Tutorial Talk: Certified Deletion

James Bartusek

UC Berkeley

Outline

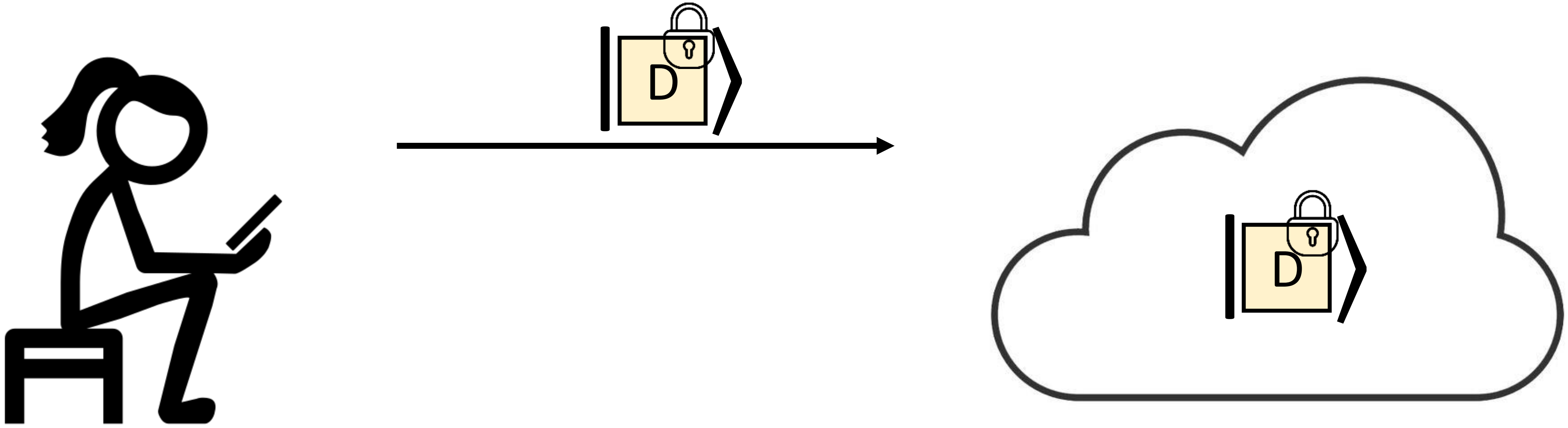
1. Basic scenario and applications
2. Recipe for constructions
3. Security
4. Certifiable deletion of programs

Outline

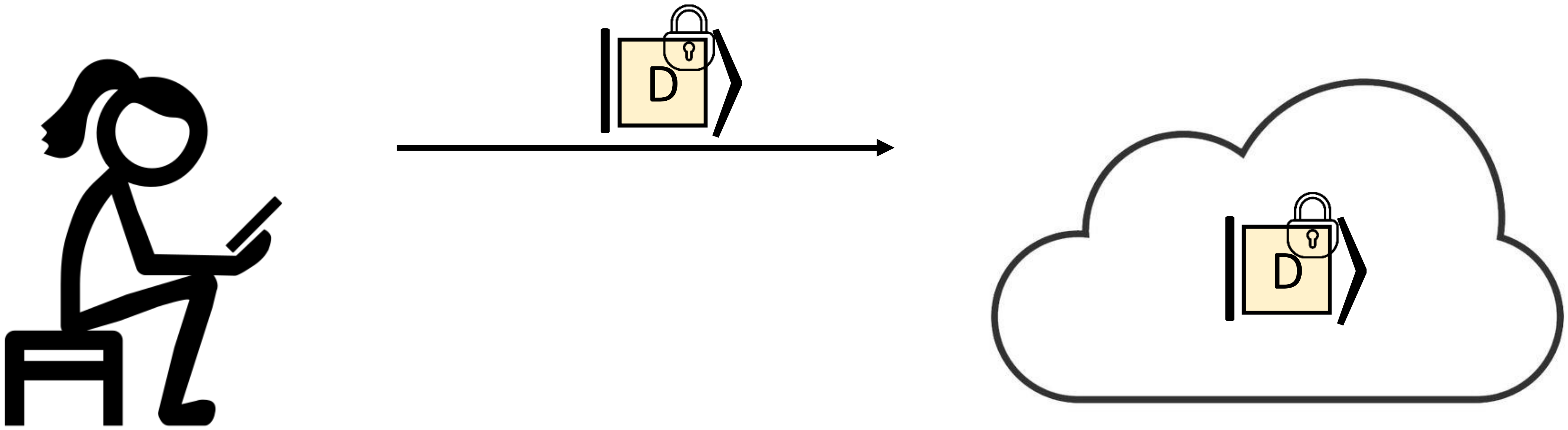
1. Basic scenario and applications
2. Recipe for constructions
3. Security
4. Certifiable deletion of programs

Certified Deletion: Cloud Storage

Certified Deletion: Cloud Storage

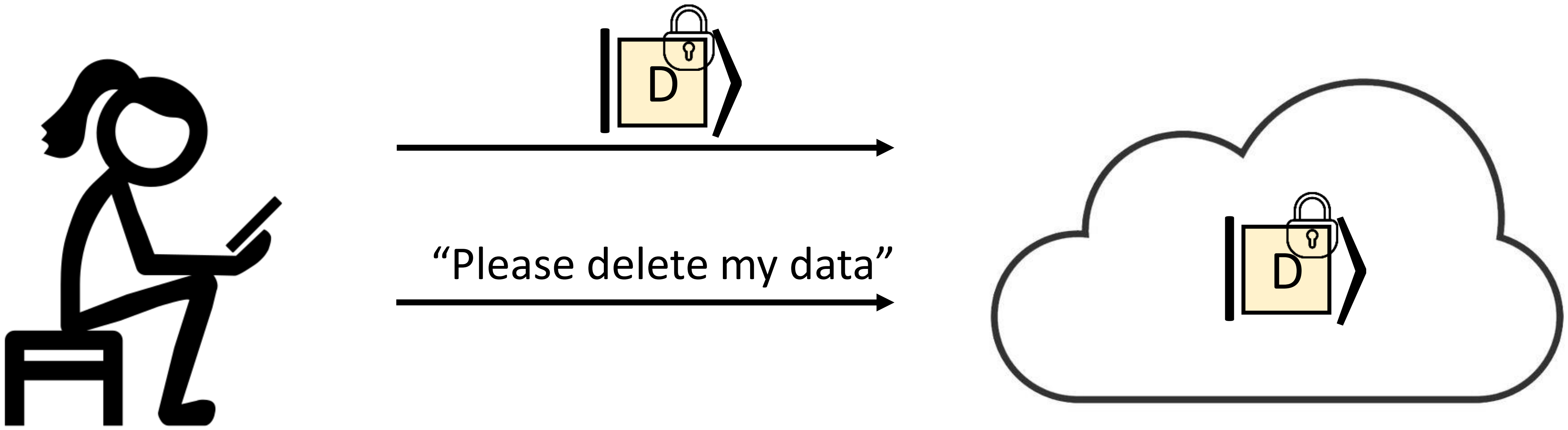


Certified Deletion: Cloud Storage



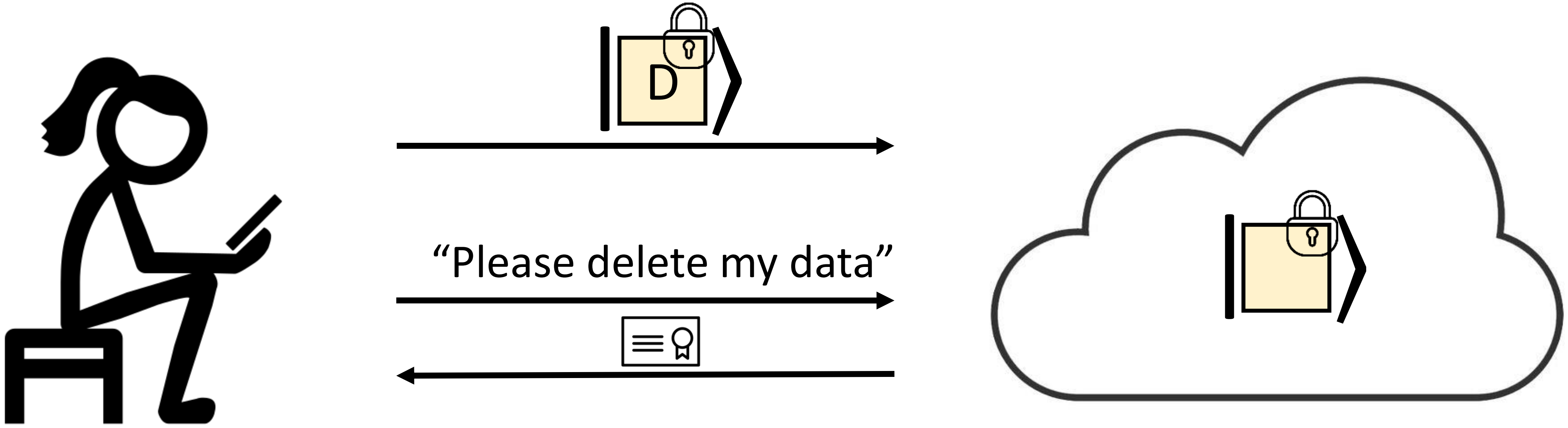
- Assumption: Malicious server cannot recover D from the encoding in polynomial time

Certified Deletion: Cloud Storage



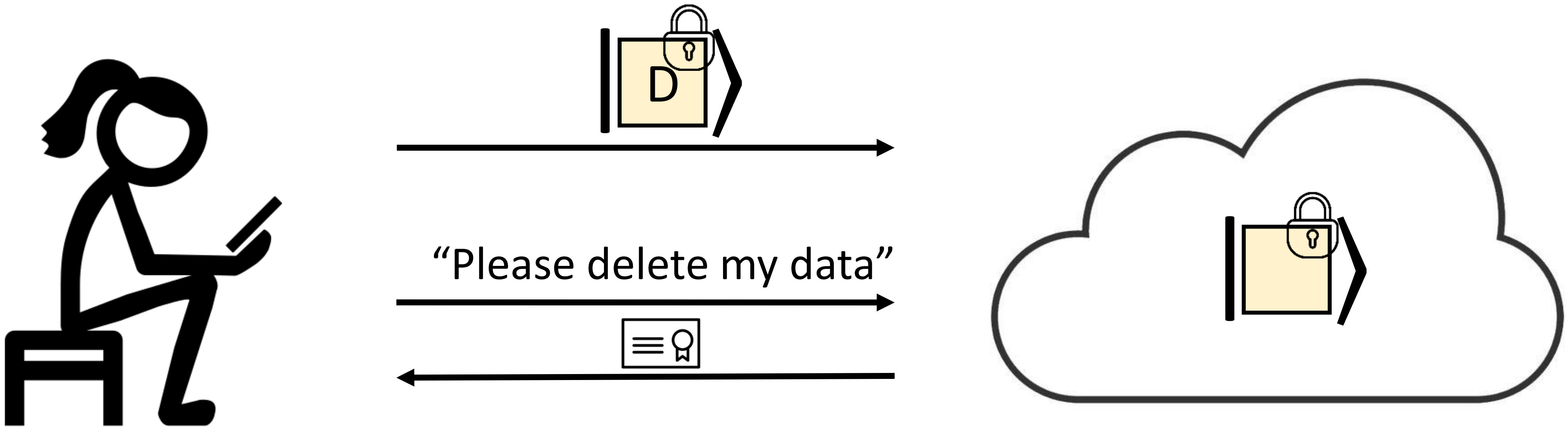
- Assumption: Malicious server cannot recover D from the encoding in polynomial time

Certified Deletion: Cloud Storage



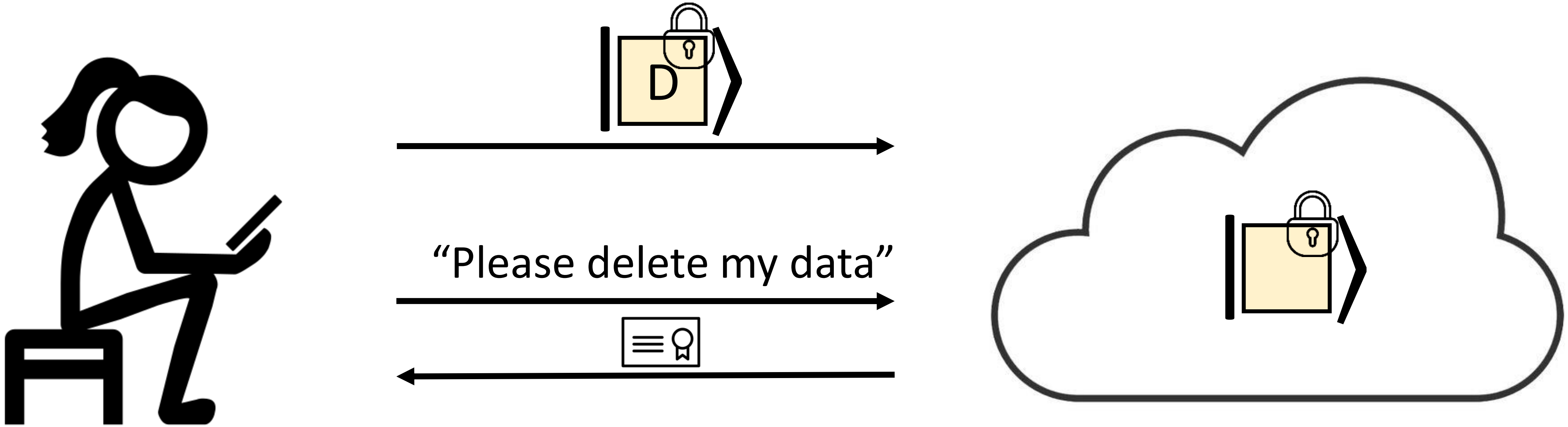
- Assumption: Malicious server cannot recover D from the encoding in polynomial time

Certified Deletion: Cloud Storage



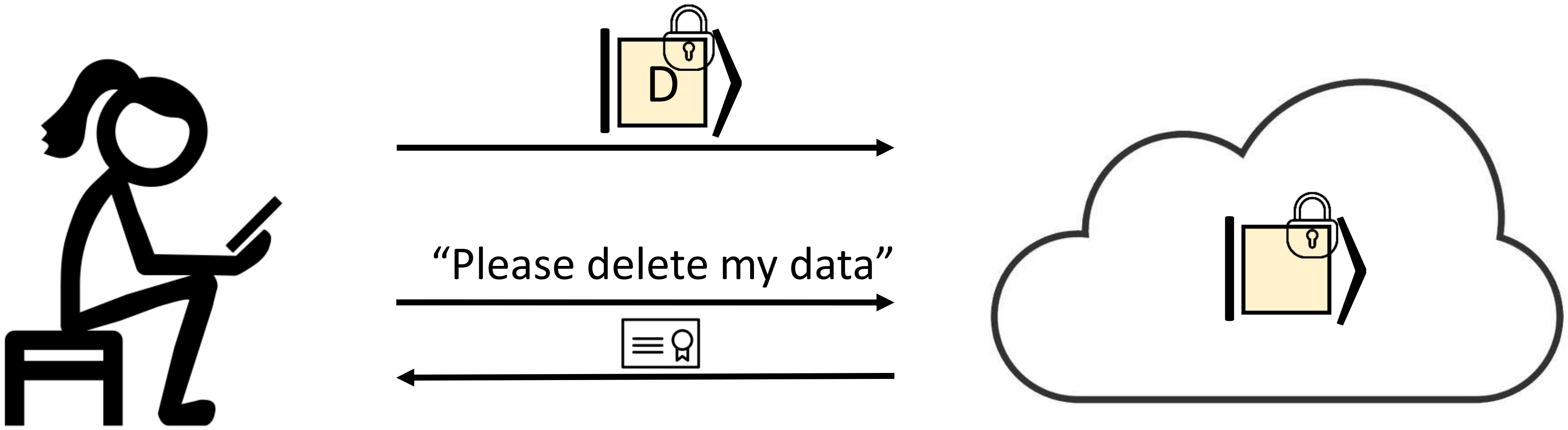
- Assumption: Malicious server cannot recover D from the encoding in polynomial time
- Goal #1: After deletion, the server won't be able to recover D even given 

Certified Deletion: Cloud Storage



- Assumption: Malicious server cannot recover D from the encoding in polynomial time
- Goal #1: After deletion, the server won’t be able to recover D even given 
- Goal #2: After deletion, the server won’t be able to recover D even given unbounded time

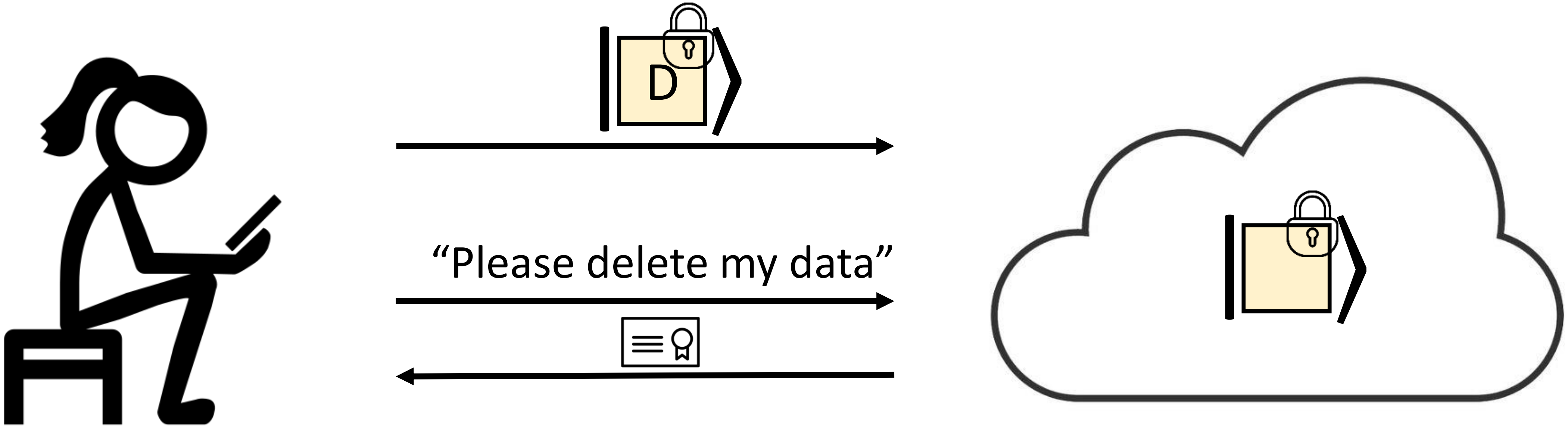
Certified Deletion: Cloud Storage



- Assumption: Malicious server cannot recover D from the encoding in polynomial time
- Goal #1: After deletion, the server won't be able to recover D even given 
- Goal #2: After deletion, the server won't be able to recover D even given unbounded time
- Requirements: encryption + unclonability

[Broadbent, Islam 20]
[Hiroka, Morimae, Nishimaki, Yamakawa 21]

Certified Deletion: Cloud Storage

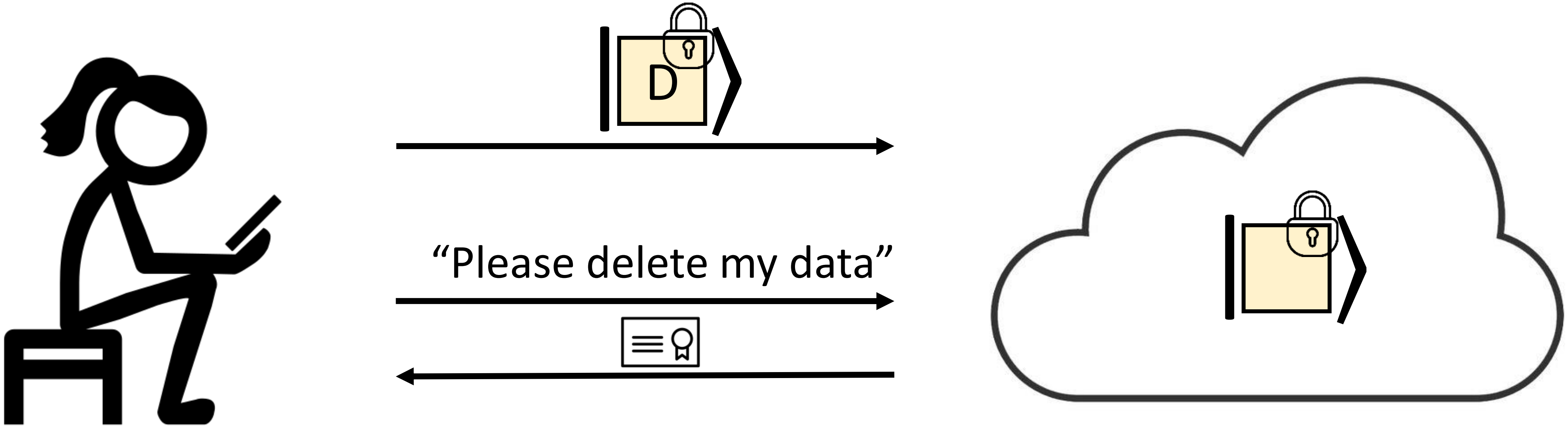


- Assumption: Malicious server cannot recover D from the encoding in polynomial time
- Goal #1: After deletion, the server won’t be able to recover D even given
- Goal #2: After deletion, the server won’t be able to recover D even given unbounded time
- Requirements: encryption + unclonability

Certified Deletion: Cloud Storage

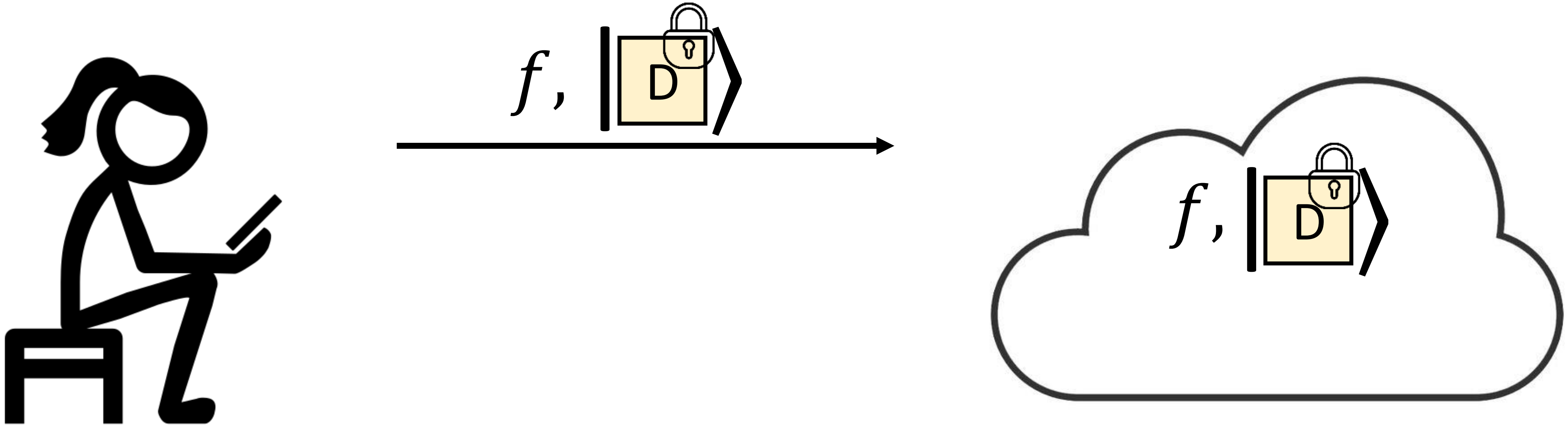
[Broadbent, Islam 20]
[Hiroka, Morimae, Nishimaki, Yamakawa 21]

Classically: [Garg, Goldwasser, Vasudevan 20]

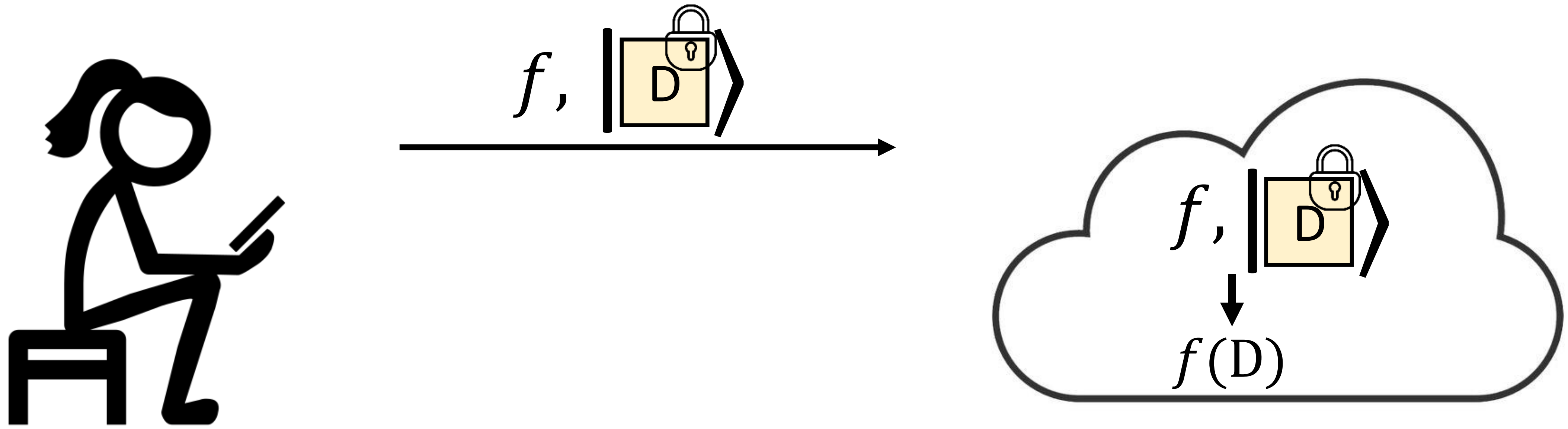


- Assumption: Malicious server cannot recover D from the encoding in polynomial time
- Goal #1: After deletion, the server won't be able to recover D even given
- Goal #2: After deletion, the server won't be able to recover D even given unbounded time
- Requirements: encryption + unclonability

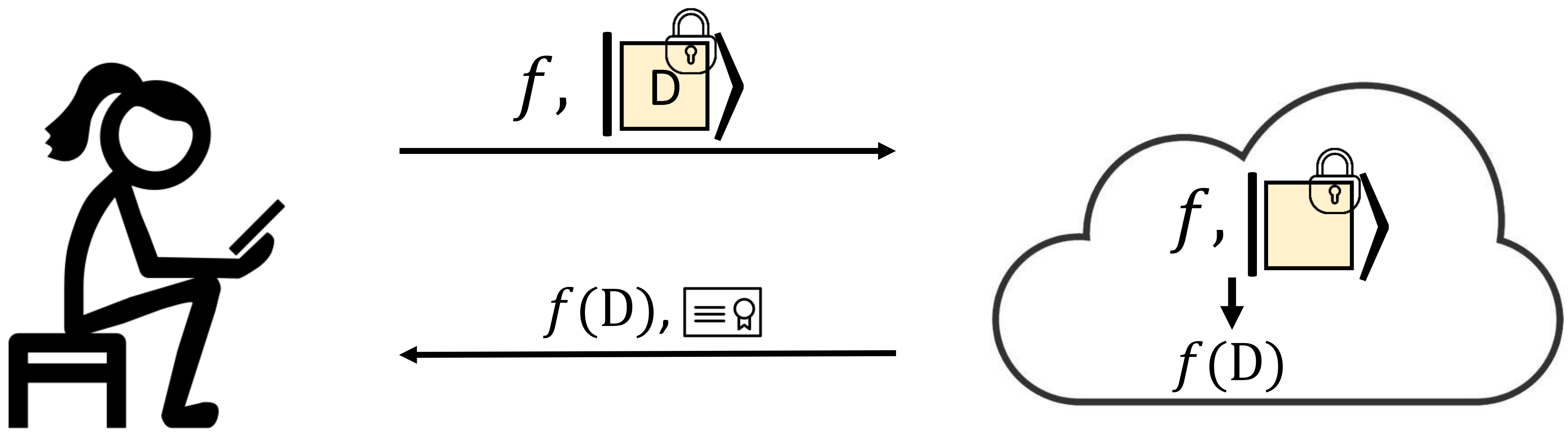
Certified Deletion: Delegation



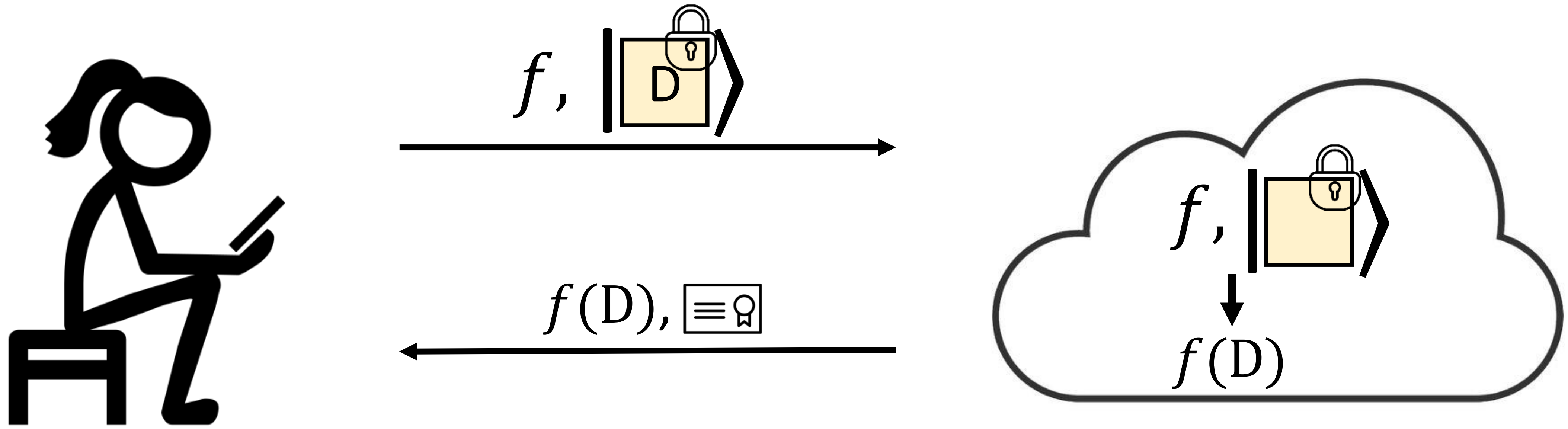
Certified Deletion: Delegation



Certified Deletion: Delegation



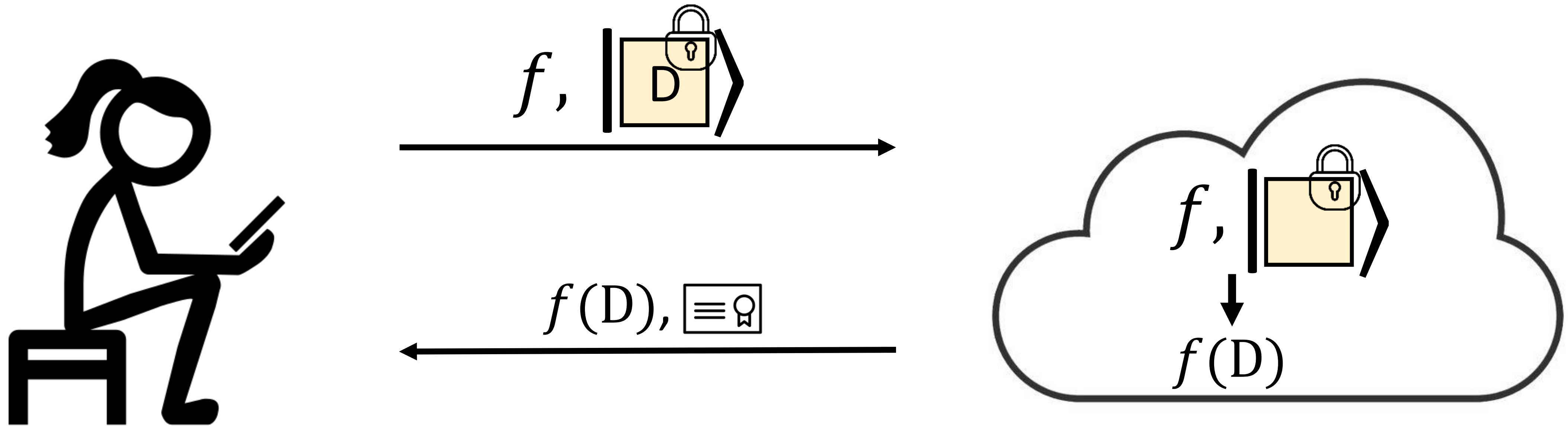
Certified Deletion: Delegation



- Server can compute and return $f(\mathrm{D})$ along with a proof [certificate icon] that they erased *all other* information about D

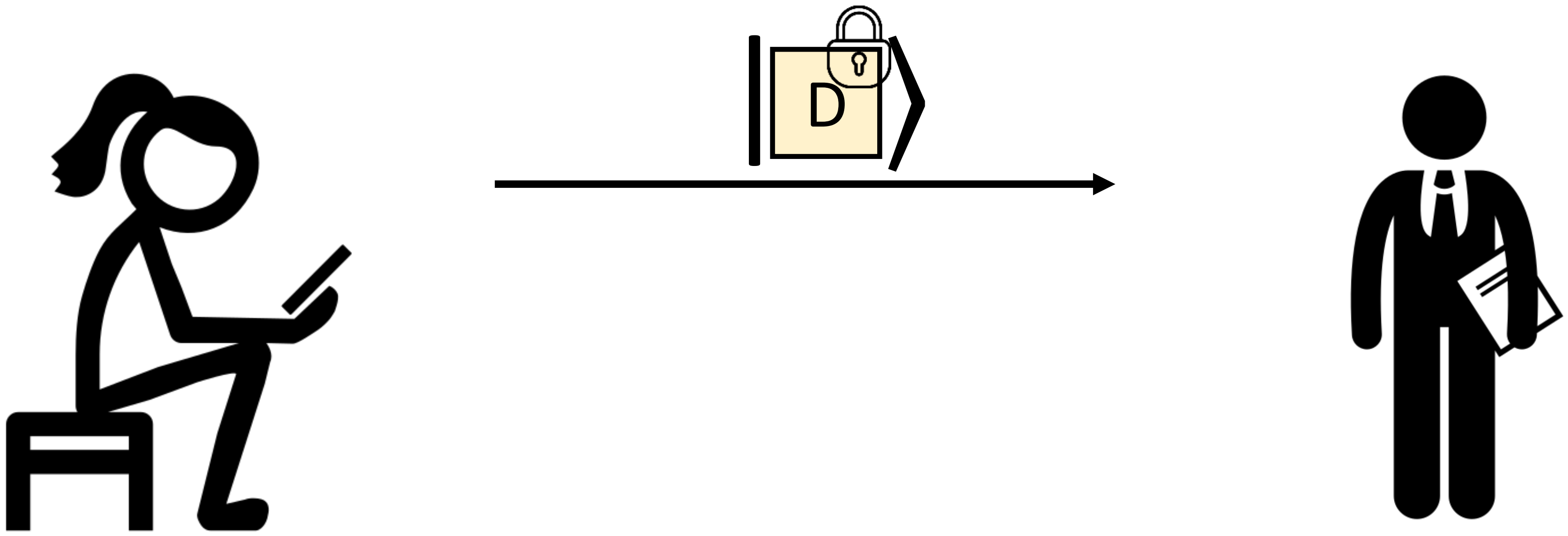
Certified Deletion: Delegation

[Broadbent, Islam 20]
[Poremba 23]
[**B**, Garg, Goyal, Khurana, Malavolta, Raizes, Roberts 23]

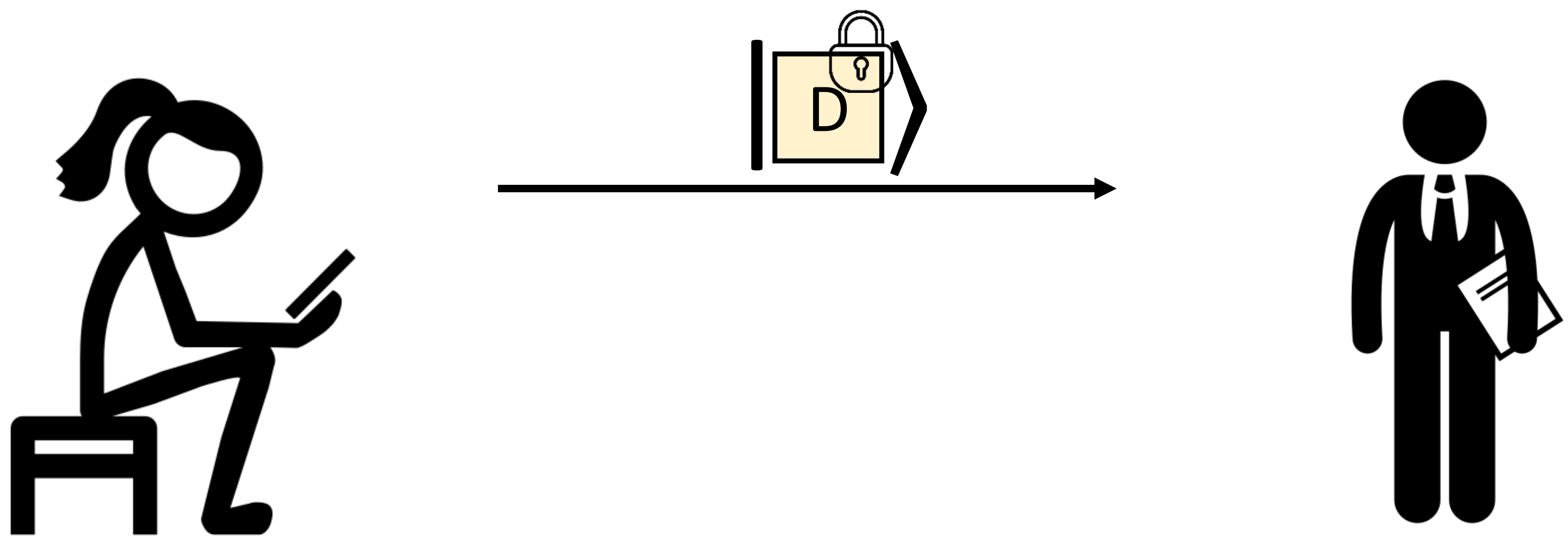


- Server can compute and return $f(\mathrm{D})$ along with a proof that they erased *all other* information about D

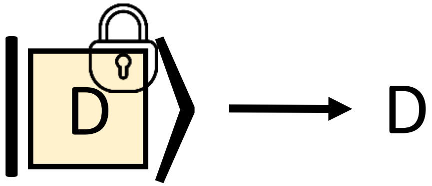
Certified Deletion: Timed-Release Encryption



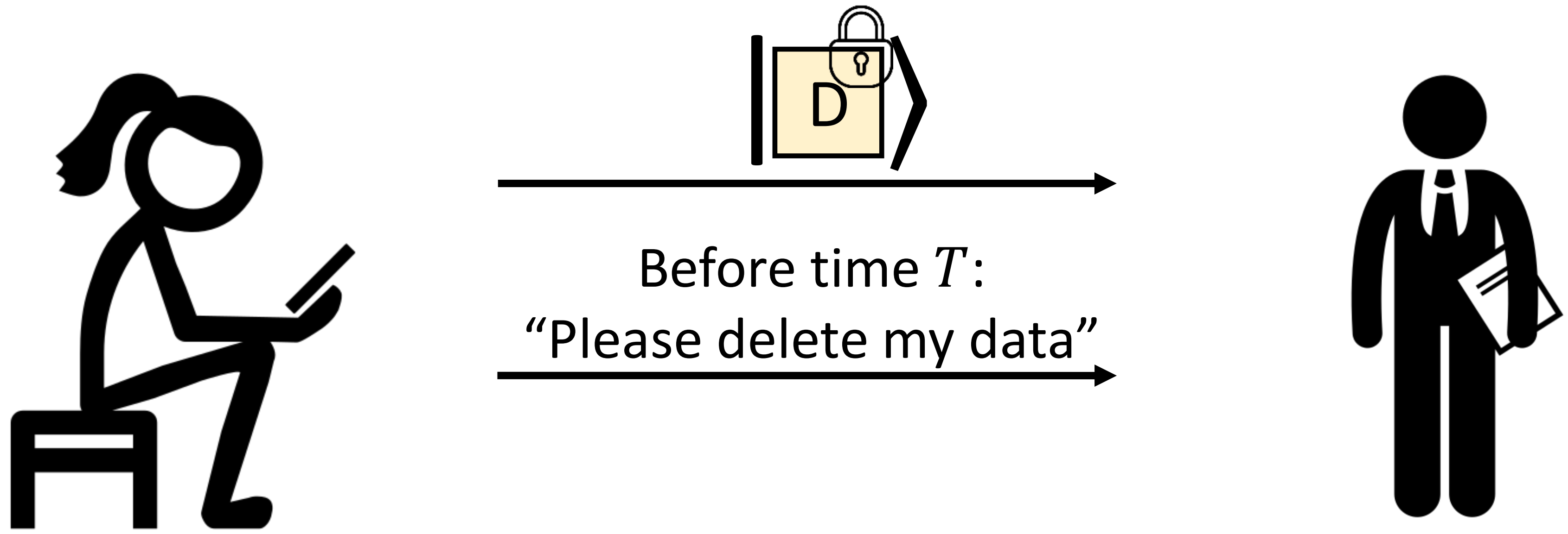
Certified Deletion: Timed-Release Encryption



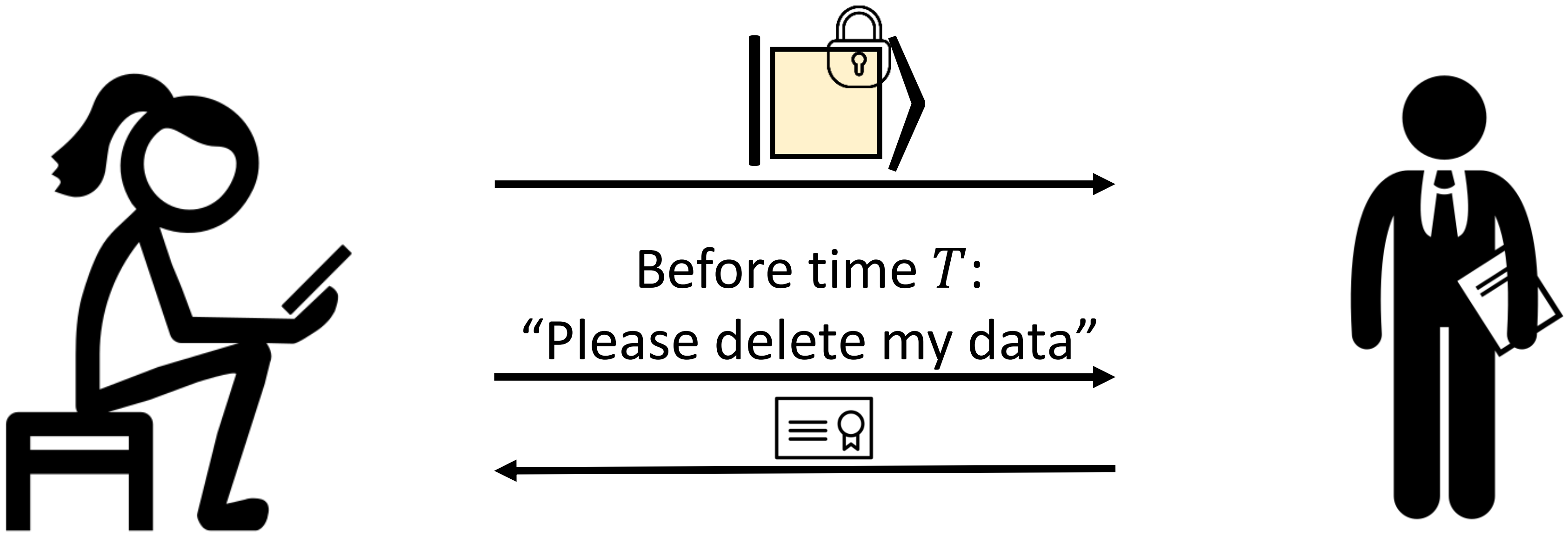
After time T:



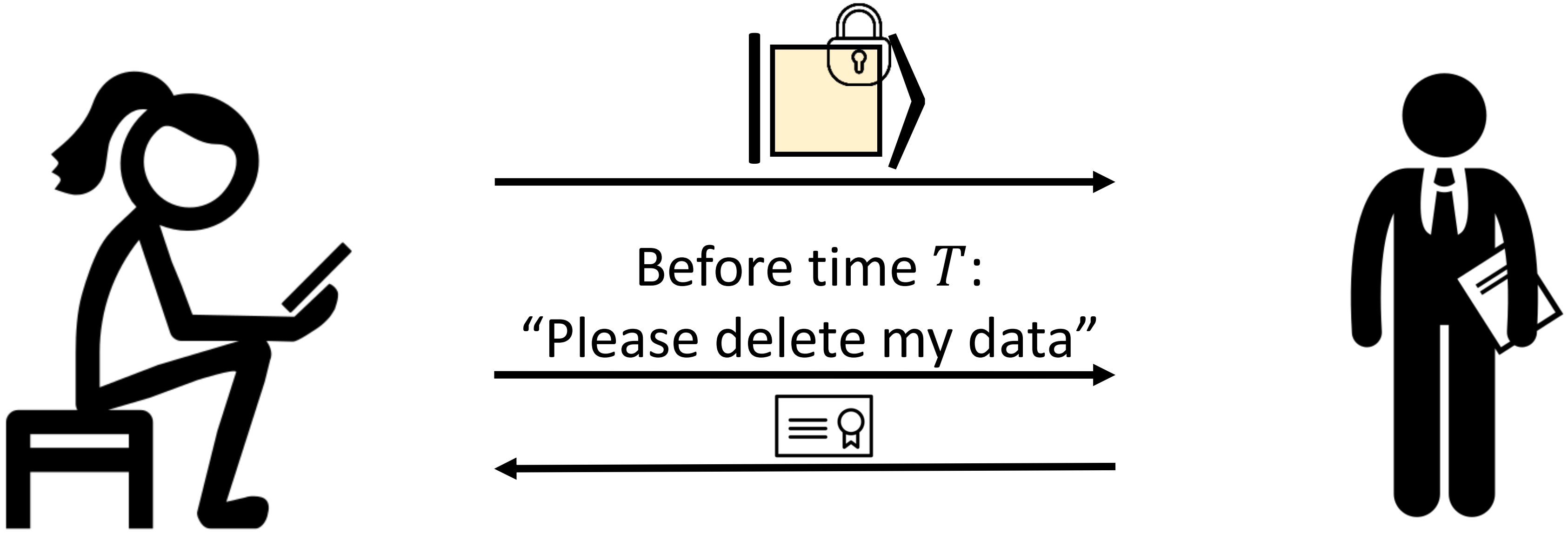
Certified Deletion: Timed-Release Encryption



Certified Deletion: Timed-Release Encryption

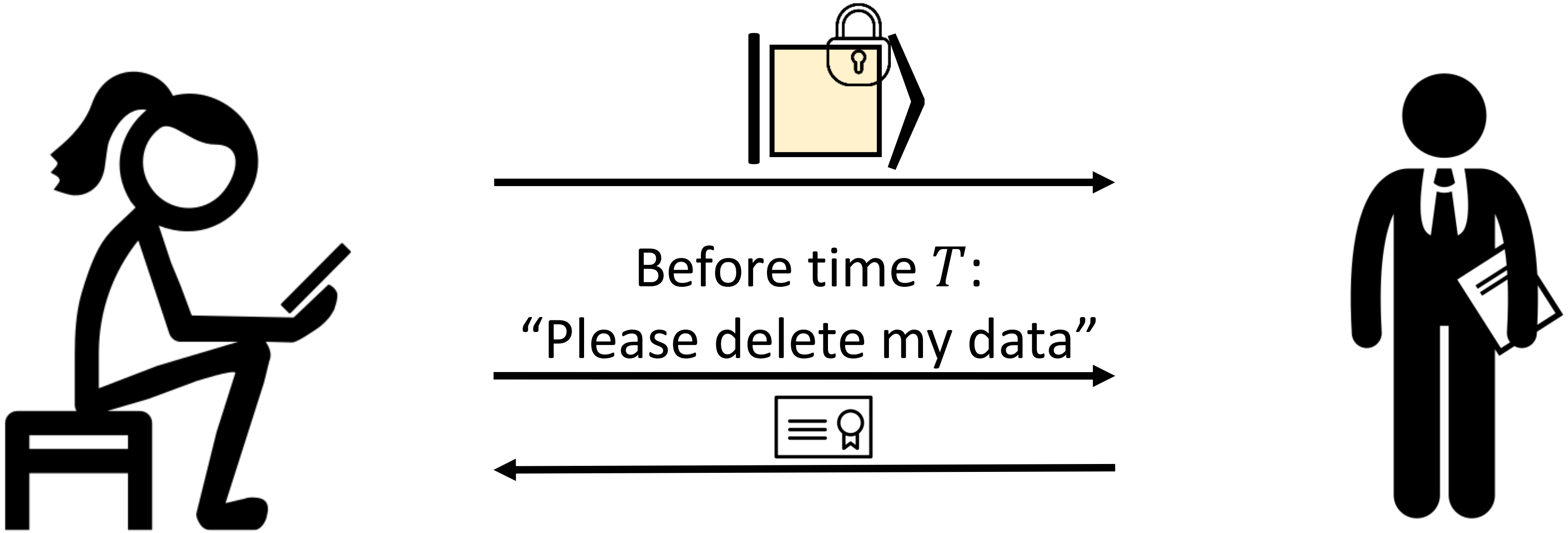


Certified Deletion: Timed-Release Encryption



- Wills

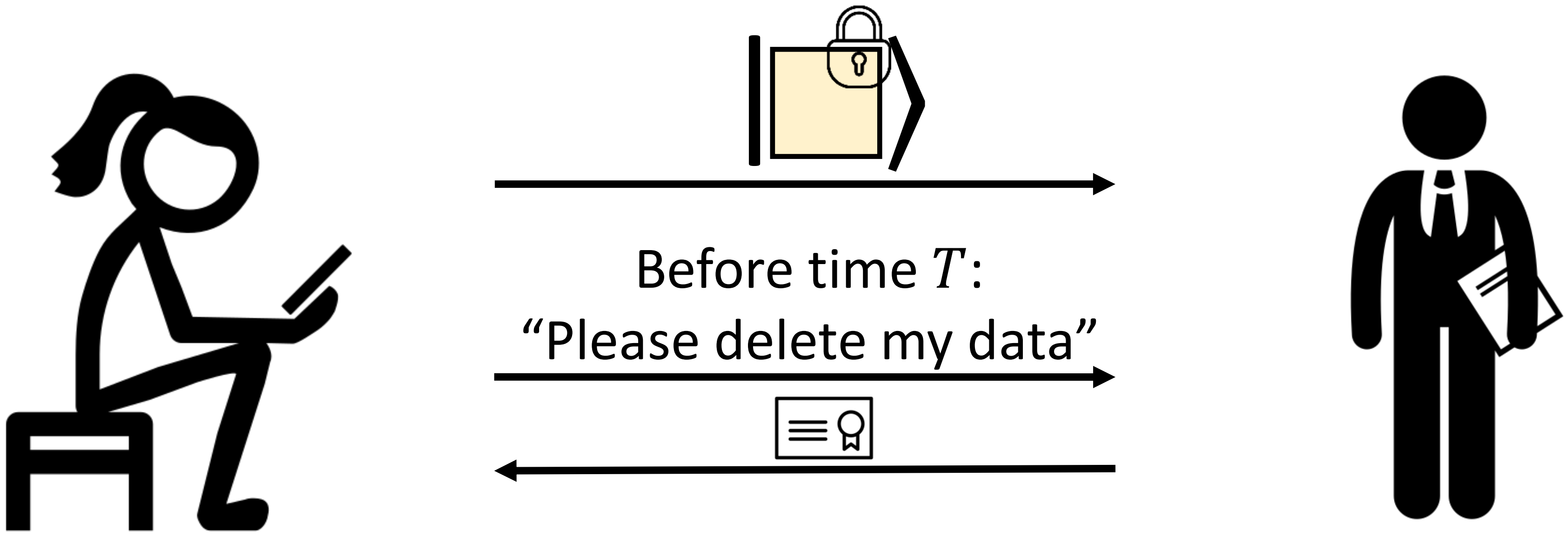
Certified Deletion: Timed-Release Encryption



- Wills
- Deposits

Certified Deletion: Timed-Release Encryption

[Unruh 13]
[**B**, Khurana 23]



- Wills
- Deposits

Outline

1. Basic scenario and applications
2. Recipe for constructions
3. Security
4. Certifiable deletion of programs

Approach

Approach

- Modularize: think about the quantum information and crypto components separately

Approach

- Modularize: think about the quantum information and crypto components separately

- Take advantage of the uncertainty principle

Approach

- Modularize: think about the quantum information and crypto components separately
- Take advantage of the uncertainty principle
- We need states that can simultaneously encode information in two conjugate bases

Approach

- Modularize: think about the quantum information and crypto components separately
- Take advantage of the uncertainty principle
- We need states that can simultaneously encode information in two conjugate bases
  - One basis will encode plaintext information

Approach

- Modularize: think about the quantum information and crypto components separately
- Take advantage of the uncertainty principle
- We need states that can simultaneously encode information in two conjugate bases
  - One basis will encode plaintext information
  - The other will encode valid deletion certificates

General Recipe

General Recipe

For a subspace $S \subset \mathbb{F}_2^n$ and vectors $x \in \mathrm{co}(S), z \in \mathrm{co}(S^\perp)$, define

$$|S_{x,z}\rangle = \frac{1}{\sqrt{|S|}} \sum_{s \in S} (-1)^{s \cdot z} |s + x\rangle$$

General Recipe

$\mathrm{co}(S)$: a set of coset representatives of S

For a subspace $S \subset \mathbb{F}_2^n$ and vectors $x \in \mathrm{co}(S), z \in \mathrm{co}(S^\perp)$, define

$$|S_{x,z}\rangle = \frac{1}{\sqrt{|S|}} \sum_{s \in S} (-1)^{s \cdot z} |s + x\rangle$$

General Recipe

$\mathrm{co}(S)$: a set of coset representatives of S

For a subspace $S \subset \mathbb{F}_2^n$ and vectors $x \in \mathrm{co}(S), z \in \mathrm{co}(S^\perp)$, define

$$|S_{x,z}\rangle = \frac{1}{\sqrt{|S|}} \sum_{s \in S} (-1)^{s \cdot z} |s + x\rangle$$

$$\mathrm{H}^{\otimes n} \updownarrow$$

$$|S^\perp_{z,x}\rangle = \frac{1}{\sqrt{|S^\perp|}} \sum_{s \in S^\perp} (-1)^{s \cdot x} |s + z\rangle$$

General Recipe

$\mathrm{co}(S)$: a set of coset representatives of S

For a subspace $S \subset \mathbb{F}_2^n$ and vectors $x \in \mathrm{co}(S), z \in \mathrm{co}(S^\perp)$, define

$$\left|S_{x,z}\right\rangle = \frac{1}{\sqrt{|S|}} \sum_{s \in S} (-1)^{s \cdot z} |s + x\rangle$$

$$\mathrm{H}^{\otimes n} \updownarrow$$

$$\left|S^\perp_{z,x}\right\rangle = \frac{1}{\sqrt{|S^\perp|}} \sum_{s \in S^\perp} (-1)^{s \cdot x} |s + z\rangle$$

Uncertainty principle: $\mathcal{A}(|S_{x,z}\rangle) \nRightarrow (s \in S + x \,, s' \in S^\perp + z)$

(if S, x, z are sufficiently random)

General Recipe

$\mathrm{co}(S)$: a set of coset representatives of S

For a subspace $S \subset \mathbb{F}_2^n$ and vectors $x \in \mathrm{co}(S), z \in \mathrm{co}(S^\perp)$, define

$$|S_{x,z}\rangle = \frac{1}{\sqrt{|S|}} \sum_{s \in S} (-1)^{s \cdot z} |s + x\rangle$$

Use x to hide the plaintext

$$\mathrm{H}^{\otimes n} \updownarrow$$

$$|S^\perp_{z,x}\rangle = \frac{1}{\sqrt{|S^\perp|}} \sum_{s \in S^\perp} (-1)^{s \cdot x} |s + z\rangle$$

Use z as certificate of deletion

Uncertainty principle: $\mathcal{A}(|S_{x,z}\rangle) \not\Rightarrow (s \in S + x \,, s' \in S^\perp + z)$

(if S, x, z are sufficiently random)

General Recipe

General Recipe

Notation

- $\mathcal{C}$: cryptosystem with decryption key sk
- $\mathcal{H}$: family of hash functions
- $\mathcal{D}$: a distribution over (S, x, z)

General Recipe

Notation

- $\mathcal{C}$: cryptosystem with decryption key sk
- $\mathcal{H}$: family of hash functions
- $\mathcal{D}$: a distribution over (S, x, z)

$\text{EncCD}(b)$:

- Sample $(S, x, z) \leftarrow \mathcal{D}$
- Sample $h \leftarrow \mathcal{H}$
- Output $|S_{x,z}\rangle, \mathcal{C}_{sk}(S, h), b \oplus h(x)$

General Recipe

Notation
- $\mathcal{C}$: cryptosystem with decryption key sk
- $\mathcal{H}$: family of hash functions
- $\mathcal{D}$: a distribution over (S, x, z)

Decryption given sk:
- Use sk to learn S and h
- Measure $|S_{x,z}\rangle$ in standard basis, and let x be the coset representative of the resulting vector
- Use $h(x)$ to learn the plaintext b

$\mathrm{EncCD}(b)$:
- Sample $(S, x, z) \leftarrow \mathcal{D}$
- Sample $h \leftarrow \mathcal{H}$
- Output $|S_{x,z}\rangle, \mathcal{C}_{sk}(S, h), b \oplus h(x)$

General Recipe

Notation

- $\mathcal{C}$: cryptosystem with decryption key sk
- $\mathcal{H}$: family of hash functions
- $\mathcal{D}$: a distribution over (S, x, z)

$\mathrm{EncCD}(b)$:

- Sample $(S, x, z) \leftarrow \mathcal{D}$
- Sample $h \leftarrow \mathcal{H}$
- Output $|S_{x,z}\rangle, \mathcal{C}_{sk}(S, h), b \oplus h(x)$

Decryption given sk:

- Use sk to learn S and h
- Measure $|S_{x,z}\rangle$ in standard basis, and let x be the coset representative of the resulting vector
- Use $h(x)$ to learn the plaintext b

Deletion:

- Measure $|S_{x,z}\rangle$ in Hadamard basis to obtain a vector π
- Verification checks that $\pi \in S^{\perp} + z$

General Recipe

One-time pad
Public-key encryption
Commitment
Timed-release encryption
…

<u>Notation</u>

- $\mathcal{C}$: cryptosystem with decryption key sk
- $\mathcal{H}$: family of hash functions
- $\mathcal{D}$: a distribution over (S, x, z)

Decryption given sk:

- Use sk to learn S and h
- Measure $|S_{x,z}\rangle$ in standard basis, and let x be the coset representative of the resulting vector
- Use $h(x)$ to learn the plaintext b

$\mathrm{EncCD}(b)$:

- Sample $(S, x, z) \leftarrow \mathcal{D}$
- Sample $h \leftarrow \mathcal{H}$
- Output $|S_{x,z}\rangle, \mathcal{C}_{sk}(S, h), b \oplus h(x)$

Deletion:

- Measure $|S_{x,z}\rangle$ in Hadamard basis to obtain a vector π
- Verification checks that $\pi \in S^{\perp} + z$

General Recipe

One-time pad
Public-key encryption
Commitment
Timed-release encryption
…

<u>Notation</u>

- $\mathcal{C}$: cryptosystem with decryption key sk
- $\mathcal{H}$: family of hash functions
- $\mathcal{D}$: a distribution over (S, x, z)

Randomness extractor with seed h

Decryption given sk:

- Use sk to learn S and h
- Measure $|S_{x,z}\rangle$ in standard basis, and let x be the coset representative of the resulting vector
- Use $h(x)$ to learn the plaintext b

$\mathrm{EncCD}(b)$:

- Sample $(S, x, z) \leftarrow \mathcal{D}$
- Sample $h \leftarrow \mathcal{H}$
- Output $|S_{x,z}\rangle, \mathcal{C}_{sk}(S, h), b \oplus h(x)$

Deletion:

- Measure $|S_{x,z}\rangle$ in Hadamard basis to obtain a vector π
- Verification checks that $\pi \in S^{\perp} + z$

General Recipe

One-time pad
Public-key encryption
Commitment
Timed-release encryption
…

Notation

- $\mathcal{C}$: cryptosystem with decryption key sk
- $\mathcal{H}$: family of hash functions
- $\mathcal{D}$: a distribution over (S, x, z)

Randomness extractor with seed h

Decryption given sk:

- Use sk to learn S and h
- Measure $|S_{x,z}\rangle$ in standard basis, and let x be the coset representative of the resulting vector
- Use $h(x)$ to learn the plaintext b

$\mathrm{EncCD}(b)$:

- Sample $(S, x, z) \leftarrow \mathcal{D}$
- Sample $h \leftarrow \mathcal{H}$
- Output $|S_{x,z}\rangle, \mathcal{C}_{sk}(S, h), b \oplus h(x)$

Deletion:

- Measure $|S_{x,z}\rangle$ in Hadamard basis to obtain a vector π
- Verification checks that $\pi \in S^{\perp} + z$

Instantiatiating the distribution over S

Optimize for…

Instantiatiating the distribution over S

Optimize for…

<u>Practicality</u>		

Instantiatiating the distribution over S

Optimize for…

Practicality		
S spanned by standard basis vectors (Wiesner/BB84 states): $\theta \leftarrow \{0,1\}^n, S = \text{span}\{e_i\}_{i:\theta_i=1}$		

Instantiatiating the distribution over S

Optimize for…

<u>Practicality</u>

S spanned by standard basis vectors (Wiesner/BB84 states):
$\theta \leftarrow \{0,1\}^n, S = \mathrm{span}\{e_i\}_{i:\theta_i=1}$

$\mathrm{H}^{\theta_1}|x_1\rangle, \ldots, \mathrm{H}^{\theta_n}|x_n\rangle,$
$\mathcal{C}_{sk}(\theta, h), b \oplus h(\{x_i\}_{i:\theta_i=0})$

Instantiatiating the distribution over S

Optimize for…

Practicality

S spanned by standard basis vectors (Wiesner/BB84 states):
$\theta \leftarrow \{0,1\}^n, S = \text{span}\{e_i\}_{i:\theta_i=1}$

$\mathrm{H}^{\theta_1}|x_1\rangle, \ldots, \mathrm{H}^{\theta_n}|x_n\rangle,$
$\mathcal{C}_{sk}(\theta, h), b \oplus h(\{x_i\}_{i:\theta_i=0})$

No entanglement required

Instantiatiating the distribution over S

Optimize for…

Practicality

S spanned by standard basis vectors (Wiesner/BB84 states): $\theta \leftarrow \{0,1\}^n, S = \text{span}\{e_i\}_{i:\theta_i=1}$

$\mathrm{H}^{\theta_1}|x_1\rangle, \dots, \mathrm{H}^{\theta_n}|x_n\rangle,$
$\mathcal{C}_{sk}(\theta, h), b \oplus h(\{x_i\}_{i:\theta_i=0})$

No entanglement required

[BI20]

Instantiatiating the distribution over S

Optimize for…

Practicality	Publicly-Verifiable Deletion	
S spanned by standard basis vectors (Wiesner/BB84 states): $\theta \leftarrow \{0,1\}^n, S = \mathrm{span}\{e_i\}_{i:\theta_i=1}$		
$\mathrm{H}^{\theta_1}\|x_1\rangle, \ldots, \mathrm{H}^{\theta_n}\|x_n\rangle, \mathcal{C}_{sk}(\theta, h), b \oplus h(\{x_i\}_{i:\theta_i=0})$		
No entanglement required		
[BI20]		

Instantiatiating the distribution over S

Optimize for…

Practicality

S spanned by standard basis vectors (Wiesner/BB84 states):
$\theta \leftarrow \{0,1\}^n, S = \mathrm{span}\{e_i\}_{i:\theta_i=1}$

$\mathrm{H}^{\theta_1}|x_1\rangle, \ldots, \mathrm{H}^{\theta_n}|x_n\rangle,$
$\mathcal{C}_{sk}(\theta, h), b \oplus h(\{x_i\}_{i:\theta_i=0})$

No entanglement required

[BI20]

Publicly-Verifiable Deletion

S has dimension $n-1$,
so $S^\perp = \{0^n, v\}$

Instantiatiating the distribution over S

Optimize for…

Practicality	Publicly-Verifiable Deletion	
S spanned by standard basis vectors (Wiesner/BB84 states): $\theta \leftarrow \{0,1\}^n, S = \mathrm{span}\{e_i\}_{i:\theta_i=1}$	S has dimension $n-1$, so $S^{\perp} = \{0^n, v\}$	
$\mathrm{H}^{\theta_1}\lvert x_1\rangle, \ldots, \mathrm{H}^{\theta_n}\lvert x_n\rangle$, $\mathcal{C}_{sk}(\theta, h), b \oplus h(\{x_i\}_{i:\theta_i=0})$	$\mathrm{H}^{\otimes n}(\lvert z\rangle + (-1)^x \lvert z+v\rangle)$, $\mathcal{C}_{sk}(v), b \oplus x$	
No entanglement required		
[BI20]		

Instantiatiating the distribution over S

Optimize for…

Practicality	Publicly-Verifiable Deletion	
S spanned by standard basis vectors (Wiesner/BB84 states): $\theta \leftarrow \{0,1\}^n, S = \mathrm{span}\{e_i\}_{i:\theta_i=1}$	S has dimension $n-1$, so $S^\perp = \{0^n, v\}$	
$\mathrm{H}^{\theta_1}\|x_1\rangle, \ldots, \mathrm{H}^{\theta_n}\|x_n\rangle$, $\mathcal{C}_{sk}(\theta, h), b \oplus h(\{x_i\}_{i:\theta_i=0})$	$\mathrm{H}^{\otimes n}(\|z\rangle + (-1)^x\|z+v\rangle)$, $\mathcal{C}_{sk}(v), b \oplus x$	
No entanglement required	Only two valid deletion certificates, so publish $\mathrm{OWF}(z), \mathrm{OWF}(z+v)$	
[BI20]		

Instantiatiating the distribution over S

Optimize for…

Practicality

S spanned by standard basis vectors (Wiesner/BB84 states):
$\theta \leftarrow \{0,1\}^n, S = \mathrm{span}\{e_i\}_{i:\theta_i=1}$

$\mathrm{H}^{\theta_1}|x_1\rangle, \dots, \mathrm{H}^{\theta_n}|x_n\rangle,$
$\mathcal{C}_{sk}(\theta, h), b \oplus h(\{x_i\}_{i:\theta_i=0})$

No entanglement required

[BI20]

Publicly-Verifiable Deletion

S has dimension $n-1$,
so $S^\perp = \{0^n, v\}$

$\mathrm{H}^{\otimes n}(|z\rangle + (-1)^x|z+v\rangle),$
$\mathcal{C}_{sk}(v), b \oplus x$

Only two valid deletion certificates, so publish
$\mathrm{OWF}(z), \mathrm{OWF}(z+v)$

[**B**KMPW23]

Instantiatiating the distribution over S

Optimize for…

Practicality

S spanned by standard basis vectors (Wiesner/BB84 states): $\theta \leftarrow \{0,1\}^n, S = \mathrm{span}\{e_i\}_{i:\theta_i=1}$

$\mathrm{H}^{\theta_1}|x_1\rangle, \ldots, \mathrm{H}^{\theta_n}|x_n\rangle,$
$\mathcal{C}_{sk}(\theta, h), b \oplus h(\{x_i\}_{i:\theta_i=0})$

No entanglement required

[BI20]

Publicly-Verifiable Deletion

S has dimension $n-1$, so $S^\perp = \{0^n, v\}$

$\mathrm{H}^{\otimes n}(|z\rangle + (-1)^x|z+v\rangle),$
$\mathcal{C}_{sk}(v), b \oplus x$

Only two valid deletion certificates, so publish $\mathrm{OWF}(z), \mathrm{OWF}(z+v)$

[**B**KMPW23]

Publicly-Verifiable Ciphertext

Instantiatiating the distribution over S

Optimize for…

Practicality	Publicly-Verifiable Deletion	Publicly-Verifiable Ciphertext
S spanned by standard basis vectors (Wiesner/BB84 states): $\theta \leftarrow \{0,1\}^n, S = \mathrm{span}\{e_i\}_{i:\theta_i=1}$	S has dimension $n-1$, so $S^\perp = \{0^n, v\}$	S uniform over all subspaces
$\mathrm{H}^{\theta_1}\|x_1\rangle, \ldots, \mathrm{H}^{\theta_n}\|x_n\rangle$, $\mathcal{C}_{sk}(\theta, h), b \oplus h(\{x_i\}_{i:\theta_i=0})$	$\mathrm{H}^{\otimes n}(\|z\rangle + (-1)^x\|z+v\rangle)$, $\mathcal{C}_{sk}(v), b \oplus x$	
No entanglement required	Only two valid deletion certificates, so publish $\mathrm{OWF}(z), \mathrm{OWF}(z+v)$	
[BI20]	[**B**KMPW23]	

Instantiatiating the distribution over S

Optimize for…

Practicality

S spanned by standard basis vectors (Wiesner/BB84 states):
$\theta \leftarrow \{0,1\}^n, S = \mathrm{span}\{e_i\}_{i:\theta_i=1}$

$\mathrm{H}^{\theta_1}|x_1\rangle, \ldots, \mathrm{H}^{\theta_n}|x_n\rangle,$
$\mathcal{C}_{sk}(\theta, h), b \oplus h(\{x_i\}_{i:\theta_i=0})$

No entanglement required

[BI20]

Publicly-Verifiable Deletion

S has dimension $n-1$,
so $S^\perp = \{0^n, v\}$

$\mathrm{H}^{\otimes n}(|z\rangle + (-1)^x|z+v\rangle),$
$\mathcal{C}_{sk}(v), b \oplus x$

Only two valid deletion certificates, so publish $\mathrm{OWF}(z), \mathrm{OWF}(z+v)$

[**B**KMPW23]

Publicly-Verifiable Ciphertext

S uniform over all subspaces

$|S_{x,z}\rangle, \mathcal{C}_{sk}(S,h), b \oplus h(x)$

Instantiatiating the distribution over S

Optimize for…

Practicality

S spanned by standard basis vectors (Wiesner/BB84 states): $\theta \leftarrow \{0,1\}^n, S = \mathrm{span}\{e_i\}_{i:\theta_i=1}$

$\mathrm{H}^{\theta_1}|x_1\rangle, \ldots, \mathrm{H}^{\theta_n}|x_n\rangle,$
$\mathcal{C}_{sk}(\theta, h), b \oplus h(\{x_i\}_{i:\theta_i=0})$

No entanglement required

[BI20]

Publicly-Verifiable Deletion

S has dimension $n-1$, so $S^\perp = \{0^n, v\}$

$\mathrm{H}^{\otimes n}(|z\rangle + (-1)^x|z+v\rangle),$
$\mathcal{C}_{sk}(v), b \oplus x$

Only two valid deletion certificates, so publish $\mathrm{OWF}(z), \mathrm{OWF}(z+v)$

[**B**KMPW23]

Publicly-Verifiable Ciphertext

S uniform over all subspaces

$|S_{x,z}\rangle, \mathcal{C}_{sk}(S,h), b \oplus h(x)$

Secure even given oracle access to $S + x$

Instantiatiating the distribution over S

Optimize for…

Practicality	Publicly-Verifiable Deletion	Publicly-Verifiable Ciphertext
S spanned by standard basis vectors (Wiesner/BB84 states): $\theta \leftarrow \{0,1\}^n, S = \text{span}\{e_i\}_{i:\theta_i=1}$	S has dimension $n-1$, so $S^\perp = \{0^n, v\}$	S uniform over all subspaces
$\text{H}^{\theta_1}\|x_1\rangle, \ldots, \text{H}^{\theta_n}\|x_n\rangle$, $\mathcal{C}_{sk}(\theta, h), b \oplus h(\{x_i\}_{i:\theta_i=0})$	$\text{H}^{\otimes n}(\|z\rangle + (-1)^x\|z+v\rangle)$, $\mathcal{C}_{sk}(v), b \oplus x$	$\|S_{x,z}\rangle, \mathcal{C}_{sk}(S,h), b \oplus h(x)$
No entanglement required	Only two valid deletion certificates, so publish $\text{OWF}(z), \text{OWF}(z+v)$	Secure even given oracle access to $S+x$
[BI20]	[**B**KMPW23]	[**B**GGKMRR23]

Outline

1. Basic scenario and applications
2. Recipe for constructions
3. Security
4. Certifiable deletion of programs

Security Game

Security Game

$\mathrm{CDExp}_{\mathcal{C},\mathcal{H},\mathcal{D},\mathcal{A}_1,\mathcal{A}_2}(b)$

- Sample $(S, x, z) \leftarrow \mathcal{D}$, $h \leftarrow \mathcal{H}$, and sk
- $\mathcal{A}_1\left(|S_{x,z}\rangle, \mathcal{C}_{sk}(S, h), b \oplus h(x)\right) \rightarrow \pi, \mathrm{st}$
- If $\pi \notin S^\perp + z$, output $b' \leftarrow \{0,1\}$
- Otherwise, output $b' \leftarrow \mathcal{A}_2(\mathrm{st}, sk)$

Security Game

$\mathrm{CDExp}_{\mathcal{C},\mathcal{H},\mathcal{D},\mathcal{A}_1,\mathcal{A}_2}(b)$

- Sample $(S, x, z) \leftarrow \mathcal{D}$, $h \leftarrow \mathcal{H}$, and sk
- $\mathcal{A}_1\left(|S_{x,z}\rangle, \mathcal{C}_{sk}(S, h), b \oplus h(x)\right) \rightarrow \pi, \mathrm{st}$
- If $\pi \notin S^\perp + z$, output $b' \leftarrow \{0,1\}$
- Otherwise, output $b' \leftarrow \mathcal{A}_2(\mathrm{st}, sk)$

Want: $\left|\Pr\left[\mathrm{CDExp}_{\mathcal{C},\mathcal{H},\mathcal{D},\mathcal{A}_1,\mathcal{A}_2}(0) = 1\right] - \Pr\left[\mathrm{CDExp}_{\mathcal{C},\mathcal{H},\mathcal{D},\mathcal{A}_1,\mathcal{A}_2}(1) = 1\right]\right| = \mathrm{negl}$

Security Game

$\text{CDExp}_{\mathcal{C},\mathcal{H},\mathcal{D},\mathcal{A}_1,\mathcal{A}_2}(b)$

- Sample $(S,x,z) \leftarrow \mathcal{D}$, $h \leftarrow \mathcal{H}$, and sk
- $\mathcal{A}_1\left(|S_{x,z}\rangle, \mathcal{C}_{sk}(S,h), b \oplus h(x)\right) \rightarrow \pi, \text{st}$
- If $\pi \notin S^\perp + z$, output $b' \leftarrow \{0,1\}$
- Otherwise, output $b' \leftarrow \mathcal{A}_2(\text{st}, sk)$

Want: $\left|\Pr\left[\text{CDExp}_{\mathcal{C},\mathcal{H},\mathcal{D},\mathcal{A}_1,\mathcal{A}_2}(0) = 1\right] - \Pr\left[\text{CDExp}_{\mathcal{C},\mathcal{H},\mathcal{D},\mathcal{A}_1,\mathcal{A}_2}(1) = 1\right]\right| = \text{negl}$

History

Security Game

$\mathrm{CDExp}_{\mathcal{C},\mathcal{H},\mathcal{D},\mathcal{A}_1,\mathcal{A}_2}(b)$

- Sample $(S, x, z) \leftarrow \mathcal{D}$, $h \leftarrow \mathcal{H}$, and sk
- $\mathcal{A}_1\left(|S_{x,z}\rangle, \mathcal{C}_{sk}(S, h), b \oplus h(x)\right) \rightarrow \pi, \mathrm{st}$
- If $\pi \notin S^{\perp} + z$, output $b' \leftarrow \{0,1\}$
- Otherwise, output $b' \leftarrow \mathcal{A}_2(\mathrm{st}, sk)$

Want: $\left|\Pr\left[\mathrm{CDExp}_{\mathcal{C},\mathcal{H},\mathcal{D},\mathcal{A}_1,\mathcal{A}_2}(0) = 1\right] - \Pr\left[\mathrm{CDExp}_{\mathcal{C},\mathcal{H},\mathcal{D},\mathcal{A}_1,\mathcal{A}_2}(1) = 1\right]\right| = \mathrm{negl}$

<u>History</u>

- [Broadbent, Islam 20]:
  - $\mathcal{C}$ one-time pad
  - $\mathcal{H}$ good randomness extractor
  - $\mathcal{D}$ Wiesner states
  - $(\mathcal{A}_1, \mathcal{A}_2)$ unbounded

Security Game

$\mathrm{CDExp}_{\mathcal{C},\mathcal{H},\mathcal{D},\mathcal{A}_1,\mathcal{A}_2}(b)$

- Sample $(S, x, z) \leftarrow \mathcal{D}$, $h \leftarrow \mathcal{H}$, and sk
- $\mathcal{A}_1\left(|S_{x,z}\rangle, \mathcal{C}_{sk}(S, h), b \oplus h(x)\right) \rightarrow \pi, \mathrm{st}$
- If $\pi \notin S^{\perp} + z$, output $b' \leftarrow \{0,1\}$
- Otherwise, output $b' \leftarrow \mathcal{A}_2(\mathrm{st}, sk)$

Want: $\left|\Pr\left[\mathrm{CDExp}_{\mathcal{C},\mathcal{H},\mathcal{D},\mathcal{A}_1,\mathcal{A}_2}(0) = 1\right] - \Pr\left[\mathrm{CDExp}_{\mathcal{C},\mathcal{H},\mathcal{D},\mathcal{A}_1,\mathcal{A}_2}(1) = 1\right]\right| = \mathrm{negl}$

History

- [Broadbent, Islam 20]:
  - $\mathcal{C}$ one-time pad
  - $\mathcal{H}$ good randomness extractor
  - $\mathcal{D}$ Wiesner states
  - $(\mathcal{A}_1, \mathcal{A}_2)$ unbounded
- [Hiroka, Morimae, Nishimaki, Yamakawa 21]:
  - $\mathcal{C}$ non-committing encryption scheme
  - $\mathcal{H}$ good randomness extractor
  - $\mathcal{D}$ Wiesner states
  - $(\mathcal{A}_1, \mathcal{A}_2)$ computationally bounded

Security Game

$\mathrm{CDExp}_{\mathcal{C},\mathcal{H},\mathcal{D},\mathcal{A}_1,\mathcal{A}_2}(b)$

- Sample $(S, x, z) \leftarrow \mathcal{D}$, $h \leftarrow \mathcal{H}$, and sk
- $\mathcal{A}_1\left(|S_{x,z}\rangle, \mathcal{C}_{sk}(S, h), b \oplus h(x)\right) \rightarrow \pi, \mathrm{st}$
- If $\pi \notin S^{\perp} + z$, output $b' \leftarrow \{0,1\}$
- Otherwise, output $b' \leftarrow \mathcal{A}_2(\mathrm{st}, sk)$

Want: $\left|\Pr\left[\mathrm{CDExp}_{\mathcal{C},\mathcal{H},\mathcal{D},\mathcal{A}_1,\mathcal{A}_2}(0) = 1\right] - \Pr\left[\mathrm{CDExp}_{\mathcal{C},\mathcal{H},\mathcal{D},\mathcal{A}_1,\mathcal{A}_2}(1) = 1\right]\right| = \mathrm{negl}$

History

- [Broadbent, Islam 20]:
  - $\mathcal{C}$ one-time pad
  - $\mathcal{H}$ good randomness extractor
  - $\mathcal{D}$ Wiesner states
  - $(\mathcal{A}_1, \mathcal{A}_2)$ unbounded
- [Hiroka, Morimae, Nishimaki, Yamakawa 21]:
  - $\mathcal{C}$ non-committing encryption scheme
  - $\mathcal{H}$ good randomness extractor
  - $\mathcal{D}$ Wiesner states
  - $(\mathcal{A}_1, \mathcal{A}_2)$ computationally bounded
- [**B**, Khurana 23]:
  - $\mathcal{C}$ semantically-secure distribution
  - $\mathcal{H}=\oplus$
  - $\mathcal{D}$ Wiesner states
  - $\mathcal{A}_1$ computationally bounded, $\mathcal{A}_2$ unbounded

Security Game

$\mathrm{CDExp}_{\mathcal{C},\mathcal{H},\mathcal{D},\mathcal{A}_1,\mathcal{A}_2}(b)$

- Sample $(S, x, z) \leftarrow \mathcal{D}$, $h \leftarrow \mathcal{H}$, and sk
- $\mathcal{A}_1\left(|S_{x,z}\rangle, \mathcal{C}_{sk}(S,h), b \oplus h(x)\right) \rightarrow \pi, \mathrm{st}$
- If $\pi \notin S^\perp + z$, output $b' \leftarrow \{0,1\}$
- Otherwise, output $b' \leftarrow \mathcal{A}_2(\mathrm{st}, sk)$

Want: $\left|\Pr\left[\mathrm{CDExp}_{\mathcal{C},\mathcal{H},\mathcal{D},\mathcal{A}_1,\mathcal{A}_2}(0) = 1\right] - \Pr\left[\mathrm{CDExp}_{\mathcal{C},\mathcal{H},\mathcal{D},\mathcal{A}_1,\mathcal{A}_2}(1) = 1\right]\right| = \mathrm{negl}$

History

- [Broadbent, Islam 20]:
  - $\mathcal{C}$ one-time pad
  - $\mathcal{H}$ good randomness extractor
  - $\mathcal{D}$ Wiesner states
  - $(\mathcal{A}_1, \mathcal{A}_2)$ unbounded
- [Hiroka, Morimae, Nishimaki, Yamakawa 21]:
  - $\mathcal{C}$ non-committing encryption scheme
  - $\mathcal{H}$ good randomness extractor
  - $\mathcal{D}$ Wiesner states
  - $(\mathcal{A}_1, \mathcal{A}_2)$ computationally bounded
- [**B**, Khurana 23]:
  - $\mathcal{C}$ semantically-secure distribution
  - $\mathcal{H}=\bigoplus$
  - $\mathcal{D}$ Wiesner states
  - $\mathcal{A}_1$ computationally bounded, $\mathcal{A}_2$ unbounded
- [**B**, Garg, Goyal, Khurana, Malavolta, Raizes, Roberts 23]
  - $\mathcal{C}$ subspace-hiding distribution
  - $\mathcal{H}=\bigoplus$
  - $\mathcal{D}$ subspace states
  - $\mathcal{A}_1$ computationally bounded, $\mathcal{A}_2$ unbounded

Security Game

$\mathrm{CDExp}_{\mathcal{C},\mathcal{H},\mathcal{D},\mathcal{A}_1,\mathcal{A}_2}(b)$

- Sample $(S, x, z) \leftarrow \mathcal{D}$, $h \leftarrow \mathcal{H}$, and sk
- $\mathcal{A}_1\left(|S_{x,z}\rangle, \mathcal{C}_{sk}(S, h), b \oplus h(x)\right) \rightarrow \pi, \mathrm{st}$
- If $\pi \notin S^{\perp} + z$, output $b' \leftarrow \{0,1\}$
- Otherwise, output $b' \leftarrow \mathcal{A}_2(\mathrm{st}, sk)$

Want: $\left|\Pr\left[\mathrm{CDExp}_{\mathcal{C},\mathcal{H},\mathcal{D},\mathcal{A}_1,\mathcal{A}_2}(0) = 1\right] - \Pr\left[\mathrm{CDExp}_{\mathcal{C},\mathcal{H},\mathcal{D},\mathcal{A}_1,\mathcal{A}_2}(1) = 1\right]\right| = \mathrm{negl}$

Note: [Unruh 13] showed similar statement for a slightly different template supporting *quantum* certificates of deletion

History

- [Broadbent, Islam 20]:
  - $\mathcal{C}$ one-time pad
  - $\mathcal{H}$ good randomness extractor
  - $\mathcal{D}$ Wiesner states
  - $(\mathcal{A}_1, \mathcal{A}_2)$ unbounded
- [Hiroka, Morimae, Nishimaki, Yamakawa 21]:
  - $\mathcal{C}$ non-committing encryption scheme
  - $\mathcal{H}$ good randomness extractor
  - $\mathcal{D}$ Wiesner states
  - $(\mathcal{A}_1, \mathcal{A}_2)$ computationally bounded
- [**B**, Khurana 23]:
  - $\mathcal{C}$ semantically-secure distribution
  - $\mathcal{H}=\bigoplus$
  - $\mathcal{D}$ Wiesner states
  - $\mathcal{A}_1$ computationally bounded, $\mathcal{A}_2$ unbounded
- [**B**, Garg, Goyal, Khurana, Malavolta, Raizes, Roberts 23]
  - $\mathcal{C}$ subspace-hiding distribution
  - $\mathcal{H}=\bigoplus$
  - $\mathcal{D}$ subspace states
  - $\mathcal{A}_1$ computationally bounded, $\mathcal{A}_2$ unbounded

Example Proof

Example Proof

- Let $\mathcal{C}$ be a computationally-hiding statistically-binding commitment
- Let $\mathcal{H} = \bigoplus$ (unseeded)
- Let $\mathcal{D}$ sample a uniformly random (S, x, z)
- Let $\mathcal{A}_1$ be computationally bounded and $\mathcal{A}_2$ be unbounded

Example Proof

- Let $\mathcal{C}$ be a computationally-hiding statistically-binding commitment
- Let $\mathcal{H} = \bigoplus$ (unseeded)
- Let $\mathcal{D}$ sample a uniformly random (S, x, z)
- Let $\mathcal{A}_1$ be computationally bounded and $\mathcal{A}_2$ be unbounded

$\underline{\mathcal{A}}$ | $\mathrm{Hyb}_0(b)$ | $\underline{\mathit{Ch}}$

Sample (S, x, z)

$\leftarrow$ $\mathrm{Com}(S), b \bigoplus_i x_i, |S_{x,z}\rangle$

$\rightarrow$ π, st

If $\pi \notin S^\perp + z$, output $|\perp\rangle\langle\perp|$

Otherwise, output st

Example Proof

- Let $\mathcal{C}$ be a computationally-hiding statistically-binding commitment
- Let $\mathcal{H} = \bigoplus$ (unseeded)
- Let $\mathcal{D}$ sample a uniformly random (S, x, z)
- Let $\mathcal{A}_1$ be computationally bounded and $\mathcal{A}_2$ be unbounded

$\underline{\mathcal{A}}$ $\mathrm{Hyb}_0(b)$ $\underline{Ch}$

Sample (S, x, z)

$\mathrm{Com}(S), b \bigoplus_i x_i, |S_{x,z}\rangle$ ←

π, st →

If $\pi \notin S^{\perp} + z$, output $|\perp\rangle\langle\perp|$

Otherwise, output st

Goal: Show that $\mathrm{TD}\big(\mathrm{Hyb}_0(0), \mathrm{Hyb}_0(1)\big) = \mathrm{negl}$

Example Proof

Hybrid 1: Delay the dependence of the experiment on b

$\underline{\mathcal{A}}$

$\mathrm{Hyb}_1(b)$

$\underline{\mathcal{Ch}}$

Sample (S, x, z)

Sample $b' \leftarrow \{0,1\}$

$\mathrm{Com}(S), b', |S_{x,z}\rangle$ ←

π, st →

If $\pi \notin S^\perp + z$, output $|\perp\rangle\langle\perp|$

If $b \bigoplus_i x_i \neq b'$, output $|\perp\rangle\langle\perp|$

Otherwise, output st

Example Proof

Hybrid 1: Delay the dependence of the experiment on b

$$\mathrm{TD}\big(\mathrm{Hyb}_1(0), \mathrm{Hyb}_1(1)\big) = \frac{1}{2}\mathrm{TD}\big(\mathrm{Hyb}_0(0), \mathrm{Hyb}_0(1)\big)$$

$\underline{\mathcal{A}}$ $\mathrm{Hyb}_1(b)$ $\underline{Ch}$

Sample (S, x, z)

Sample $b' \leftarrow \{0,1\}$

$\mathrm{Com}(S), b', |S_{x,z}\rangle$ ←

π, st →

If $\pi \notin S^\perp + z$, output $|\perp\rangle\langle\perp|$

If $b \bigoplus_i x_i \neq b'$, output $|\perp\rangle\langle\perp|$

Otherwise, output st

Example Proof

Hybrid 1: Delay the dependence of the experiment on b

$$\mathrm{TD}\big(\mathrm{Hyb}_1(0), \mathrm{Hyb}_1(1)\big) = \frac{1}{2}\mathrm{TD}\big(\mathrm{Hyb}_0(0), \mathrm{Hyb}_0(1)\big)$$

$\mathcal{A}$ | $\mathrm{Hyb}_1(b)$ | $\mathcal{Ch}$

Sample (S, x, z)

Sample $b' \leftarrow \{0,1\}$

$\leftarrow$ $\mathrm{Com}(S), b', |S_{x,z}\rangle$

$\rightarrow$ π, st

If $\pi \notin S^{\perp} + z$, output $|\perp\rangle\langle\perp|$

If $b \oplus_i x_i \neq b'$, output $|\perp\rangle\langle\perp|$

Otherwise, output st

Remains to show that x has a lot of conditional min-entropy

Example Proof

Want to show: If $\mathcal{A}(|S_{x,z}\rangle, \mathrm{Com}(S))$ outputs $\pi \in S^{\perp} + z$, then x has a lot of conditional min-entropy

Example Proof

Want to show: If $\mathcal{A}(|S_{x,z}\rangle, \mathrm{Com}(S))$ outputs $\pi \in S^{\perp} + z$, then x has a lot of conditional min-entropy

$$\mathcal{A}(|S_{x,z}\rangle, \mathrm{Com}(S)) \to \pi$$

Example Proof

Want to show: If $\mathcal{A}(|S_{x,z}\rangle, \mathrm{Com}(S))$ outputs $\pi \in S^\perp + z$, then x has a lot of conditional min-entropy

$$\mathcal{A}(|S_{x,z}\rangle, \mathrm{Com}(S)) \to \pi$$

↓ Purify

$$\sum_{x \in \mathrm{co}(S)} |x\rangle \quad \mathcal{A}(|S_{x,z}\rangle, \mathrm{Com}(S)) \to \pi$$

Example Proof

Want to show: If $\mathcal{A}(|S_{x,z}\rangle, \mathrm{Com}(S))$ outputs $\pi \in S^{\perp} + z$, then x has a lot of conditional min-entropy

$$\mathcal{A}(|S_{x,z}\rangle, \mathrm{Com}(S)) \to \pi$$

Purify

$$\sum_{x \in \mathrm{co}(S)} |x\rangle \qquad \mathcal{A}(|S_{x,z}\rangle, \mathrm{Com}(S)) \to \pi$$

For $x \in \mathrm{co}(S)$: $\mathrm{U}_S|x\rangle \to \sum_{v \in S^{\perp}} (-1)^{v \cdot x}|v\rangle$

For $v \in S^{\perp}$: $\mathrm{U}_S^{\dagger}|v\rangle \to \sum_{x \in \mathrm{co}(S)} (-1)^{v \cdot x}|x\rangle$

Example Proof

Want to show: If $\mathcal{A}(|S_{x,z}\rangle, \mathrm{Com}(S))$ outputs $\pi \in S^\perp + z$, then x has a lot of conditional min-entropy

$$\mathcal{A}(|S_{x,z}\rangle, \mathrm{Com}(S)) \to \pi$$

↓ Purify

$$\sum_{x \in \mathrm{co}(S)} |x\rangle \quad \mathcal{A}(|S_{x,z}\rangle, \mathrm{Com}(S)) \to \pi$$

↓ U_S

$$\sum_{v \in S^\perp} |v\rangle \quad \mathcal{A}\left(\sum_{w} (-1)^{w \cdot (z+v)} |w\rangle, \mathrm{Com}(S)\right) \to \pi$$

For $x \in \mathrm{co}(S)$: $\mathrm{U}_S|x\rangle \to \sum_{v \in S^\perp} (-1)^{v \cdot x} |v\rangle$

For $v \in S^\perp$: $\mathrm{U}_S^\dagger|v\rangle \to \sum_{x \in \mathrm{co}(S)} (-1)^{v \cdot x} |x\rangle$

Example Proof

Want to show: If $\mathcal{A}(|S_{x,z}\rangle, \mathrm{Com}(S))$ outputs $\pi \in S^\perp + z$, then x has a lot of conditional min-entropy

$$\mathcal{A}(|S_{x,z}\rangle, \mathrm{Com}(S)) \to \pi$$

$\downarrow$ Purify

$$\sum_{x \in \mathrm{co}(S)} |x\rangle \qquad \mathcal{A}(|S_{x,z}\rangle, \mathrm{Com}(S)) \to \pi$$

$\downarrow$ U_S

$$\sum_{v \in S^\perp} |v\rangle \qquad \mathcal{A}(\mathrm{H}^{\otimes n}|v + z\rangle, \mathrm{Com}(S)) \to \pi$$

For $x \in \mathrm{co}(S)$: $\mathrm{U}_S|x\rangle \to \sum_{v \in S^\perp} (-1)^{v \cdot x}|v\rangle$

For $v \in S^\perp$: $\mathrm{U}_S^\dagger|v\rangle \to \sum_{x \in \mathrm{co}(S)} (-1)^{v \cdot x}|x\rangle$

Example Proof

Want to show: If $\mathcal{A}(|S_{x,z}\rangle, \mathrm{Com}(S))$ outputs $\pi \in S^\perp + z$, then x has a lot of conditional min-entropy

$$\mathcal{A}(|S_{x,z}\rangle, \mathrm{Com}(S)) \to \pi$$

$\downarrow$ Purify

$$\sum_{x \in \mathrm{co}(S)} |x\rangle \quad \mathcal{A}(|S_{x,z}\rangle, \mathrm{Com}(S)) \to \pi$$

$\downarrow \mathrm{U}_S$

$$\sum_{v \in S^\perp} |v\rangle \quad \mathcal{A}(\mathrm{H}^{\otimes n}|v+z\rangle, \mathrm{Com}(S)) \to \pi$$

For $x \in \mathrm{co}(S)$: $\mathrm{U}_S|x\rangle \to \sum_{v \in S^\perp} (-1)^{v \cdot x}|v\rangle$

For $v \in S^\perp$: $\mathrm{U}_S^\dagger|v\rangle \to \sum_{x \in \mathrm{co}(S)} (-1)^{v \cdot x}|x\rangle$

Claim: if $\mathcal{A}$ given random $v + z$ and outputs $\pi \in S^\perp + z$, then $\pi = v + z$ with overwhelming probability (over S, z)

Example Proof

Want to show: If $\mathcal{A}(|S_{x,z}\rangle, \mathrm{Com}(S))$ outputs $\pi \in S^\perp + z$, then x has a lot of conditional min-entropy

$$\mathcal{A}(|S_{x,z}\rangle, \mathrm{Com}(S)) \to \pi$$

$\downarrow$ Purify

$$\sum_{x \in \mathrm{co}(S)} |x\rangle \quad \mathcal{A}(|S_{x,z}\rangle, \mathrm{Com}(S)) \to \pi$$

$\downarrow$ U_S

$$\sum_{v \in S^\perp} |v\rangle \quad \mathcal{A}(\mathrm{H}^{\otimes n}|v + z\rangle, \mathrm{Com}(S)) \to \pi$$

$\downarrow$ Project

$$|\pi - z\rangle$$

For $x \in \mathrm{co}(S)$: $\mathrm{U}_S|x\rangle \to \sum_{v \in S^\perp} (-1)^{v \cdot x}|v\rangle$

For $v \in S^\perp$: $\mathrm{U}_S^\dagger|v\rangle \to \sum_{x \in \mathrm{co}(S)} (-1)^{v \cdot x}|x\rangle$

Claim: if $\mathcal{A}$ given random $v + z$ and outputs $\pi \in S^\perp + z$, then $\pi = v + z$ with overwhelming probability (over S, z)

Example Proof

Want to show: If $\mathcal{A}(|S_{x,z}\rangle, \mathrm{Com}(S))$ outputs $\pi \in S^\perp + z$, then x has a lot of conditional min-entropy

$$\mathcal{A}(|S_{x,z}\rangle, \mathrm{Com}(S)) \rightarrow \pi$$

↓ Purify

$$\sum_{x \in \mathrm{co}(S)} |x\rangle \quad \mathcal{A}(|S_{x,z}\rangle, \mathrm{Com}(S)) \rightarrow \pi$$

↓ U_S

$$\sum_{v \in S^\perp} |v\rangle \quad \mathcal{A}(\mathrm{H}^{\otimes n}|v+z\rangle, \mathrm{Com}(S)) \rightarrow \pi$$

↓ Project

$$|\pi - z\rangle$$

↓ $\mathrm{U}_S^\dagger$

$$\sum_{x \in \mathrm{co}(S)} (-1)^{(\pi - z)\cdot x} |x\rangle$$

For $x \in \mathrm{co}(S)$: $\mathrm{U}_S|x\rangle \rightarrow \sum_{v \in S^\perp} (-1)^{v \cdot x}|v\rangle$

For $v \in S^\perp$: $\mathrm{U}_S^\dagger|v\rangle \rightarrow \sum_{x \in \mathrm{co}(S)} (-1)^{v \cdot x}|x\rangle$

Claim: if $\mathcal{A}$ given random $v + z$ and outputs $\pi \in S^\perp + z$, then $\pi = v + z$ with overwhelming probability (over S, z)

Example Proof

Want to show: If $\mathcal{A}(|S_{x,z}\rangle, \mathrm{Com}(S))$ outputs $\pi \in S^{\perp} + z$, then x has a lot of conditional min-entropy

$$\mathcal{A}(|S_{x,z}\rangle, \mathrm{Com}(S)) \to \pi$$

$\downarrow$ Purify

$$\sum_{x \in \mathrm{co}(S)} |x\rangle \qquad \mathcal{A}(|S_{x,z}\rangle, \mathrm{Com}(S)) \to \pi$$

$\downarrow$ U_S

$$\sum_{v \in S^{\perp}} |v\rangle \qquad \mathcal{A}(\mathrm{H}^{\otimes n}|v + z\rangle, \mathrm{Com}(S)) \to \pi$$

$\downarrow$ Project

$$|\pi - z\rangle$$

$\downarrow$ $\mathrm{U}_S^{\dagger}$

$$\sum_{x \in \mathrm{co}(S)} (-1)^{(\pi - z)\cdot x} |x\rangle$$

For $x \in \mathrm{co}(S)$: $\mathrm{U}_S|x\rangle \to \sum_{v \in S^{\perp}} (-1)^{v \cdot x}|v\rangle$

For $v \in S^{\perp}$: $\mathrm{U}_S^{\dagger}|v\rangle \to \sum_{x \in \mathrm{co}(S)} (-1)^{v \cdot x}|x\rangle$

Claim: if $\mathcal{A}$ given random $v + z$ and outputs $\pi \in S^{\perp} + z$, then $\pi = v + z$ with overwhelming probability (over S, z)

Measuring gives a uniformly random $x \in \mathrm{co}(S)$, *independent* of $\mathcal{A}$'s view

Outline

1. Basic scenario and applications
2. Recipe for constructions
3. Security
4. Certifiable deletion of programs

Plan

- (Indistinguishability) obfuscation with certified deletion
- Applications
- Comparison with other notions

copy-protection

copy-detection

revocable crypto / key leasing

secure software leasing

Obfuscation with Certified Deletion

Obfuscation with Certified Deletion

Rough goal:

Obfuscation with Certified Deletion

Rough goal:

- Encode a program f into a deletable quantum state

Obfuscation with Certified Deletion

Rough goal:
- Encode a program f into a deletable quantum state
- Before deletion, the program is useful in some way, after deletion it is not

Obfuscation with Certified Deletion

Rough goal:

- Encode a program f into a deletable quantum state
- Before deletion, the program is useful in some way, after deletion it is not

Candidate construction:

[**B**GGKMRR23]

$\left|S_{x,z}\right\rangle, \mathrm{CObf}(\mathrm{P}[S, f \oplus x])$

$\mathrm{P}\left[S, \tilde{f}\right](y, v)$:

- Let x be the coset of S that v belongs to
- Let $f = \tilde{f} \oplus x$
- Output $f(y)$

Obfuscation with Certified Deletion

A "one-way" compiler that scrambles the description of a circuit while maintaining its functionality

Rough goal:

- Encode a program f into a deletable quantum state
- Before deletion, the program is useful in some way, after deletion it is not

Candidate construction: [**B**GGKMRR23]

$$|S_{x,z}\rangle, \mathrm{CObf}(\mathrm{P}[S, f \oplus x])$$

$\mathrm{P}[S, \tilde{f}](y, v)$:

- Let x be the coset of S that v belongs to
- Let $f = \tilde{f} \oplus x$
- Output $f(y)$

Obfuscation with Certified Deletion

Rough goal:

- Encode a program f into a deletable quantum state
- Before deletion, the program is useful in some way, after deletion it is not

Candidate construction:

[**B**GGKMRR23]

$|S_{x,z}\rangle, \mathrm{CObf}(\mathrm{P}[S, f \oplus x])$

$\mathrm{P}[S, \tilde{f}](y, v)$:

- Let x be the coset of S that v belongs to
- Let $f = \tilde{f} \oplus x$
- Output $f(y)$

Correctness: Given any input y, evaluate $\mathrm{Obf}(\mathrm{P}[S, f \oplus x])$ on y and in superposition over $S + x$ to learn $f(y)$

Obfuscation with Certified Deletion

Rough goal:

- Encode a program f into a deletable quantum state
- Before deletion, the program is useful in some way, after deletion it is not

Candidate construction:

[**B**GGKMRR23]

$|S_{x,z}\rangle, \mathrm{CObf}(\mathrm{P}[S, f \oplus x])$

$\mathrm{P}[S, \tilde{f}](y, v)$:

- Let x be the coset of S that v belongs to
- Let $f = \tilde{f} \oplus x$
- Output $f(y)$

Issue with security: By querying on different $v \notin S + x$, can potentially learn evaluations of functions whose description is related to f

Obfuscation with Certified Deletion

Rough goal:

- Encode a program f into a deletable quantum state
- Before deletion, the program is useful in some way, after deletion it is not

Candidate construction:

[**B**GGKMRR23]

$|S_{x,z}\rangle, \mathrm{CObf}(\mathrm{P}[S, f \oplus x])$

$\mathrm{P}[S, \tilde{f}](y, v)$:

- Let x be the coset of S that v belongs to
- Let $f = \tilde{f} \oplus x$
- Output $f(y)$

Solution: P should only accept *authentic* vectors v derived from the state $|S_{x,z}\rangle$

Obfuscation with Certified Deletion

Rough goal:

- Encode a program f into a deletable quantum state
- Before deletion, the program is useful in some way, after deletion it is not

Candidate construction:

[**B**GGKMRR23]

$$|S_{x,z}\rangle, \mathrm{CObf}(\mathrm{P}[S, T, u, f \oplus x])$$

$\mathrm{P}[S, T, u, \tilde{f}](y, v)$:

- Abort if $v \notin T + u$
- Let x be the coset of S that v belongs to
- Let $f = \tilde{f} \oplus x$
- Output $f(y)$

Solution: P should only accept *authentic* vectors v derived from the state $|S_{x,z}\rangle$

Define authentic vectors via a random superspace $T + u \supset S + x$

Obfuscation with Certified Deletion

Rough goal:

- Encode a program f into a deletable quantum state
- Before deletion, the program is useful in some way, after deletion it is not

Candidate construction: [**B**GGKMRR23]

$$|S_{x,z}\rangle, \mathrm{CObf}(\mathrm{P}[S, T, u, f \oplus x])$$

$\mathrm{P}[S, T, u, \tilde{f}](y, v)$:

- Abort if $v \notin T + u$
- Let x be the coset of S that v belongs to
- Let $f = \tilde{f} \oplus x$
- Output $f(y)$

Solution: P should only accept *authentic* vectors v derived from the state $|S_{x,z}\rangle$
Define authentic vectors via a random superspace $T + u \supset S + x$
Hard for the adversary to query on any authentic vector not in $S + x$

Obfuscation with Certified Deletion

If COb is modeled as a classical oracle:

- Before deletion, evaluator can use the oracle to learn $f(y)$ for any y of their choice
- After deletion (outputting $v \in S^{\perp} + z$), the evaluator cannot learn anything else from the oracle even given unbounded queries

Candidate construction:
[**B**GGKMRR23]

$|S_{x,z}\rangle, \mathrm{CObf}(\mathrm{P}[S, T, u, f \oplus x])$

$\mathrm{P}[S, T, u, \tilde{f}](y, v)$:

- Abort if $v \notin T + u$
- Let x be the coset of S that v belongs to
- Let $f = \tilde{f} \oplus x$
- Output $f(y)$

Solution: P should only accept *authentic* vectors v derived from the state $|S_{x,z}\rangle$
Define authentic vectors via a random superspace $T + u \supset S + x$
Hard for the adversary to query on any authentic vector not in $S + x$

Without Oracles…

Without Oracles…

Indistinguishability obfuscation

Without Oracles…

Indistinguishability obfuscation

- For any two functionally equivalent circuits $\mathrm{C}_0, \mathrm{C}_1, \mathrm{Obf}(\mathrm{C}_0) \approx_c \mathrm{Obf}(\mathrm{C}_1)$

Without Oracles…

Indistinguishability obfuscation with certified deletion

- For any two functionally equivalent circuits $\mathrm{C}_0, \mathrm{C}_1, \mathrm{Obf}(\mathrm{C}_0) \approx_c \mathrm{Obf}(\mathrm{C}_1)$, and after deletion $\mathrm{Obf}(\mathrm{C}_0) \approx_s \mathrm{Obf}(\mathrm{C}_1)$

Without Oracles…

Indistinguishability obfuscation with certified deletion

- For any two functionally equivalent circuits $\mathrm{C}_0, \mathrm{C}_1, \mathrm{Obf}(\mathrm{C}_0) \approx_c \mathrm{Obf}(\mathrm{C}_1)$, and after deletion $\mathrm{Obf}(\mathrm{C}_0) \approx_s \mathrm{Obf}(\mathrm{C}_1)$

Satisfied by a slightly modified construction

Without Oracles…

Indistinguishability obfuscation with certified deletion

- For any two functionally equivalent circuits $\mathrm{C}_0, \mathrm{C}_1, \mathrm{Obf}(\mathrm{C}_0) \approx_c \mathrm{Obf}(\mathrm{C}_1)$, and after deletion $\mathrm{Obf}(\mathrm{C}_0) \approx_s \mathrm{Obf}(\mathrm{C}_1)$

Satisfied by a slightly modified construction

Seems like a weak guarantee, but (*differing inputs*) iO with CD are useful tools:

Without Oracles…

Indistinguishability obfuscation with certified deletion

- For any two functionally equivalent circuits $\mathrm{C}_0, \mathrm{C}_1, \mathrm{Obf}(\mathrm{C}_0) \approx_c \mathrm{Obf}(\mathrm{C}_1)$, and after deletion $\mathrm{Obf}(\mathrm{C}_0) \approx_s \mathrm{Obf}(\mathrm{C}_1)$

Satisfied by a slightly modified construction

Seems like a weak guarantee, but (*differing inputs*) iO with CD are useful tools:

- Two-message delegation with certified deletion

Without Oracles…

Indistinguishability obfuscation with certified deletion

- For any two functionally equivalent circuits $\mathrm{C}_0, \mathrm{C}_1, \mathrm{Obf}(\mathrm{C}_0) \approx_c \mathrm{Obf}(\mathrm{C}_1)$, and after deletion $\mathrm{Obf}(\mathrm{C}_0) \approx_s \mathrm{Obf}(\mathrm{C}_1)$

Satisfied by a slightly modified construction

Seems like a weak guarantee, but (*differing inputs*) iO with CD are useful tools:

- Two-message delegation with certified deletion
- A generic compiler from encryption to encryption with *revocable secret keys*

Encryption with Revocable / Deletable Secret Keys

- $\text{Gen} \rightarrow \text{pk}, \text{vk}, |\text{sk}\rangle$
- $\text{Enc}(\text{pk}, m) \rightarrow \text{ct}$
- $\text{Dec}(|\text{sk}\rangle, \text{ct}) \rightarrow m$
- $\text{Del}(|\text{sk}\rangle) \rightarrow \text{cert}$
- $\text{Ver}(\text{vk}, \text{cert}) \rightarrow \top/\bot$

Encryption with Revocable / Deletable Secret Keys

- $\mathrm{Gen} \to \mathrm{pk}, \mathrm{vk}, |\mathrm{sk}\rangle$
- $\mathrm{Enc}(\mathrm{pk}, m) \to \mathrm{ct}$
- $\mathrm{Dec}(|\mathrm{sk}\rangle, \mathrm{ct}) \to m$
- $\mathrm{Del}(|\mathrm{sk}\rangle) \to \mathrm{cert}$
- $\mathrm{Ver}(\mathrm{vk}, \mathrm{cert}) \to \top/\bot$

Deletion security: ciphertexts generated after successful deletion of $|\mathrm{sk}\rangle$ are semantically secure

Encryption with Revocable / Deletable Secret Keys

- $\mathrm{Gen} \rightarrow \mathrm{pk}, \mathrm{vk}, |\mathrm{sk}\rangle$
- $\mathrm{Enc}(\mathrm{pk}, m) \rightarrow \mathrm{ct}$
- $\mathrm{Dec}(|\mathrm{sk}\rangle, \mathrm{ct}) \rightarrow m$
- $\mathrm{Del}(|\mathrm{sk}\rangle) \rightarrow \mathrm{cert}$
- $\mathrm{Ver}(\mathrm{vk}, \mathrm{cert}) \rightarrow \top/\bot$

Deletion security: ciphertexts generated after successful deletion of $|\mathrm{sk}\rangle$ are semantically secure

Simple compiler: $|\mathrm{sk}\rangle = \mathrm{iOCD}(\mathrm{Dec}(\mathrm{sk}, \cdot))$ [**B**GGMKRR23]

Encryption with Revocable / Deletable Secret Keys

- $\text{Gen} \to \text{pk}, \text{vk}, |\text{sk}\rangle$
- $\text{Enc}(\text{pk}, m) \to \text{ct}$
- $\text{Dec}(|\text{sk}\rangle, \text{ct}) \to m$
- $\text{Del}(|\text{sk}\rangle) \to \text{cert}$
- $\text{Ver}(\text{vk}, \text{cert}) \to \top/\bot$

Deletion security: ciphertexts generated after successful deletion of $|\text{sk}\rangle$ are semantically secure

Simple compiler: $|\text{sk}\rangle = \text{iOCD}(\text{Dec}(\text{sk}, \cdot))$ [**B**GGMKRR23]

Gives *publicly-verifiable revocation* if iOCD is publicly verifiable

Encryption with Revocable / Deletable Secret Keys

- $\mathrm{Gen} \rightarrow \mathrm{pk}, \mathrm{vk}, |\mathrm{sk}\rangle$
- $\mathrm{Enc}(\mathrm{pk}, m) \rightarrow \mathrm{ct}$
- $\mathrm{Dec}(|\mathrm{sk}\rangle, \mathrm{ct}) \rightarrow m$
- $\mathrm{Del}(|\mathrm{sk}\rangle) \rightarrow \mathrm{cert}$
- $\mathrm{Ver}(\mathrm{vk}, \mathrm{cert}) \rightarrow \top/\bot$

Deletion security: ciphertexts generated after successful deletion of $|\mathrm{sk}\rangle$ are semantically secure

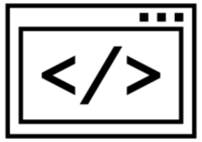
Simple compiler: $|\mathrm{sk}\rangle = \mathrm{iOCD}(\mathrm{Dec}(\mathrm{sk}, \cdot))$ [**B**GGMKRR23]

Gives *publicly-verifiable revocation* if iOCD is publicly verifiable

Privately-verifiable revocation from standard assumptions: [Kitagawa, Nishimaki 22], [Agarwal, Kitagawa, Nishimaki, Yamada, Yamakawa 23], [Ananth, Poremba, Vaikuntanathan 23]

Related Notions

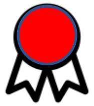
Hard for the adversary to produce…



“working” copy of a program

certificate derived from program

publicly verifiable

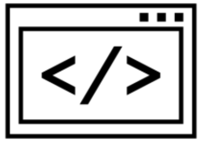


privately verifiable

Related Notions

Hard for the adversary to produce…

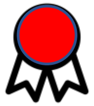
Copy Protection: [Aaronson 09]		



"working" copy of a program

certificate derived from program

publicly verifiable



privately verifiable

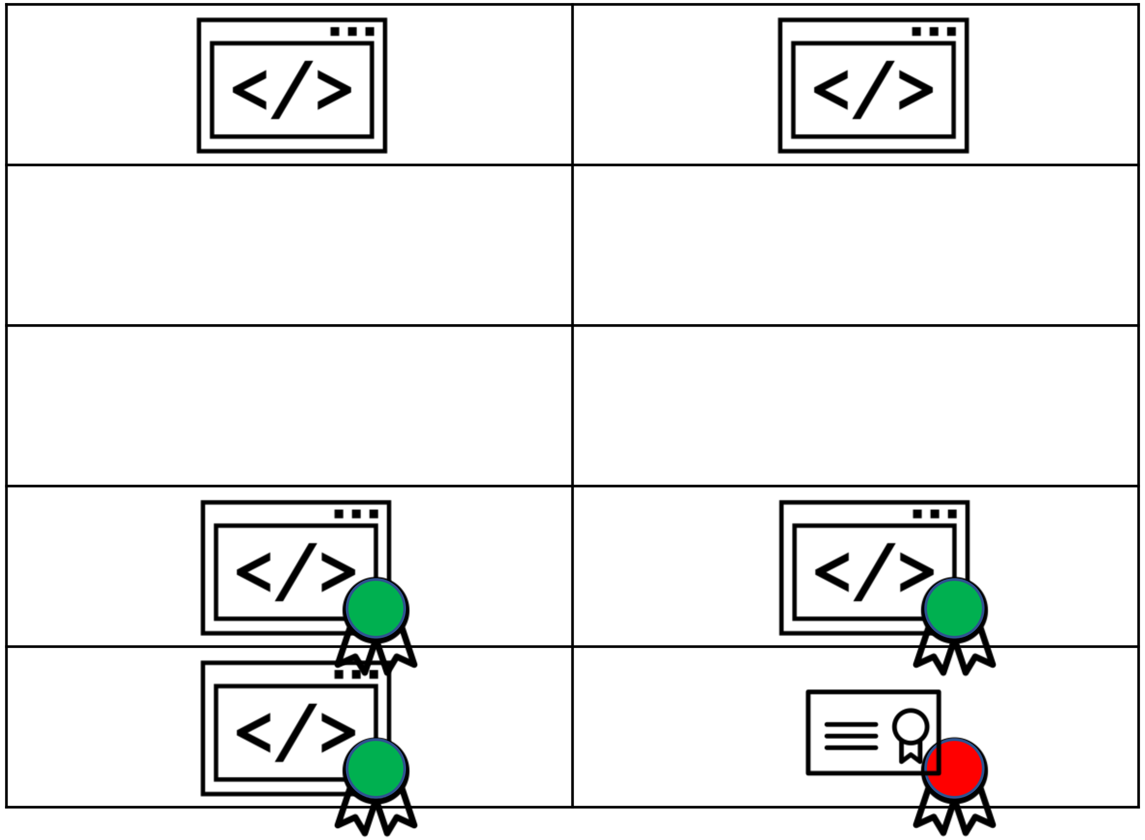
Related Notions

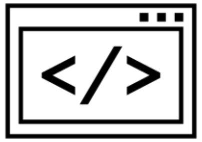
Hard for the adversary to produce…

Copy Protection:
[Aaronson 09]

Copy Detection / Infinite-Term Secure Software Leasing:
[Ananth, La Placa 21], [Aaronson, Liu, Liu, Zhandry, Zhang 22]

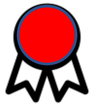
Finite-Term Secure Software Leasing:
[AL21]




“working” copy of a program


certificate derived from program


publicly verifiable


privately verifiable

Related Notions

Hard for the adversary to produce…

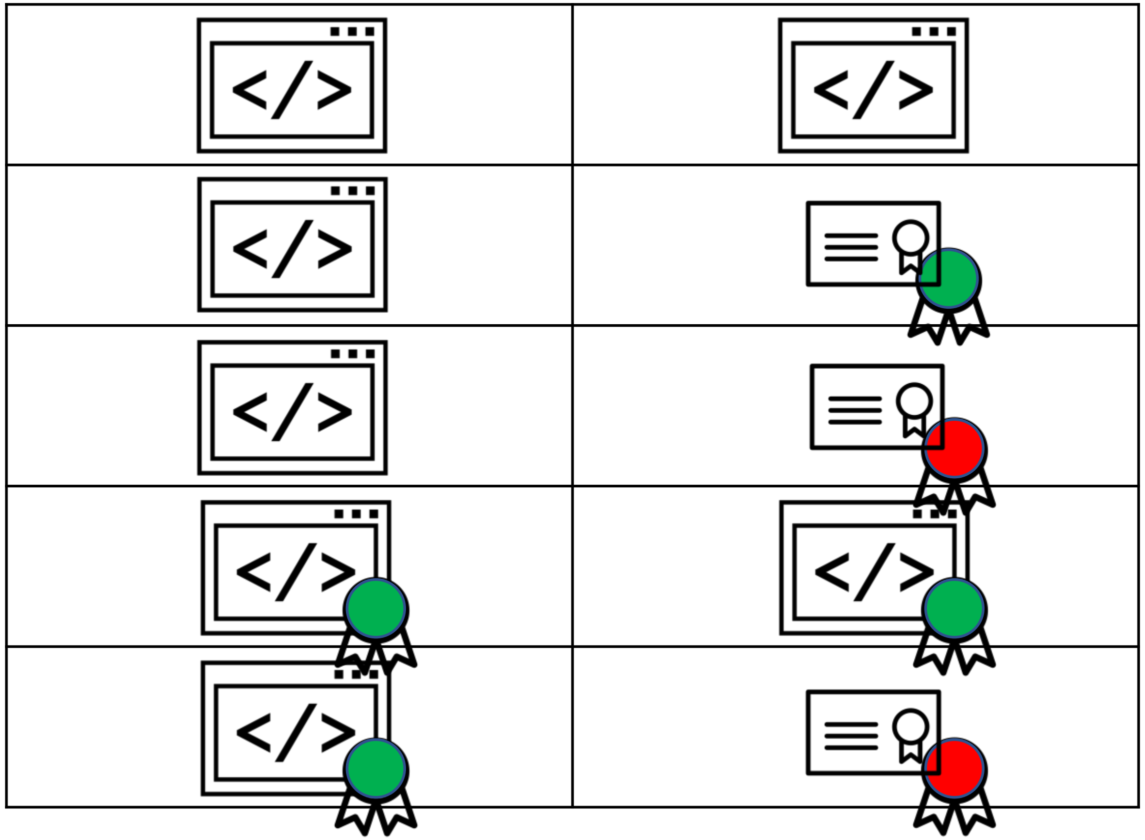
Copy Protection:
[Aaronson 09]

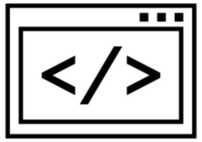
Publicly-Verifiable Deletion / Revocation:
[**B**GGKMRR23]

Privately-Verifiable Deletion / Revocation:
[KN22], [AKNYY23], [APV23]

Copy Detection / Infinite-Term Secure Software Leasing:
[Ananth, La Placa 21], [Aaronson, Liu, Liu, Zhandry, Zhang 22]

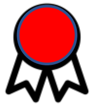
Finite-Term Secure Software Leasing:
[AL21]



 “working” copy of a program

 certificate derived from program

 publicly verifiable

 privately verifiable

Future Directions

Future Directions

- Prove the security of $\mathrm{CDExp}_{\mathcal{C},\mathcal{H},\mathcal{D},\mathcal{A}_1,\mathcal{A}_2}$ when

Future Directions

- Prove the security of $\mathrm{CDExp}_{\mathcal{C},\mathcal{H},\mathcal{D},\mathcal{A}_1,\mathcal{A}_2}$ when
  - Encoding super-logarithmic bits per subspace state

Future Directions

- Prove the security of $\mathrm{CDExp}_{\mathcal{C},\mathcal{H},\mathcal{D},\mathcal{A}_1,\mathcal{A}_2}$ when
  - Encoding super-logarithmic bits per subspace state
  - $\mathcal{C}$ is any semantically-secure distribution and $\mathcal{H}$ is a good *seeded* randomness extractor

Future Directions

- Prove the security of $\mathrm{CDExp}_{\mathcal{C},\mathcal{H},\mathcal{D},\mathcal{A}_1,\mathcal{A}_2}$ when
  - Encoding super-logarithmic bits per subspace state
  - $\mathcal{C}$ is any semantically-secure distribution and $\mathcal{H}$ is a good *seeded* randomness extractor

- Robustness to noise (beyond one-time pad [BI20])

Future Directions

- Prove the security of $\mathrm{CDExp}_{\mathcal{C},\mathcal{H},\mathcal{D},\mathcal{A}_1,\mathcal{A}_2}$ when
  - Encoding super-logarithmic bits per subspace state
  - $\mathcal{C}$ is any semantically-secure distribution and $\mathcal{H}$ is a good *seeded* randomness extractor

- Robustness to noise (beyond one-time pad [BI20])

- Publicly-verifiable revocation/deletion without post-quantum iO

Future Directions

- Prove the security of $\mathrm{CDExp}_{\mathcal{C},\mathcal{H},\mathcal{D},\mathcal{A}_1,\mathcal{A}_2}$ when
  - Encoding super-logarithmic bits per subspace state
  - $\mathcal{C}$ is any semantically-secure distribution and $\mathcal{H}$ is a good *seeded* randomness extractor
- Robustness to noise (beyond one-time pad [BI20])
- Publicly-verifiable revocation/deletion without post-quantum iO
- More rigorous understanding of the relationship between unclonable primitives from previous slide ([Ananth, Kaleoglu, Liu 23])