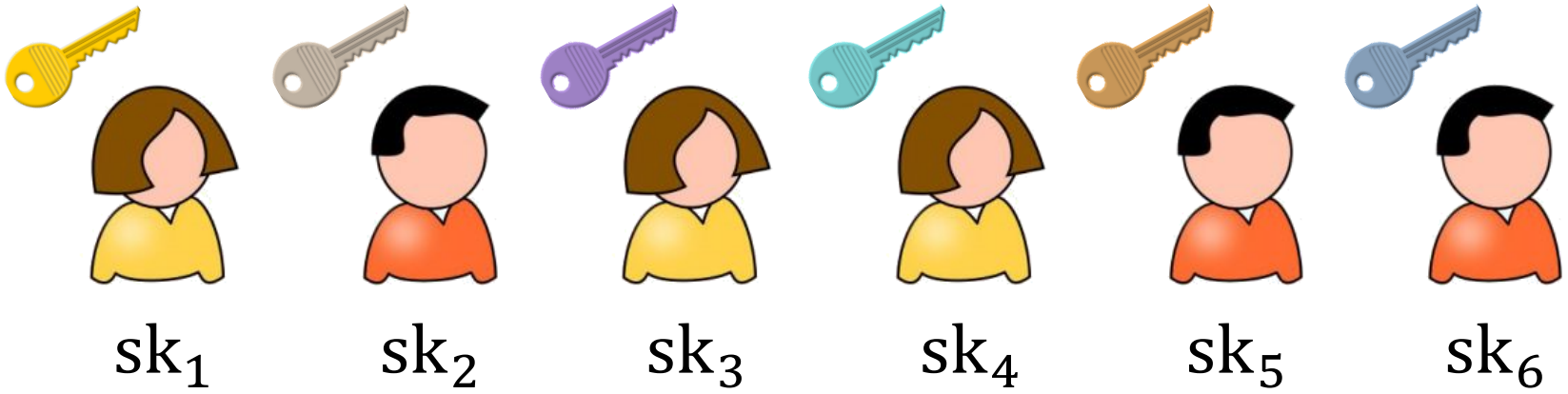
Distributed Broadcast Encryption from Lattices

Jeffrey Champion and David Wu

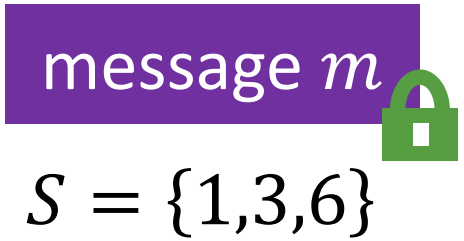
Broadcast Encryption

[FN93]



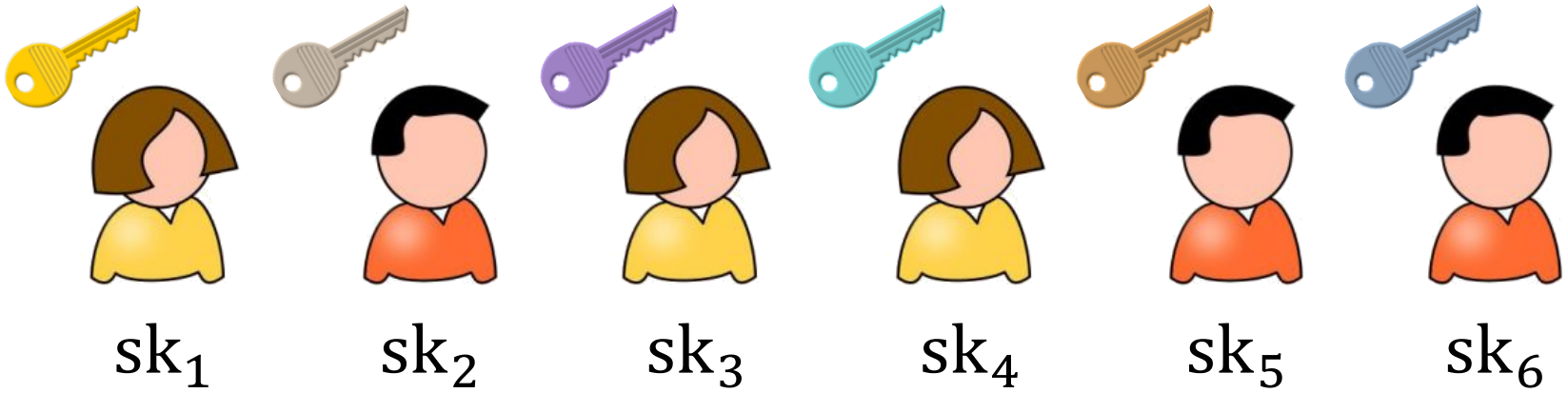
Broadcast Encryption

[FN93]



$S = \{1,3,6\}$

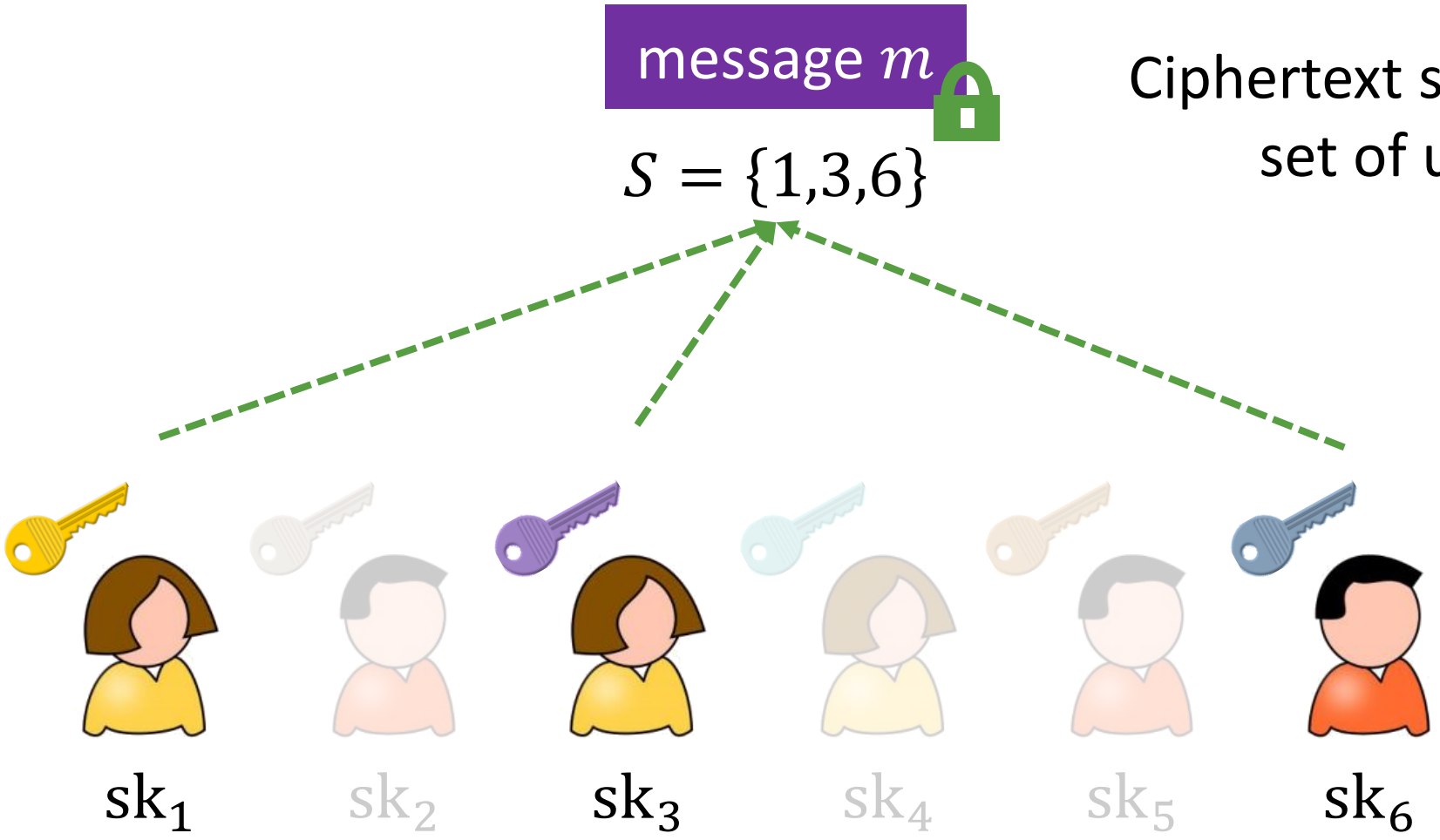
Ciphertext specifies a set of users



Broadcast Encryption

[FN93]

Functionality: Users in the set can decrypt



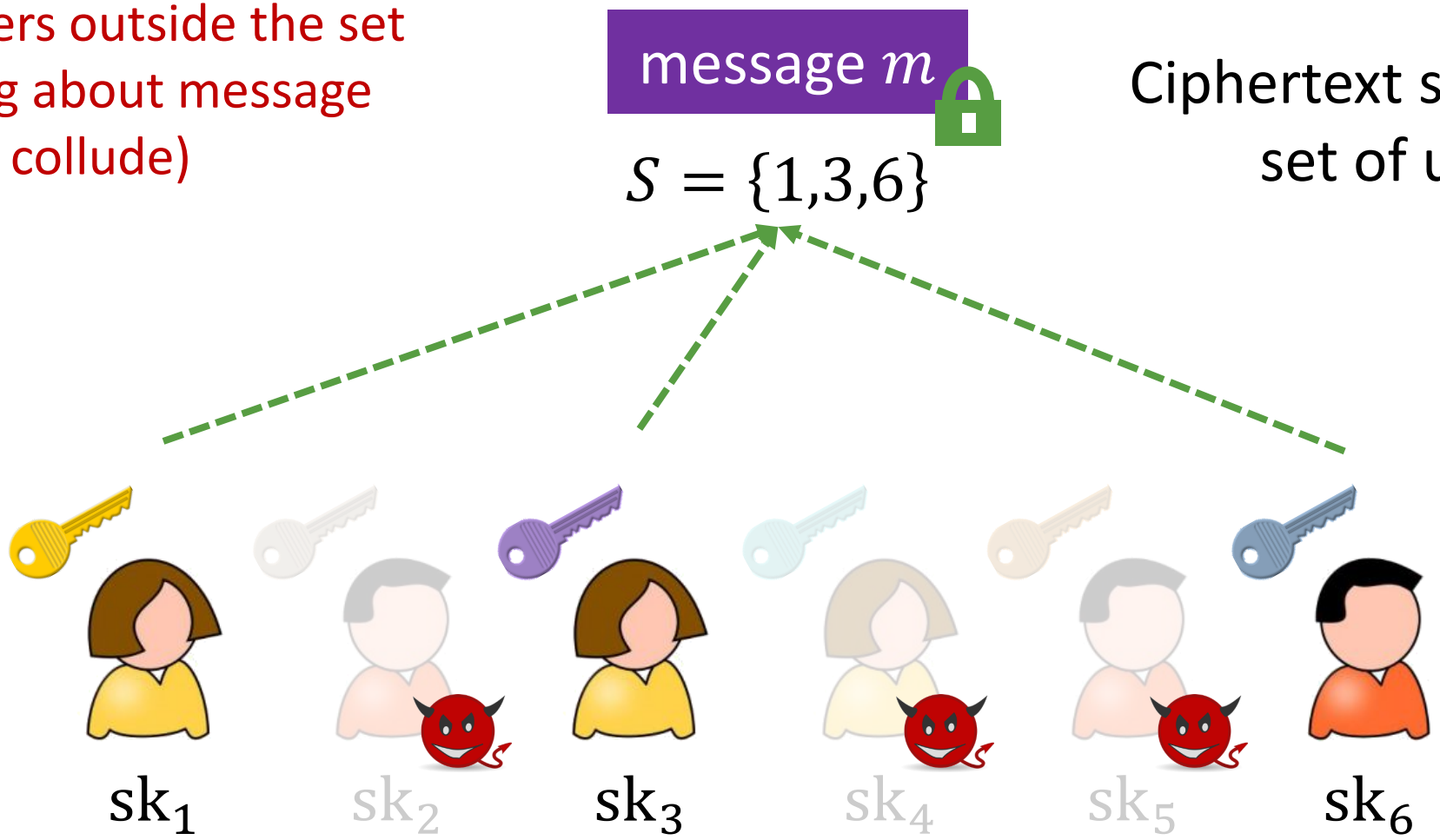
Ciphertext specifies a set of users

Broadcast Encryption

[FN93]

Functionality: Users in the set can decrypt

Security: Users outside the set learn nothing about message (even if they collude)



Ciphertext specifies a set of users

Broadcast Encryption

[FN93]

Functionality: Users in the set can decrypt

Security: Users outside the set learn nothing about message (even if they collude)

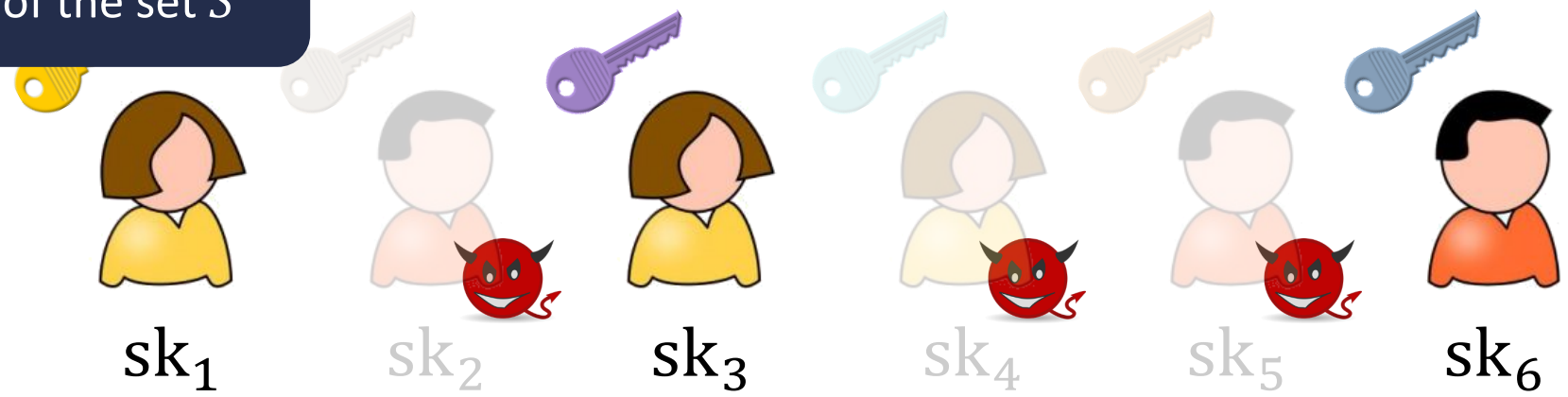
Efficiency: $|\mathrm{ct}| = |m| + \mathrm{poly}(\lambda, \log|S|)$

message m

$S = \{1,3,6\}$

Ciphertext specifies a set of users

Note: decryption requires knowledge of the set S

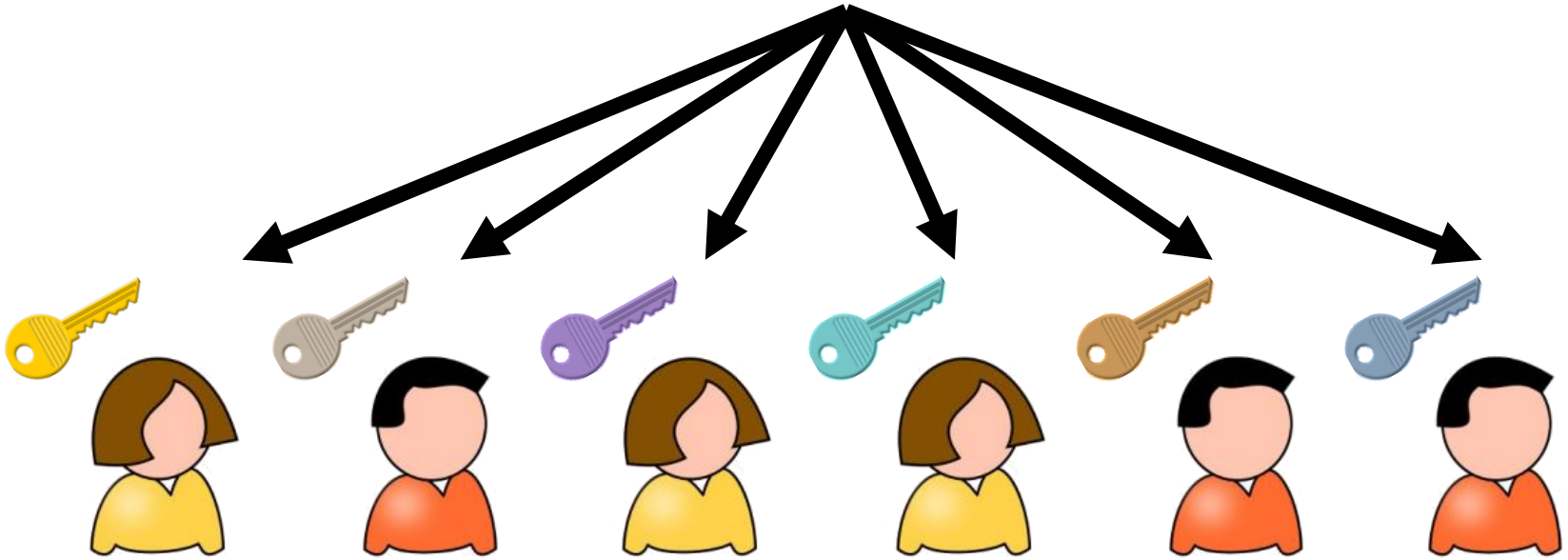


Broadcast Encryption

[FN93]

Central **trusted** authority generates keys

master secret key



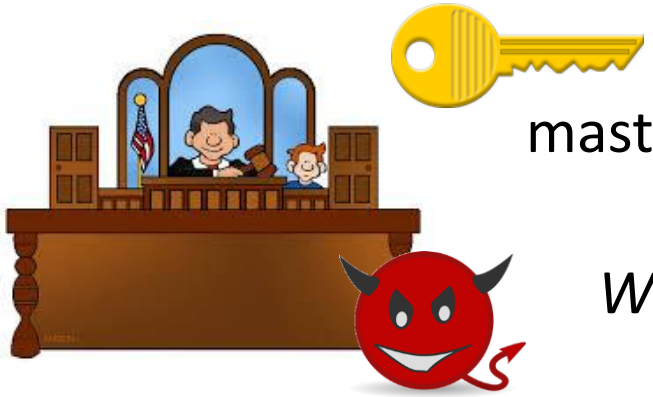
sk_1 sk_2 sk_3 sk_4 sk_5 sk_6

Broadcast Encryption

[FN93]

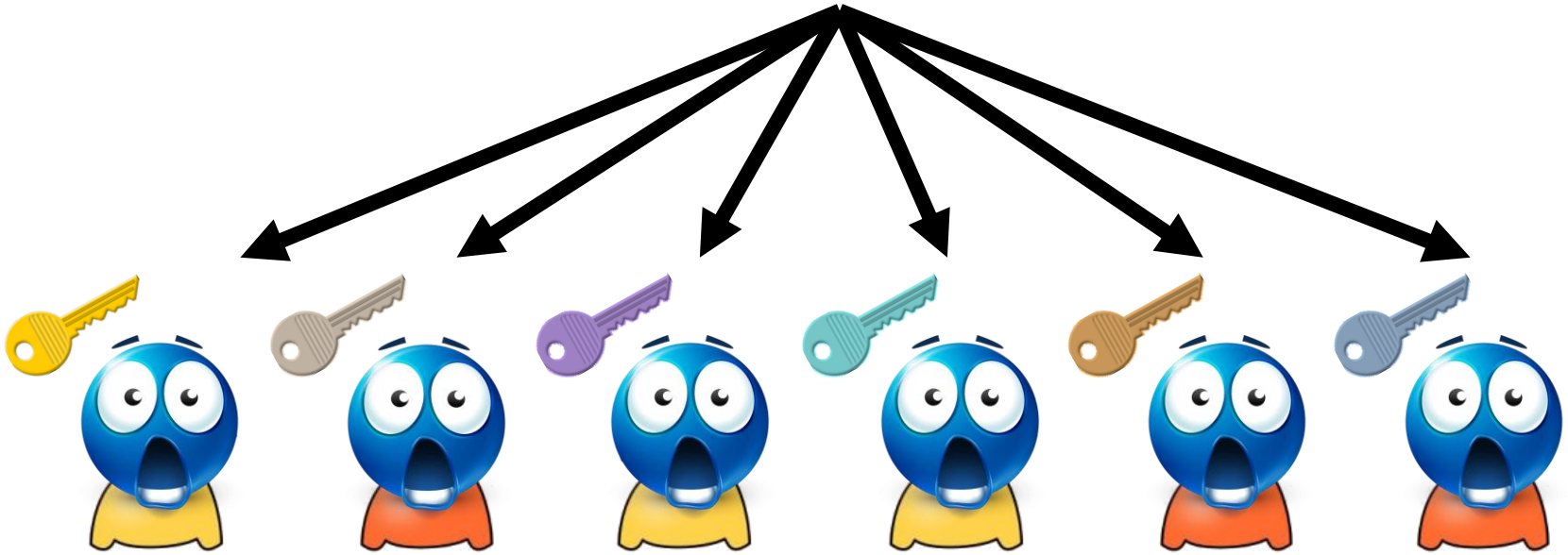
Central **trusted** authority generates keys

Central point of failure



master secret key

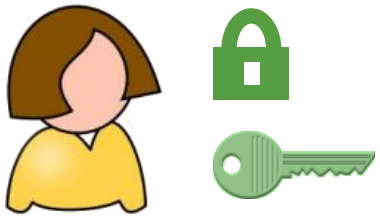
What if the key issuer is compromised?



sk_1 sk_2 sk_3 sk_4 sk_5 sk_6

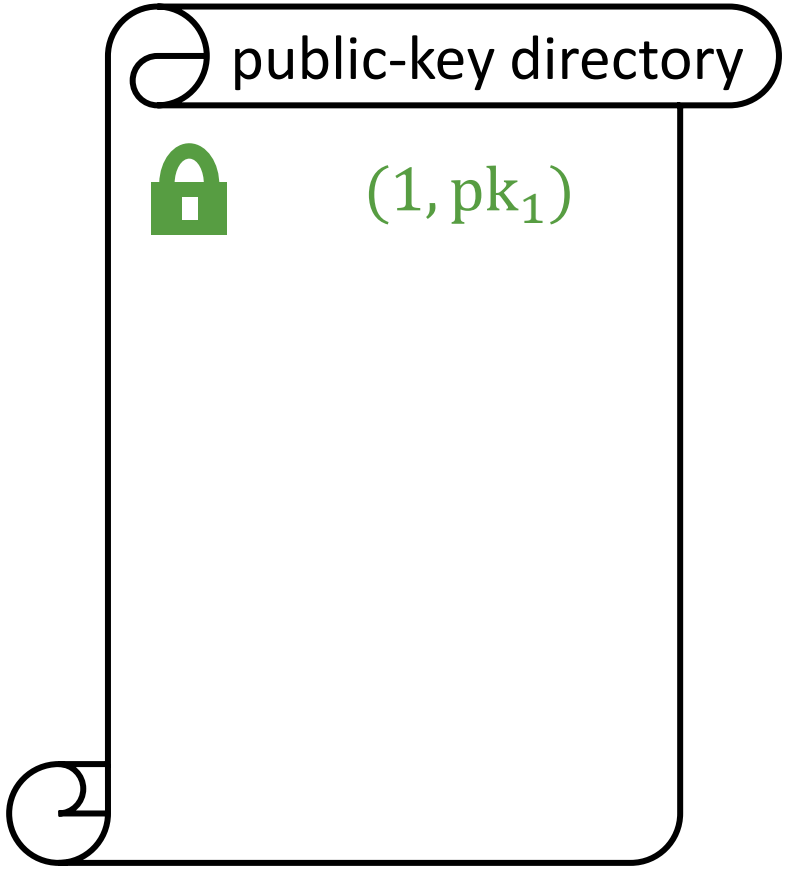
Distributed Broadcast Encryption (DBE)

[BZ14]



pk_1

sk_1

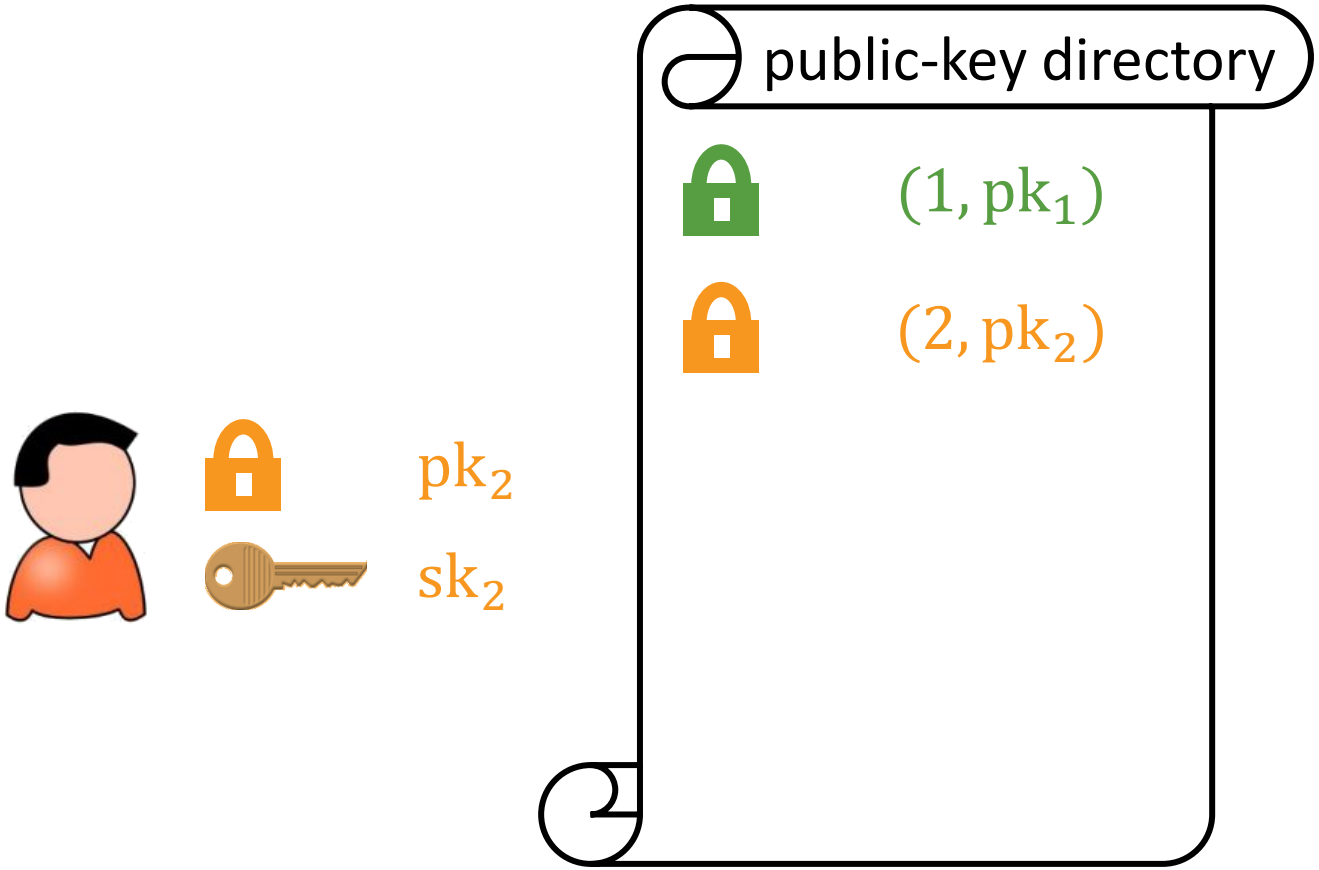


Users generate public/private keys independently (as in public-key encryption)

Broadcast encryption without a central authority

Distributed Broadcast Encryption (DBE)

[BZ14]

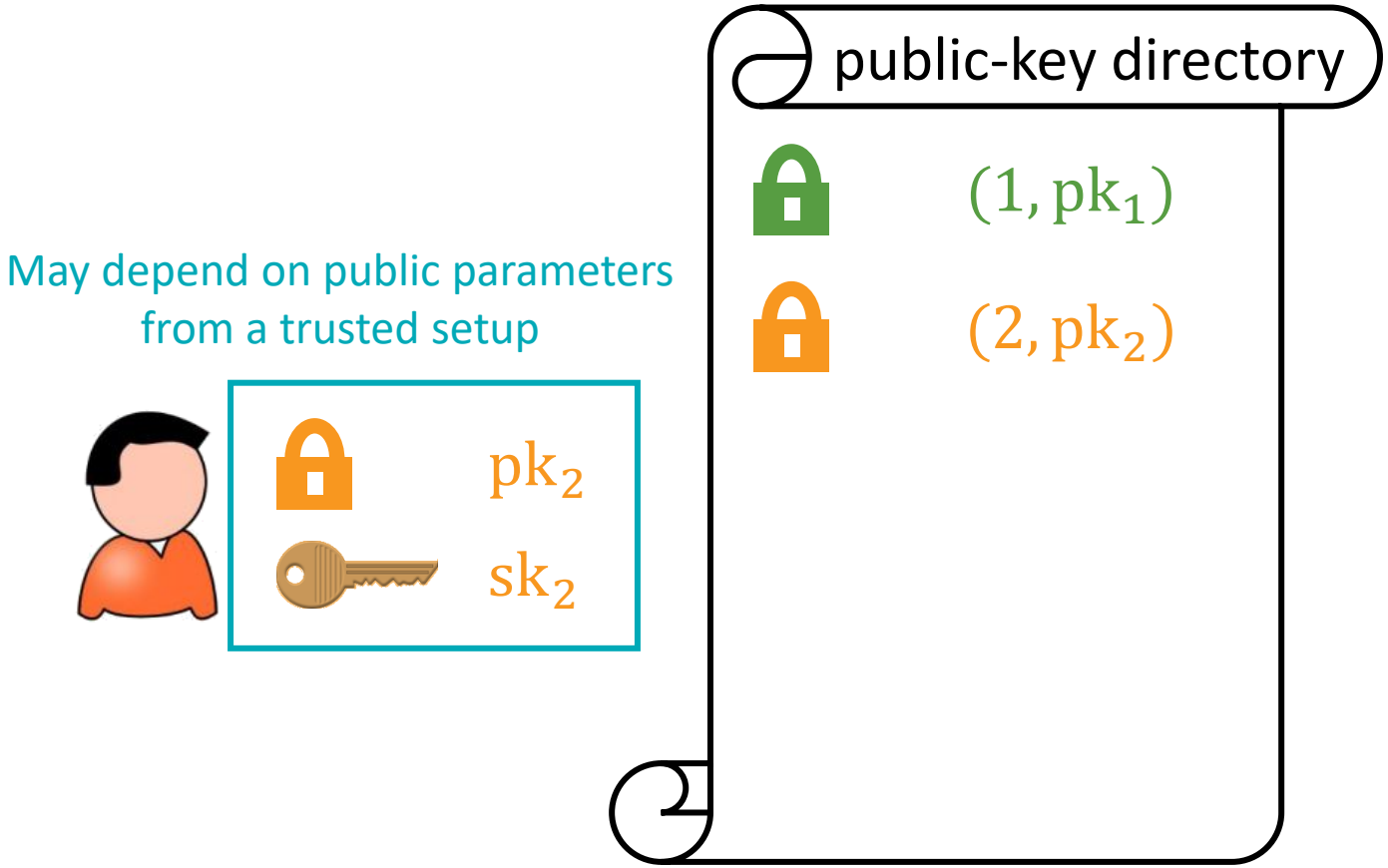


Users generate public/private keys independently (as in public-key encryption)

Broadcast encryption without a central authority

Distributed Broadcast Encryption (DBE)

[BZ14]

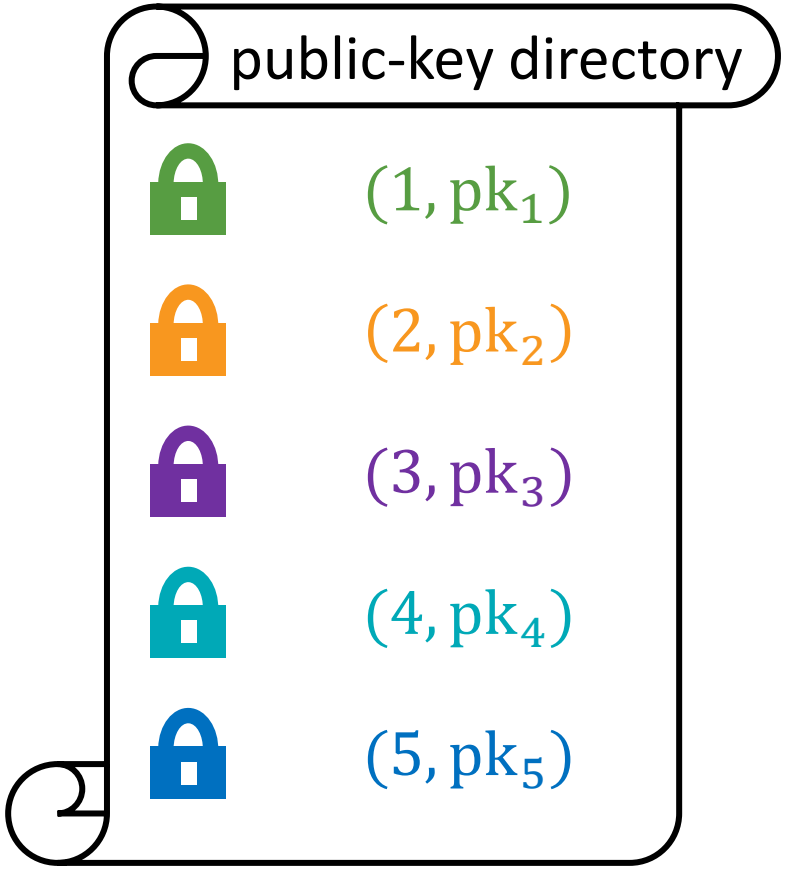


Users generate public/private keys independently (as in public-key encryption)

Broadcast encryption without a central authority

Distributed Broadcast Encryption (DBE)

[BZ14]



public parameters

$$\mathrm{Encrypt}(\mathrm{pp}, \{\mathrm{pk}_i\}_{i \in S}, m) \to \mathrm{ct}$$

$$\mathrm{Decrypt}(\mathrm{pp}, \{\mathrm{pk}_i\}_{i \in S}, \mathrm{sk}, \mathrm{ct}) \to m$$

Efficiency: $|\mathrm{ct}| = |m| + \mathrm{poly}(\lambda, \log|S|)$

Security: Users outside the set learn nothing about message (even if they collude)

Broadcast encryption without a central authority

Constructions of DBE

- Indistinguishability obfuscation (and OWF) [BZ14]
- Witness encryption (and leveled HE) [FWW23]
- Registered attribute-based encryption [FWW23]
- Pairing-based assumptions (BDHE or k-Lin) [KMW23, GKPW24]

Constructions of DBE

- Indistinguishability obfuscation (and OWF) [BZ14]
- Witness encryption (and leveled HE) [FWW23]
- Registered attribute-based encryption [FWW23]
- Pairing-based assumptions (BDHE or k-Lin) [KMW23, GKPW24]

Constructions from lattice assumptions?

Lattice-Based Distributed Broadcast

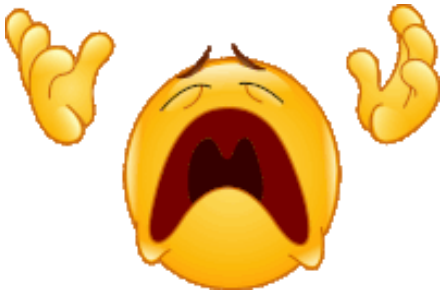
LWE?

[Reg05]

Lattice-Based Distributed Broadcast

LWE?
[Reg05]

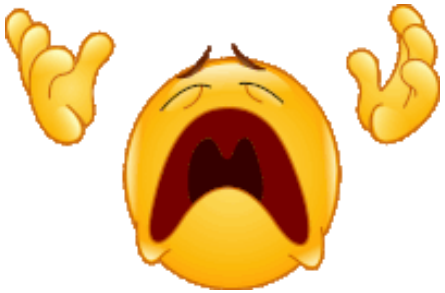
No centralized broadcast after ~20 years



Lattice-Based Distributed Broadcast

LWE?
[Reg05]

No centralized broadcast after ~20 years



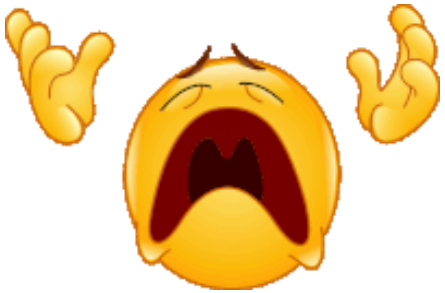
Evasive LWE:
[Tsa22, Wee22]

Public-coin: centralized broadcast [Wee22]
Private-coin: DBE (via WE [Tsa22, VWW22])

Lattice-Based Distributed Broadcast

LWE?
[Reg05]

No centralized broadcast after ~20 years



Evasive LWE:
[Tsa22, Wee22]

Public-coin: centralized broadcast [Wee22]

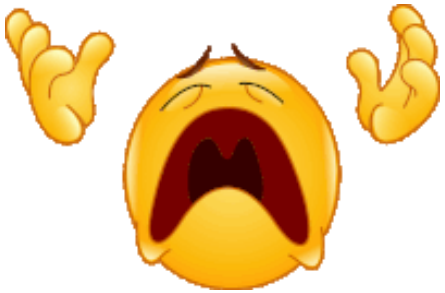
Private-coin: DBE (via WE [Tsa22, VWW22])

Assumption broken and partly patched [BÜW24]

Lattice-Based Distributed Broadcast

LWE?
[Reg05]

No centralized broadcast after ~20 years

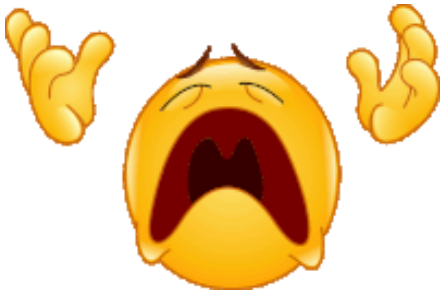


Evasive LWE:
[Tsa22, Wee22]

Public-coin: centralized broadcast [Wee22]

Private-coin: DBE (via WE [Tsa22, VWW22])

Assumption broken and partly patched [BÜW24]



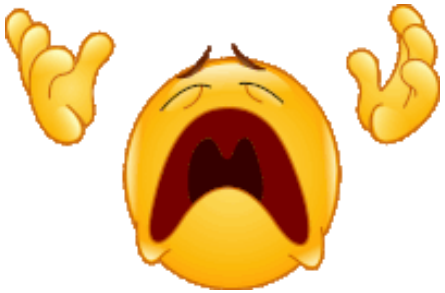
ℓ-succinct LWE:
[Wee24]

Centralized broadcast via succinct ABE [Wee24]

Lattice-Based Distributed Broadcast

LWE?
[Reg05]

No centralized broadcast after ~20 years

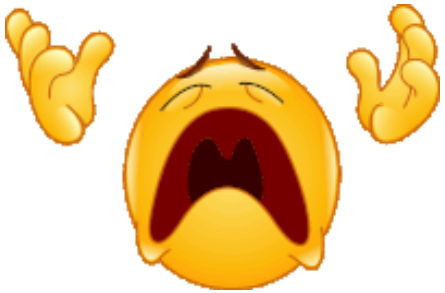


Evasive LWE:
[Tsa22, Wee22]

Public-coin: centralized broadcast [Wee22]

Private-coin: DBE (via WE [Tsa22, VWW22])

Assumption broken and partly patched [BÜW24]



ℓ-succinct LWE:
[Wee24]

Centralized broadcast via succinct ABE [Wee24]

This work: DBE from ℓ-succinct LWE

ℓ-Succinct LWE Assumption

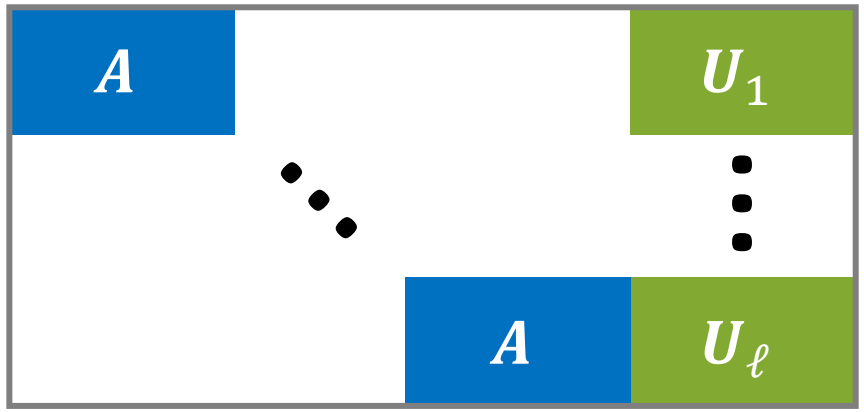
[Wee24]

LWE is hard with respect to $\boldsymbol{A}$ given a "fresh" trapdoor for a related matrix $\boldsymbol{D}_\ell$:

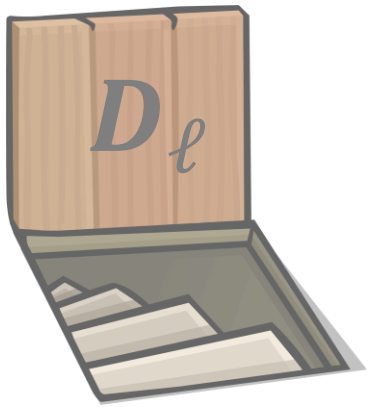
$$\boldsymbol{s}^{\mathrm{T}} \boldsymbol{A} + \boldsymbol{e}^{\mathrm{T}} \approx \boldsymbol{z}^{\mathrm{T}}$$

$\boldsymbol{A}, \boldsymbol{U}_i, \boldsymbol{s}, \boldsymbol{z}$ are uniform

given $\boldsymbol{D}_\ell =$



and



ℓ-Succinct LWE Assumption

[Wee24]

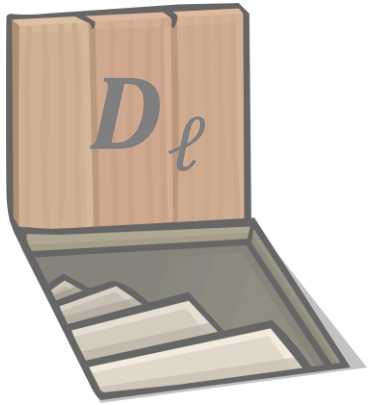
LWE is hard with respect to $\boldsymbol{A}$ given a "fresh" trapdoor for a related matrix $\boldsymbol{D}_\ell$:

$$\boldsymbol{s}^{\mathrm{T}} \boldsymbol{A} + \boldsymbol{e}^{\mathrm{T}} \approx \boldsymbol{z}^{\mathrm{T}}$$

$\boldsymbol{A}, \boldsymbol{U}_i, \boldsymbol{s}, \boldsymbol{z}$ are uniform

Implied by LWE if width of $\boldsymbol{U}_i$ scales with ℓ and by public-coin evasive LWE when $\boldsymbol{U}_i$ are narrow

given $\boldsymbol{D}_\ell = \begin{bmatrix} \boldsymbol{A} & & & \boldsymbol{U}_1 \\ & \ddots & & \vdots \\ & & \boldsymbol{A} & \boldsymbol{U}_\ell \end{bmatrix}$ and



ℓ-Succinct LWE Assumption

[Wee24]

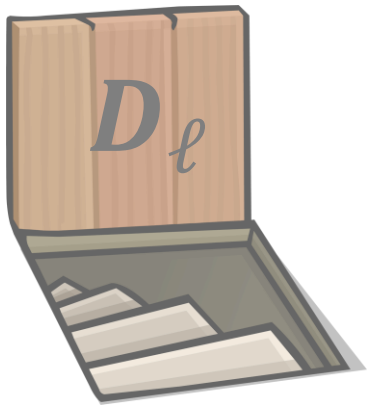
LWE is hard with respect to $\boldsymbol{A}$ given a "fresh" trapdoor for a related matrix $\boldsymbol{D}_\ell$:

$$\boldsymbol{s}^{\mathrm{T}} \boldsymbol{A} + \boldsymbol{e}^{\mathrm{T}} \approx \boldsymbol{z}^{\mathrm{T}}$$

$\boldsymbol{A}, \boldsymbol{U}_i, \boldsymbol{s}, \boldsymbol{z}$ are uniform

Implied by LWE if width of $\boldsymbol{U}_i$ scales with ℓ and by public-coin evasive LWE when $\boldsymbol{U}_i$ are narrow

given $\boldsymbol{D}_\ell = \begin{bmatrix} \boldsymbol{A} & & & \boldsymbol{U}_1 \\ & \ddots & & \vdots \\ & & \boldsymbol{A} & \boldsymbol{U}_\ell \end{bmatrix}$ and



Falsifiable and instance-independent unlike evasive LWE

ℓ-Succinct LWE Assumption

[Wee24]

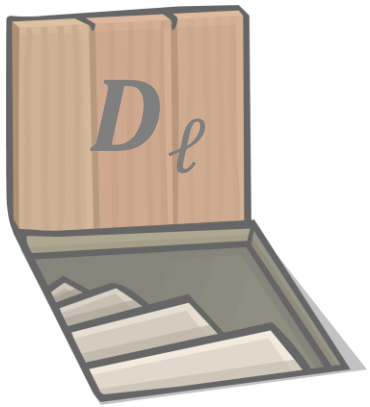
LWE is hard with respect to $\boldsymbol{A}$ given a "fresh" trapdoor for a related matrix $\boldsymbol{D}_\ell$:

$$\boldsymbol{s}^{\mathrm{T}} \boldsymbol{A} + \boldsymbol{e}^{\mathrm{T}} \approx \boldsymbol{z}^{\mathrm{T}}$$

$\boldsymbol{A}, \boldsymbol{U}_i, \boldsymbol{s}, \boldsymbol{z}$ are uniform

Implied by LWE if width of $\boldsymbol{U}_i$ scales with ℓ and by public-coin evasive LWE when $\boldsymbol{U}_i$ are narrow

given $\boldsymbol{D}_\ell = \begin{bmatrix} \boldsymbol{A} & & & \boldsymbol{U}_1 \\ & \ddots & & \vdots \\ & & \boldsymbol{A} & \boldsymbol{U}_\ell \end{bmatrix}$ and

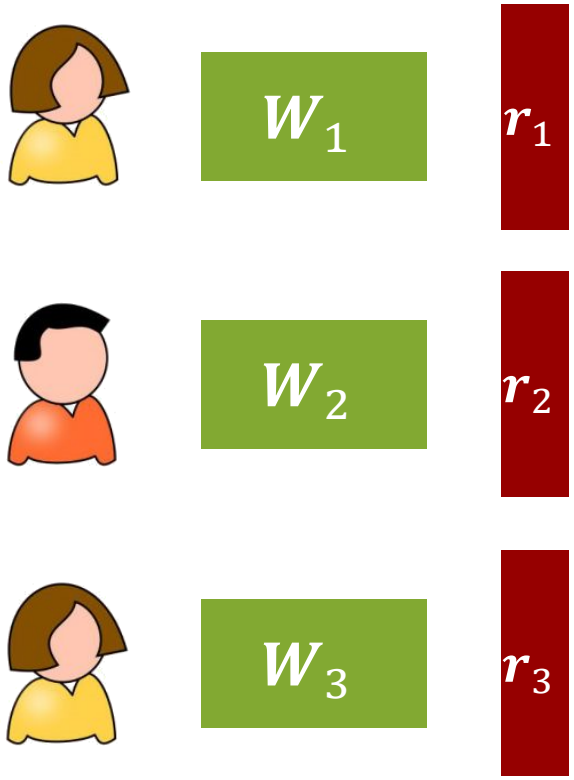


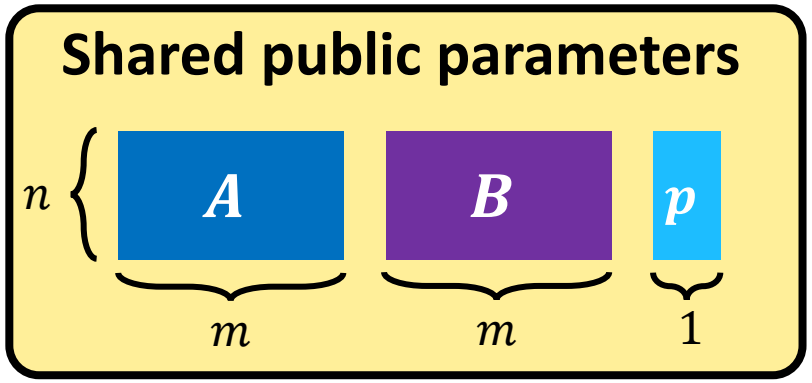
Falsifiable and instance-independent unlike evasive LWE

Also yields ABE with short ciphertexts [Wee24] and functional commitments [WW23]

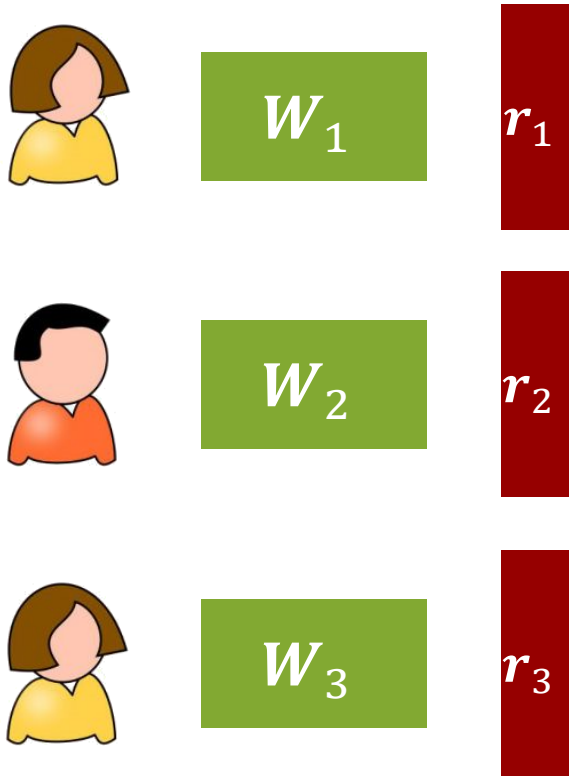
Starting Point: Centralized Broadcast Encryption

Starting Point: Centralized Broadcast Encryption

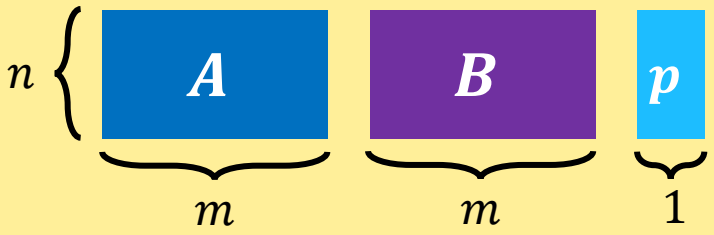




Starting Point: Centralized Broadcast Encryption



Shared public parameters



Ciphertext encrypting a bit $b \in \{0,1\}$ to a set $S \subseteq [\ell]$:

plain public-key encryption terms

$$s^T \left(B + \sum_{i \in S} W_i \right)$$

broadcast term

Noise omitted

Starting Point: Centralized Broadcast Encryption

Public parameters

$\boldsymbol{A}$ $\boldsymbol{W}_i$

$\boldsymbol{B}$ $\boldsymbol{r}_i$ $i \in [\ell]$

$\boldsymbol{p}$

Encryption of $b \in \{0,1\}$ to $S \subseteq [\ell]$:

$$\boldsymbol{s}^{\mathrm{T}} \boldsymbol{A} \qquad \underbrace{\boldsymbol{s}^{\mathrm{T}} \boldsymbol{p}}_{\text{masks } b}$$

$$\boldsymbol{s}^{\mathrm{T}} \left(\boldsymbol{B} + \sum_{j \in S} \boldsymbol{W}_j \right)$$

Starting Point: Centralized Broadcast Encryption

Public parameters

$\boldsymbol{A}$ $\boldsymbol{W}_i$

$\boldsymbol{B}$ $\boldsymbol{r}_i$ $i \in [\ell]$

$\boldsymbol{p}$

Goal: user $i \in S$ should be able to uniquely compute $\boldsymbol{s}^{\mathrm{T}} \boldsymbol{p}$

Encryption of $b \in \{0,1\}$ to $S \subseteq [\ell]$:

$\boldsymbol{s}^{\mathrm{T}} \boldsymbol{A}$ $\underbrace{\boldsymbol{s}^{\mathrm{T}} \boldsymbol{p}}_{\text{masks } b}$

$$\boldsymbol{s}^{\mathrm{T}} \left(\boldsymbol{B} + \sum_{j \in S} \boldsymbol{W}_j \right)$$

Starting Point: Centralized Broadcast Encryption

Public parameters

A $\quad W_i$

B $\quad r_i \quad i \in [\ell]$

p

Goal: user $i \in S$ should be able to uniquely compute $s^{\mathrm{T}} p$

Secret key for user i: should derive a short vector y_S such that

$$A\, y_S = p + B\, r_i + \sum_{j \in S} W_j\, r_i$$

Encryption of $b \in \{0,1\}$ to $S \subseteq [\ell]$:

$$s^{\mathrm{T}} A \qquad \underbrace{s^{\mathrm{T}} p}_{\text{masks } b}$$

$$s^{\mathrm{T}} \left(B + \sum_{j \in S} W_j \right)$$

Starting Point: Centralized Broadcast Encryption

Public parameters

$\boldsymbol{A}$ $\boldsymbol{W}_i$

$\boldsymbol{B}$ $\boldsymbol{r}_i$ $i \in [\ell]$

$\boldsymbol{p}$

Encryption of $b \in \{0,1\}$ to $S \subseteq [\ell]$:

$$\boldsymbol{s}^{\mathrm{T}} \boldsymbol{A} \qquad \underbrace{\boldsymbol{s}^{\mathrm{T}} \boldsymbol{p}}_{\text{masks } b}$$

$$\boldsymbol{s}^{\mathrm{T}} \left(\boldsymbol{B} + \sum_{j \in S} \boldsymbol{W}_j \right)$$

Goal: user $i \in S$ should be able to uniquely compute $\boldsymbol{s}^{\mathrm{T}} \boldsymbol{p}$

Secret key for user $\boldsymbol{i}$: should derive a short vector $\boldsymbol{y}_S$ such that

$$\boldsymbol{A} \boldsymbol{y}_S = \boldsymbol{p} + \boldsymbol{B} \boldsymbol{r}_i + \sum_{j \in S} \boldsymbol{W}_j \boldsymbol{r}_i$$

Decryption: User i can now compute $\boldsymbol{s}^{\mathrm{T}} \boldsymbol{p}$ by the subtraction

$$\boldsymbol{s}^{\mathrm{T}} \boldsymbol{A} \boldsymbol{y}_S - \boldsymbol{s}^{\mathrm{T}} \left(\boldsymbol{B} + \sum_{j \in S} \boldsymbol{W}_j \right) \boldsymbol{r}_i$$

Requires $\boldsymbol{r}_i$ be short when noise is included

Starting Point: Centralized Broadcast Encryption

Public parameters

A $\quad W_i$

B $\quad r_i \quad i \in [\ell]$

p

Encryption of $b \in \{0,1\}$ to $S \subseteq [\ell]$:

$s^{\mathrm{T}} A \qquad \underbrace{s^{\mathrm{T}} p}_{\text{masks } b}$

$$s^{\mathrm{T}} \left(B + \sum_{j \in S} W_j \right)$$

Goal: user $i \in S$ should be able to uniquely compute $s^{\mathrm{T}} p$

Secret key for user i: short vectors $y_{j,i}$ such that

$$A\, y_{i,i} = p + B\, r_i + W_i\, r_i$$

$$A\, y_{j,i} = W_j\, r_i \qquad j \in [\ell], j \neq i$$

When $i \in S$, derive $y_S = \sum_{j \in S} y_{j,i}$

Simplifying Secret Keys

Public parameters

A $\quad W_i$

B $\quad r_i \quad i \in [\ell]$

p

Encryption of $b \in \{0,1\}$ to $S \subseteq [\ell]$:

$s^T \; A \qquad \underbrace{s^T \; p}_{\text{masks } b}$

$$s^T \left(B + \sum_{j \in S} W_j \right)$$

Secret key for user i: short vectors $y_{j,i}$ such that

$$A \; y_{i,i} = p + B \; r_i + W_i \; r_i$$

$$A \; y_{j,i} = W_j \; r_i \qquad j \in [\ell], j \neq i$$

Does not map A to p !

Simplifying Secret Keys

Public parameters

A W_i B r_i $y_{j,i}$ p

$i, j \in [\ell]$
$j \neq i$

Encryption of $b \in \{0,1\}$ to $S \subseteq [\ell]$:

$$s^{\mathrm{T}} A \qquad \underbrace{s^{\mathrm{T}} p}_{\text{masks } b}$$

$$s^{\mathrm{T}} \left(B + \sum_{j \in S} W_j \right)$$

Secret key for user i: short vector $y_{i,i}$ such that

$$A\, y_{i,i} = p + B\, r_i + W_i\, r_i$$

Simplifying Secret Keys

Public parameters

A $\quad W_i$

B $\quad r_i \quad y_{j,i}$ $\quad i, j \in [\ell]$, $j \neq i$

p

Secret key for user i: short vector $y_{i,i}$ such that

$$A\, y_{i,i} = p + B\, r_i + W_i\, r_i$$

Encryption of $b \in \{0,1\}$ to $S \subseteq [\ell]$:

$$s^{\mathrm{T}} A \qquad \underbrace{s^{\mathrm{T}} p}_{\text{masks } b}$$

$$s^{\mathrm{T}} \left(B + \sum_{j \in S} W_j \right)$$

This is a **centralized** broadcast encryption scheme

Sampling $y_{i,j}$ requires knowledge of the trapdoor for A

Distributed Key Generation

Challenge: No one can know a trapdoor for $\boldsymbol{A}$

Distributed Key Generation

Challenge: No one can know a trapdoor for $\boldsymbol{A}$

Approach: User i will generate $\boldsymbol{W}_i$ and short $\boldsymbol{y}_{i,j}$ given public parameters

Distributed Key Generation

Challenge: No one can know a trapdoor for $\boldsymbol{A}$

Approach: User i will generate $\boldsymbol{W}_i$ and short $\boldsymbol{y}_{i,j}$ given public parameters

Secret key for user $\boldsymbol{i}$: short vector $\boldsymbol{y}_{i,i}$ such that

$$\boldsymbol{A}\,\boldsymbol{y}_{i,i} = \boldsymbol{p} + \boldsymbol{B}\,\boldsymbol{r}_i + \boldsymbol{W}_i\,\boldsymbol{r}_i$$

"Cross-term" for distinct users $\boldsymbol{i}$ and $\boldsymbol{j}$: short vector $\boldsymbol{y}_{i,j}$ such that

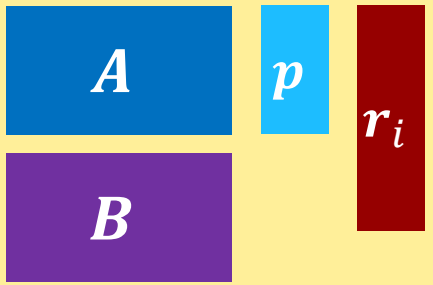
$$\boldsymbol{A}\,\boldsymbol{y}_{i,j} = \boldsymbol{W}_i\,\boldsymbol{r}_j$$

Distributed Key Generation

Challenge: No one can know a trapdoor for $\boldsymbol{A}$

Approach: User i will generate $\boldsymbol{W}_i$ and short $\boldsymbol{y}_{i,j}$ given public parameters

Public parameters

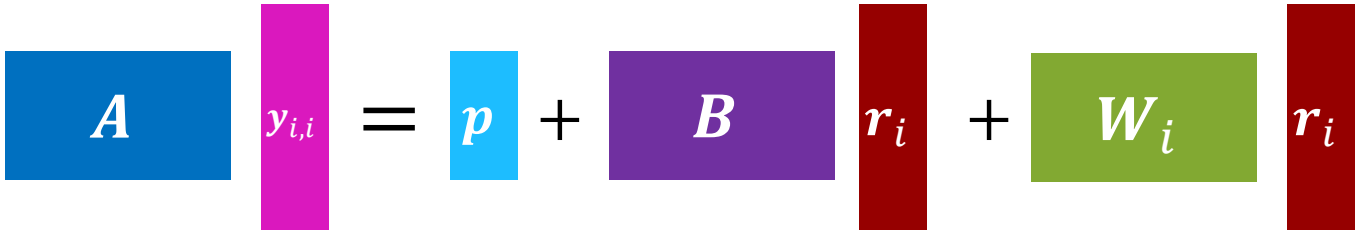


$i \in [\ell]$

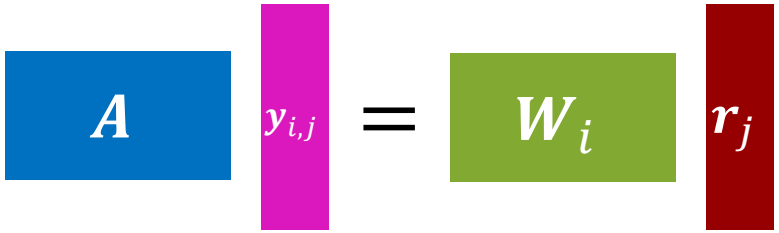
Public key for user $\boldsymbol{i}$:

$j \in [\ell]$
$j \neq i$

Secret key for user $\boldsymbol{i}$: short vector $\boldsymbol{y}_{i,i}$ such that

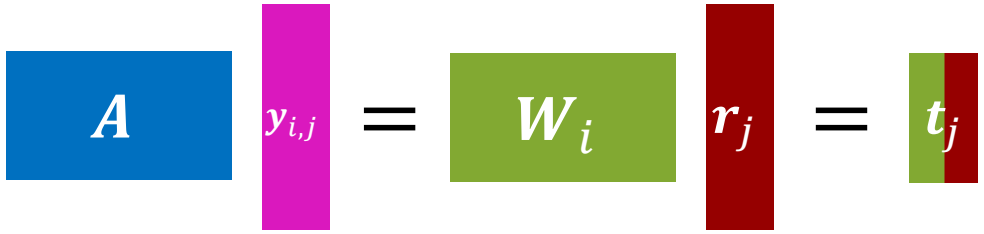


"Cross-term" for distinct users $\boldsymbol{i}$ and $\boldsymbol{j}$: short vector $\boldsymbol{y}_{i,j}$ such that



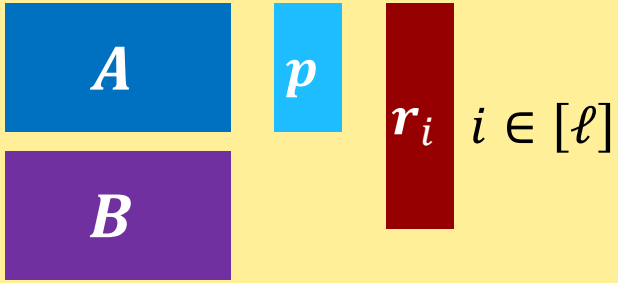
Distributed Key Generation

Goal: Generate $\boldsymbol{W}_i$ and $\boldsymbol{y}_{i,j}$ for $j \in [\ell]$ without a trapdoor for $\boldsymbol{A}$



Removed $\boldsymbol{p} + \boldsymbol{B}\boldsymbol{r}_i$ for simplicity

Public parameters



Distributed Key Generation

Goal: Generate $\boldsymbol{W}_i$ and $\boldsymbol{y}_{i,j}$ for $j \in [\ell]$ without a trapdoor for $\boldsymbol{A}$

$$\boldsymbol{A}\ \boldsymbol{y}_{i,j} = \boldsymbol{W}_i\ \boldsymbol{r}_j = \boldsymbol{t}_j$$

Removed $\boldsymbol{p} + \boldsymbol{B}\boldsymbol{r}_i$ for simplicity

Approach: Use a random trapdoor for a matrix $\boldsymbol{V}_\ell$ related to $\boldsymbol{A}$

Public parameters

$\boldsymbol{A}$ $\boldsymbol{p}$ $\boldsymbol{r}_i$ $i \in [\ell]$

$\boldsymbol{B}$ $\boldsymbol{V}_\ell$

Distributed Key Generation

Goal: Generate $\boldsymbol{W}_i$ and $\boldsymbol{y}_{i,j}$ for $j \in [\ell]$ without a trapdoor for $\boldsymbol{A}$

$$\boldsymbol{A}\ \boldsymbol{y}_{i,j} = \boldsymbol{W}_i\ \boldsymbol{r}_j = \boldsymbol{t}_j$$

Removed $\boldsymbol{p} + \boldsymbol{B}\boldsymbol{r}_i$ for simplicity

Public parameters

$\boldsymbol{A}$ $\boldsymbol{p}$ $\boldsymbol{r}_i$ $i \in [\ell]$

$\boldsymbol{B}$ $\boldsymbol{V}_\ell$

Approach: Use a random trapdoor for a matrix $\boldsymbol{V}_\ell$ related to $\boldsymbol{A}$

$$\underbrace{\begin{bmatrix} \boldsymbol{A} & & \\ & \ddots & \\ & & \boldsymbol{A} \end{bmatrix}}_{\boldsymbol{V}_\ell} \begin{bmatrix} \boldsymbol{y}_{i,1} \\ \vdots \\ \boldsymbol{y}_{i,\ell} \end{bmatrix} = \begin{bmatrix} \boldsymbol{t}_1 \\ \vdots \\ \boldsymbol{t}_\ell \end{bmatrix}$$

Distributed Key Generation

Goal: Generate $\boldsymbol{W}_i$ and $\boldsymbol{y}_{i,j}$ for $j \in [\ell]$ without a trapdoor for $\boldsymbol{A}$

$$\boldsymbol{A}\ \boldsymbol{y}_{i,j} = \boldsymbol{W}_i\ \boldsymbol{r}_j = \boldsymbol{t}_j$$

Removed $\boldsymbol{p} + \boldsymbol{B}\boldsymbol{r}_i$ for simplicity

Public parameters

$\boldsymbol{A}$ $\boldsymbol{p}$ $\boldsymbol{r}_i$ $i \in [\ell]$

$\boldsymbol{B}$ $\boldsymbol{V}_\ell$

Approach: Use a random trapdoor for a matrix $\boldsymbol{V}_\ell$ related to $\boldsymbol{A}$

$$\underbrace{\begin{bmatrix} \boldsymbol{A} & & \\ & \ddots & \\ & & \boldsymbol{A} \end{bmatrix}}_{\boldsymbol{V}_\ell} \begin{bmatrix} \boldsymbol{y}_{i,1} \\ \vdots \\ \boldsymbol{y}_{i,\ell} \end{bmatrix} = \begin{bmatrix} \boldsymbol{t}_1 \\ \vdots \\ \boldsymbol{t}_\ell \end{bmatrix}$$

Trapdoor for $\boldsymbol{V}_\ell$ used to sample short solutions to this equation

Distributed Key Generation

Goal: Generate $\boldsymbol{W}_i$ and $\boldsymbol{y}_{i,j}$ for $j \in [\ell]$ without a trapdoor for $\boldsymbol{A}$

$$\boldsymbol{A}\ \boldsymbol{y}_{i,j} = \boldsymbol{W}_i\ \boldsymbol{r}_j = \boldsymbol{t}_j$$

Removed $\boldsymbol{p} + \boldsymbol{B}\boldsymbol{r}_i$ for simplicity

Public parameters

$\boldsymbol{A}$ $\boldsymbol{p}$ $\boldsymbol{r}_i$ $i \in [\ell]$

$\boldsymbol{B}$ $\boldsymbol{V}_\ell$

Approach: Use a random trapdoor for a matrix $\boldsymbol{V}_\ell$ related to $\boldsymbol{A}$

Each block could be **any** vector $\boldsymbol{h}$!

$$\begin{bmatrix} \boldsymbol{A} & & \\ & \ddots & \\ & & \boldsymbol{A} \end{bmatrix} \begin{bmatrix} \boldsymbol{y}_{i,1} \\ \vdots \\ \boldsymbol{y}_{i,\ell} \end{bmatrix} = \begin{bmatrix} \boldsymbol{t}_1 \\ \vdots \\ \boldsymbol{t}_\ell \end{bmatrix}$$

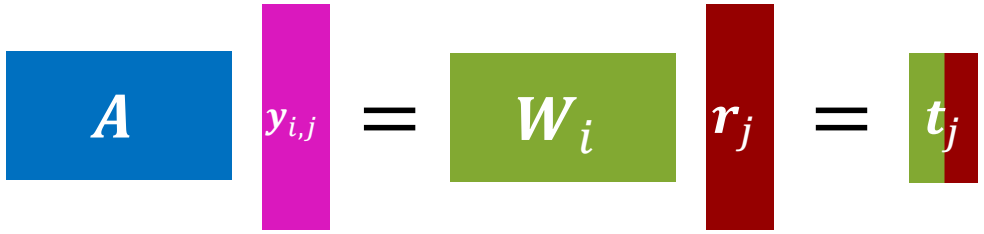
$\boldsymbol{V}_\ell$

Trapdoor for $\boldsymbol{V}_\ell$ leaks trapdoor for $\boldsymbol{A}$!

Trapdoor for $\boldsymbol{V}_\ell$ used to sample short solutions to this equation

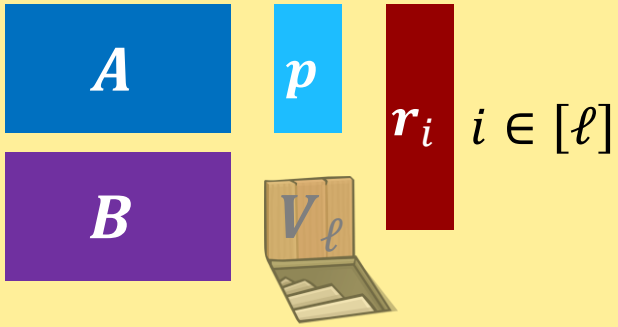
Distributed Key Generation

Goal: Generate $\boldsymbol{W}_i$ and $\boldsymbol{y}_{i,j}$ for $j \in [\ell]$ without a trapdoor for $\boldsymbol{A}$



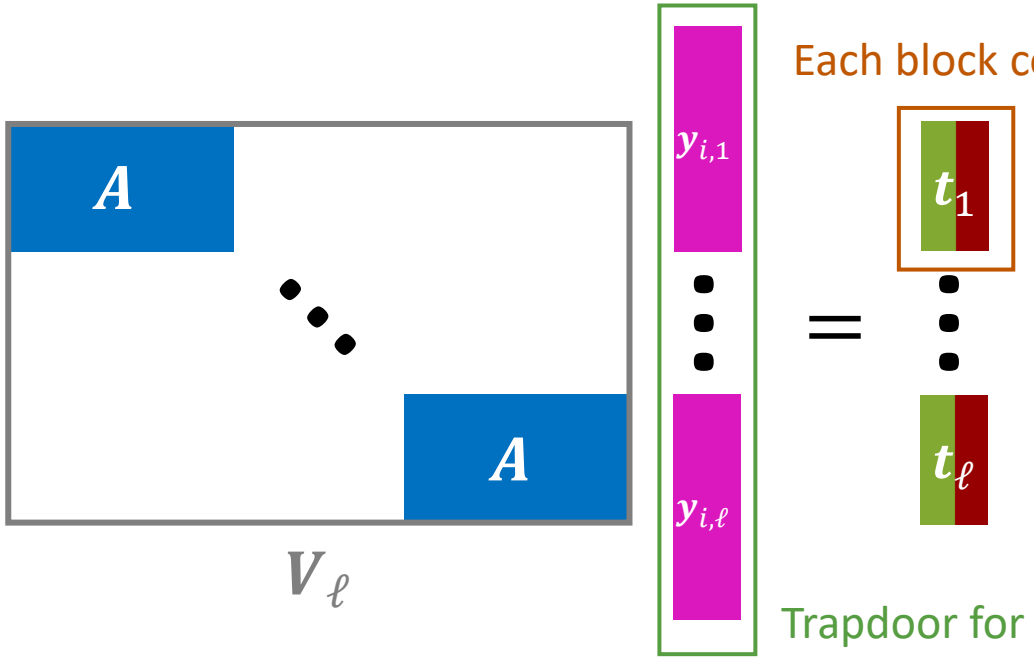
Removed $\boldsymbol{p} + \boldsymbol{B}\boldsymbol{r}_i$ for simplicity

Public parameters



Approach: Use a random trapdoor for a matrix $\boldsymbol{V}_\ell$ related to $\boldsymbol{A}$

Each block could be **any** vector $\boldsymbol{h}$!



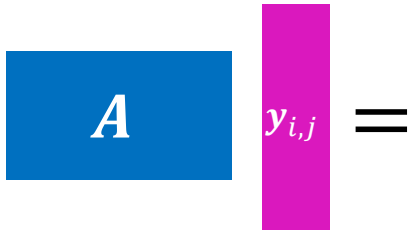
Trapdoor for $\boldsymbol{V}_\ell$ leaks trapdoor for $\boldsymbol{A}$!

Want $\boldsymbol{A}\boldsymbol{y}_{i,j} = \boldsymbol{W}_i\boldsymbol{r}_j + \boldsymbol{h}$ for any choice of $\boldsymbol{h}$ on right

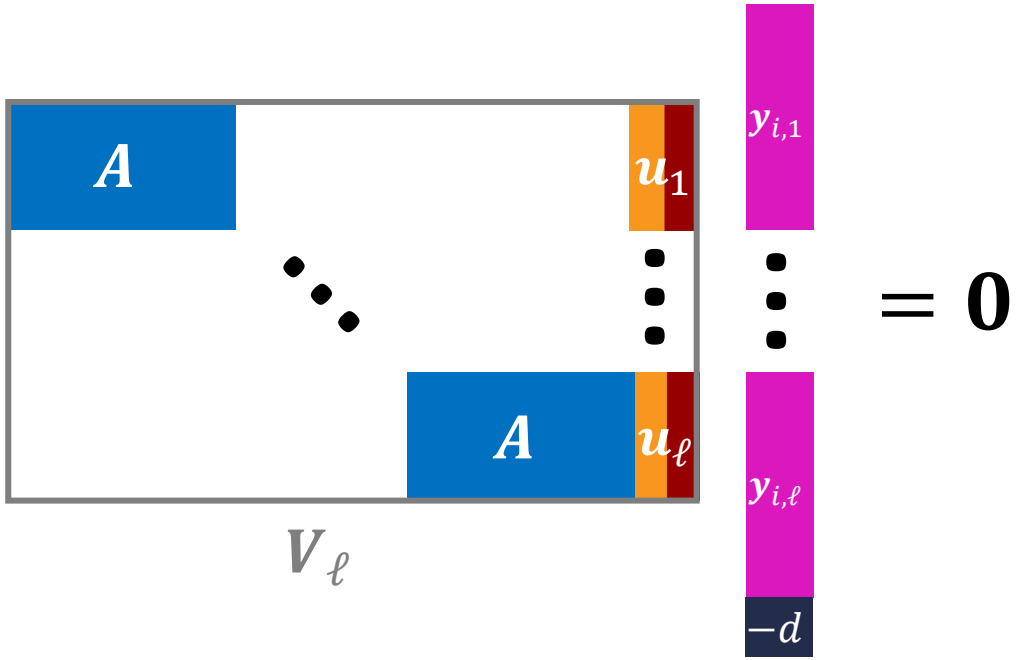
Trapdoor for $\boldsymbol{V}_\ell$ used to sample short solutions to this equation

Distributed Key Generation

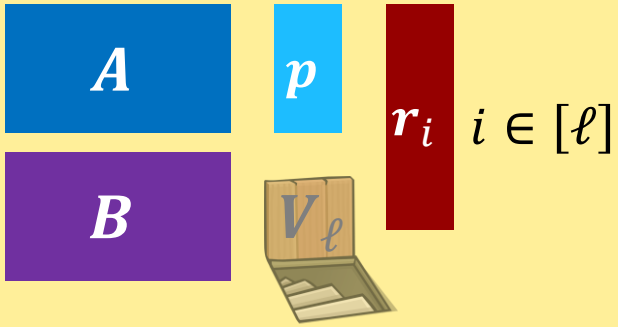
Goal: Generate W_i and $y_{i,j}$ for $j \in [\ell]$ without a trapdoor for A



Approach: Use a random trapdoor for a matrix V_ℓ related to A

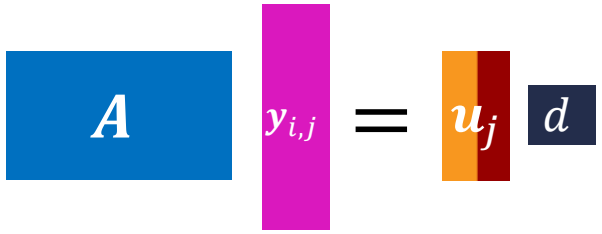


Public parameters

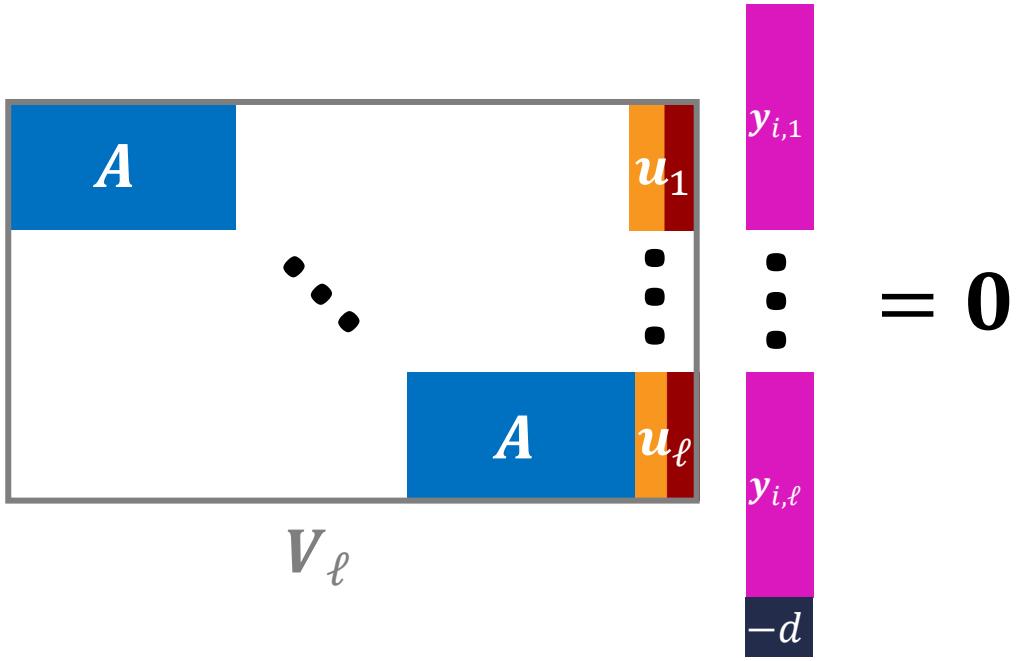


Distributed Key Generation

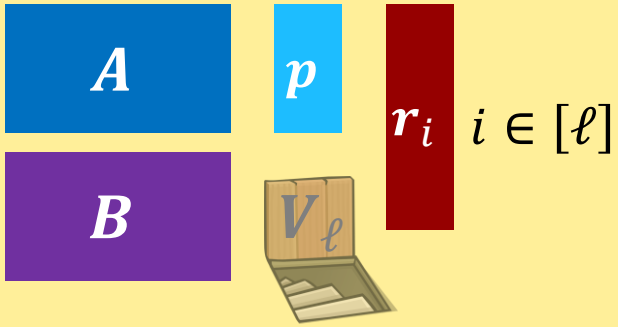
Goal: Generate W_i and $y_{i,j}$ for $j \in [\ell]$ without a trapdoor for A



Approach: Use a random trapdoor for a matrix V_ℓ related to A

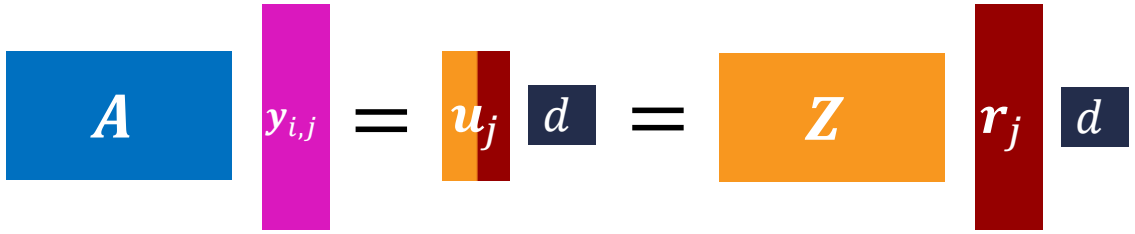


Public parameters

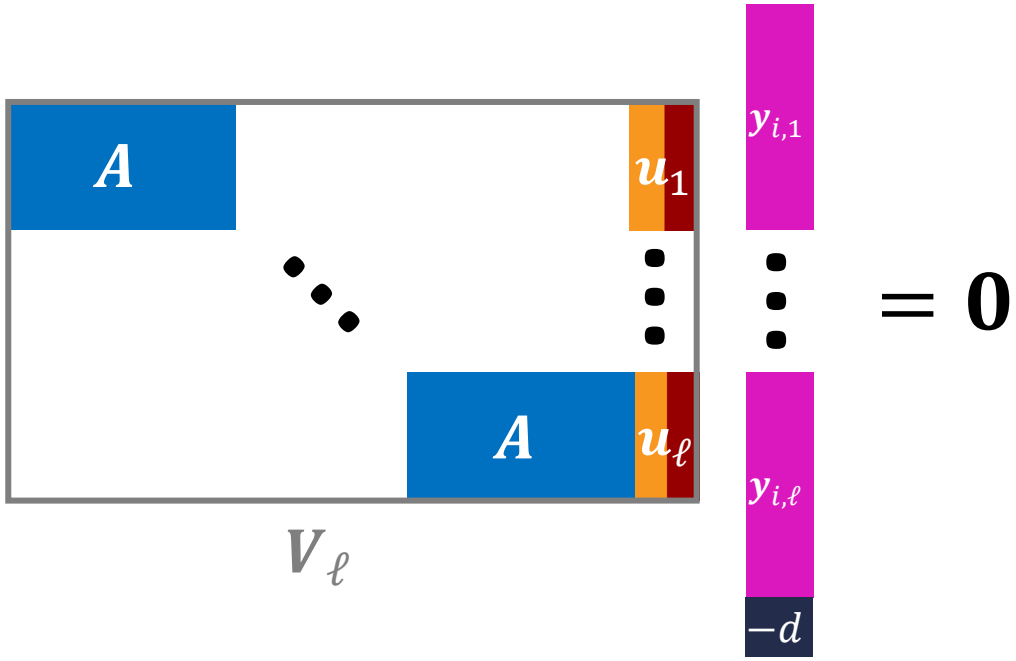


Distributed Key Generation

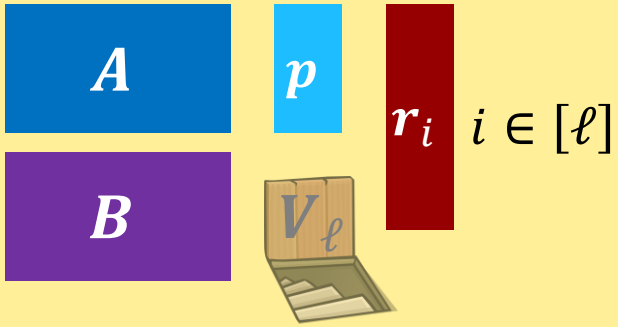
Goal: Generate W_i and $y_{i,j}$ for $j \in [\ell]$ without a trapdoor for A



Approach: Use a random trapdoor for a matrix V_ℓ related to A



Public parameters



Distributed Key Generation

Goal: Generate W_i and $y_{i,j}$ for $j \in [\ell]$ without a trapdoor for A

$$A \; y_{i,j} = u_j \; d = Z \; r_j \; d = Z \; d \; r_j$$

Approach: Use a random trapdoor for a matrix V_ℓ related to A

$$\begin{bmatrix} A & & & u_1 \\ & \ddots & & \vdots \\ & & A & u_\ell \end{bmatrix} \begin{bmatrix} y_{i,1} \\ \vdots \\ y_{i,\ell} \\ -d \end{bmatrix} = 0$$

V_ℓ

Public parameters

A p r_i $i \in [\ell]$

B V_ℓ

Distributed Key Generation

Goal: Generate W_i and $y_{i,j}$ for $j \in [\ell]$ without a trapdoor for A

$$A \; y_{i,j} = u_j \; d = Z \; r_j \; d = Z \; d \; r_j$$

Approach: Use a random trapdoor for a matrix V_ℓ related to A

$$\begin{bmatrix} A & & & u_1 \\ & \ddots & & \vdots \\ & & A & u_\ell \end{bmatrix} \begin{bmatrix} y_{i,1} \\ \vdots \\ y_{i,\ell} \\ -d \end{bmatrix} = 0$$

V_ℓ

Set $W_i = Z \; d$

Public parameters

A p r_i $i \in [\ell]$

B V_ℓ

Distributed Key Generation

Goal: Generate W_i and $y_{i,j}$ for $j \in [\ell]$ without a trapdoor for A

$$A \; y_{i,j} = u_j \; d = Z \; r_j \; d = Z \; d \; r_j$$

Approach: Use a random trapdoor for a matrix V_ℓ related to A

$$\begin{bmatrix} A & & & u_1 \\ & \ddots & & \vdots \\ & & A & u_\ell \end{bmatrix} \begin{bmatrix} y_{i,1} \\ \vdots \\ y_{i,\ell} \\ -d \end{bmatrix} = 0$$

V_ℓ

Set $W_i = Z \; d$

Small number of possible W_i ⇒ high chance of collisions!

Public parameters

A p r_i $i \in [\ell]$

B V_ℓ

Distributed Key Generation

Goal: Generate W_i and $y_{i,j}$ for $j \in [\ell]$ without a trapdoor for A

$$A \; y_{i,j} = u_j \; d = Z \; r_j \; d = Z \; d \; r_j$$

Approach: Use a random trapdoor for a matrix V_ℓ related to A

$$\begin{bmatrix} A & & & u_1 \\ & \ddots & & \vdots \\ & & A & u_\ell \end{bmatrix} \begin{bmatrix} y_{i,1} \\ \vdots \\ y_{i,\ell} \\ -d \end{bmatrix} = 0$$

V_ℓ

Set $W_i = Z \; d$

Small number of possible $W_i \Rightarrow$ high chance of collisions!

Solution: generate many independent Z_j, add column in V_ℓ for each, and set $W_i = \sum Z_j \cdot d_j$

Public parameters

A p r_i $i \in [\ell]$

B V_ℓ

Distributed Key Generation

Goal: Generate W_i and $y_{i,j}$ for $j \in [\ell]$ without a trapdoor for A

$$A \; y_{i,j} = u_j \; d = Z \; r_j \; d = Z \; d \; r_j$$

Approach: Use a random trapdoor for a matrix V_ℓ related to A

$$\begin{bmatrix} A & & & u_1 \\ & \ddots & & \vdots \\ & & A & u_\ell \end{bmatrix} \begin{bmatrix} y_{i,1} \\ \vdots \\ y_{i,\ell} \\ -d \end{bmatrix} = 0$$

V_ℓ

Set $W_i = Z \; d$

Small number of possible $W_i \Rightarrow$ high chance of collisions!

Solution: generate many independent Z_j, add column in V_ℓ for each, and set $W_i = \sum Z_j \cdot d_j$

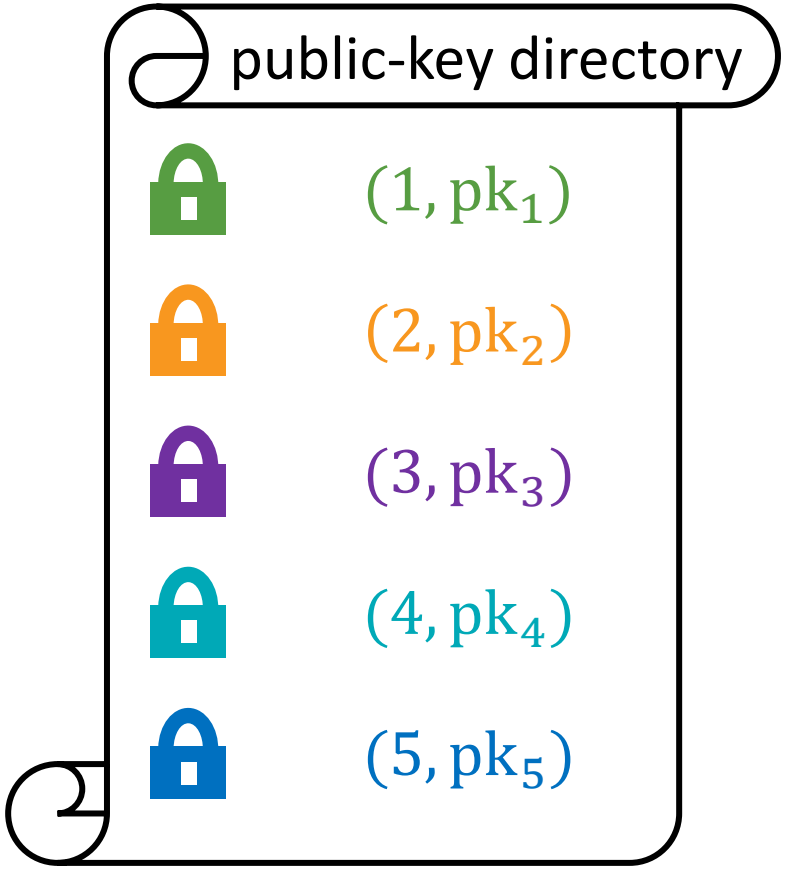
With enough Z_j, trapdoor for V_ℓ can be derived from ℓ'-succinct LWE trapdoor!

Public parameters

A p r_i $i \in [\ell]$

B V_ℓ

Summary



Broadcast encryption without a central authority

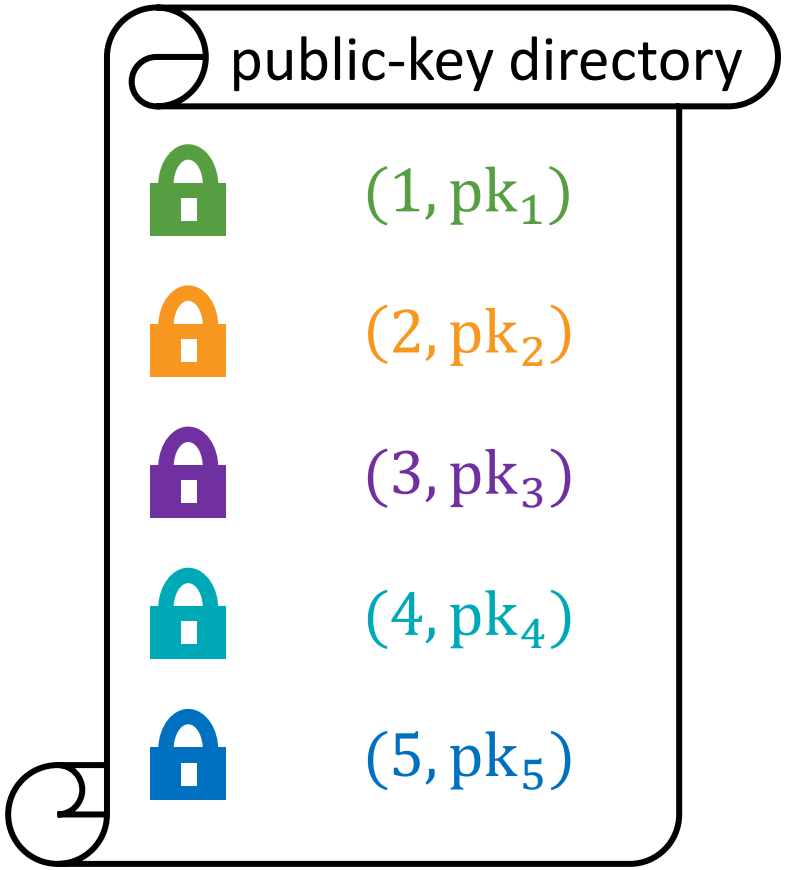
Selectively-secure distributed broadcast encryption for ℓ users from ℓ'-succinct LWE where $\ell' \geq \ell \cdot O(\lambda \log \ell)$

Public parameter size: $\ell^2 \cdot \mathrm{poly}(\lambda, \log \ell)$

User public key size: $\ell \cdot \mathrm{poly}(\lambda, \log \ell)$

Ciphertext size: $\mathrm{poly}(\lambda, \log \ell)$

Summary



Broadcast encryption without a central authority

Selectively-secure distributed broadcast encryption for ℓ users from ℓ'-succinct LWE where $\ell' \geq \ell \cdot O(\lambda \log \ell)$

Public parameter size: $\ell^2 \cdot \mathrm{poly}(\lambda, \log \ell)$

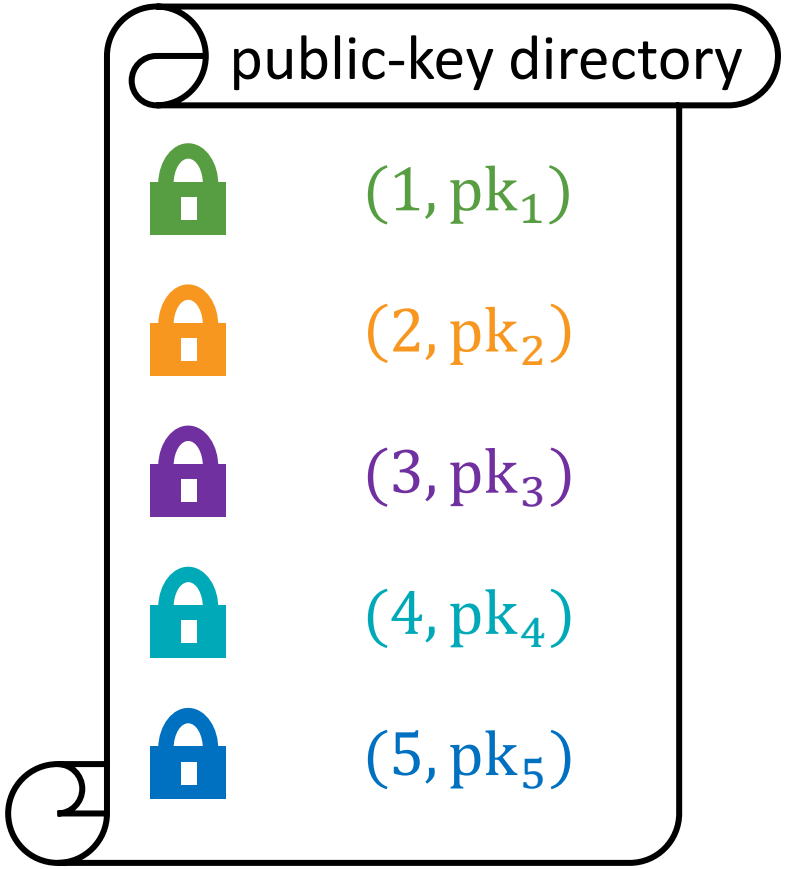
User public key size: $\ell \cdot \mathrm{poly}(\lambda, \log \ell)$

Ciphertext size: $\mathrm{poly}(\lambda, \log \ell)$

Open problems:

- Proving security from plain LWE
- Cryptanalysis and more applications of ℓ-succinct LWE

Summary



Broadcast encryption without a central authority

Selectively-secure distributed broadcast encryption for ℓ users from ℓ'-succinct LWE where $\ell' \geq \ell \cdot O(\lambda \log \ell)$

Public parameter size: $\ell^2 \cdot \mathrm{poly}(\lambda, \log \ell)$

User public key size: $\ell \cdot \mathrm{poly}(\lambda, \log \ell)$

Ciphertext size: $\mathrm{poly}(\lambda, \log \ell)$

Open problems:

- Proving security from plain LWE
- Cryptanalysis and more applications of ℓ-succinct LWE

Upcoming work: registered ABE for circuits from ℓ-succinct LWE in the random oracle model

- Requires simulating challenge ciphertexts w.r.t *malicious* keys
- Techniques generalize to obtain *adaptively-secure* DBE

Thanks for listening!

https://eprint.iacr.org/2024/1417