The Pseudorandom Oracle Model and Ideal Obfuscation

Aayush Jain   Rachel Lin   Ji Luo 罗辑   Daniel Wichs

Carnegie Mellon University   UNIVERSITY of WASHINGTON   Northeastern University   NTT Research

Program Obfuscation (for Circuits)

C

```
basicFun = Function[{Typed[pixel0, "ComplexReal64"]},
    Module[{iters = 1, maxIters = 1000, pixel = pixel0},
      While[iters < maxIters && Abs[pixel] < 2,
        pixel = pixel^2 + pixel0;
        iters++
      ];
      iters]];
```

$\mathrm{Obf}(\cdot)$

$\tilde{C}$

Correctness.

$\mathrm{Obf}(C)(x) = C(x)$

Security.

$\mathrm{Obf}(C)$ is "***unintelligible***"

Indistinguishability Obfuscation ($i\mathcal{O}$)

$$|C_0| = |C_1| \text{ and } \forall x : C_0(x) = C_1(x) \quad \Longrightarrow \quad \mathrm{Obf}(C_0) \approx \mathrm{Obf}(C_1)$$

Indistinguishability Obfuscation ($i\mathcal{O}$)

$$|C_0| = |C_1| \text{ and } \forall x: C_0(x) = C_1(x) \implies \mathrm{Obf}(C_0) \approx \mathrm{Obf}(C_1)$$

very powerful primitive

deniable encryption, short signatures, injective TDF, NIZK, OT [SW]
FHE [ABFGGTW] FE for P [GGHRSW] 2-round MPC [GGHR]
succinct garbled RAM [CH] (+ many more!)

Indistinguishability Obfuscation ($i\mathcal{O}$)

$$|C_0| = |C_1| \text{ and } \forall x: C_0(x) = C_1(x) \quad \Longrightarrow \quad \mathrm{Obf}(C_0) \approx \mathrm{Obf}(C_1)$$

- **very powerful primitive**

  deniable encryption, short signatures, injective TDF, NIZK, OT [SW]
  FHE [ABFGGTW] FE for P [GGHRSW] 2-round MPC [GGHR]
  succinct garbled RAM [CH] (+ many more!)

- **from well-studied assumptions [JLS]**

Indistinguishability Obfuscation ($i\mathcal{O}$)

$$|C_0| = |C_1| \text{ and } \forall x: C_0(x) = C_1(x) \implies \mathrm{Obf}(C_0) \approx \mathrm{Obf}(C_1)$$

- ✅ **very powerful primitive**

  deniable encryption, short signatures, injective TDF, NIZK, OT [SW]
  FHE [ABFGGTW] FE for P [GGHRSW] 2-round MPC [GGHR]
  succinct garbled RAM [CH] (+ many more!)

- ✅ **from well-studied assumptions [JLS]**

🤔 **What else to desire?**

⚠️ **weak, unintuitive security guarantees**

⚠️ **unclear security for natural usages**

⚠️ **convoluted techniques for applications**

Natural Applications of Obfuscation

Cryptographic Program

secret-key encryption + obfuscation $\Longrightarrow$ **public**-key encryption [DH]

$$\mathrm{sk} = \mathrm{skeK}$$
$$\mathrm{pk} = \mathrm{Obf}(\mathrm{skeEnc}(\mathrm{skeK}, \cdot))$$

Natural Applications of Obfuscation

Cryptographic Program

secret-key encryption + obfuscation $\Longrightarrow$ **public**-key encryption [DH]

$$\mathrm{sk} = \mathrm{skeK}$$
$$\mathrm{pk} = \mathrm{Obf}(\mathrm{skeEnc}(\mathrm{skeK}, \cdot))$$

maybe $i\mathcal{O}$ + techniques (e.g., puncturing)

Natural Applications of Obfuscation

Cryptographic Program

secret-key encryption + obfuscation $\Longrightarrow$ public-key encryption [DH]

$\mathrm{sk} = \mathrm{skeK}$

$\mathrm{pk} = \mathrm{Obf}(\mathrm{skeEnc}(\mathrm{skeK}, \cdot))$

maybe $i\mathcal{O}$ + techniques (e.g., puncturing)

Non-Cryptographic Programs

software vulnerability patches

programs with copyright protection

machine learning models

...

Natural Applications of Obfuscation

Cryptographic Program

secret-key encryption + obfuscation $\Longrightarrow$ **public**-key encryption [DH]

$$\mathrm{sk} = \mathrm{skeK}$$
$$\mathrm{pk} = \mathrm{Obf}(\mathrm{skeEnc}(\mathrm{skeK}, \cdot))$$

maybe $i\mathcal{O}$ + techniques (e.g., puncturing)

Non-Cryptographic Programs

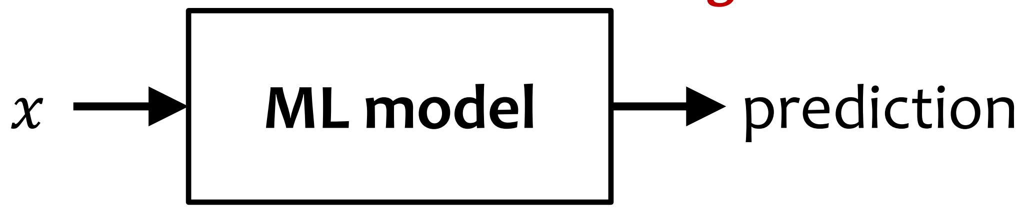
software vulnerability patches

programs with copyright protection

machine learning models

…

Model parameters could **leak sensitive training data.**



Natural Applications of Obfuscation

Cryptographic Program

secret-key encryption + obfuscation $\Longrightarrow$ public-key encryption [DH]

$$\mathsf{sk} = \mathsf{skeK}$$
$$\mathsf{pk} = \mathsf{Obf}(\mathsf{skeEnc}(\mathsf{skeK}, \cdot))$$

maybe $i\mathcal{O}$ + techniques (e.g., puncturing)

Non-Cryptographic Programs

software vulnerability patches
programs with copyright protection
machine learning models
…

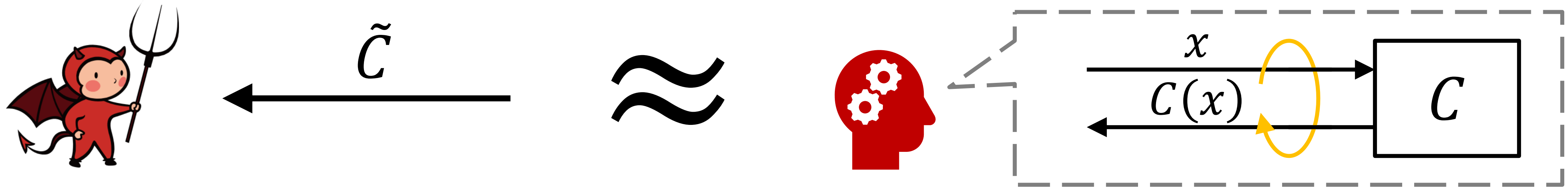
? ? ?

Model parameters could
leak sensitive training data.

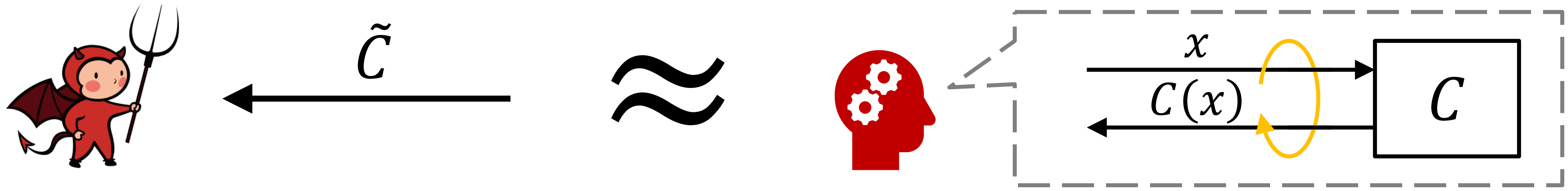
x → **ML model** → prediction

Want. $\mathsf{Obf}(\text{ML model})$ **hides training data.**
(If they cannot be extracted from prediction…)

Simulation-Secure Obfuscation



Simulation-Secure Obfuscation



Ideal Obfuscation.

$\exists \mathcal{S} \quad \forall C:$

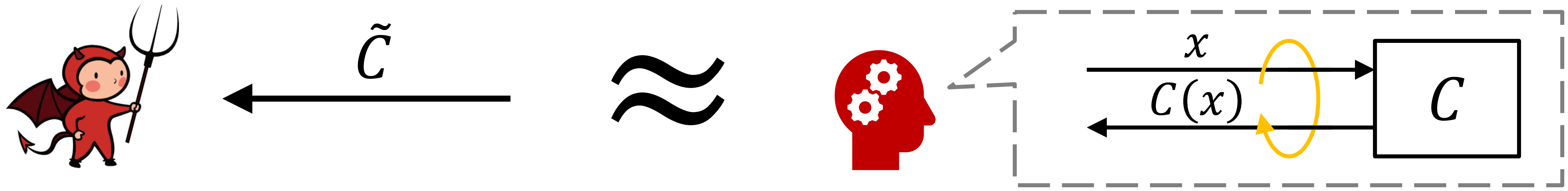
$\mathsf{Obf}(C) \approx \mathcal{S}^C(1^{|C|}, 1^{|x|}).$

Virtual Black-Box Obfuscation.

$\forall$ **one–bit** $\mathcal{A} \quad \exists \mathcal{S}_{\mathcal{A}} \quad \forall C:$

$\mathcal{A}(\mathsf{Obf}(C)) \approx \mathcal{S}_{\mathcal{A}}^C(1^{|C|}, 1^{|x|}).$

Simulation-Secure Obfuscation



Ideal Obfuscation.

$\exists \mathcal{S} \quad \forall C:$

$\mathrm{Obf}(C) \approx \mathcal{S}^{C}(1^{|C|}, 1^{|x|}).$

✖ impossible for unlearnable circuits

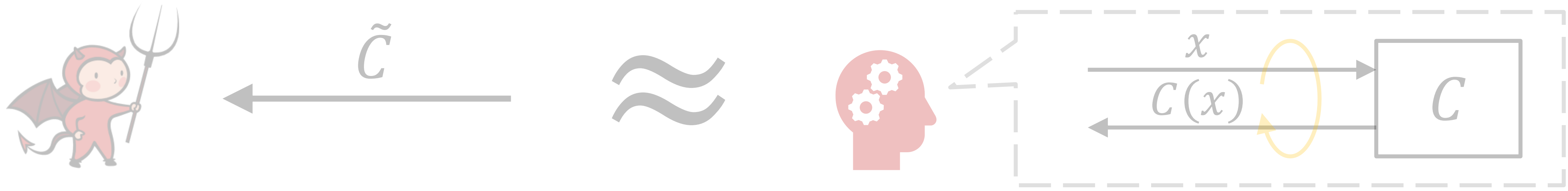
Virtual Black-Box Obfuscation.

$\forall$ one–bit $\mathcal{A} \quad \exists \mathcal{S}_{\mathcal{A}} \quad \forall C:$

$\mathcal{A}(\mathrm{Obf}(C)) \approx \mathcal{S}_{\mathcal{A}}^{C}(1^{|C|}, 1^{|x|}).$

⚠ VBB not possible in general [BGIRSVY]
(contrived "self-eating" programs)

Simulation-Secure Obfuscation



Ideal Obfuscation.

$$\exists \mathcal{S} \quad \forall C:$$
$$\mathrm{Obf}(C) \approx \mathcal{S}^C(1^{|C|}, 1^{|x|}).$$

✗ impossible for unlearnable circuits

Virtual Black-Box Obfuscation.

$\forall$ **one–bit** $\mathcal{A} \quad \exists \mathcal{S}_{\mathcal{A}} \quad \forall C:$
$$\mathcal{A}(\mathrm{Obf}(C)) \approx \mathcal{S}_{\mathcal{A}}^C(1^{|C|}, 1^{|x|}).$$

⚠ VBB not possible in general [BGIRSVY]
(contrived "self-eating" programs)

✓ **strong, intuitive security guarantees**

✓ **simple, intuitive designs in applications**

Simulation-Secure Obfuscation

Ideal Obfuscation.

$\exists \mathcal{S} \quad \forall C$:

$\mathrm{Obf}(C) \approx \mathcal{S}^{C}(1^{|C|}, 1^{|x|})$.

❌ impossible for unlearnable circuits

Virtual Black-Box Obfuscation.

$\forall$ **one–bit** $\mathcal{A} \quad \exists \mathcal{S}_{\mathcal{A}} \quad \forall C$:

$\mathcal{A}(\mathrm{Obf}(C)) \approx \mathcal{S}_{\mathcal{A}}^{C}(1^{|C|}, 1^{|x|})$.

⚠ VBB not possible in general [BGIRSVY]
(contrived "self-eating" programs)

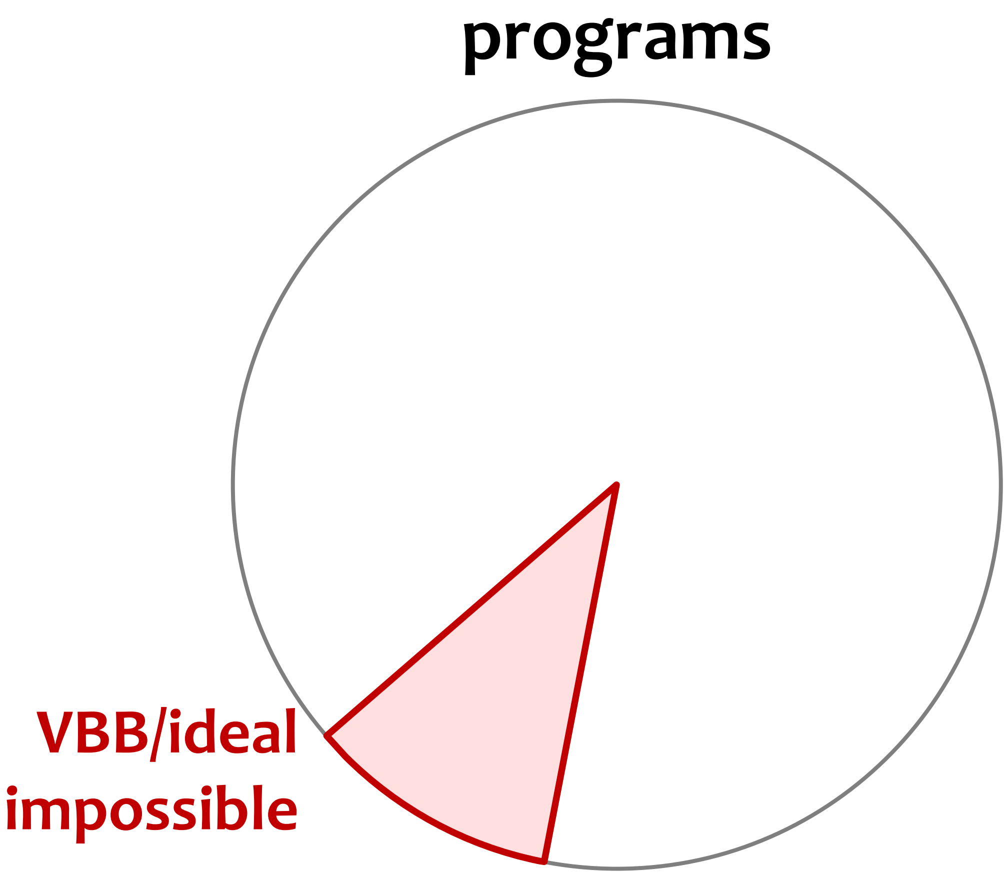
✅ **strong, intuitive security guarantees**

✅ **simple, intuitive designs in applications**

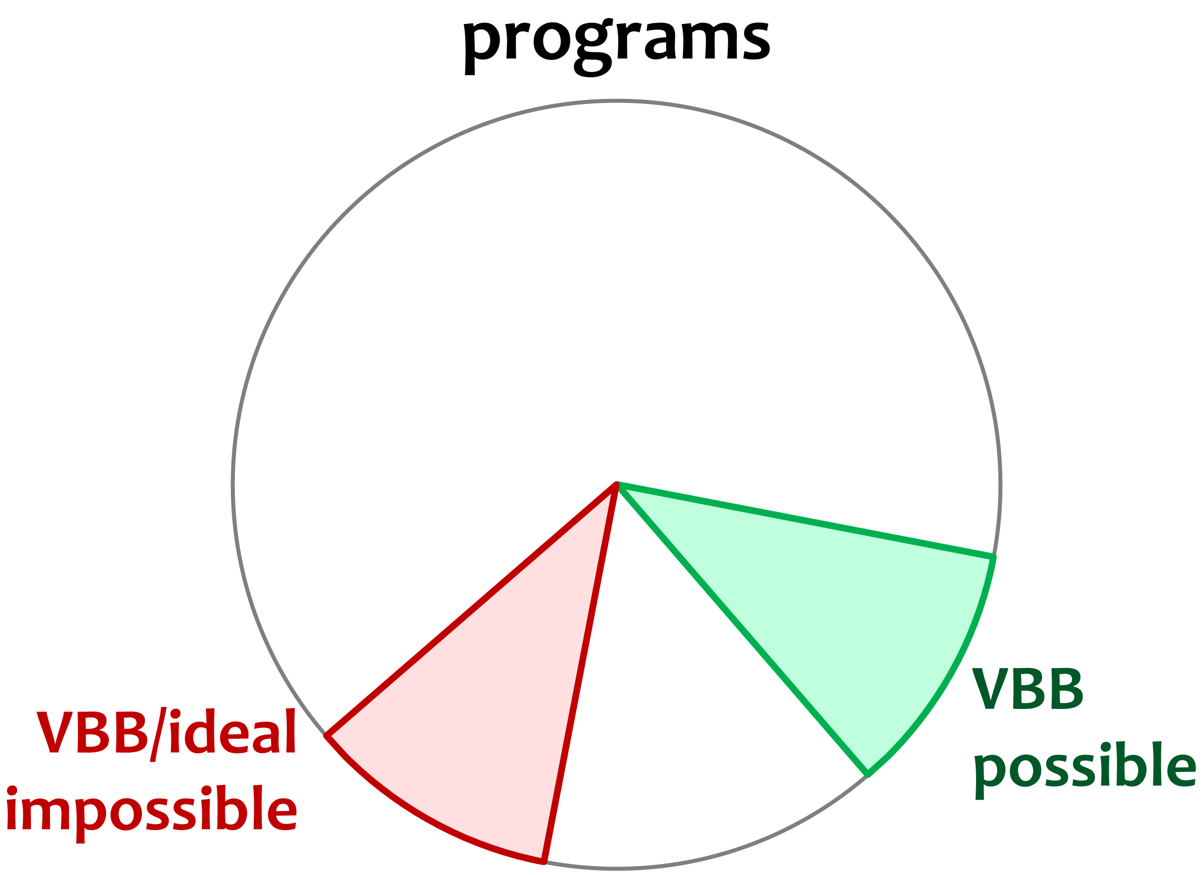
✅ **only path to certain (plausible) applications**

DE-PIR [BIPW,LMW] FHE for RAM [HHWW,LMW]
VGB obfuscation [BC] FE with optimal Dec time [ACFQ]
OT from binary erasure channel [AIKNPPR]
public-coin $di\mathcal{O}$, input-unbounded obfuscation for TM [IPS]
input-hiding obfuscation for evasive functions [BBCKPS]
extractable WE [GKPVZ] wiretap-channel coding [IKLS,IJLSZ]
refuting dream XOR lemma [BIKSW]
strong protection for "non-cryptographic" programs

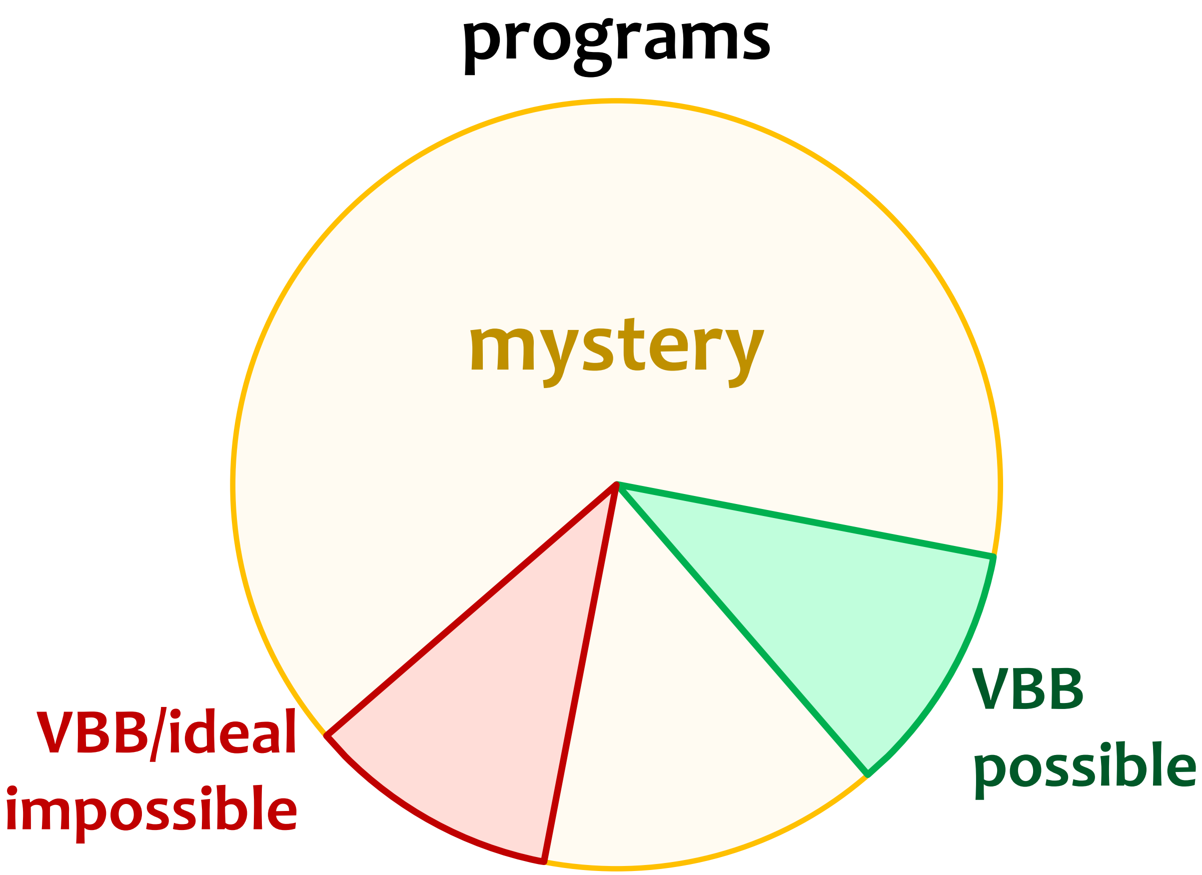
Current State of Obfuscation



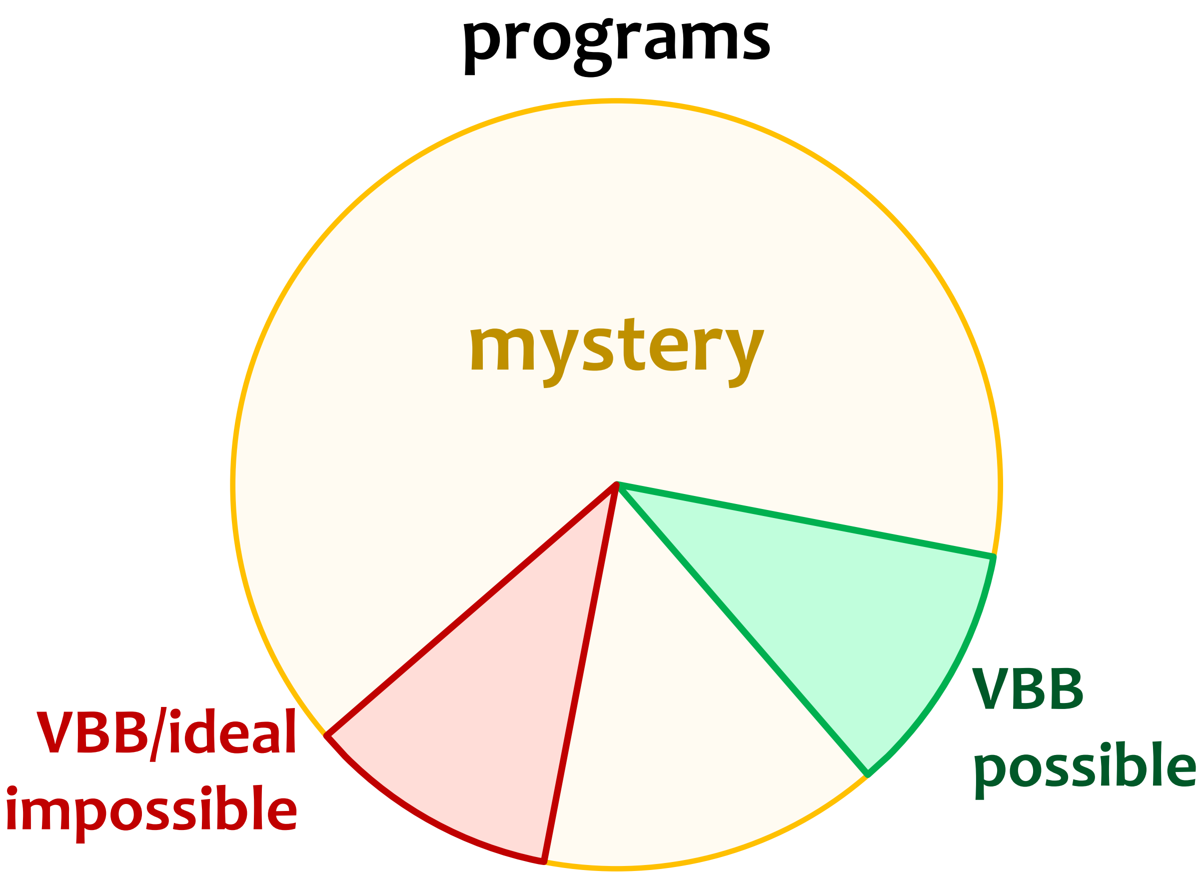
Current State of Obfuscation

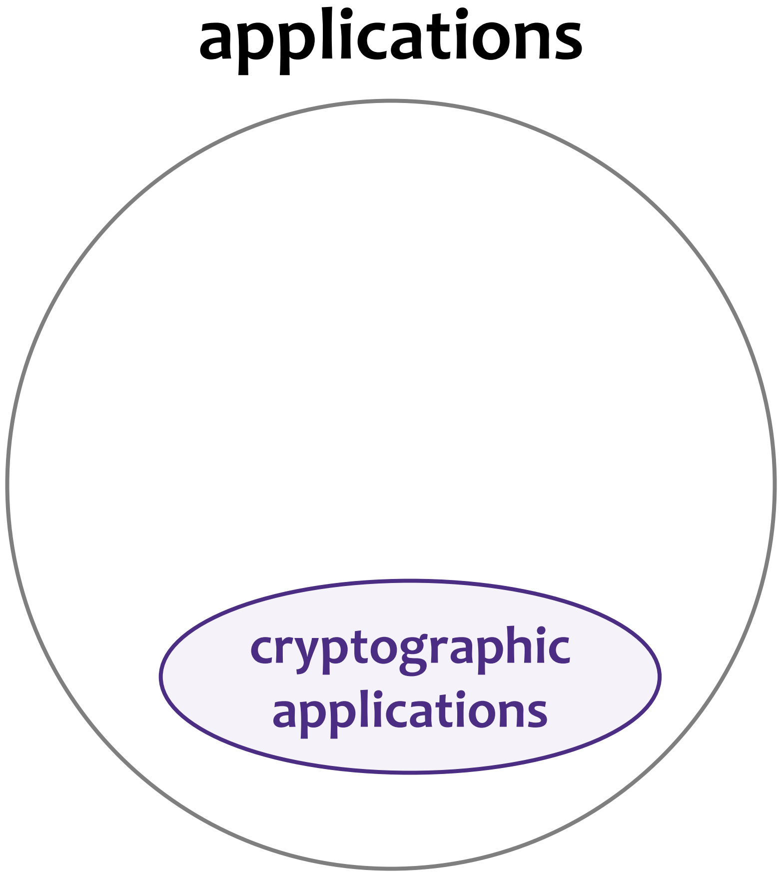


Current State of Obfuscation

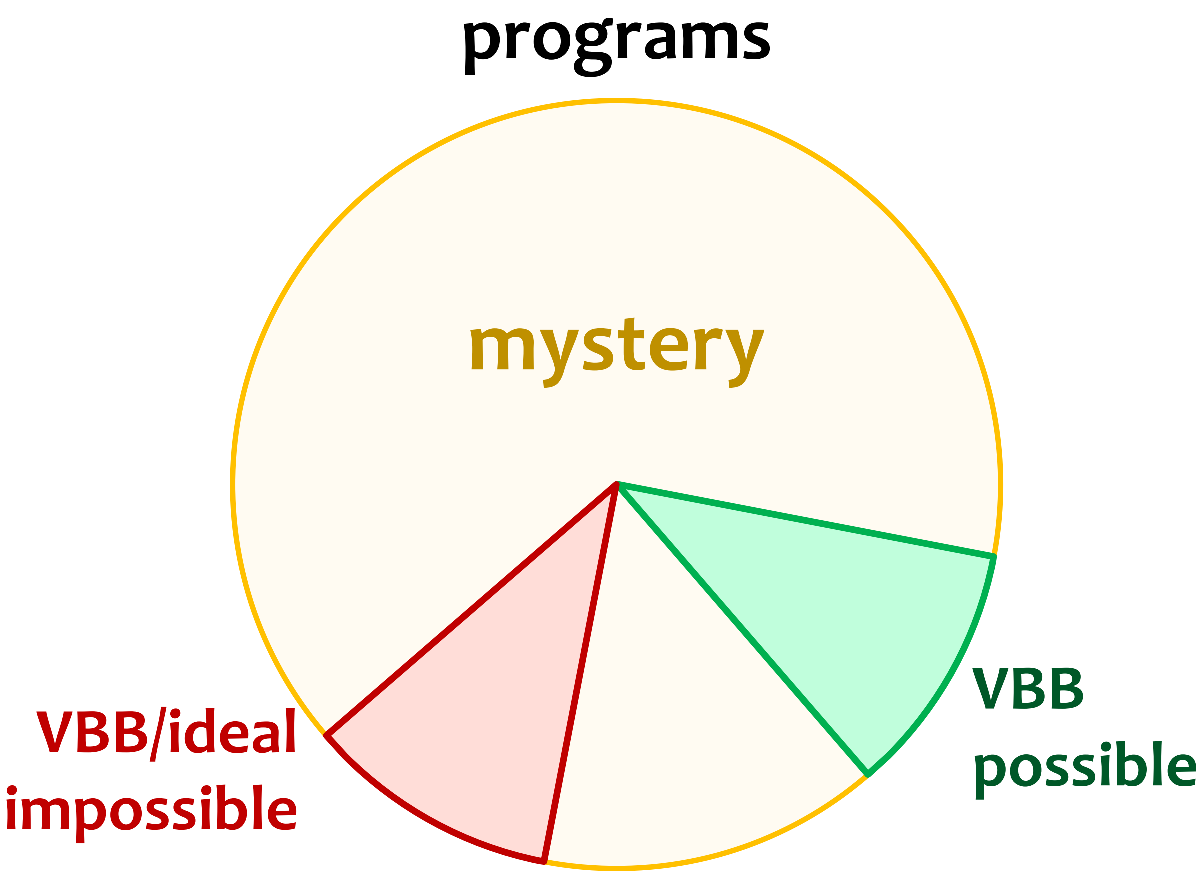


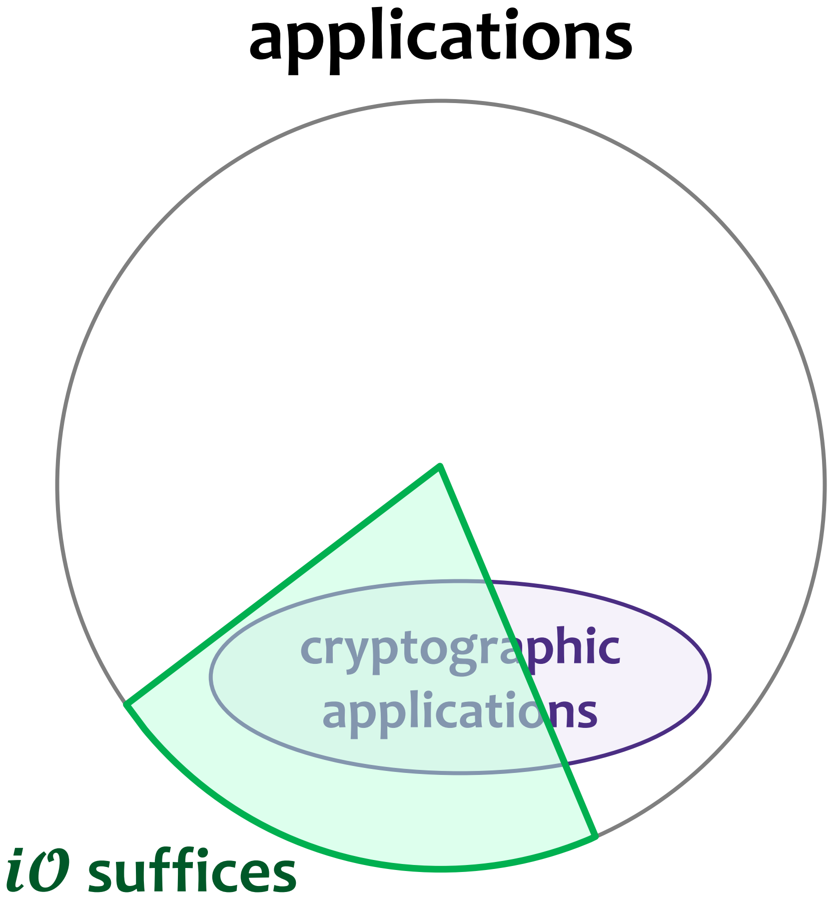
Current State of Obfuscation



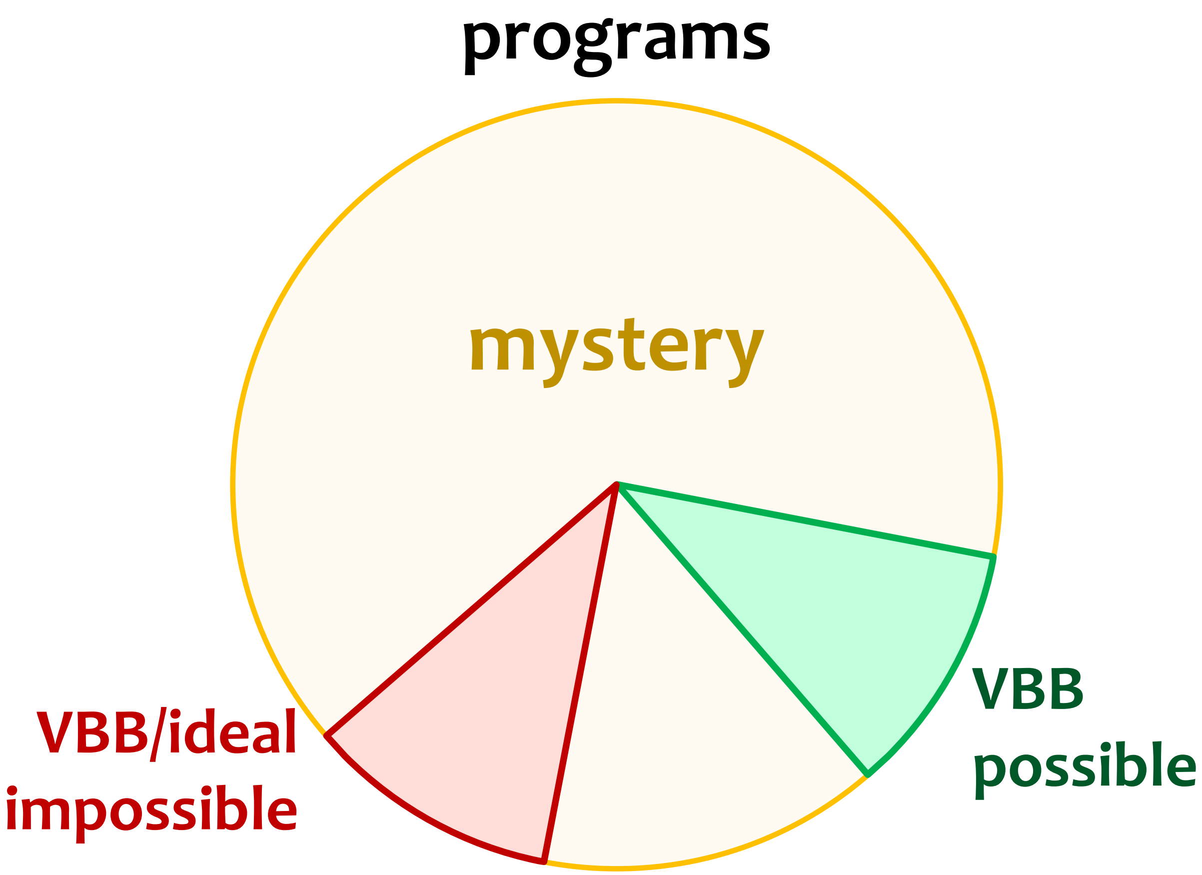


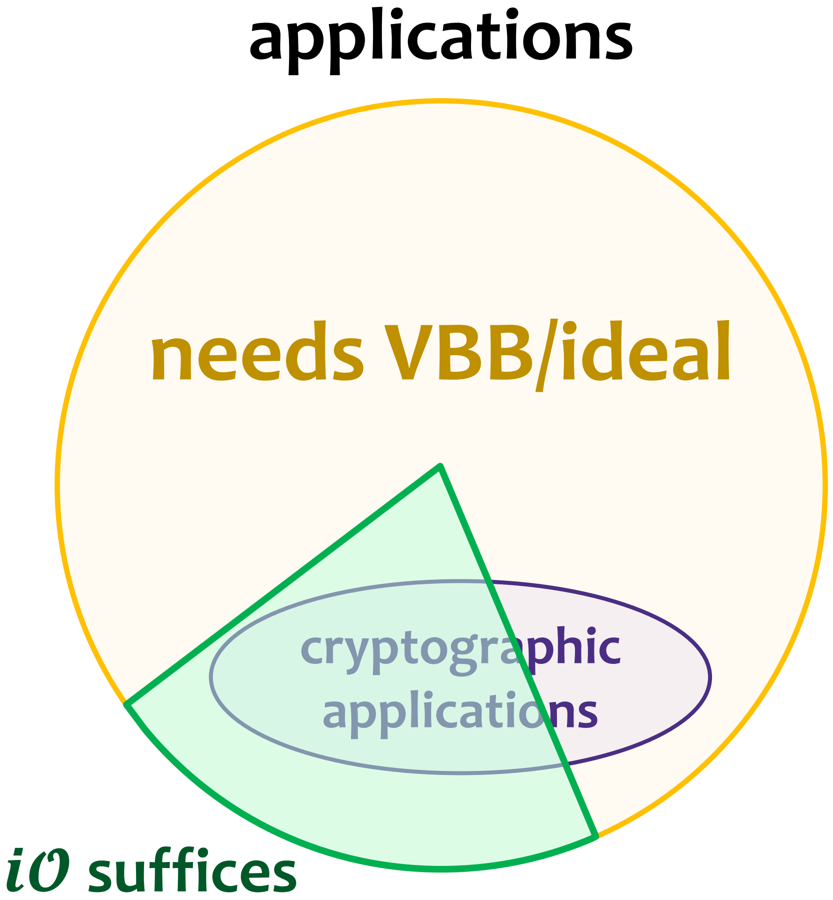
Current State of Obfuscation





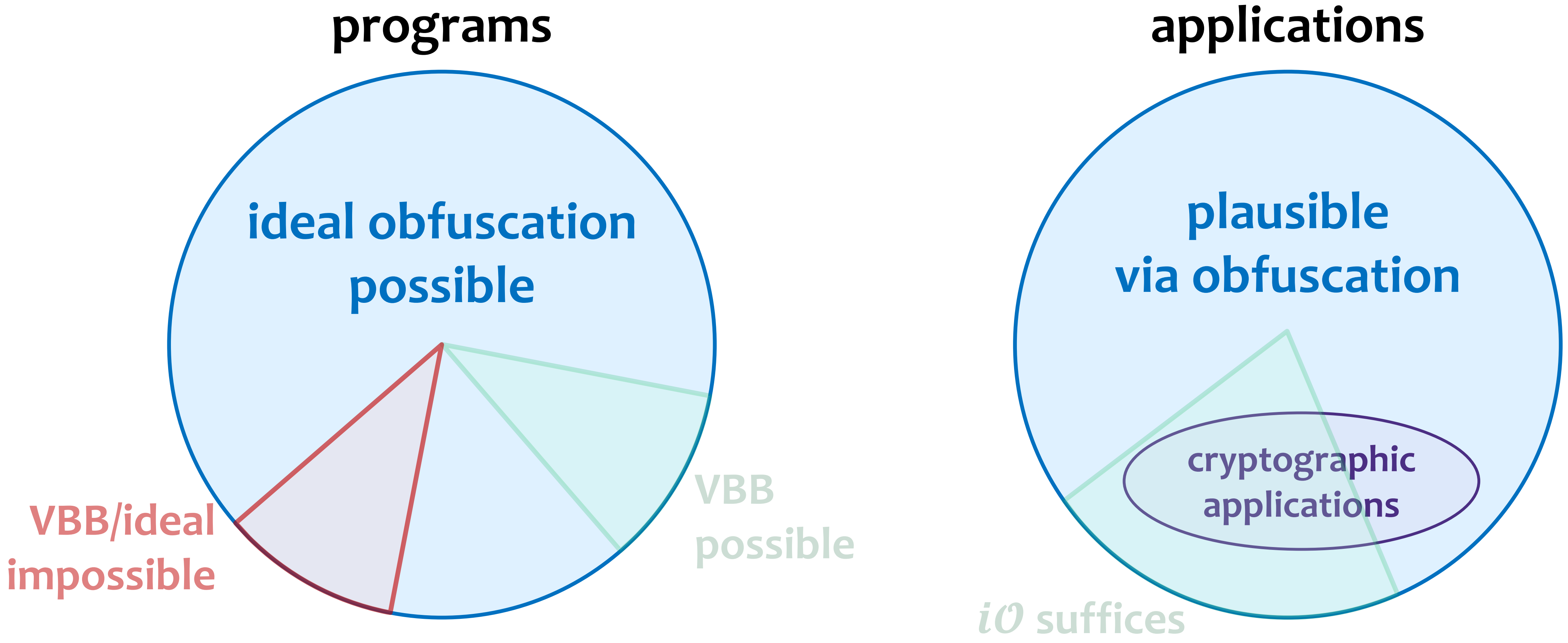
Current State of Obfuscation





Motivation

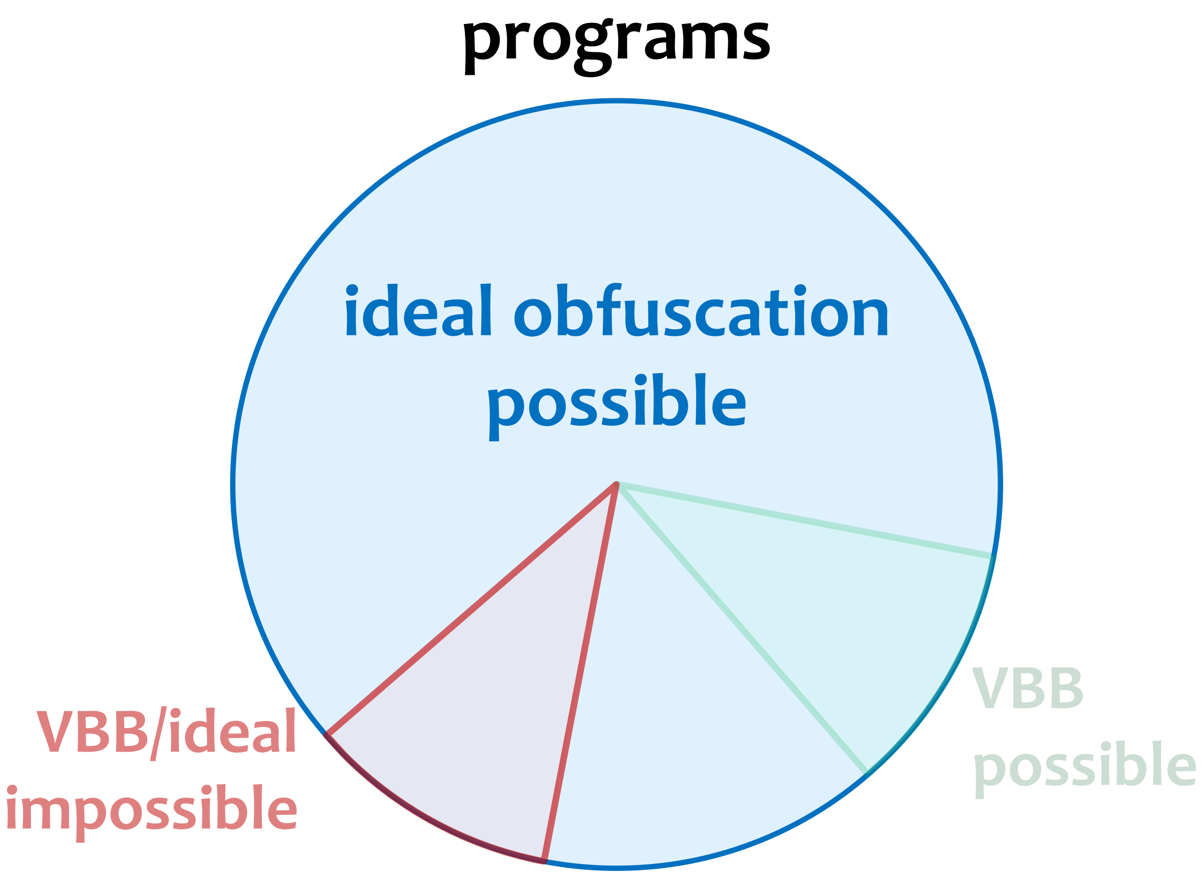
Natural Heuristics. For **natural** programs, **ideal** obfuscation is **possible**!
Natural applications of **ideal** obfuscation are **plausible**!

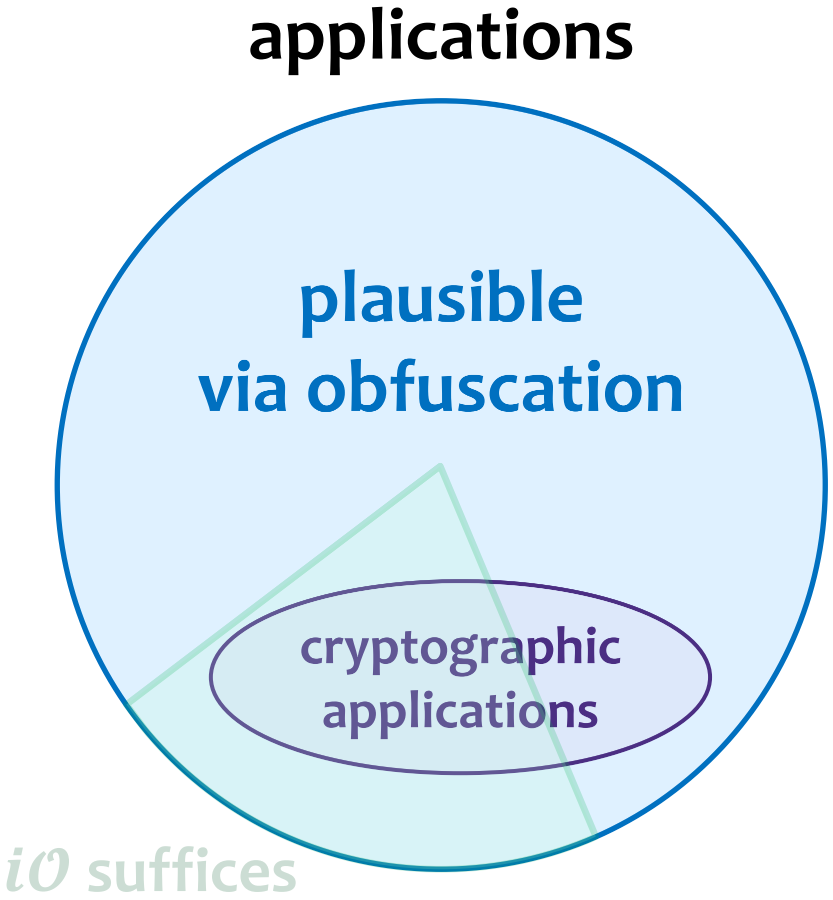


Motivation

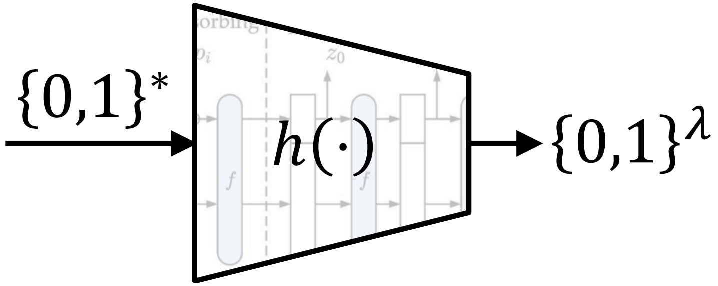
? Can we justify the natural heuristics?

Natural Heuristics. For **natural** programs, **ideal** obfuscation is **possible**!
Natural applications of **ideal** obfuscation are **plausible**!



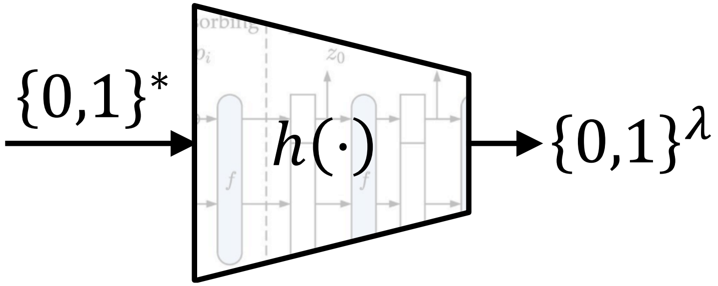


Hash Functions



basic, useful primitive

Hash Functions



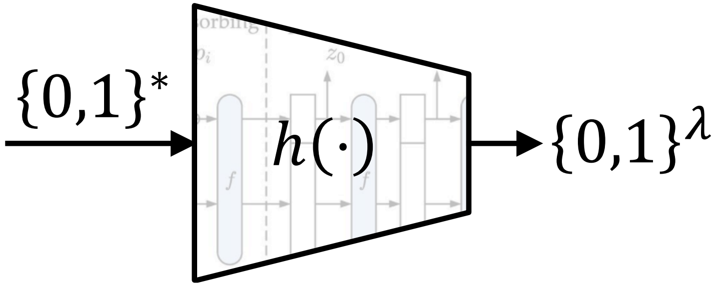
$$\boxed{\mathrm{Sign}(\mathrm{sk}, m) \stackrel{\mathrm{def}}{=} \mathrm{Sign}_\lambda\big(\mathrm{sk}, h(m)\big)}$$

UF-CMA $\Longleftarrow$ UF-CMA + CR

basic, useful primitive

- one-wayness
- second-preimage resistance
- collision resistance

Hash Functions



basic, useful primitive

- one-wayness
- second-preimage resistance
- collision resistance

$$\mathrm{Sign}(\mathrm{sk}, m) \stackrel{\text{def}}{=} \mathrm{Sign}_\lambda(\mathrm{sk}, h(m))$$

✅ UF-CMA $\Longleftarrow$ UF-CMA + CR

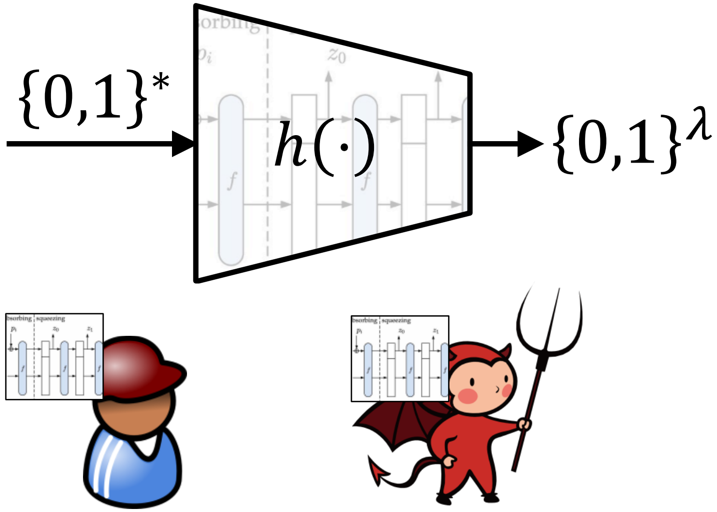
$$\mathrm{Sign}(\mathrm{sk}, m) \stackrel{\text{def}}{=} \mathrm{TDP}^{-1}(\mathrm{sk}, h(m))$$

What assumption for h?

intuition of security **beyond complexity assumptions**

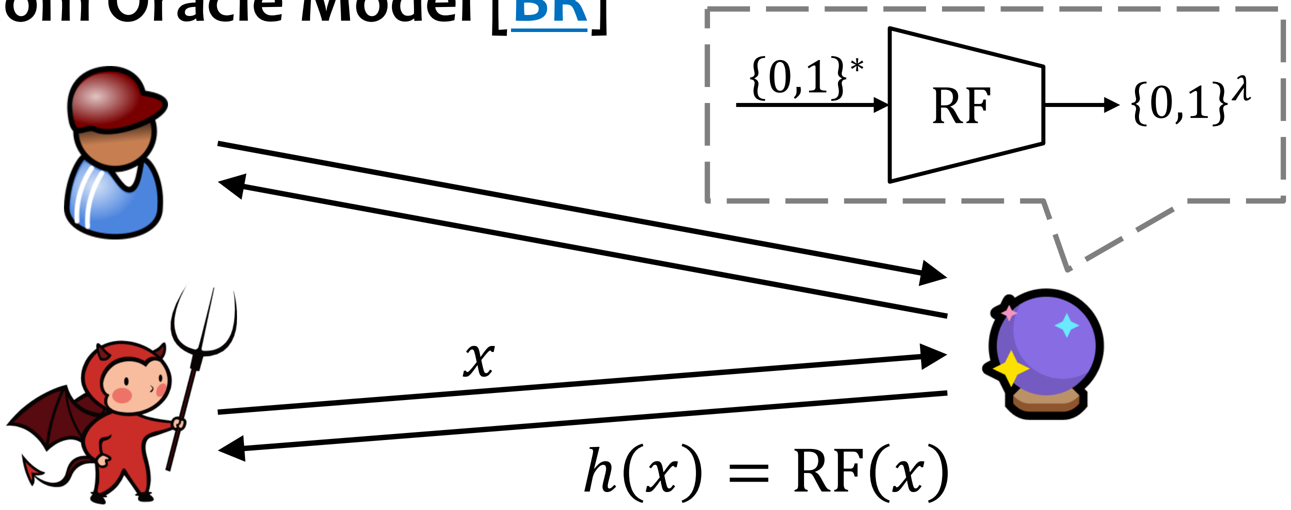
Hash Functions: Idealization

Standard Model



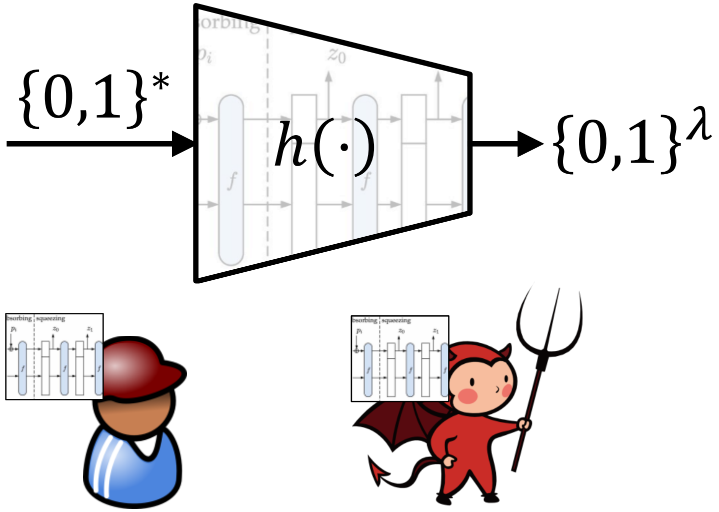
- one-wayness
- second-preimage resistance
- collision resistance

Random Oracle Model [BR]



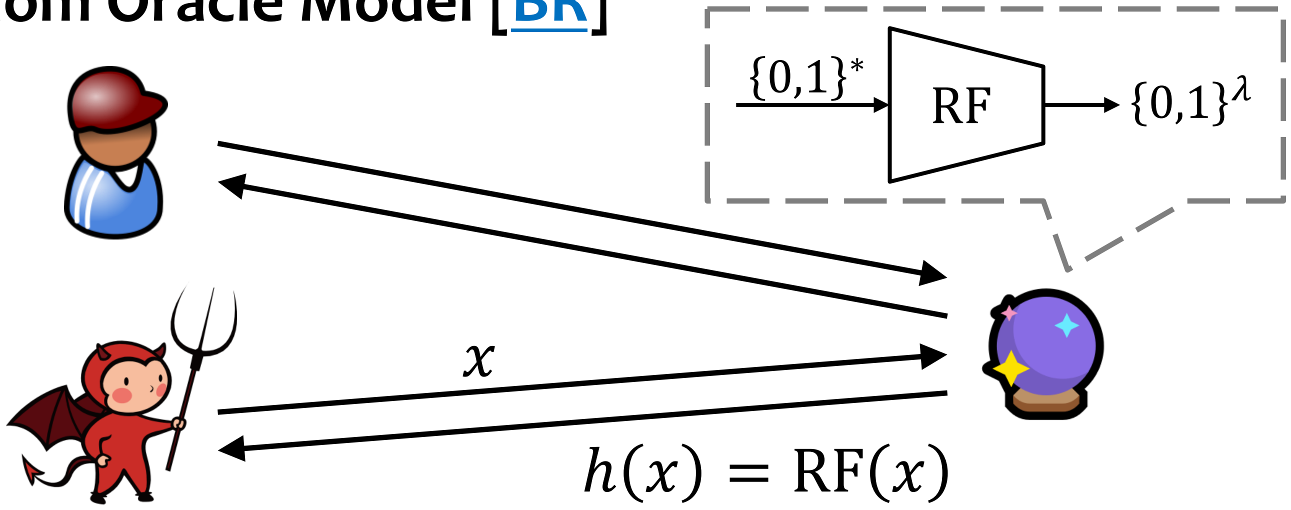
Hash Functions: Idealization

Standard Model



- one-wayness
- second-preimage resistance
- collision resistance

Random Oracle Model [BR]

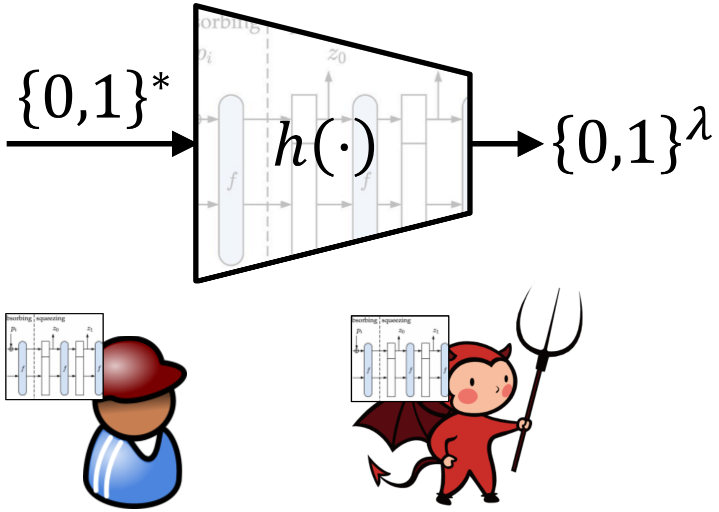


✓ **practically used and more efficient schemes**

Schnorr signature[Schnorr] RSA-OAEP[BR] TLS[KPW,DFGS,DJ] ...

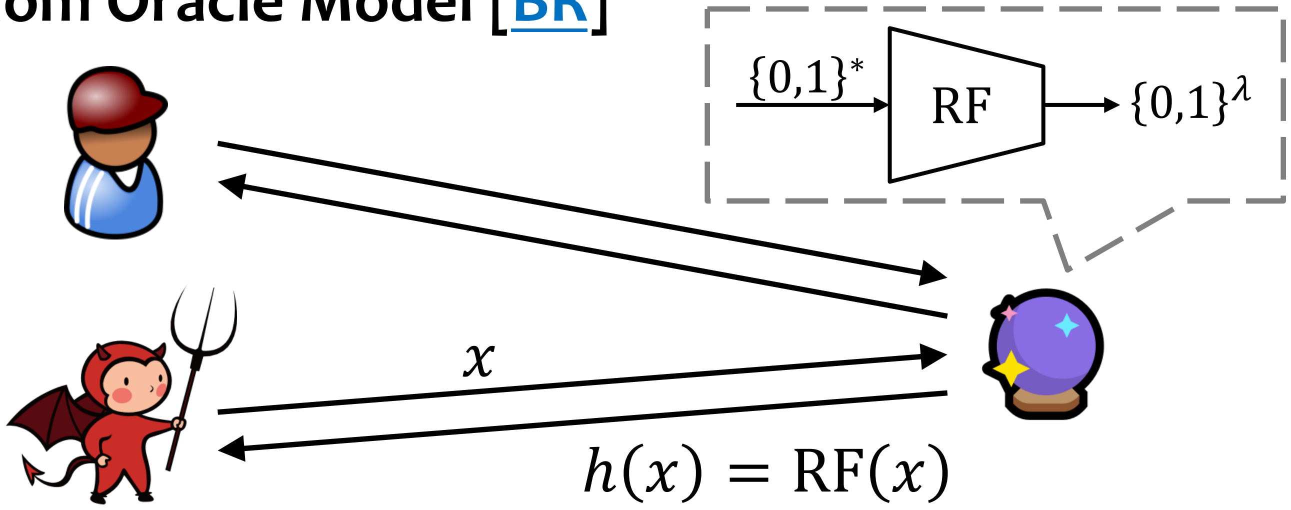
Hash Functions: Idealization

Standard Model



- one-wayness
- second-preimage resistance
- collision resistance

Random Oracle Model [BR]



practically used and more efficient schemes

Schnorr signature[Schnorr] RSA-OAEP[BR] TLS[KPW,DFGS,DJ] …

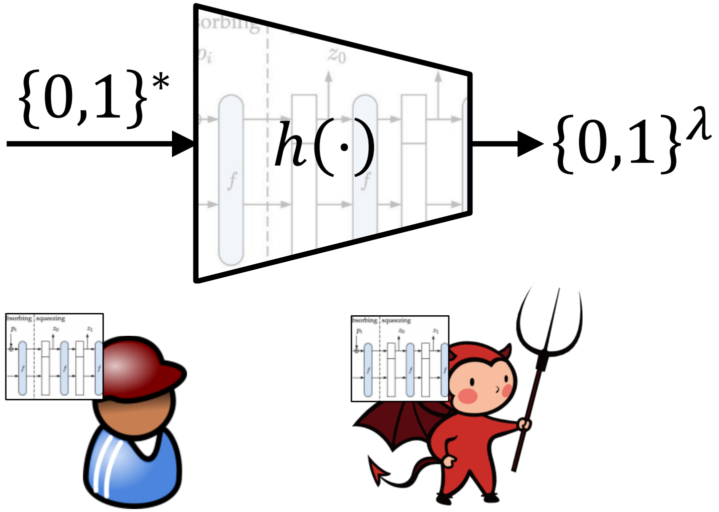
(contrived) uninstantiability results [CGH]

proof in ROM better than no proof at all

problematic if cannot write proof in ROM

Hash Functions: Idealization

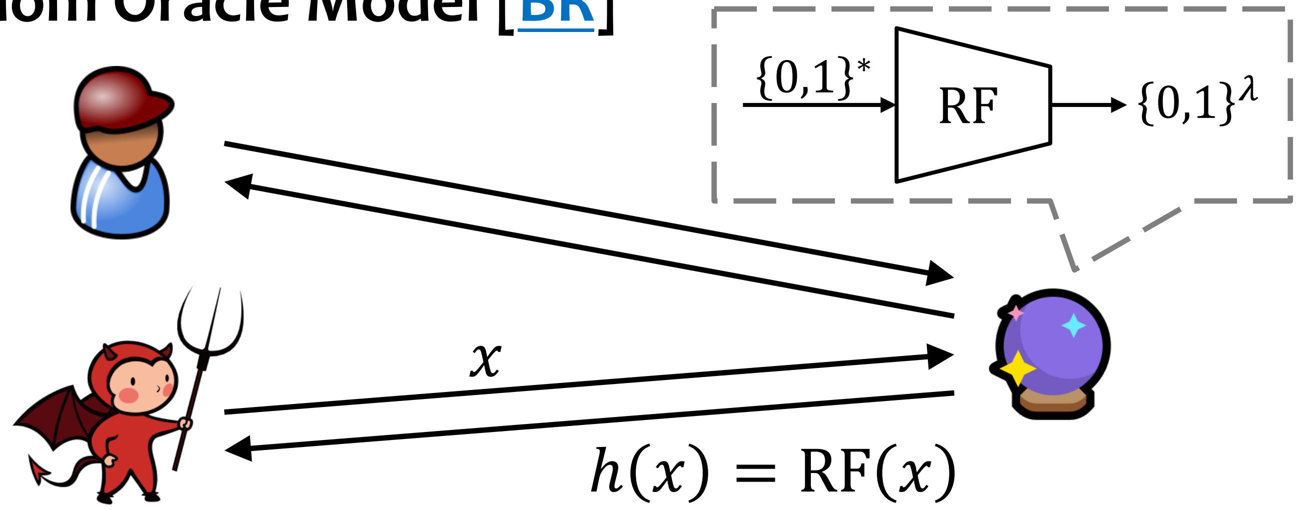
Standard Model



- one-wayness
- second-preimage resistance
- collision resistance

- correlation intractability

Random Oracle Model [BR]



practically used and more efficient schemes

Schnorr signature[Schnorr] RSA-OAEP[BR] TLS[KPW,DFGS,DJ] …

(contrived) uninstantiability results [CGH]

proof in ROM better than no proof at all

problematic if cannot write proof in ROM

precursor to standard-model version

Question

$$\textbf{CRHF} \xrightarrow{\text{idealize}} \textbf{random oracle}$$

$$i\mathcal{O} \xrightarrow[\text{idealize}]{} \textbf{ideal obfuscation}$$

Question

$$\begin{tikzcd}[column sep=huge, row sep=huge]
\textbf{CRHF} \arrow[r, "\text{idealize}"] & \textbf{random oracle} \\
i\mathcal{O} \arrow[r, "\text{idealize}"'] & \textbf{ideal obfuscation} \arrow[u, "\mathrm{RO}(\cdot) = \mathrm{Obf}\big(F(k,\cdot)\big) \text{ for PRF } F"]
\end{tikzcd}$$

Question

$$\begin{array}{ccc} \textbf{CRHF} & \xrightarrow{\text{idealize}} & \textbf{random oracle} \\ & & \\ i\mathcal{O} & \xrightarrow[\text{idealize}]{} & \textbf{ideal obfuscation} \end{array}$$

$\mathrm{RO}(\cdot) = \mathrm{Obf}\big(F(k,\cdot)\big)$ for PRF F

?

Question

$$\begin{tikzcd}[column sep=huge, row sep=huge]
\textbf{CRHF} \arrow[r, "\text{idealize}"] & \textbf{random oracle} \arrow[d, shift left, "\times"] \\
i\mathcal{O} \arrow[r, "\text{idealize}"'] & \textbf{ideal obfuscation} \arrow[u, shift left, "\mathrm{RO}(\cdot) = \mathrm{Obf}\big(F(k,\cdot)\big) \text{ for PRF } F"]
\end{tikzcd}$$

Black-box use of hash functions does **not** help building ideal obfuscation. [CKP]

Question

Can **non-black-box** use of hash functions help? ?

CRHF $\xrightarrow{\text{idealize}}$ **random oracle**

$\mathrm{RO}(\cdot) = \mathrm{Obf}\big(F(k,\cdot)\big)$ for PRF F

Black-box use of hash functions does **not** help building ideal obfuscation. [CKP]

$i\mathcal{O}$ $\xrightarrow{\text{idealize}}$ **ideal obfuscation**

Result

morally ROM but more **flexible**

CRHF $\xrightarrow{\text{idealize}}$ **pseudorandom oracle**

$i\mathcal{O}$ $\xrightarrow[\text{idealize}]{}$ **ideal obfuscation**

Result

morally ROM but more **flexible**

CRHF $\xrightarrow{\text{idealize}}$ **pseudorandom oracle**

Theorem

Assuming polynomially secure FE for circuits, there exists ideal obfuscation in the pseudorandom oracle model.

+ FE

$i\mathcal{O}$ $\xrightarrow[\text{idealize}]{}$ **ideal obfuscation**

Result

CRHF $\xrightarrow{\text{idealize}}$ morally ROM but more **flexible** **pseudorandom oracle**

Theorem

Assuming polynomially secure FE for circuits, there exists ideal obfuscation in the pseudorandom oracle model.

- **Justifies** all downstream applications.
- Hope of $i\mathcal{O}$ from **polynomial security**.
- Multiple interpretations. (**heuristics** / bootstrapping / hardware tokens)

+ FE

$i\mathcal{O}$ $\xrightarrow[\text{idealize}]{}$ **ideal obfuscation**

Take-Home Message

CRHF $\xrightarrow{\text{idealize}}$ **pseudorandom oracle**

morally ROM but more **flexible**

Reduction of Heuristics

accept heuristics of **ideal hash functions**
(+ accept Pr$\mathcal{O}$M as a good model of it)
$\Longrightarrow$ **accept** heuristics of **ideal obfuscation**

+ FE

$i\mathcal{O}$ $\xrightarrow[\text{idealize}]{}$ **ideal obfuscation**

Take-Home Message

CRHF $\xrightarrow{\text{idealize}}$ **pseudorandom oracle**

morally ROM but more **flexible**

Reduction of Heuristics

accept heuristics of **ideal hash functions**
(+ accept Pr$\mathcal{O}$M as a good model of it)
$\Longrightarrow$ **accept** heuristics of **ideal obfuscation**

+ FE

> *Heuristically assuming **ideal obfuscation***
> *is **not "crazier" than***
> *heuristically assuming **ideal hash functions**.*

$i\mathcal{O}$ $\xrightarrow{\text{idealize}}$ **ideal obfuscation**

Pseudo**r**andom **O**racle **M**odel ($\mathrm{Pr}\mathcal{O}\mathrm{M}$)

Two aspects of the model.

- h looks like a **random function**
- h has (short) **code**

Pseudorandom **O**racle **M**odel (Pr$\mathcal{O}$M)

Two aspects of the model.

- h looks like a **random function**
- h has (short) **code**

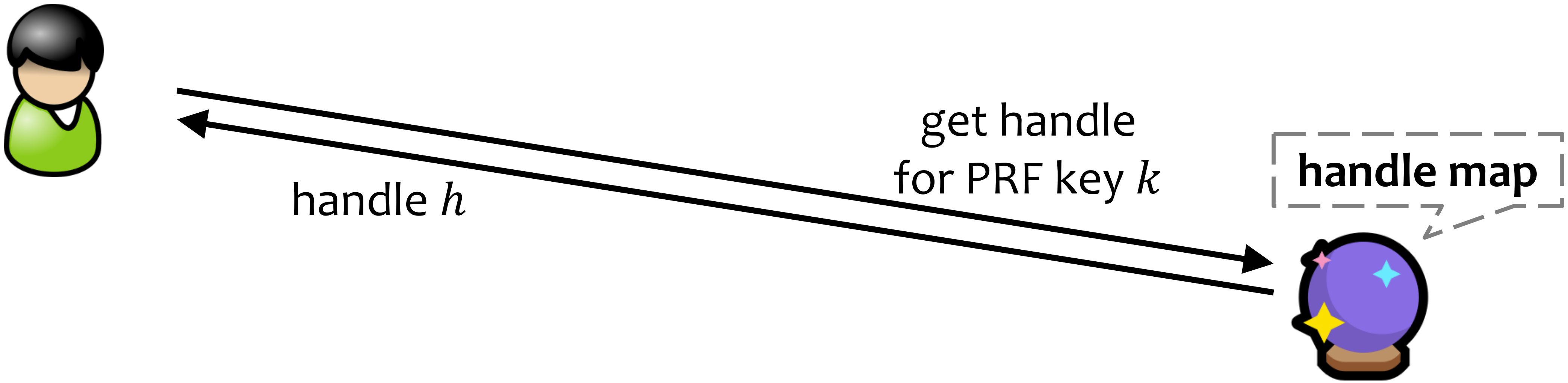
Pr$\mathcal{O}$M for PRF F

Pseudorandom **O**racle **M**odel (Pr$\mathcal{O}$M)

Two aspects of the model.

- h looks like a **random function**
- h has (short) **code**

Pr$\mathcal{O}$M for PRF F

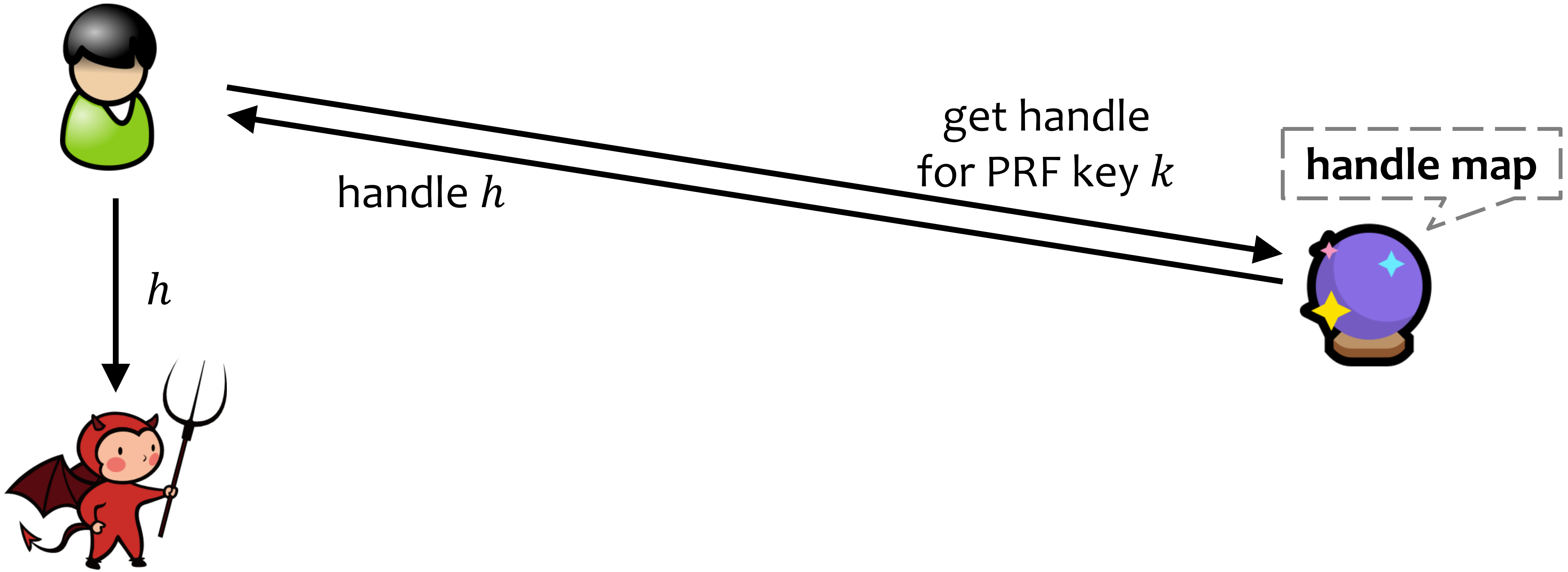


Pseudorandom **O**racle **M**odel (Pr$\mathcal{O}$M)

Two aspects of the model.

- h looks like a **random function**
- h has (short) **code**

Pr$\mathcal{O}$M for PRF F

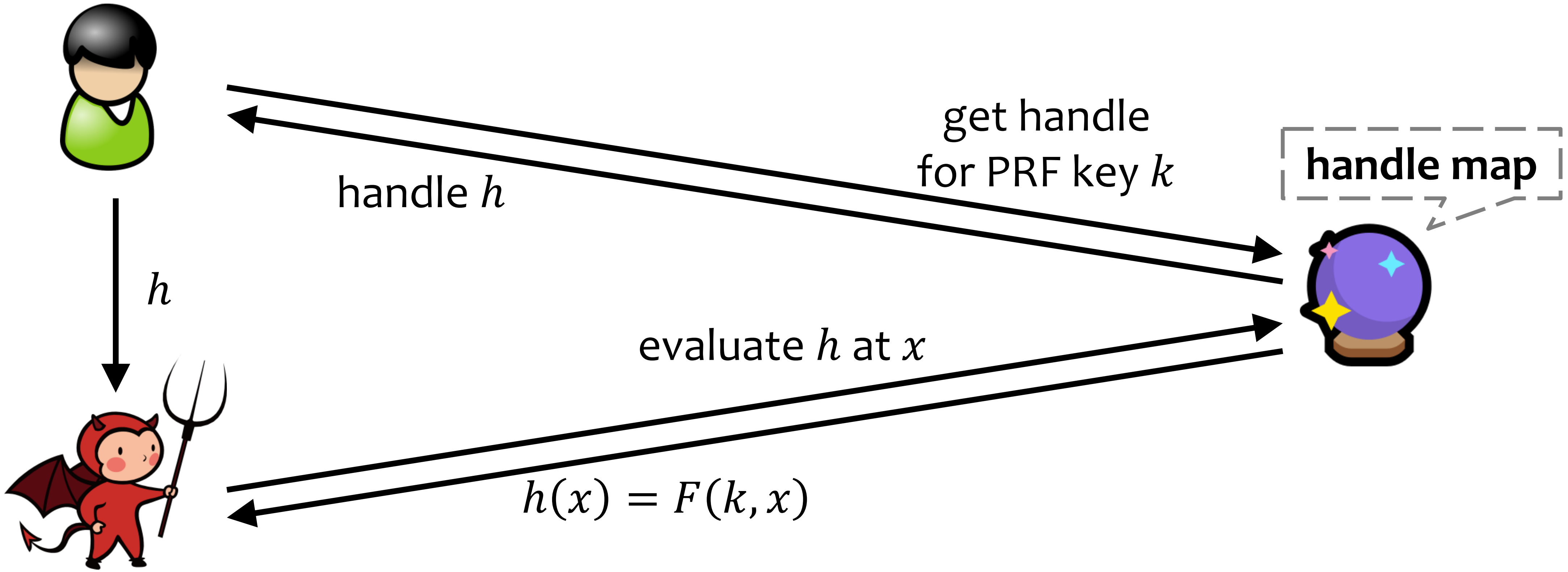


Pseudorandom **O**racle **M**odel (Pr$\mathcal{O}$M)

Two aspects of the model.

- h looks like a **random function**
- h has (short) **code**

Pr$\mathcal{O}$M for PRF F

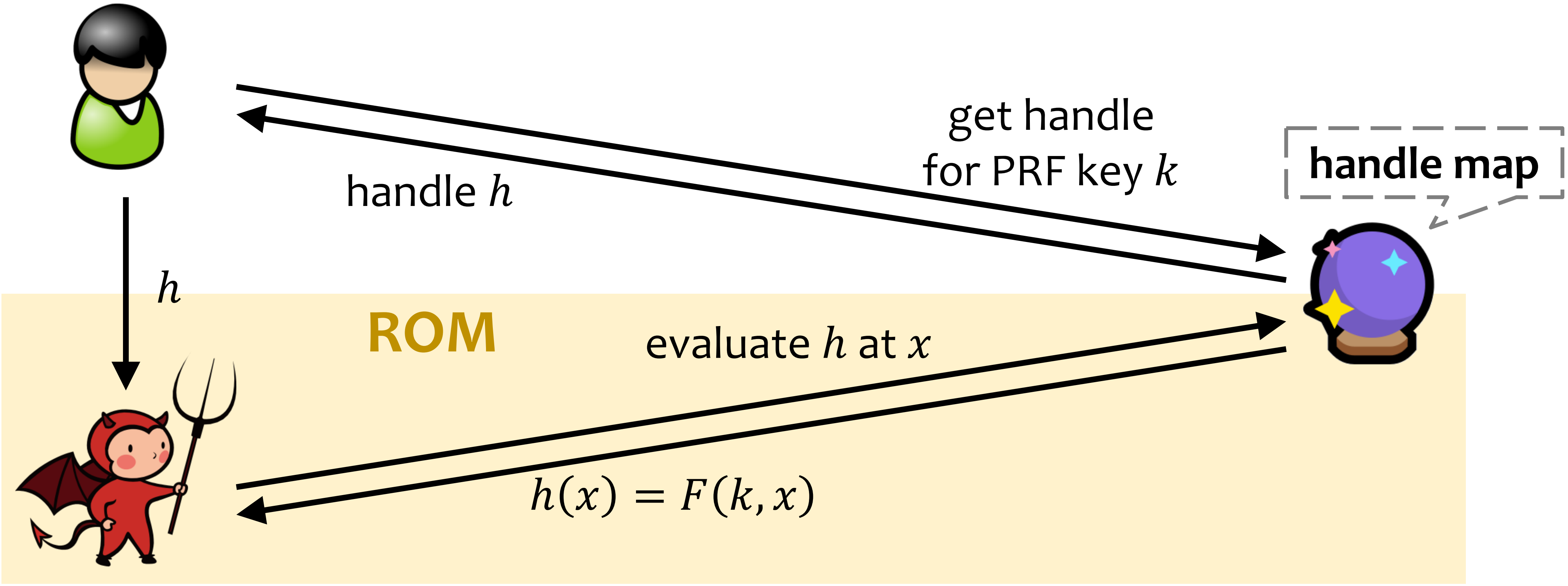


Pseudorandom **O**racle **M**odel (Pr$\mathcal{O}$M)

Pr$\mathcal{O}$M for PRF F

Two aspects of the model.

- h looks like a **random function**
- h has (short) **code**

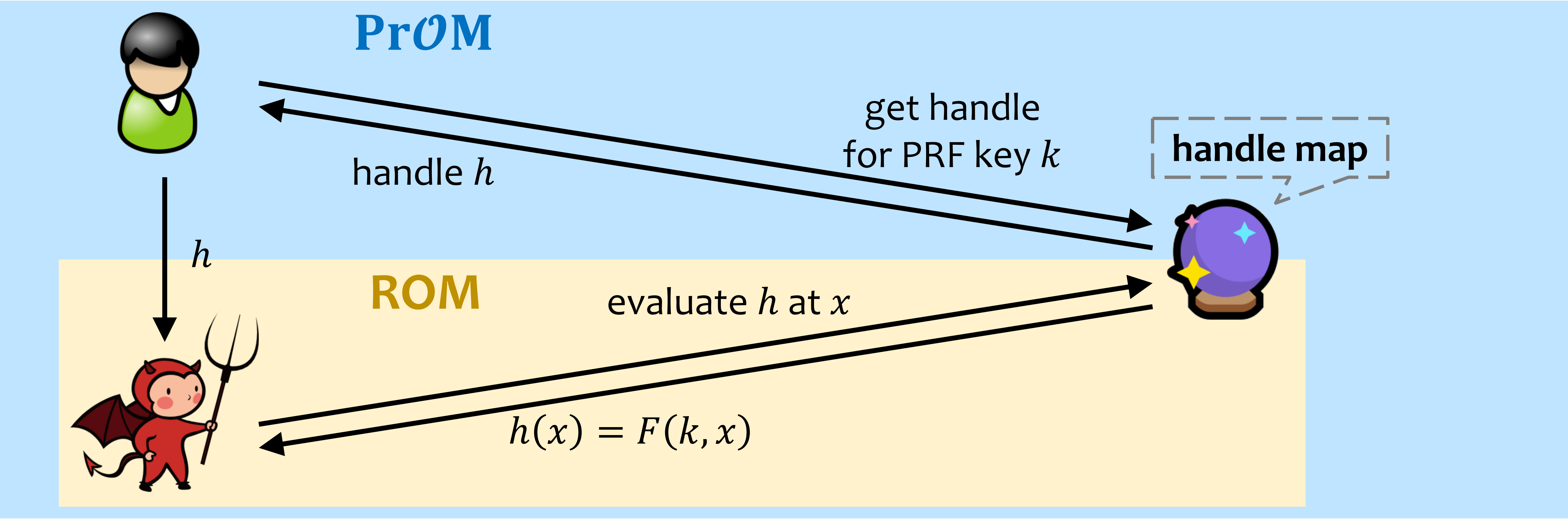


Pseudorandom **O**racle **M**odel (Pr$\mathcal{O}$M)

Pr$\mathcal{O}$M for PRF F

Two aspects of the model.

- h looks like a **random function**
- h has (short) **code**



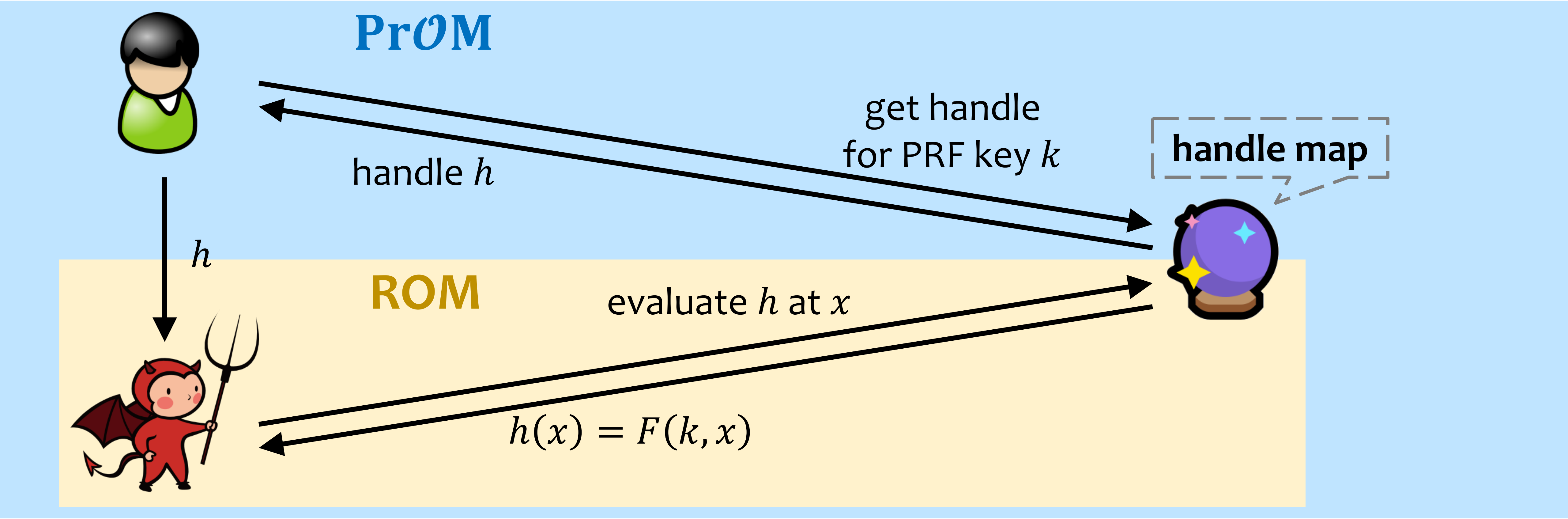
Pseudorandom **O**racle **M**odel (Pr$\mathcal{O}$M)

Two aspects of the model.

- h looks like a **random function**
- h has (short) **code**

Pr$\mathcal{O}$M for PRF F

But $h(x)$ is <u>not</u> random if k is present?!

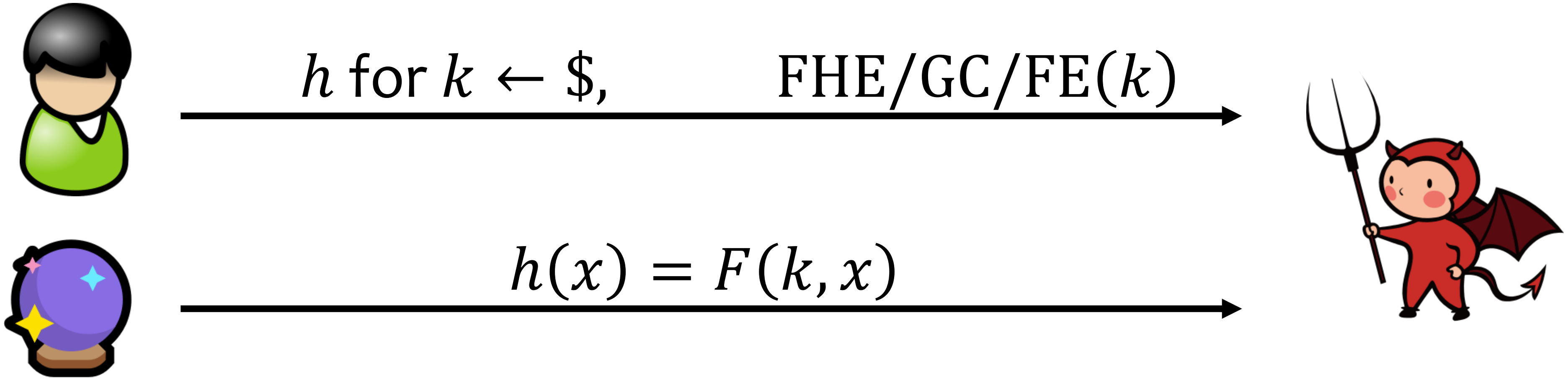


Basic Recipe of Using Pr$\mathcal{O}$M

Basic Recipe of Using Pr$\mathcal{O}$M

$$\xrightarrow{\quad h \text{ for } k \leftarrow \$, \qquad \mathrm{FHE/GC/FE}(k) \quad}$$

Basic Recipe of Using Pr$\mathcal{O}$M



Basic Recipe of Using Pr$\mathcal{O}$M

h for $k \leftarrow \$$, $\qquad \mathrm{FHE/GC/FE}(k)$

$h(x) = F(k, x)$

⚠ ***h* cannot be monitored or programmed**

Basic Recipe of Using Pr$\mathcal{O}$M

h for $k \leftarrow \$$, FHE/GC/FE(k)

$h(x) = F(k, x)$

⚠ **h cannot be monitored or programmed**

$\approx$

h for $k \leftarrow \$$, $\mathrm{Sim}(\{F(k, x_i)\}_i)$

$h(x) = F(k, x)$

(FHE/GC/FE security)

Basic Recipe of Using Pr$\mathcal{O}$M

h for $k \leftarrow \$$, FHE/GC/FE(k)

$h(x) = F(k, x)$

⚠ ***h* cannot be monitored or programmed**

$\approx$

h for $k \leftarrow \$$, $\mathrm{Sim}(\{F(k, x_i)\}_i)$

$h(x) = F(k, x)$

(FHE/GC/FE security)

$\approx$

h, $\mathrm{Sim}(\{\mathrm{RF}(x_i)\}_i)$

$h(x) = \mathrm{RF}(x)$

(PRF security of F)

Basic Recipe of Using Pr$\mathcal{O}$M

h for $k \leftarrow \$$, FHE/GC/FE(k) $\longrightarrow$

$h(x) = F(k, x)$ $\longrightarrow$

⚠ ***h* cannot be monitored or programmed**

$\approx$

h for $k \leftarrow \$$, $\mathrm{Sim}(\{F(k, x_i)\}_i)$ $\longrightarrow$

$h(x) = F(k, x)$

(FHE/GC/FE security)

$\approx$

h, $\mathrm{Sim}(\{\mathrm{RF}(x_i)\}_i)$ $\longrightarrow$

$h(x) = \mathrm{RF}(x)$

✓ ***h* can be monitored and programmed**

(PRF security of F)

Basic Recipe of Using Pr$\mathcal{O}$M

Limited Use of Code

must ≈ hybrid with only black-box use

h for $k \leftarrow \$$, FHE/GC/FE(k) $\longrightarrow$

$h(x) = F(k, x)$ $\longrightarrow$

⚠ ***h* cannot be monitored or programmed**

$\approx$

h for $k \leftarrow \$$, $\mathrm{Sim}(\{F(k, x_i)\}_i)$ $\longrightarrow$ **(FHE/GC/FE security)**

$h(x) = F(k, x)$

$\approx$

h, $\mathrm{Sim}(\{\mathrm{RF}(x_i)\}_i)$ $\longrightarrow$ **(PRF security of F)**

$h(x) = \mathrm{RF}(x)$

✓ ***h* can be monitored and programmed**

Instantiating Pr$\mathcal{O}$M

Rule of Thumb. good for ROM $\Longrightarrow$ good for Pr$\mathcal{O}$M

Instantiating Pr$\mathcal{O}$M

Rule of Thumb. good for ROM $\Longrightarrow$ good for Pr$\mathcal{O}$M

Example. Let $h = k$ be random and $F(k, x) = \mathrm{SHA3}(k \parallel x)$

Rationale. Good hash functions are "**self-obfuscated PRF**".

Instantiating Pr$\mathcal{O}$M

Rule of Thumb. good for ROM $\Longrightarrow$ good for Pr$\mathcal{O}$M

Example. Let $h = k$ be random and $F(k, x) = \mathrm{SHA3}(k \parallel x)$

Rationale. Good hash functions are "**self-obfuscated PRF**".

Pr$\mathcal{O}$M does **not demand more than ROM**.

— the only meaningful thing to do with a prefix is to evaluate the function at its extensions.

Ideal Obfuscation

$\mathcal{O}$: oracle of idealized model (e.g., $\Pr\mathcal{O}^F$)

C (no oracle)

```
basicFun = Function[{Typed[pixel0, "ComplexReal64"]},
    Module[{iters = 1, maxIters = 1000, pixel = pixel0},
      While[iters < maxIters && Abs[pixel] < 2,
        pixel = pixel^2 + pixel0;
        iters++
      ];
      iters]];
```

$\mathrm{Obf}^{\mathcal{O}}(\cdot)$

$\tilde{C}^{\bullet}$

$$\forall C, x: \quad \Pr\left[\tilde{C}^{\bullet} \leftarrow \mathrm{Obf}^{\mathcal{O}}(C) \ : \ \tilde{C}^{\mathcal{O}}(x) = C(x)\right] = 1.$$

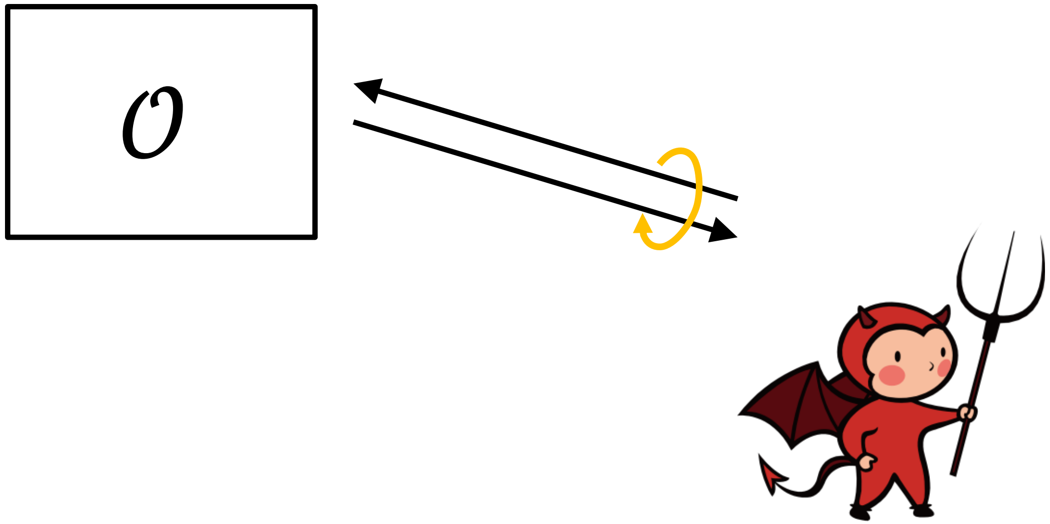
Ideal Obfuscation: Security

Real **Simulation**

Ideal Obfuscation: Security

Real

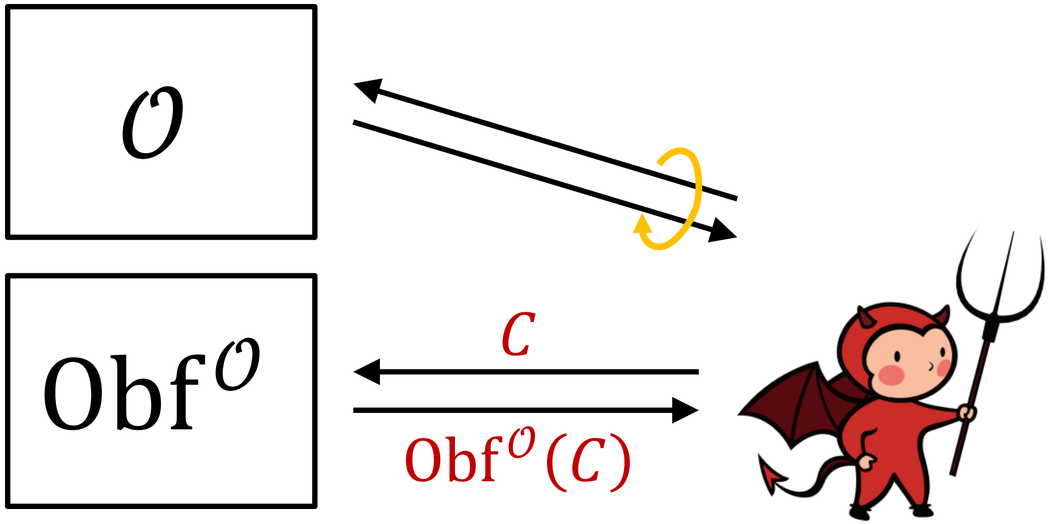
Simulation



Ideal Obfuscation: Security

Real

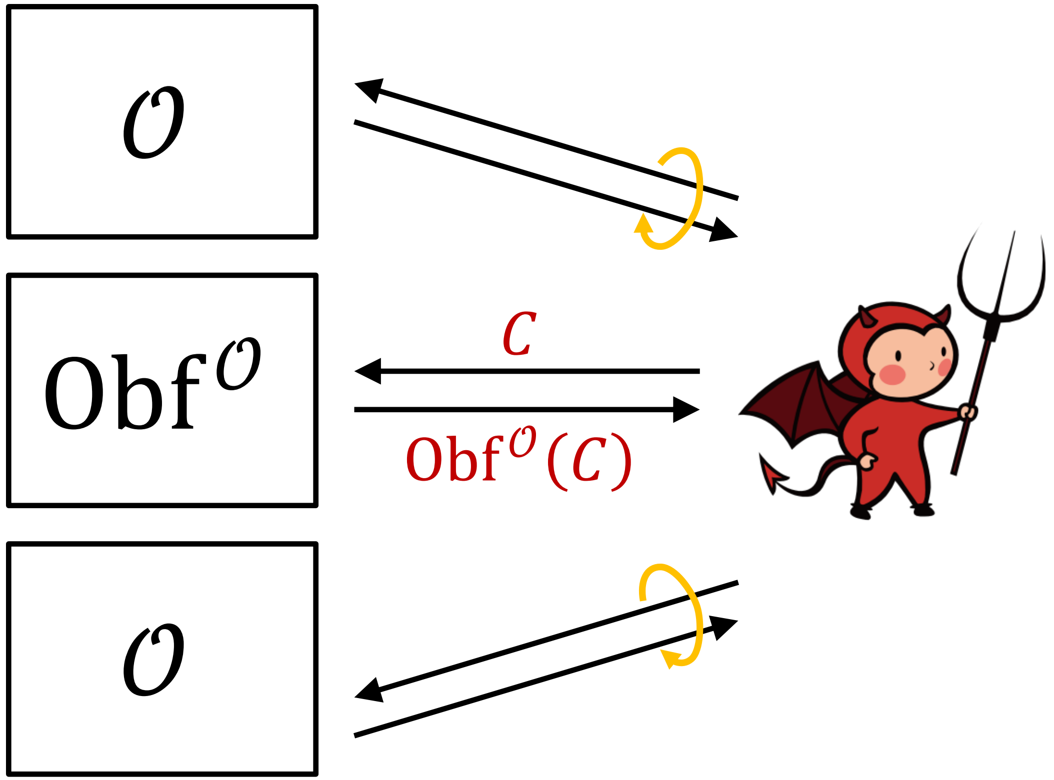
Simulation



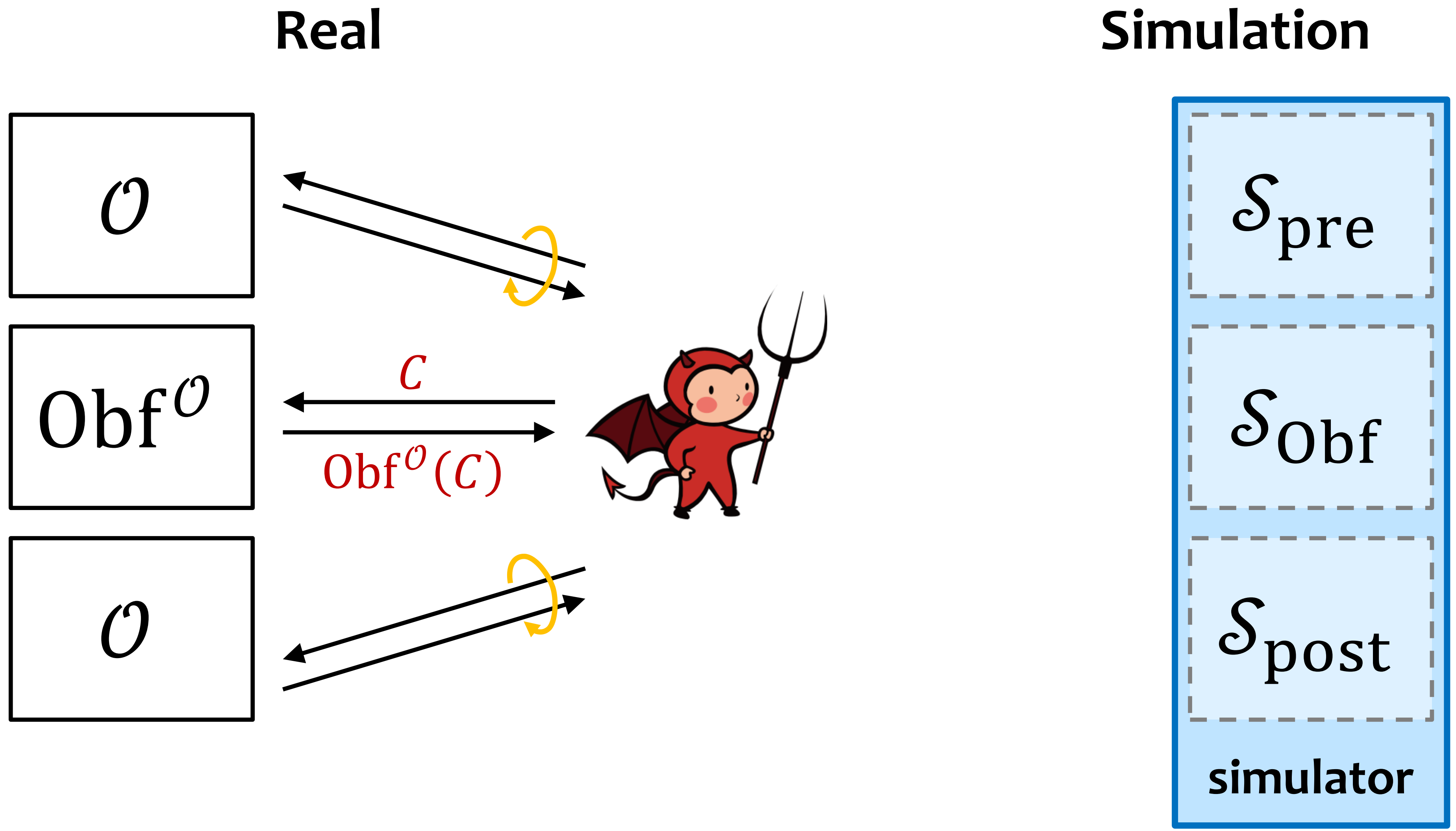
Ideal Obfuscation: Security

Real

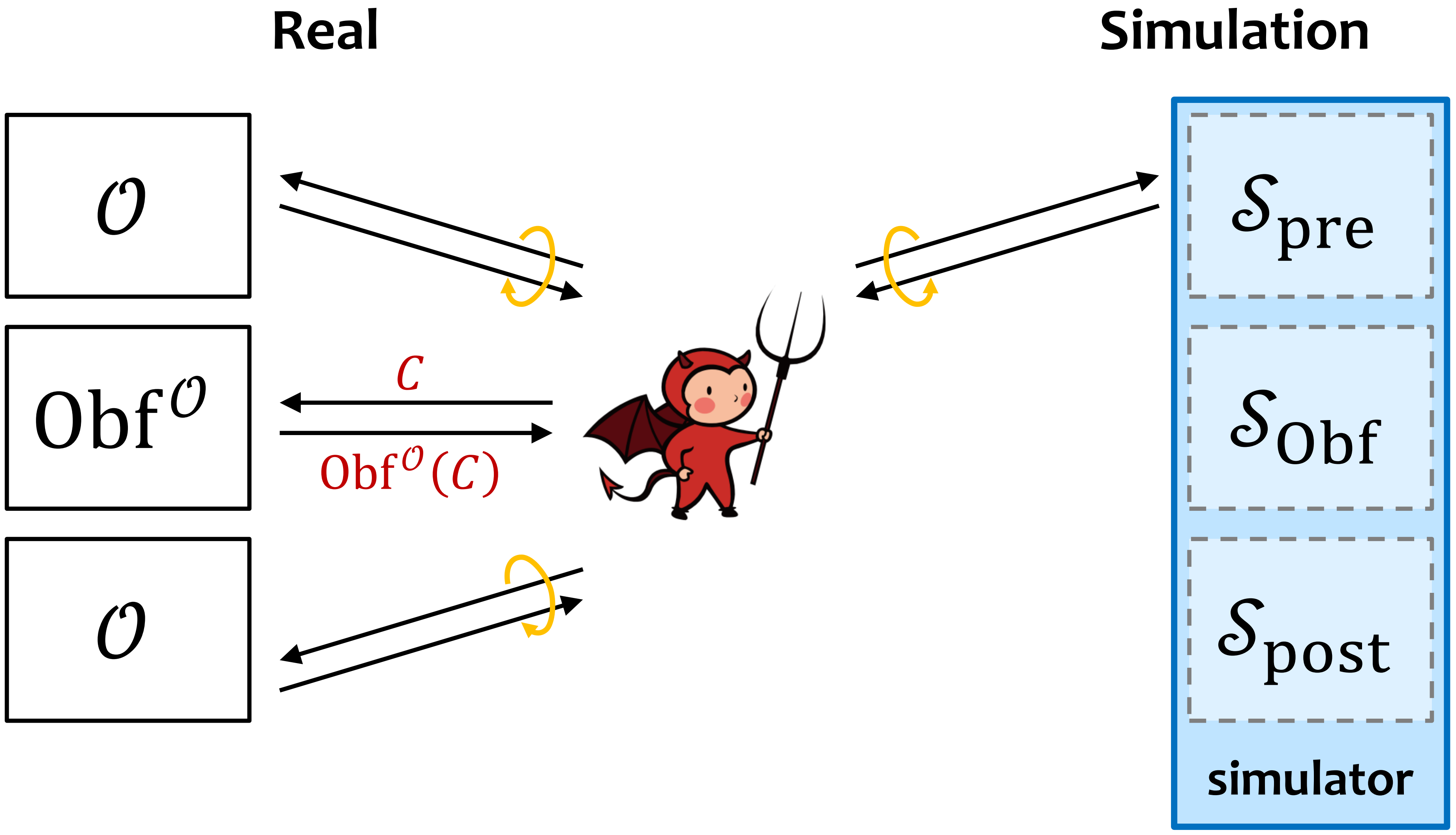
Simulation



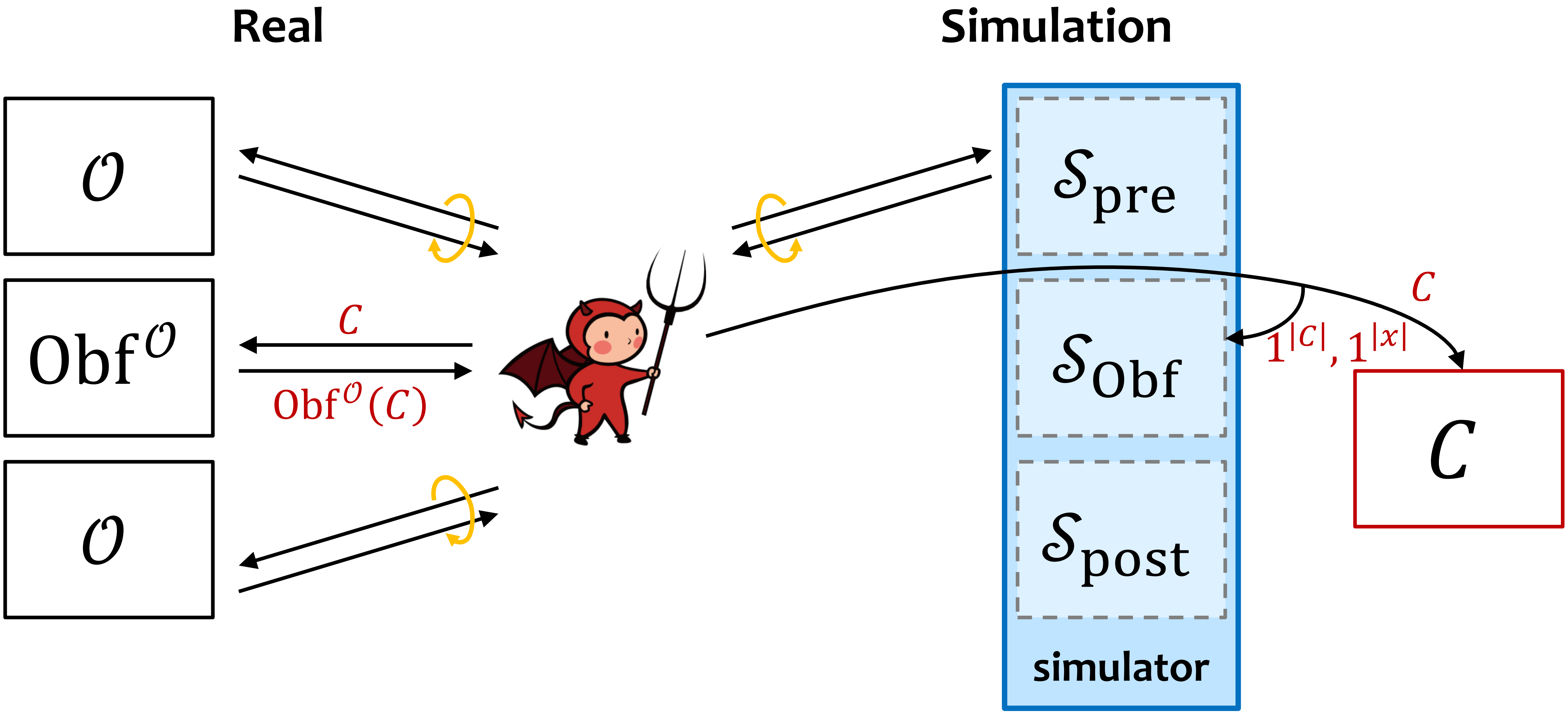
Ideal Obfuscation: Security



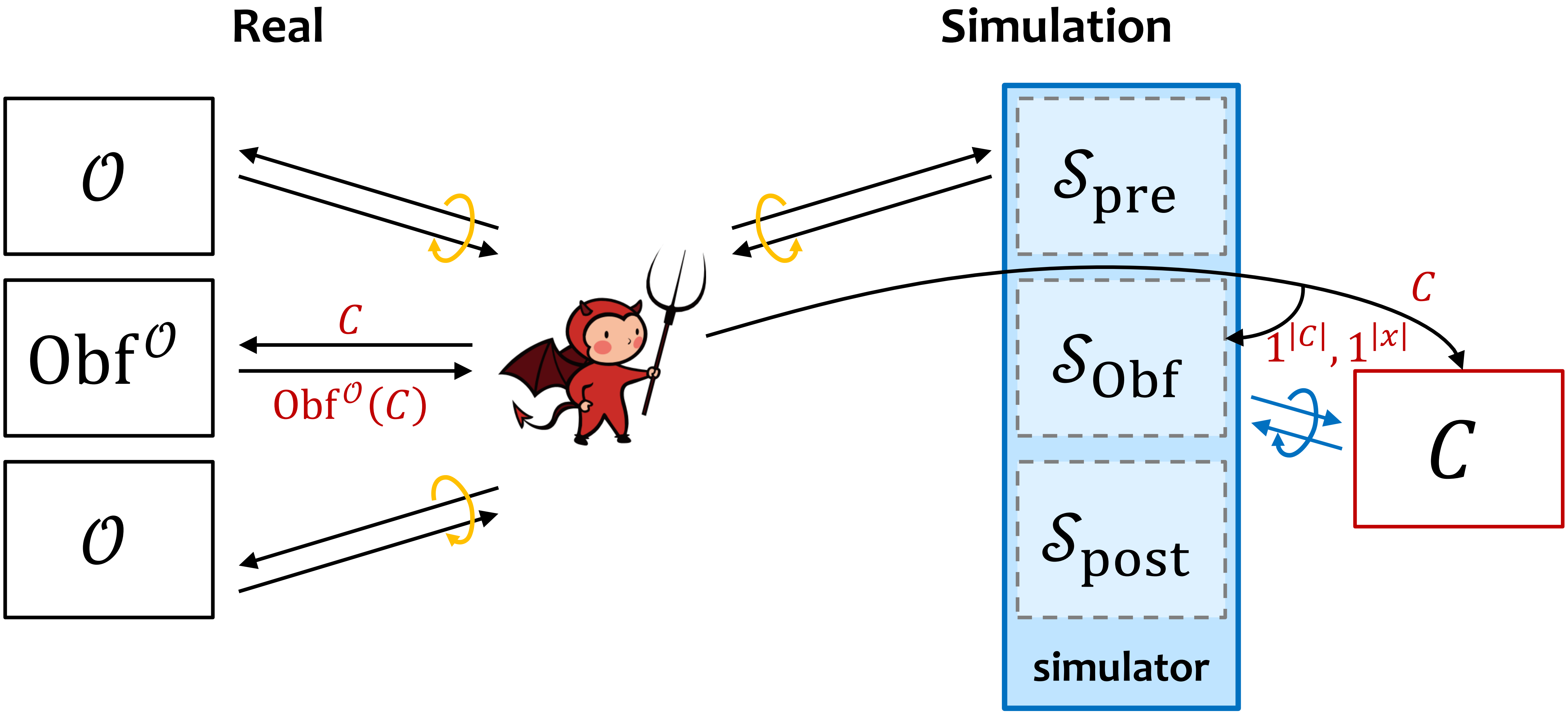
Ideal Obfuscation: Security



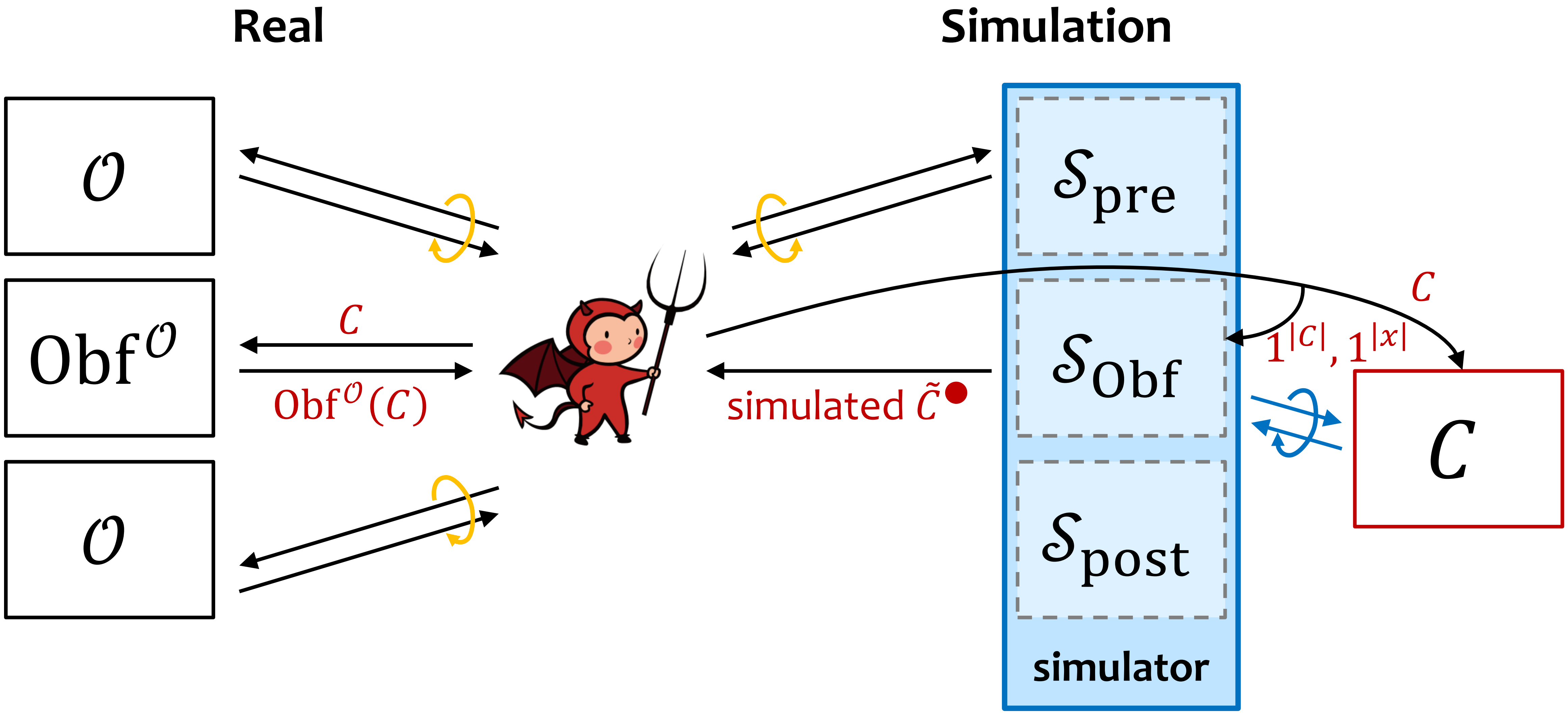
Ideal Obfuscation: Security



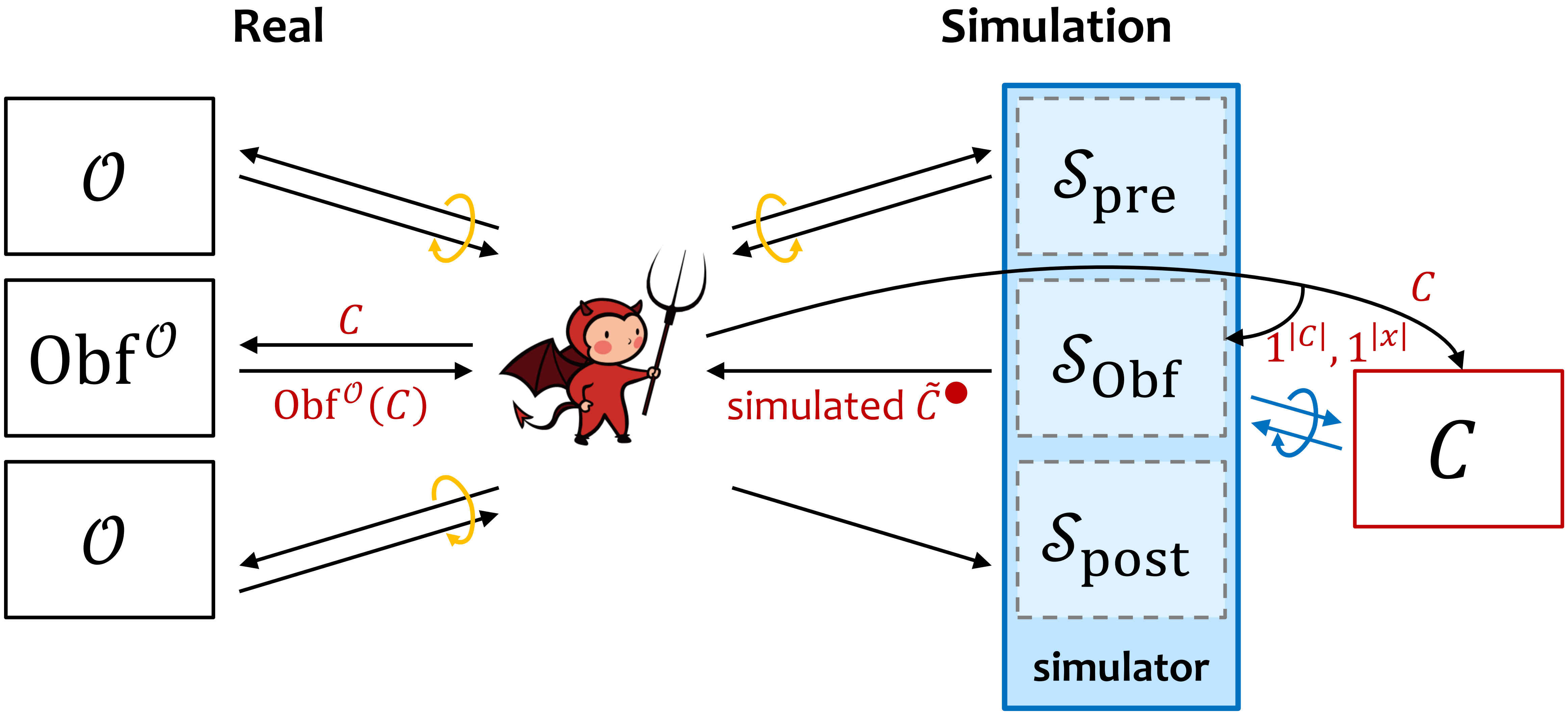
Ideal Obfuscation: Security



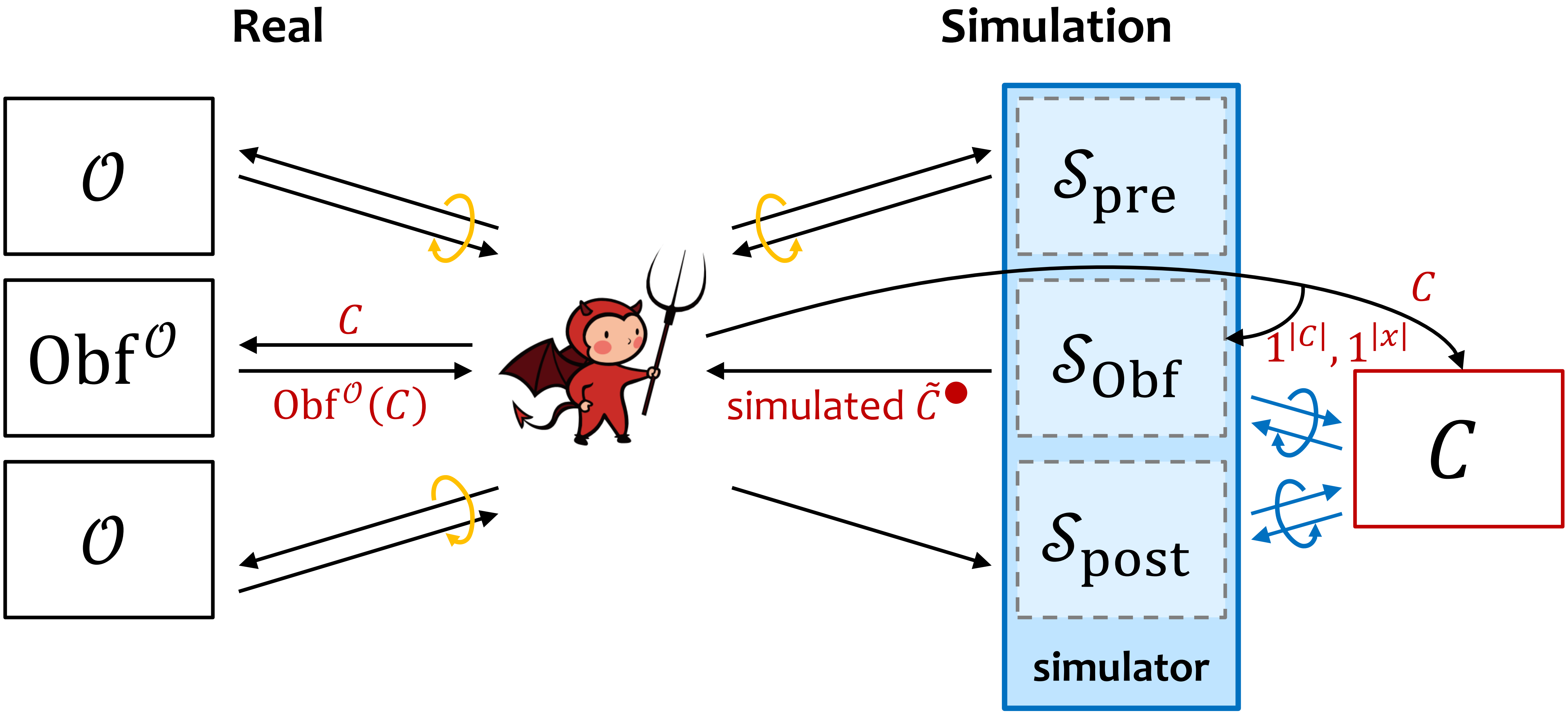
Ideal Obfuscation: Security



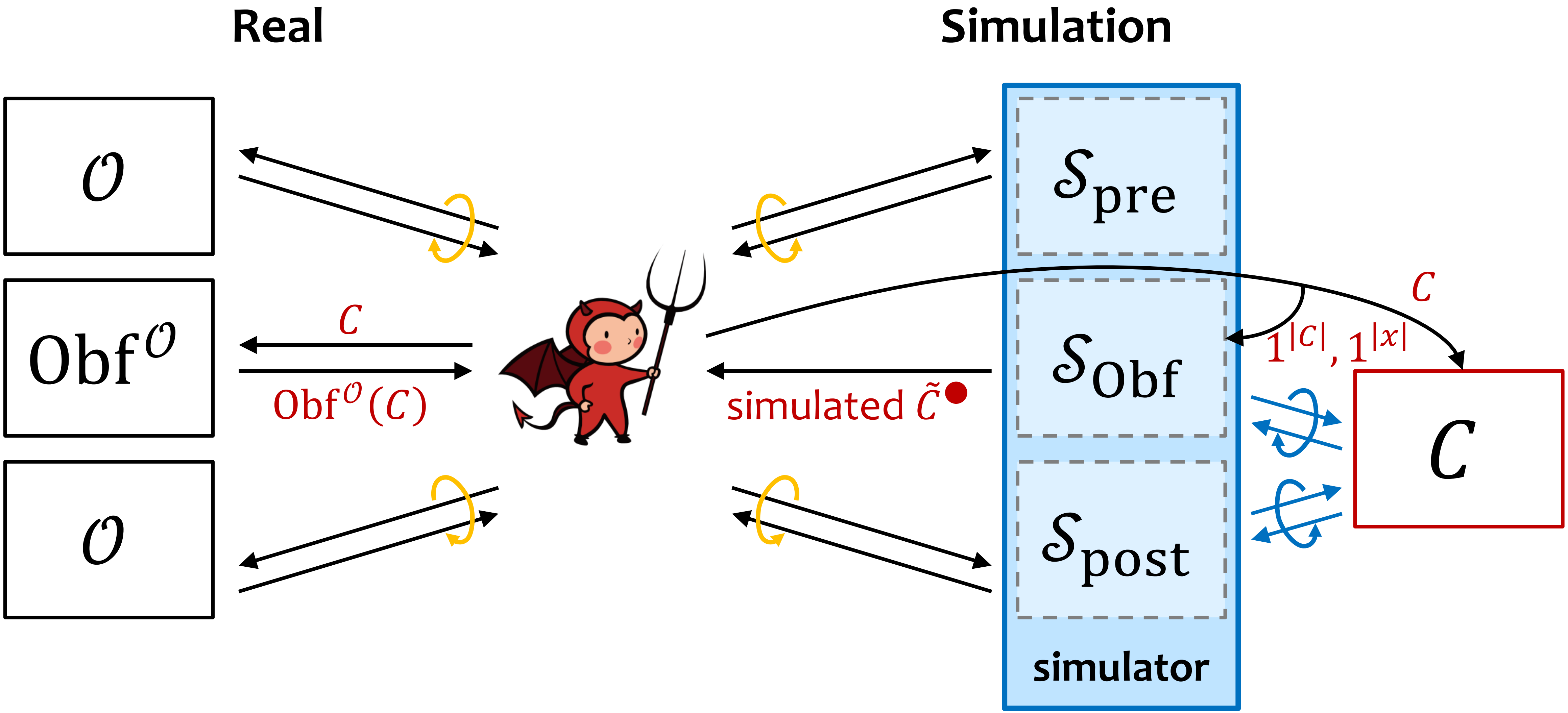
Ideal Obfuscation: Security



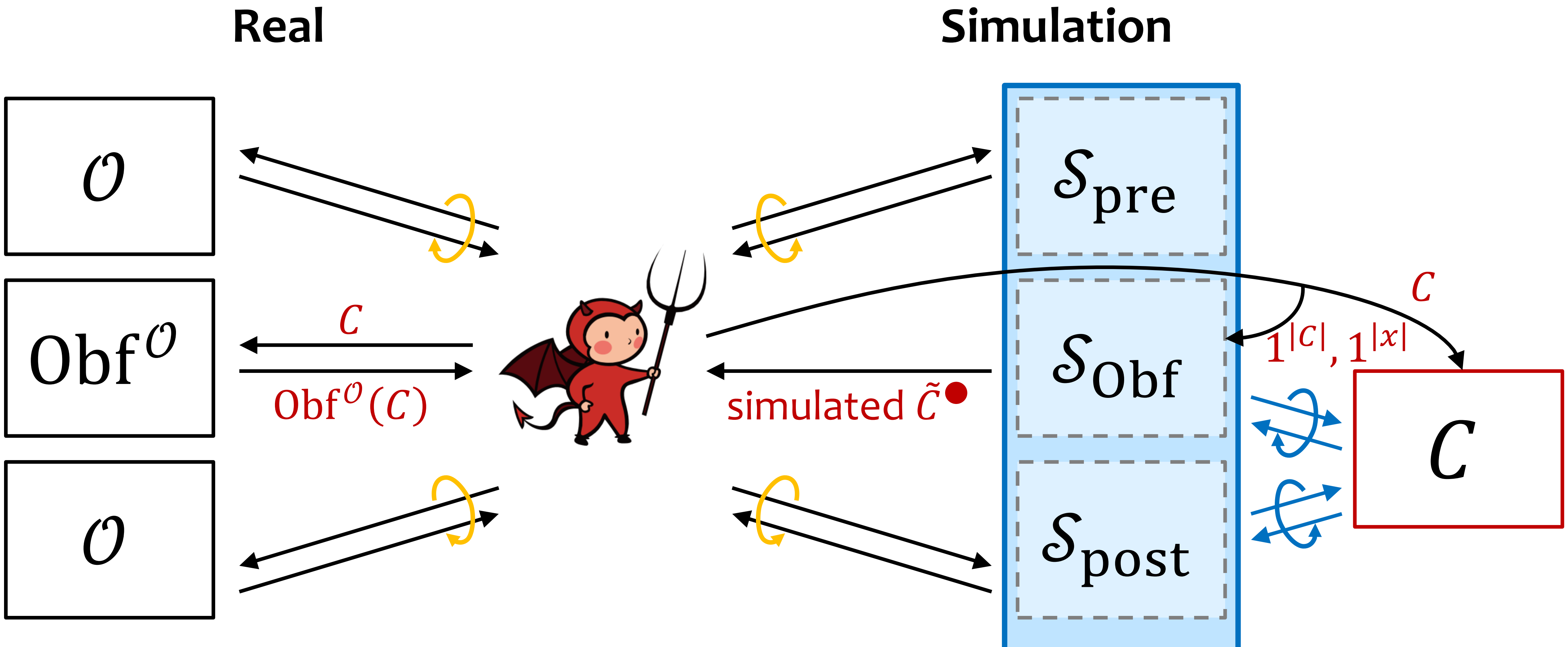
Ideal Obfuscation: Security



Ideal Obfuscation: Security



Ideal Obfuscation: Security



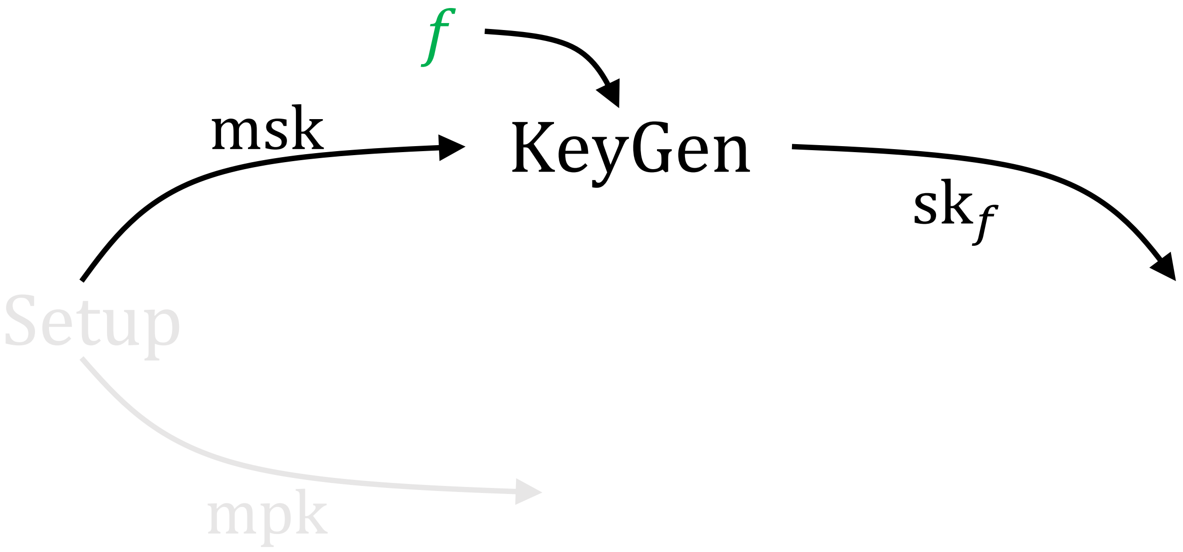
$$\left(\mathcal{O}, \mathrm{Obf}^{\mathcal{O}}(C), \mathcal{O}\right) \approx \left(\mathcal{S}_{\mathrm{pre}}, \mathcal{S}_{\mathrm{Obf}}^{C}, \mathcal{S}_{\mathrm{post}}^{C}\right)$$

(Standard-Model) Functional Encryption

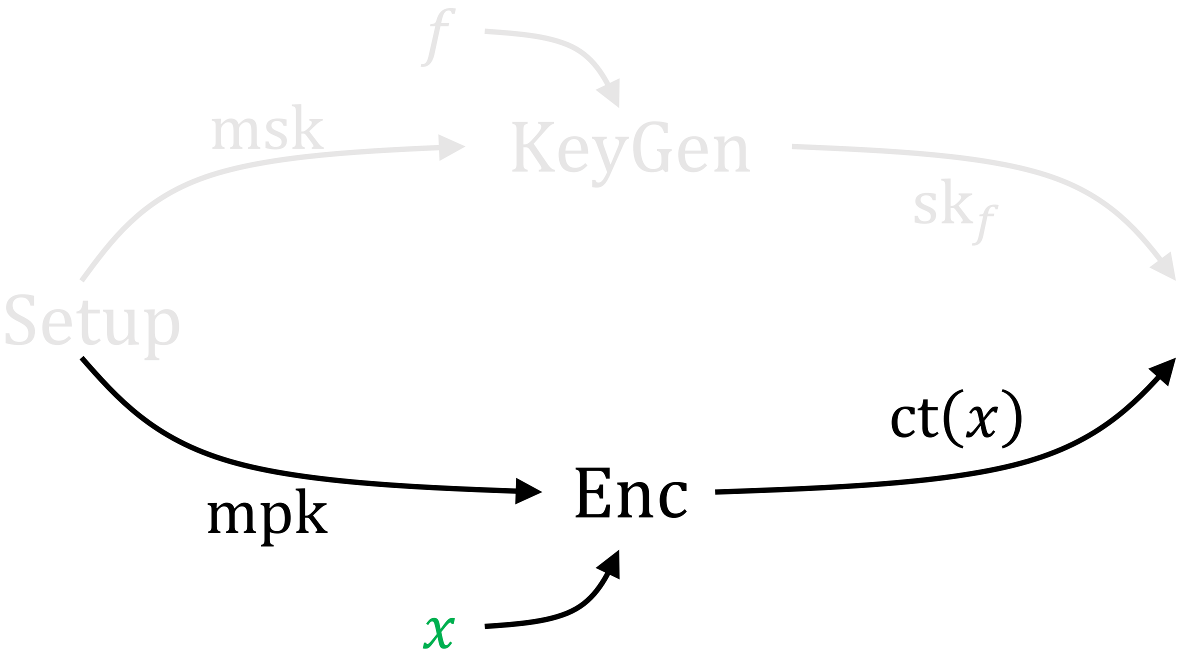
(Standard-Model) Functional Encryption

$$\text{Setup} \begin{cases} \xrightarrow{\text{msk}} \\ \xrightarrow[\text{mpk}]{} \end{cases}$$

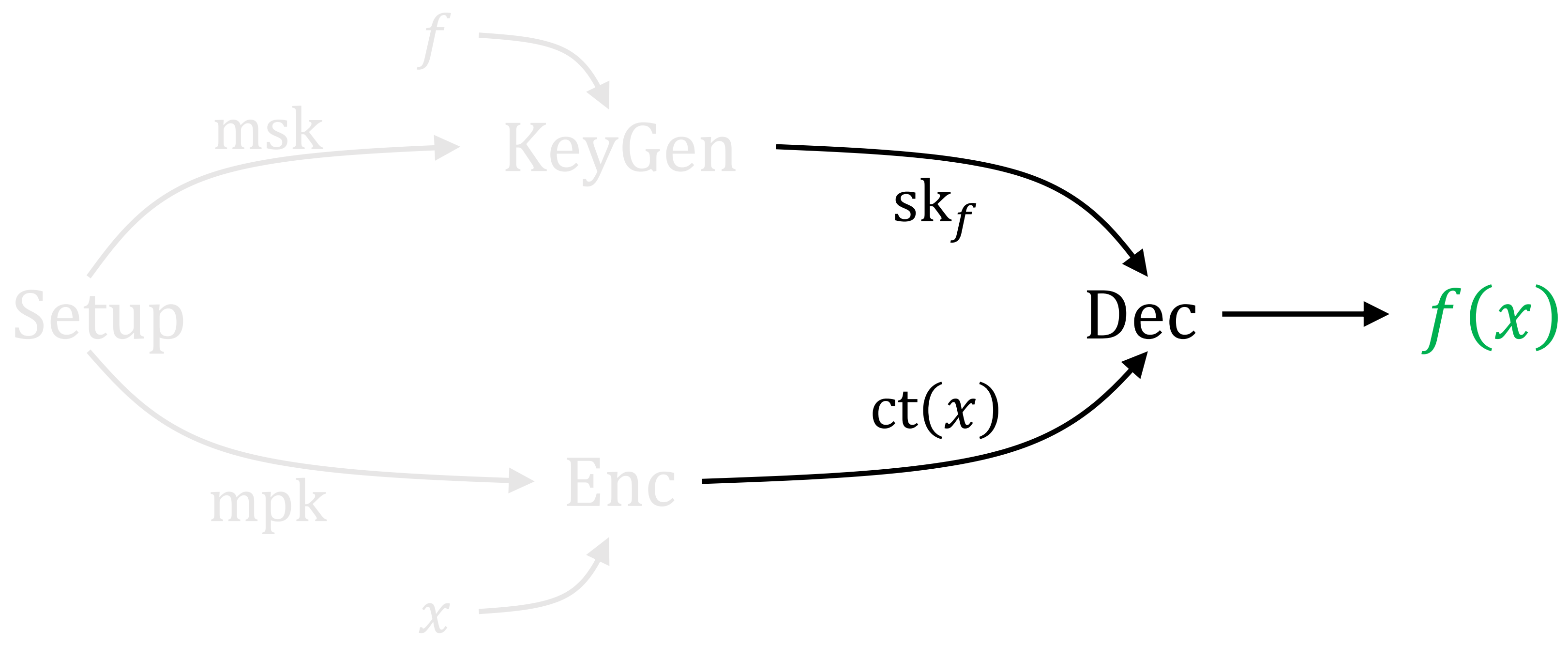
(Standard-Model) Functional Encryption



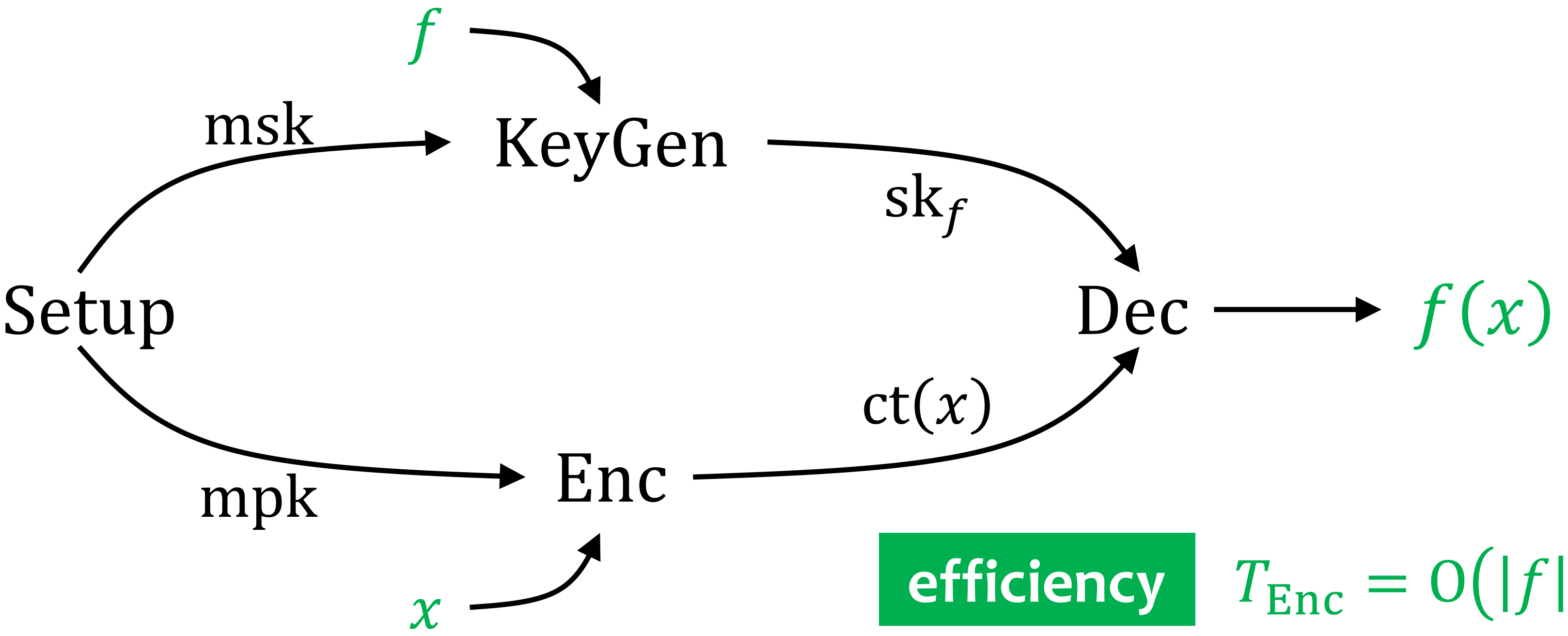
(Standard-Model) Functional Encryption



(Standard-Model) Functional Encryption



(Standard-Model) Functional Encryption



efficiency $T_{\mathrm{Enc}} = O\left(|f|^{1-\alpha} + |x|^{2-\alpha}\right)$

(Standard-Model) Functional Encryption

$$
\begin{tikzcd}
 & f \arrow[d, bend left] & & \\
 & \mathrm{KeyGen} \arrow[dr, bend left, "\mathrm{sk}_f"] & & \\
\mathrm{Setup} \arrow[ur, bend left, "\mathrm{msk}"] \arrow[dr, bend right, "\mathrm{mpk}"'] & & \mathrm{Dec} \arrow[r] & f(x) \\
 & \mathrm{Enc} \arrow[ur, bend right, "\mathrm{ct}(x)"'] & & \\
 & x \arrow[u, bend right] & &
\end{tikzcd}
$$

efficiency $T_{\mathrm{Enc}} = O\left(|f|^{1-\alpha} + |x|^{2-\alpha}\right)$

adaptive x_0, x_1 chosen after seeing $\mathrm{mpk}, \mathrm{sk}_f$

Security. if $f(x_0) = f(x_1)$ then $\left(\mathrm{mpk}, \mathrm{sk}_f, \mathrm{ct}(x_0)\right) \approx \left(\mathrm{mpk}, \mathrm{sk}_f, \mathrm{ct}(x_1)\right)$

(Standard-Model) Functional Encryption

$$
\begin{tikzcd}
 & \color{green}{f} \arrow[d] & & \\
 & \mathrm{KeyGen} \arrow[dr, bend left, "\mathrm{sk}_f"] & & \\
\mathrm{Setup} \arrow[ur, bend left, "\mathrm{msk}"] \arrow[dr, bend right, "\mathrm{mpk}"'] & & \mathrm{Dec} \arrow[r] & \color{green}{f(x)} \\
 & \mathrm{Enc} \arrow[ur, bend right, "\mathrm{ct}(x)"'] & & \\
 & \color{green}{x} \arrow[u] & &
\end{tikzcd}
$$

efficiency $T_{\mathrm{Enc}} = O\left(|f|^{1-\alpha} + |x|^{2-\alpha}\right)$

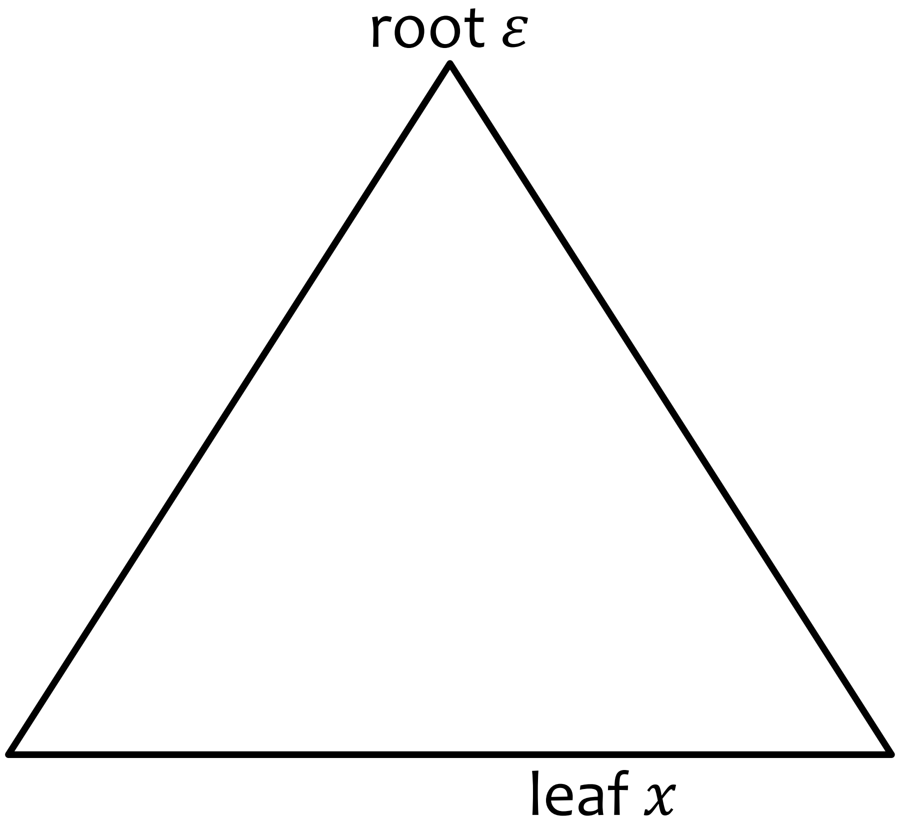
adaptive x_0, x_1 chosen after seeing $\mathrm{mpk}, \mathrm{sk}_f$

Security. if $f(x_0) = f(x_1)$ then $\left(\mathrm{mpk}, \mathrm{sk}_f, \mathrm{ct}(x_0)\right) \approx \left(\mathrm{mpk}, \mathrm{sk}_f, \mathrm{ct}(x_1)\right)$

Fact. Such FE follows from "obfuscation-minimum" FE, which can be based on well-studied assumptions. [ABSV, GVW, GS, LM, AJS, BV, AS, KNTY, JLL, JLS]

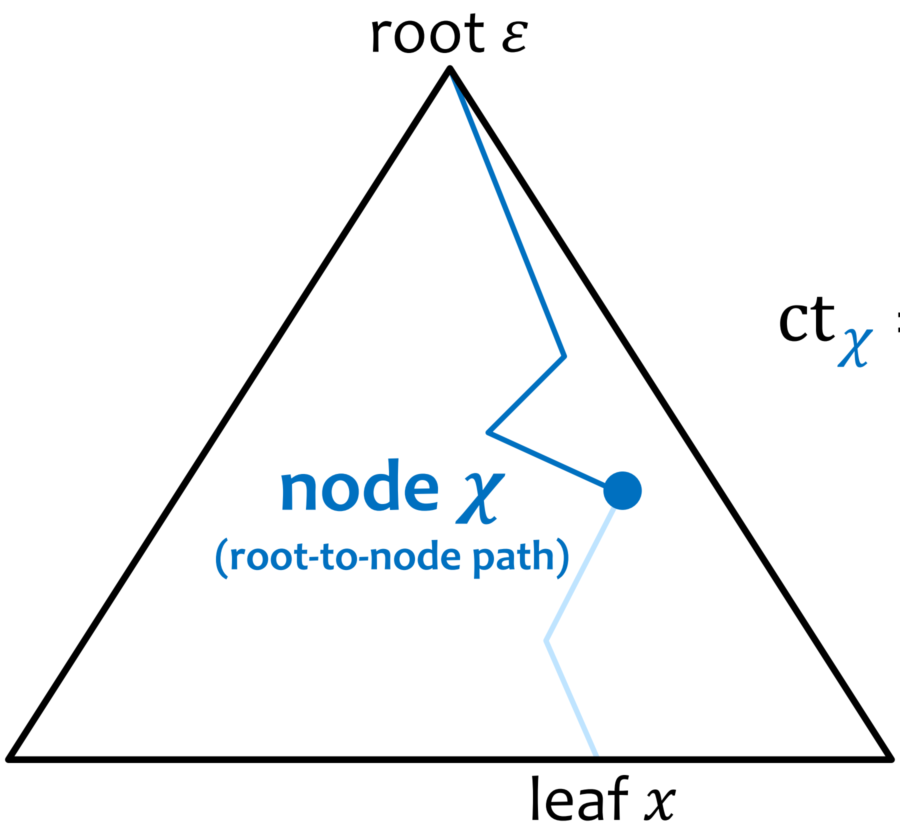
$i\mathcal{O}$ from FE: Quick Recap

perfect binary tree (leaf ⇔ input)



$i\mathcal{O}$ from FE: Quick Recap

perfect binary tree (leaf ⇔ input)



ct_χ = FE ciphertext of $(C, \chi, \dots)$ for $\chi \in \{0,1\}^{\leq D}$

$i\mathcal{O}$ from FE: Traversal and Evaluation

$\mathrm{ct}_{\chi}(C, \chi, \ldots)$

traverse/evaluate = decrypt using FE key sk_f

$\mathrm{ct}_{\varepsilon}$

$i\mathcal{O}$ from FE: Traversal and Evaluation

$\mathrm{ct}_{\chi}(C, \chi, \ldots)$

traverse/evaluate = decrypt using FE key sk_f

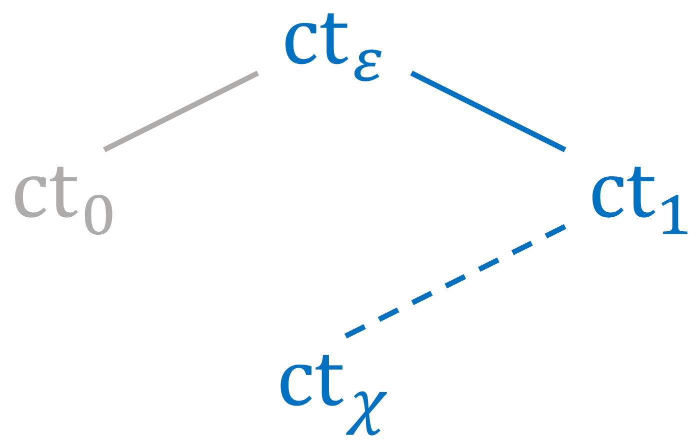
$\mathrm{ct}_{\varepsilon}$

ct_0 ct_1

$i\mathcal{O}$ from FE: Traversal and Evaluation

$\mathrm{ct}_\chi(C, \chi, \dots)$

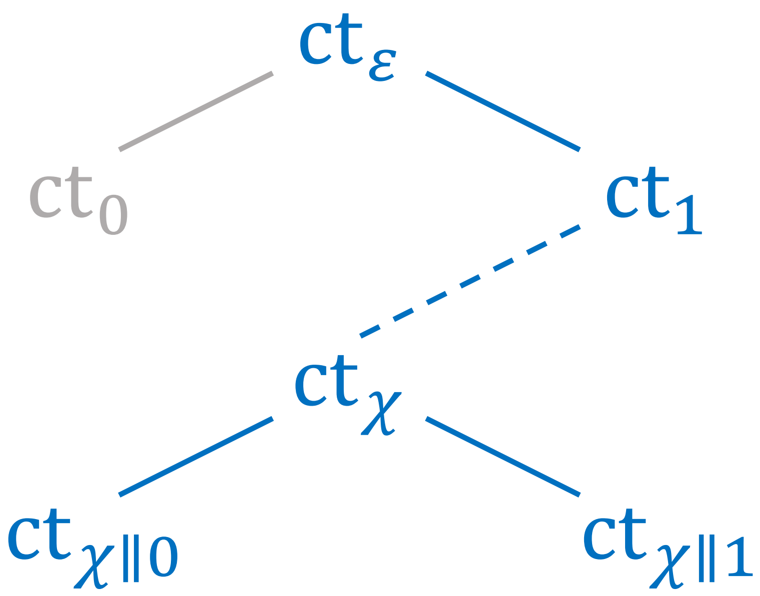
traverse/evaluate = decrypt using FE key sk_f



$i\mathcal{O}$ from FE: Traversal and Evaluation

$\mathrm{ct}_\chi(C, \chi, \ldots)$

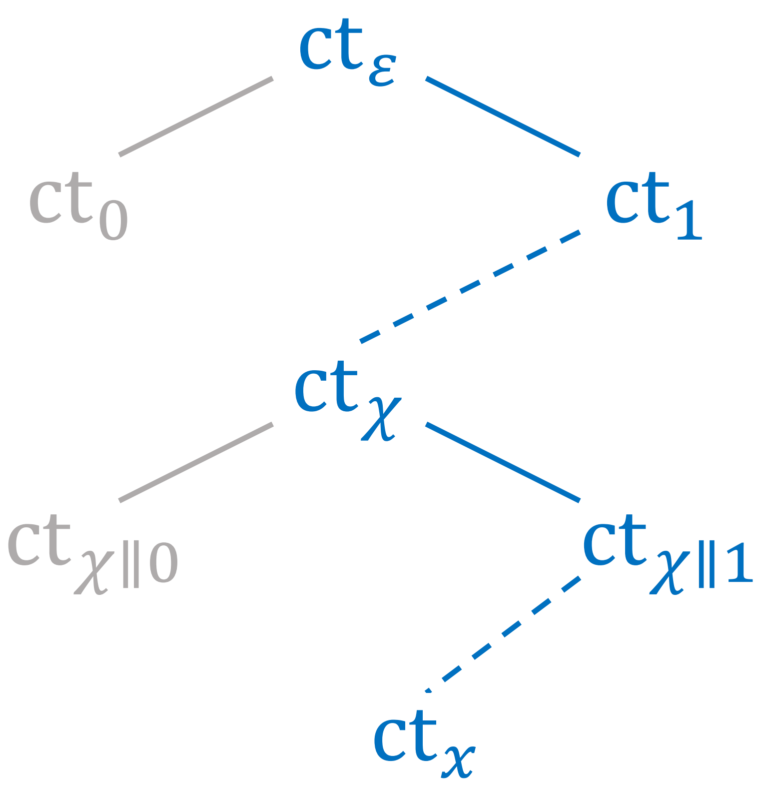
traverse/evaluate = decrypt using FE key sk_f



$i\mathcal{O}$ from FE: Traversal and Evaluation

$\mathrm{ct}_\chi(C, \chi, \ldots)$

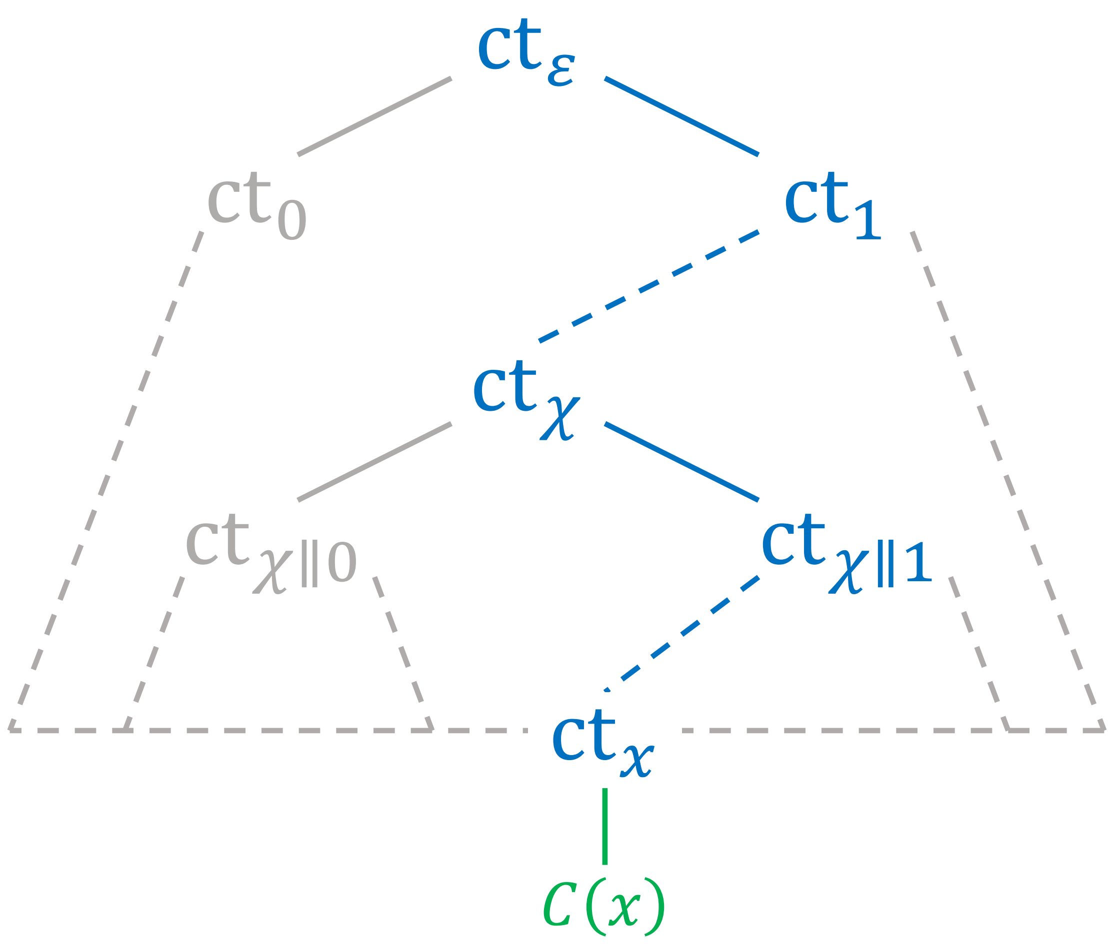
traverse/evaluate = decrypt using FE key sk_f



$i\mathcal{O}$ from FE: Traversal and Evaluation

$\mathrm{ct}_\chi(C, \chi, \dots)$

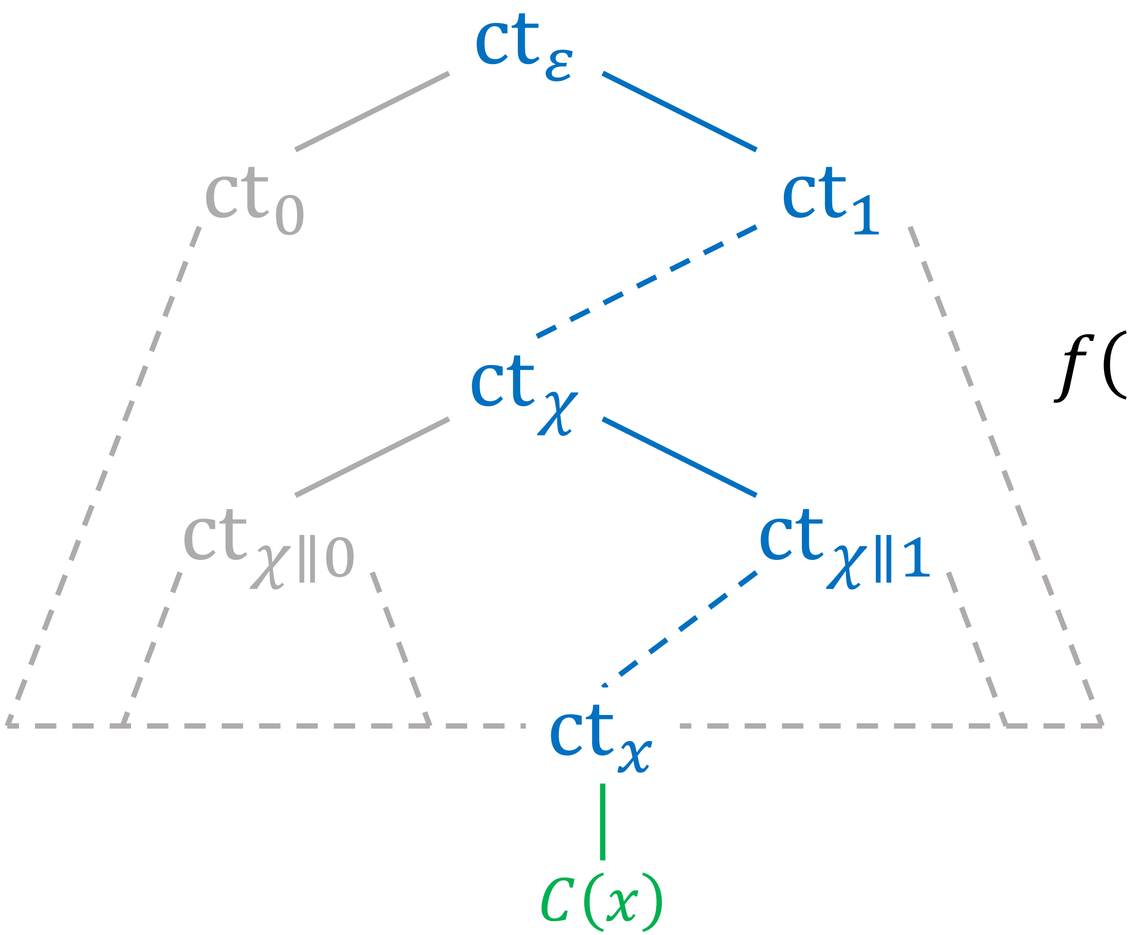
traverse/evaluate = decrypt using FE key sk_f



$i\mathcal{O}$ from FE: Traversal and Evaluation

$\mathrm{ct}_{\chi}(C, \chi, \dots)$

traverse/evaluate = decrypt using FE key sk_f

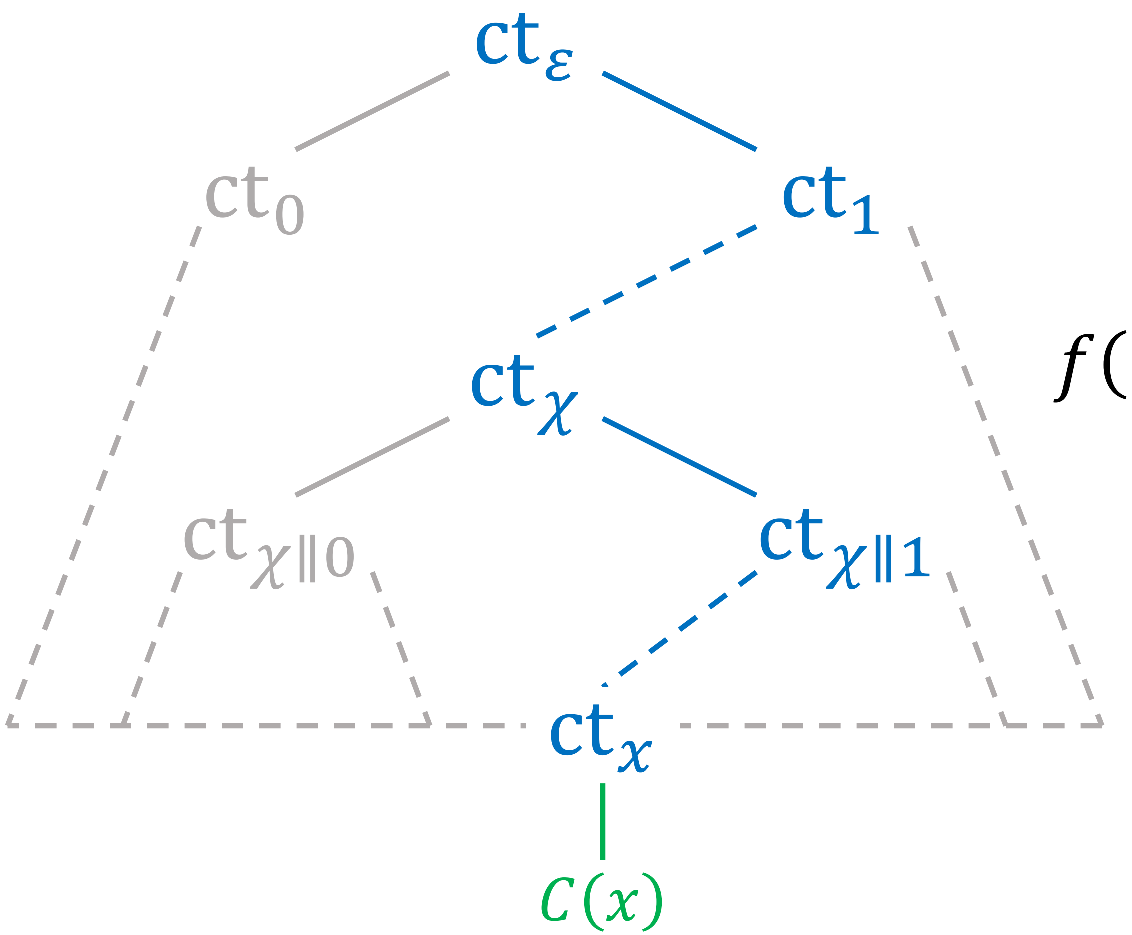


$f(C, \chi, \dots) = (\mathrm{ct}_{\chi\|0}, \mathrm{ct}_{\chi\|1})$ when $|\chi| < D$

$i\mathcal{O}$ from FE: Traversal and Evaluation

$\mathrm{ct}_{\chi}(C, \chi, \dots)$

traverse/evaluate = decrypt using FE key sk_f



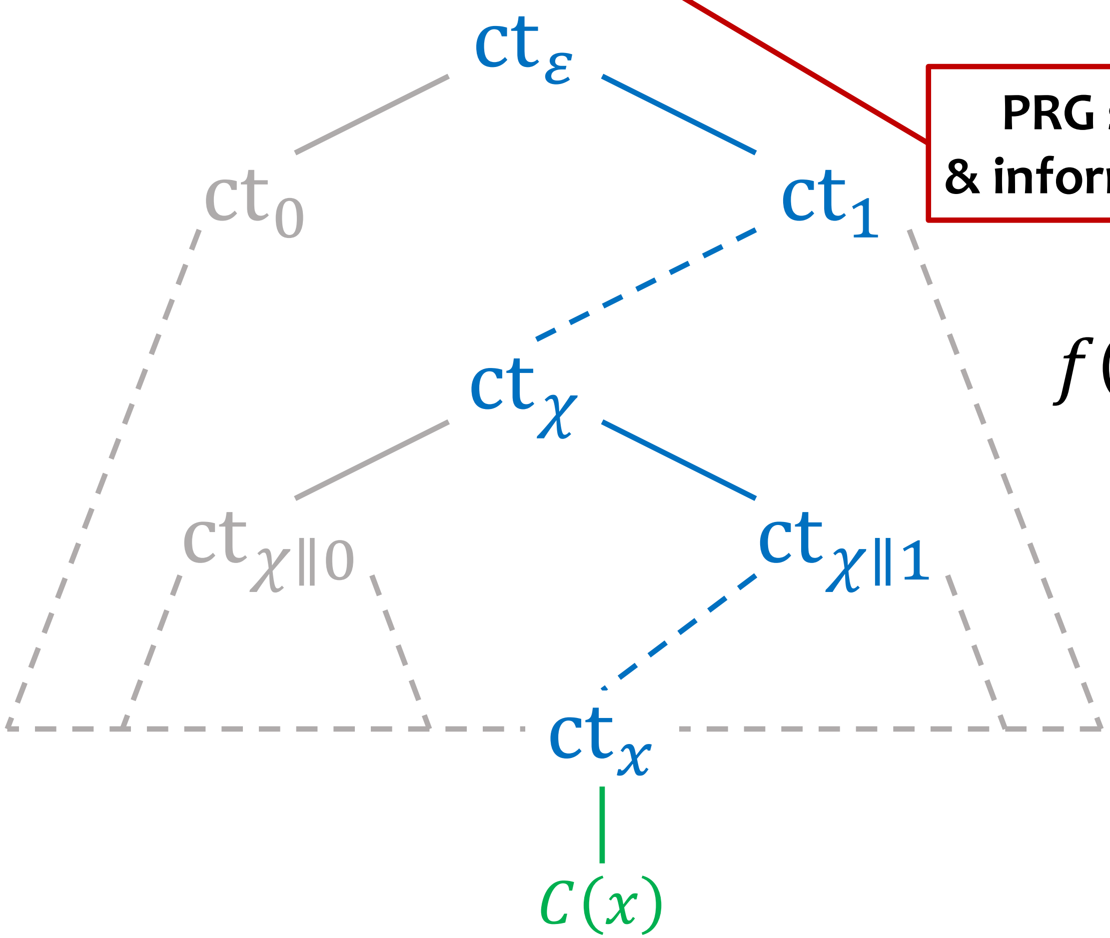
$f(C, \chi, \dots) = (\mathrm{ct}_{\chi\|0}, \mathrm{ct}_{\chi\|1})$ when $|\chi| < D$

$f(C, \chi, \dots) = C(\chi)$ when $|\chi| = D$

$i\mathcal{O}$ from FE: Traversal and Evaluation

$\mathrm{ct}_\chi(C, \chi, \boxed{\ldots})$

traverse/evaluate = decrypt using FE key sk_f



PRG seed / PRF key & information for proof

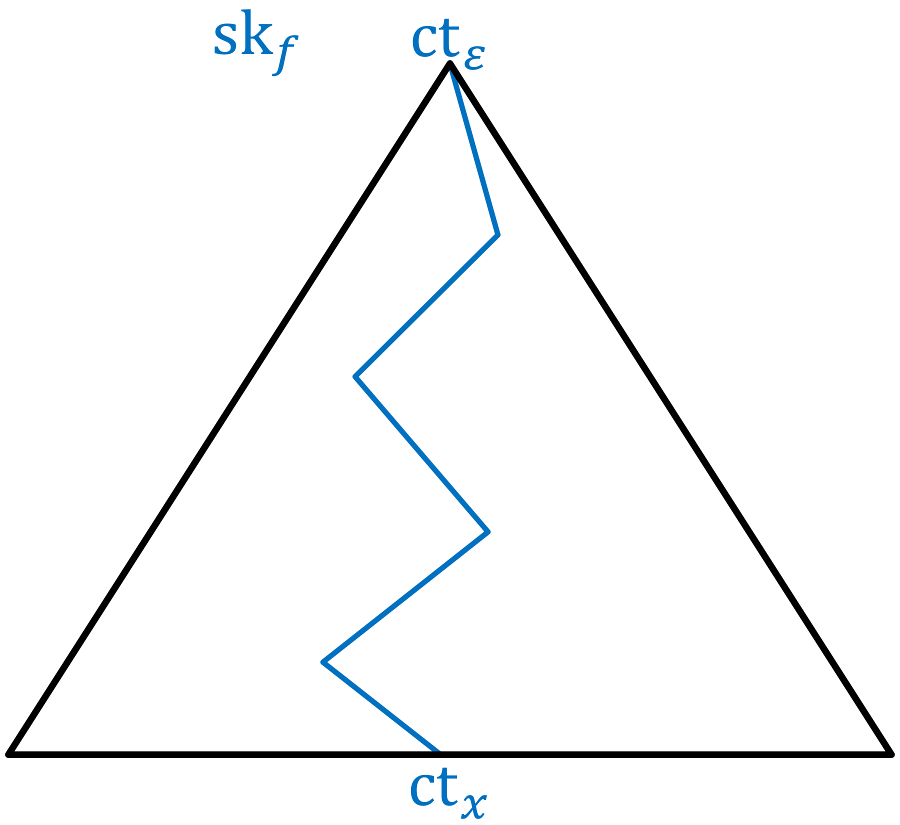
$f(C, \chi, \boxed{\ldots}) = \boxed{(\mathrm{ct}_{\chi\|0}, \mathrm{ct}_{\chi\|1})}$ when $|\chi| < D$

needs FE. Enc randomness

$f(C, \chi, \ldots) = C(\chi)$ when $|\chi| = D$

$i\mathcal{O}$ from FE

$\mathrm{ct}_\chi(C, \chi)$



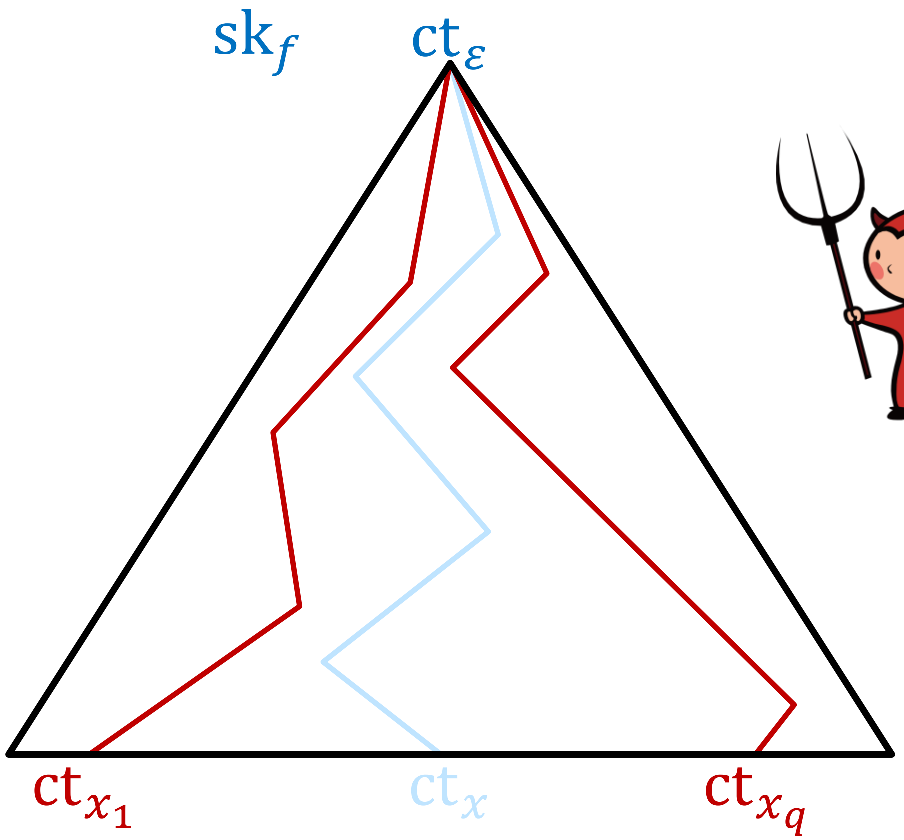
$\mathrm{sk}_f : (C, \chi) \mapsto (\mathrm{ct}_{\chi\|0}, \mathrm{ct}_{\chi\|1}) / C(\chi)$

$\mathrm{Obf}(C) = \left(\mathrm{sk}_f, \mathrm{ct}_\varepsilon\right)$

$i\mathcal{O}$ from FE

$\mathsf{ct}_\chi(C, \chi)$

$\mathsf{sk}_f : (C, \chi) \mapsto (\mathsf{ct}_{\chi\|0}, \mathsf{ct}_{\chi\|1}) / C(\chi)$



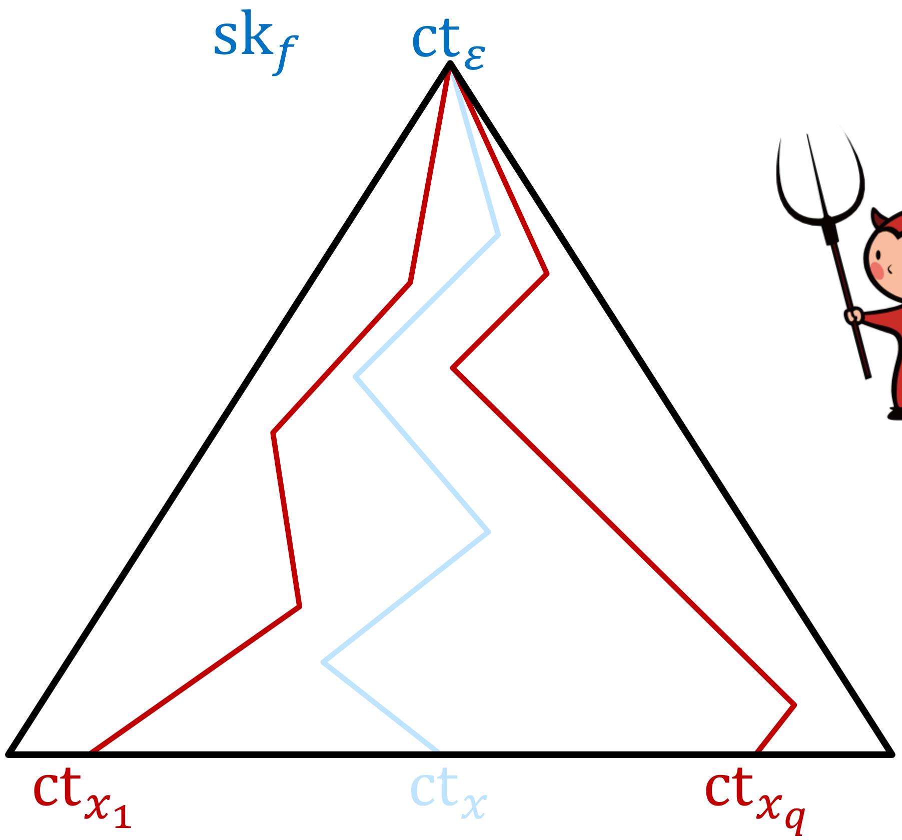
$$\mathsf{Obf}(C) = \left(\mathsf{sk}_f, \mathsf{ct}_\varepsilon\right)$$

← could evaluate at many inputs.

$i\mathcal{O}$ from FE

$\mathsf{ct}_\chi(C,\chi)$

$\mathsf{sk}_f : (C,\chi) \mapsto (\mathsf{ct}_{\chi\|0}, \mathsf{ct}_{\chi\|1}) / C(\chi)$



$$\mathsf{Obf}(C) = \left(\mathsf{sk}_f, \mathsf{ct}_\varepsilon\right)$$

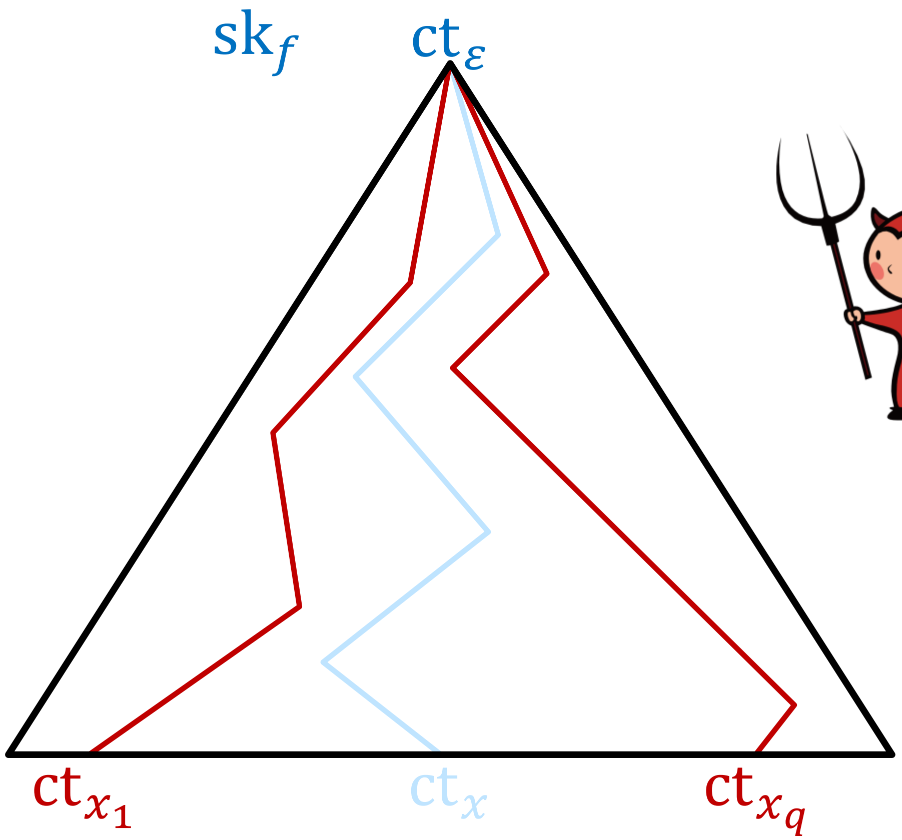
← could evaluate at many inputs.
Do **not know where** it explores.

Simulation. Cannot hardwire **all** evaluations into **short** $\mathsf{Obf}(C)$.

Proof. Must go over **all** x, incurring **exponential loss.**

$i\mathcal{O}$ from FE

$\mathrm{ct}_\chi(C, \chi)$



$\mathrm{sk}_f \colon (C, \chi) \mapsto (\mathrm{ct}_{\chi\|0}, \mathrm{ct}_{\chi\|1}) / C(\chi)$

$$\mathrm{Obf}(C) = (\mathrm{sk}_f, \mathrm{ct}_\varepsilon)$$

← could evaluate at many inputs.
Do **not know where** it explores.

Simulation. Cannot hardwire **all** evaluations into **short** $\mathrm{Obf}(C)$.

Proof. Must go over **all** x, incurring **exponential loss.**

Idealized models could help?

Simplified Ideas in ROM

$\mathrm{ct}_{\chi}(C, \chi)$ $\qquad$ $\mathrm{sk}_f\colon (C, \chi) \mapsto h(\chi) \oplus \left((\mathrm{ct}_{\chi\|0}, \mathrm{ct}_{\chi\|1}) / C(\chi) \right)$

$\mathrm{ct}_{\varepsilon}$

Simplified Ideas in ROM

$\mathrm{ct}_\chi(C, \chi)$

$\mathrm{sk}_f\colon (C, \chi) \mapsto h(\chi) \oplus \big((\mathrm{ct}_{\chi\|0}, \mathrm{ct}_{\chi\|1})/C(\chi)\big)$

ct_ε

ct_0 ct_1

Simplified Ideas in ROM

$\mathrm{ct}_\chi(C, \chi)$

$\mathrm{sk}_f : (C, \chi) \mapsto h(\chi) \oplus \big((\mathrm{ct}_{\chi\|0}, \mathrm{ct}_{\chi\|1}) / C(\chi)\big)$

must query $h(\varepsilon)$
and XOR to get

ct_ε

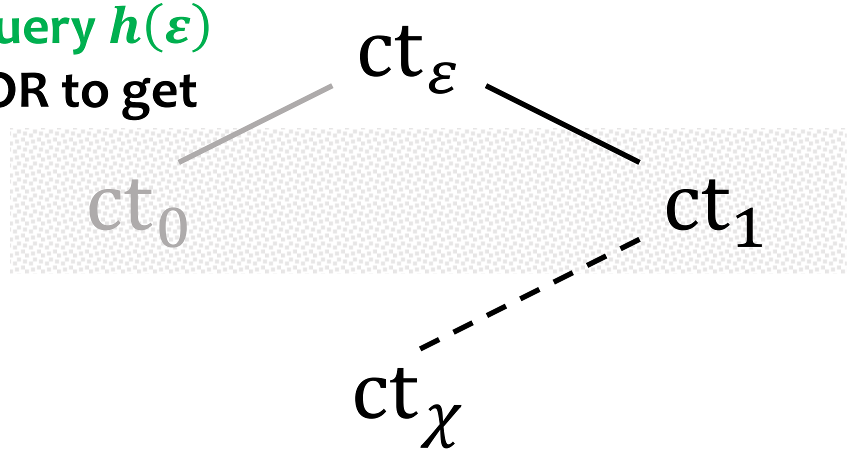
ct_0 $\qquad\qquad$ ct_1

Simplified Ideas in ROM

$\mathrm{ct}_\chi(C,\chi)$

$\mathrm{sk}_f\colon (C,\chi) \mapsto h(\chi) \oplus \left((\mathrm{ct}_{\chi\|0}, \mathrm{ct}_{\chi\|1})/C(\chi)\right)$

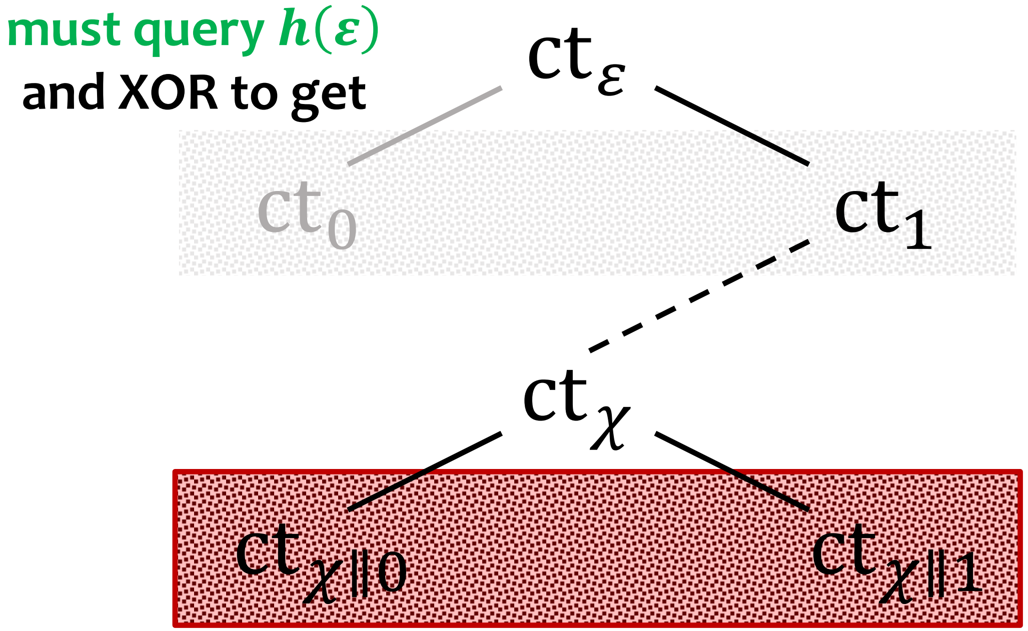
must query $h(\varepsilon)$
and XOR to get



Simplified Ideas in ROM

$\mathrm{ct}_\chi(C, \chi)$

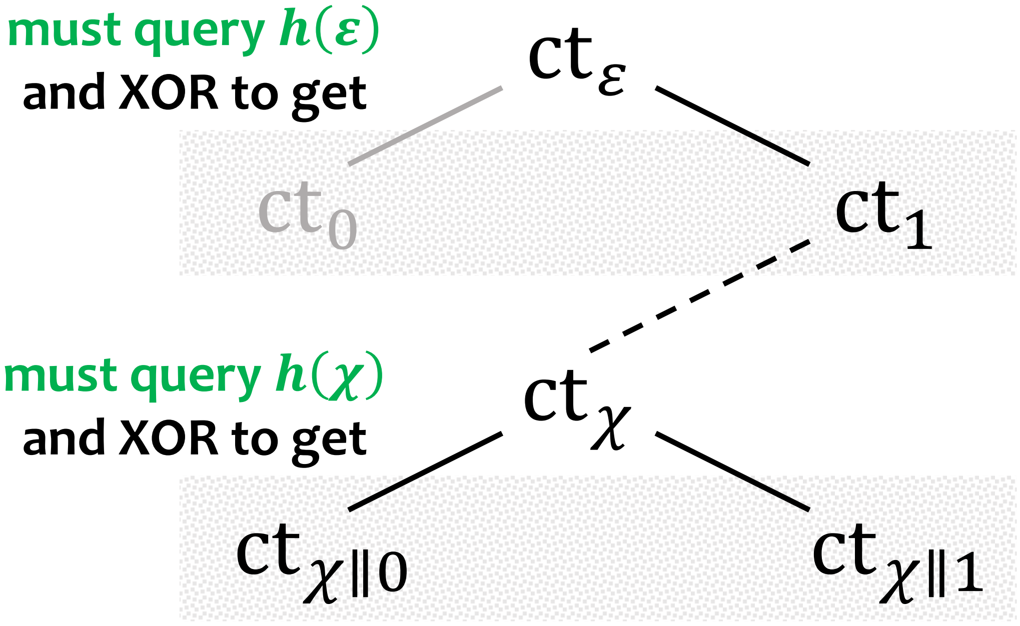
$\mathrm{sk}_f\colon (C, \chi) \mapsto h(\chi) \oplus \big((\mathrm{ct}_{\chi\|0}, \mathrm{ct}_{\chi\|1})/C(\chi)\big)$



Simplified Ideas in ROM

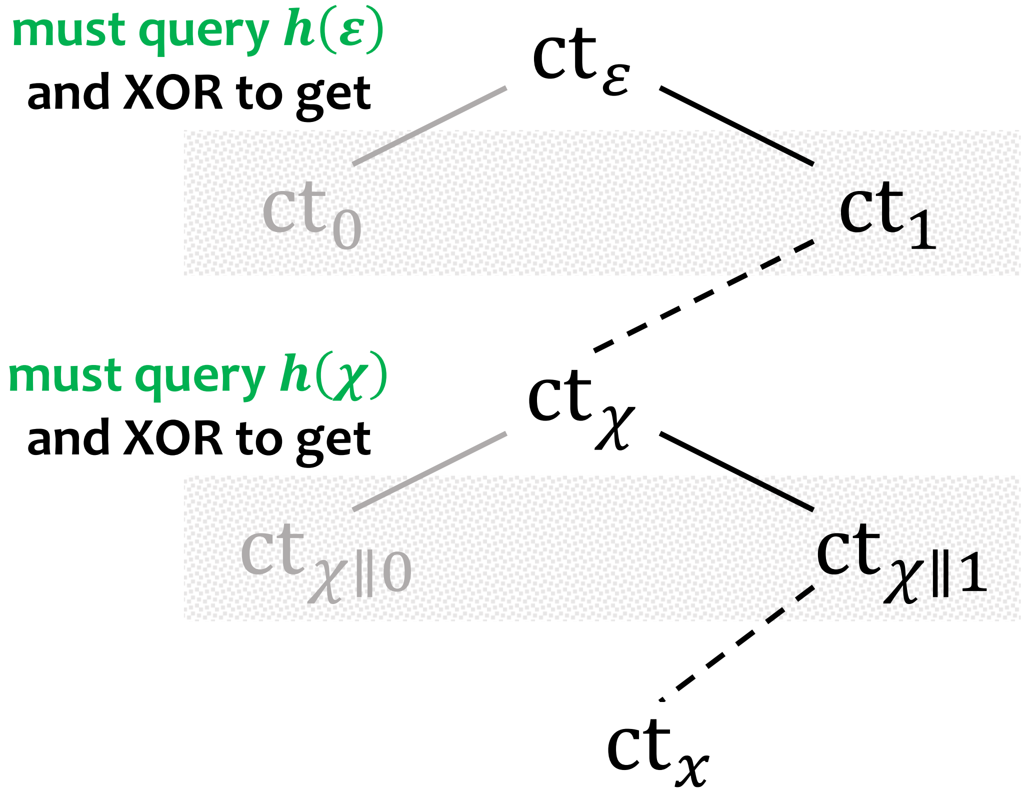
$\mathrm{ct}_\chi(C, \chi)$

$\mathrm{sk}_f\colon (C, \chi) \mapsto h(\chi) \oplus \big((\mathrm{ct}_{\chi\|0}, \mathrm{ct}_{\chi\|1})/C(\chi)\big)$



Simplified Ideas in ROM

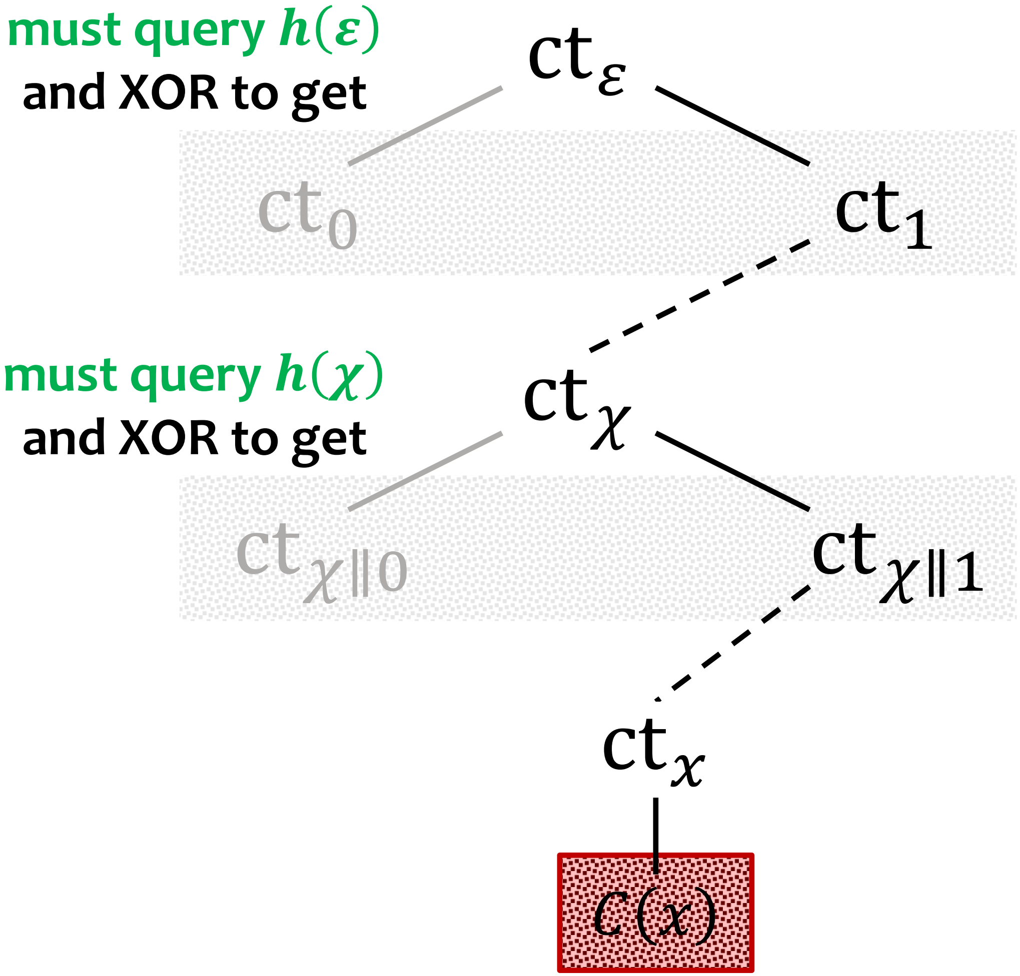
$\mathrm{ct}_\chi(C,\chi)$ $\quad\quad$ $\mathrm{sk}_f\colon (C,\chi) \mapsto h(\chi) \oplus \big((\mathrm{ct}_{\chi\|0}, \mathrm{ct}_{\chi\|1})/C(\chi)\big)$



Simplified Ideas in ROM

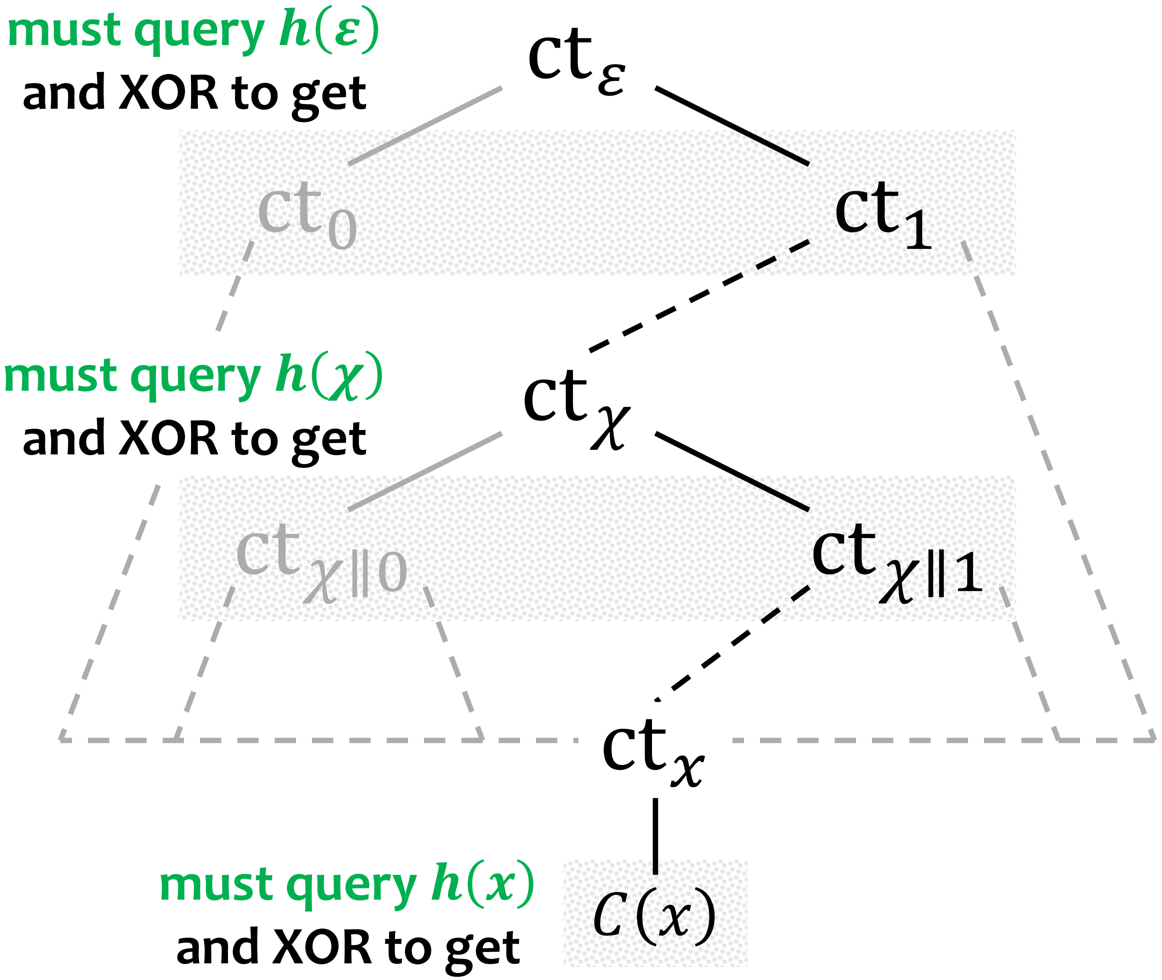
$\mathsf{ct}_\chi(C,\chi)$

$\mathsf{sk}_f\colon (C,\chi) \mapsto h(\chi) \oplus \big((\mathsf{ct}_{\chi\|0}, \mathsf{ct}_{\chi\|1})/C(\chi)\big)$



Simplified Ideas in ROM

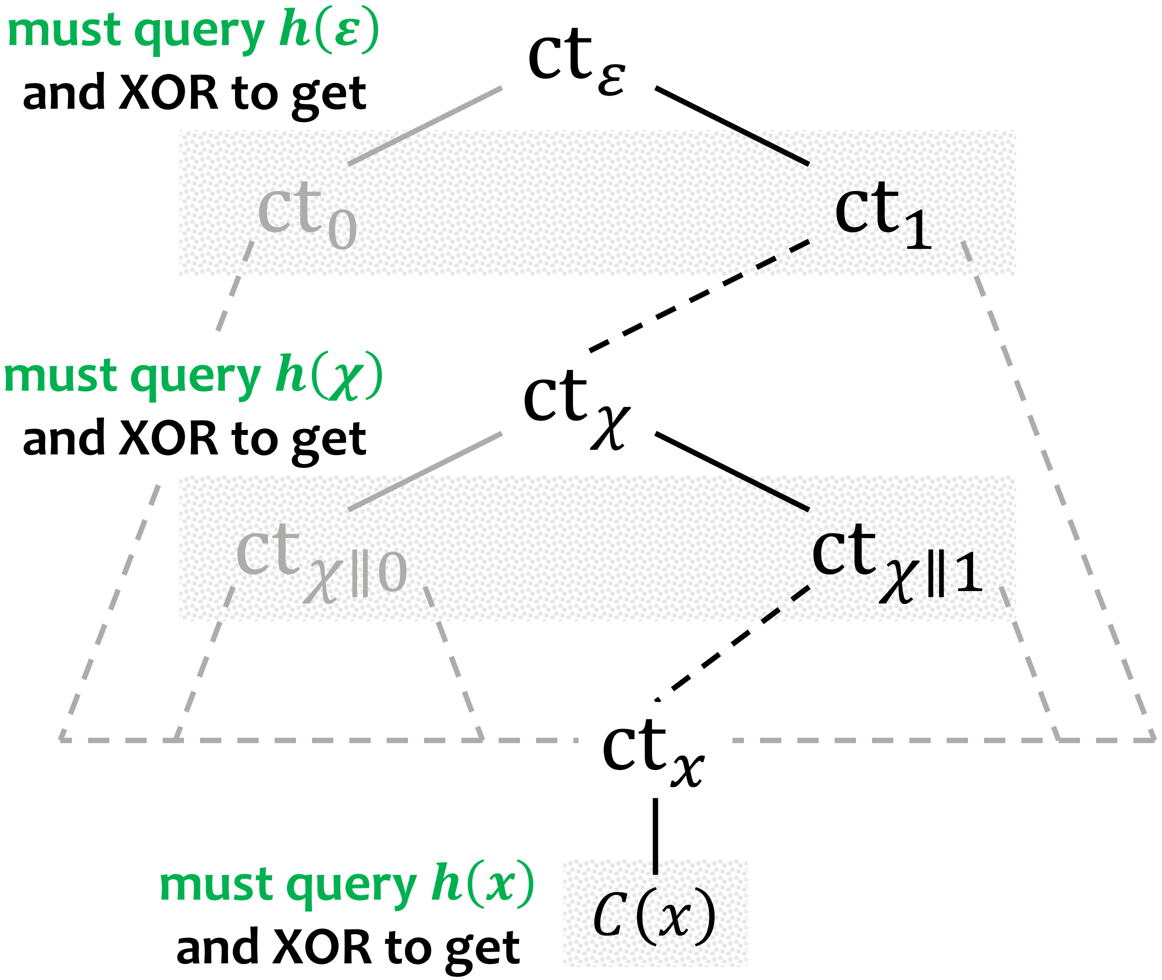
$\mathrm{ct}_\chi(C,\chi)$ $\quad\quad$ $\mathrm{sk}_f\colon (C,\chi) \mapsto h(\chi) \oplus \big((\mathrm{ct}_{\chi\|0}, \mathrm{ct}_{\chi\|1})/C(\chi)\big)$



Simplified Ideas in ROM

$\mathsf{ct}_\chi(C,\chi)$

$\mathsf{sk}_f\colon (C,\chi) \mapsto h(\chi) \oplus \big((\mathsf{ct}_{\chi\|0}, \mathsf{ct}_{\chi\|1})/C(\chi)\big)$

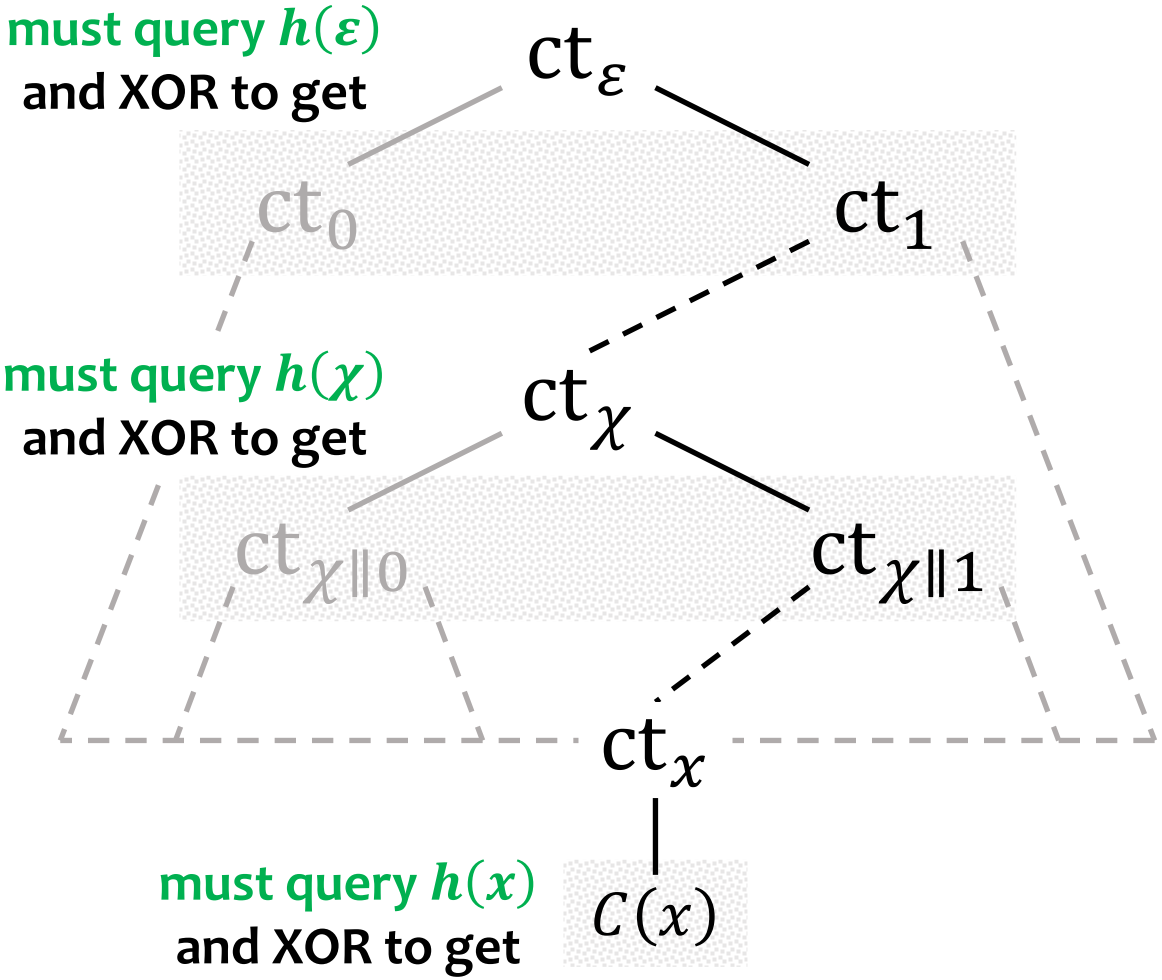


$(\mathsf{ct}_{\chi\|0}, \mathsf{ct}_{\chi\|1})/C(\chi)$ is **hidden**
if $h(\chi)$ is **not queried.**

Simplified Ideas in ROM

$\mathrm{ct}_\chi(C, \chi)$

$\mathrm{sk}_f : (C, \chi) \mapsto h(\chi) \oplus \big((\mathrm{ct}_{\chi\|0}, \mathrm{ct}_{\chi\|1}) / C(\chi)\big)$



$(\mathrm{ct}_{\chi\|0}, \mathrm{ct}_{\chi\|1}) / C(\chi)$ is **hidden**
if $h(\chi)$ is **not queried.**

Observe RO queries
to know exploration path.

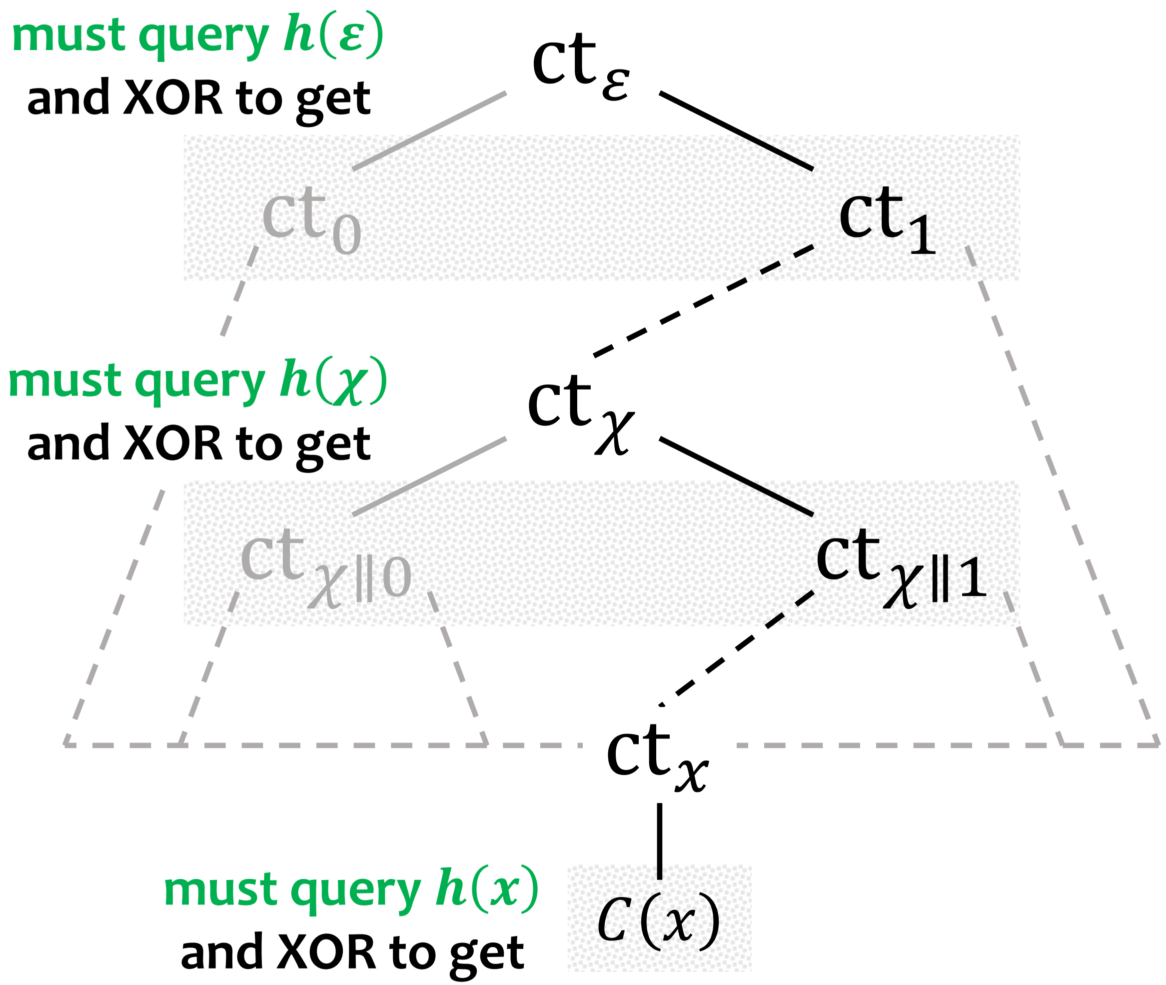
Program RO responses
to hardwire $C(x)$.

Simplified Ideas in ROM

$\mathrm{ct}_\chi(C, \chi)$

$\mathrm{sk}_f\colon (C, \chi) \mapsto \boxed{h(\chi)} \oplus \big((\mathrm{ct}_{\chi\|0}, \mathrm{ct}_{\chi\|1}) / C(\chi)\big)$

✖ cannot query RO in circuit sent to $\mathbf{FE.KeyGen}$



$(\mathrm{ct}_{\chi\|0}, \mathrm{ct}_{\chi\|1}) / C(\chi)$ is **hidden** if $h(\chi)$ is **not queried.**

Observe RO queries to know exploration path.

Program RO responses to hardwire $C(x)$.

Simplified Fix in $\mathsf{Pr}\mathcal{O}\mathsf{M}$

$$\mathrm{ct}_{\chi}(C, \chi, k) \qquad \mathrm{sk}_f\colon (C, \chi, k) \mapsto F(k, \chi) \oplus \left((\mathrm{ct}_{\chi\|0}, \mathrm{ct}_{\chi\|1})/C(\chi)\right)$$

$$\tilde{C} = \left(\mathrm{sk}_f, \mathrm{ct}_{\varepsilon}, h\right)$$

Simplified Fix in $\mathrm{Pr}\mathcal{O}\mathrm{M}$

obfuscator can **use code of *F* with *k***

$$\mathrm{ct}_\chi(C, \chi, k) \qquad \mathrm{sk}_f\colon (C, \chi, k) \mapsto F(k, \chi) \oplus \big((\mathrm{ct}_{\chi\|0}, \mathrm{ct}_{\chi\|1})/C(\chi)\big)$$

$$\tilde{C} = \left(\mathrm{sk}_f, \mathrm{ct}_\varepsilon, h\right)$$

Simplified Fix in Pr$\mathcal{O}$M

obfuscator can **use code of F with k**

$\mathrm{ct}_\chi(C, \chi, k)$

$\mathrm{sk}_f \colon (C, \chi, k) \mapsto F(k, \chi) \oplus \left((\mathrm{ct}_{\chi\|0}, \mathrm{ct}_{\chi\|1})/C(\chi)\right)$

$\tilde{C} = \left(\mathrm{sk}_f, \mathrm{ct}_\varepsilon, h\right)$

$\mathrm{Dec}\left(\mathrm{sk}_f, \mathrm{ct}_\chi\right) \oplus h(\chi) = (\mathrm{ct}_{\chi\|0}, \mathrm{ct}_{\chi\|1})/C(\chi)$

evaluator can **call evaluation oracle with h**

Simplified Simulator

$$\mathrm{ct}_\chi \begin{cases} C, \chi, k \\ \sigma_\chi \end{cases} \qquad \mathrm{sk}_f \begin{cases} (C, \chi, k) \mapsto F(k, \chi) \oplus \left((\mathrm{ct}_{\chi\|0}, \mathrm{ct}_{\chi\|1})/C(\chi)\right) \\ \sigma_\chi \mapsto G(\sigma_\chi) \end{cases}$$

$$\tilde{C} = \left(\mathrm{sk}_f, \mathrm{ct}_\varepsilon, h\right) \qquad \mathrm{Dec}\left(\mathrm{sk}_f, \mathrm{ct}_\chi\right) \oplus h(\chi) = (\mathrm{ct}_{\chi\|0}, \mathrm{ct}_{\chi\|1})/C(\chi)$$

Dec invariant preserved during simulation

Simplified Simulator

$$\mathrm{ct}_\chi \begin{cases} C, \chi, k \\ \sigma_\chi \end{cases} \qquad \mathrm{sk}_f \begin{cases} (C, \chi, k) \mapsto F(k, \chi) \oplus \left((\mathrm{ct}_{\chi\|0}, \mathrm{ct}_{\chi\|1})/C(\chi)\right) \\ \sigma_\chi \mapsto G(\sigma_\chi) \end{cases}$$

$$\tilde{C} = \left(\mathrm{sk}_f, \mathrm{ct}_\varepsilon, h\right) \qquad \mathrm{Dec}\left(\mathrm{sk}_f, \mathrm{ct}_\chi\right) \oplus h(\chi) = (\mathrm{ct}_{\chi\|0}, \mathrm{ct}_{\chi\|1})/C(\chi)$$

Dec invariant preserved during simulation

Simulation.

- ct_χ is FE ciphertext of σ_χ
- $h(\chi)$ is programmed as $G(\sigma_\chi) \oplus \left((\mathrm{ct}_{\chi\|0}, \mathrm{ct}_{\chi\|1})/C(\chi)\right)$

Simplified Simulator

$$\mathrm{ct}_\chi \begin{cases} C, \chi, k \\ \sigma_\chi \end{cases} \qquad \mathrm{sk}_f \begin{cases} (C, \chi, k) \mapsto F(k, \chi) \oplus \left((\mathrm{ct}_{\chi\|0}, \mathrm{ct}_{\chi\|1}) / C(\chi) \right) \\ \sigma_\chi \mapsto G(\sigma_\chi) \end{cases}$$

$$\tilde{C} = \left(\mathrm{sk}_f, \mathrm{ct}_\varepsilon, h \right) \qquad \mathrm{Dec}\left(\mathrm{sk}_f, \mathrm{ct}_\chi \right) \oplus h(\chi) = (\mathrm{ct}_{\chi\|0}, \mathrm{ct}_{\chi\|1}) / C(\chi)$$

Dec invariant preserved during simulation

Simulation.

- ct_χ is FE ciphertext of σ_χ

  $\tilde{C} = \left(\mathrm{sk}_f, \mathrm{ct}_\varepsilon, h \right)$ does not use C
- $h(\chi)$ is programmed as $G(\sigma_\chi) \oplus \left((\mathrm{ct}_{\chi\|0}, \mathrm{ct}_{\chi\|1}) / C(\chi) \right)$

  $C(x)$ is queried upon query to $h(x)$

Simplified Simulator

$$\text{ct}_\chi \begin{cases} C, \chi, k \\ \sigma_\chi \end{cases} \qquad \text{sk}_f \begin{cases} (C, \chi, k) \mapsto F(k, \chi) \oplus \big((\text{ct}_{\chi\|0}, \text{ct}_{\chi\|1})/C(\chi)\big) \\ \sigma_\chi \mapsto G(\sigma_\chi) \end{cases}$$

$$\tilde{C} = \big(\text{sk}_f, \text{ct}_\varepsilon, h\big) \qquad \text{Dec}\big(\text{sk}_f, \text{ct}_\chi\big) \oplus h(\chi) = (\text{ct}_{\chi\|0}, \text{ct}_{\chi\|1})/C(\chi)$$

Dec invariant preserved during simulation

Simulation.

- ct_χ is FE ciphertext of σ_χ
  $\tilde{C} = \big(\text{sk}_f, \text{ct}_\varepsilon, h\big)$ does not use C
- $h(\chi)$ is programmed as $G(\sigma_\chi) \oplus \big((\text{ct}_{\chi\|0}, \text{ct}_{\chi\|1})/C(\chi)\big)$
  $C(x)$ is queried upon query to $h(x)$

⚠ **More tricks in actual scheme to make proof work**

Pseudorandom Oracle Model

(novel model for **ideal** hash functions **with code**)

Future. Other uses of $\mathsf{Pr}\mathcal{O}\mathsf{M}$?

+ Functional Encryption $\Longrightarrow$ **Ideal Obfuscation**

Thanks!

ia.cr/2022/1204

luoji@cs.washington.edu / luoji.bio

Alternative Interpretations

Hardware. tokens **for PRF** (implements $\mathrm{Pr}\mathcal{O}\mathrm{M}$)
$\implies$ ideal obfuscation **for all circuits**

Bootstrapping. ideal obfuscation **for PRF** (candidate: hash functions)
$\implies$ ideal obfuscation **for all circuits**

Alternative Interpretations

Hardware. tokens **for PRF** (implements Pr$\mathcal{O}$M)
$\implies$ ideal obfuscation **for all circuits**

Bootstrapping. ideal obfuscation **for PRF** (candidate: hash functions)
$\implies$ ideal obfuscation **for all circuits**

Previous.

- tokens for **more complex** functionality [GISVW, DMMN, BCGHKR, NFRCLSG]
- bootstrapping from **more complex underlying class** of circuits [GGHRSW, A, CLTV]
    - no candidate / requires generic multilinear maps

Recent Models of Hash Functions [Z, CCS, CCGCS]

Pr$\mathcal{O}$M. **Enables applications by modelling use of code.**

Recent Models of Hash Functions [Z, CCS, CCGCS]

Augmented ROM. Transforms secure in AROM
avoid known proofs of uninstantiability.
Gets stronger proofs by capturing contrivance.

Pr$\mathcal{O}$M. **Enables applications by modelling use of code.**

Recent Models of Hash Functions [Z, CCS, CCGCS]

Augmented ROM. Transforms secure in AROM
avoid known proofs of uninstantiability.
Gets stronger proofs by capturing contrivance.

Low-Degree ROM. SNARK in $\mathcal{O}$-model for $\mathrm{NP}^{\mathcal{O}}$.
Candidate instantiation is obfuscating algebraic PRF.
Enables applications by adding arithmetic structure.

Pr$\mathcal{O}$M. **Enables applications by modelling use of code.**

Recent Models of Hash Functions [Z, CCS, CCGCS]

Augmented ROM. Transforms secure in AROM
avoid known proofs of uninstantiability.
Gets stronger proofs by capturing contrivance.

Low-Degree ROM. SNARK in $\mathcal{O}$-model for $\mathrm{NP}^{\mathcal{O}}$.
Candidate instantiation is obfuscating algebraic PRF.
Enables applications by adding arithmetic structure.

Arithmetized ROM. PCD in $\mathcal{O}$-model for computation in $\mathcal{O}$-model.
Candidate instantiation is hash functions.
Enables applications by modelling very specific use of code and adding arithmetic structure. (^ SAT reduction)

Pr$\mathcal{O}$M. **Enables applications by modelling use of code.**

Relation with Best-Possible Obfuscation [BR]

Best-Possible. $\mathcal{A}(i\mathcal{O}(C_0)) \approx \mathcal{S}(C_1)$.

Folklore. $i\mathcal{O}(\text{padded}(C)) \approx i\mathcal{O}(\text{Obf}(C))$.
LHS hides whatever RHS hides.
$i\mathcal{O}$ **not less secure** than any Obf.

Relation with Best-Possible Obfuscation [BR]

Best-Possible. $\mathcal{A}(i\mathcal{O}(C_0)) \approx \mathcal{S}(C_1)$.

Folklore. $i\mathcal{O}(\text{padded}(C)) \approx i\mathcal{O}(\text{Obf}(C))$.
LHS hides whatever RHS hides.
$i\mathcal{O}$ **not less secure** than any Obf.

Our result justifies this precondition!

$$\exists\ \text{Obf hiding } \mathcal{A}(C) \implies i\mathcal{O} \text{ hides } \mathcal{A}(C)$$

Relation with Best-Possible Obfuscation [BR]

Best-Possible. $\mathcal{A}(i\mathcal{O}(C_0)) \approx \mathcal{S}(C_1)$.

Folklore. $i\mathcal{O}(\text{padded}(C)) \approx i\mathcal{O}(\text{Obf}(C))$.
LHS hides whatever RHS hides.
$i\mathcal{O}$ **not less secure** than any Obf.

Our result justifies this precondition!

$\exists$ Obf hiding $\mathcal{A}(C) \implies i\mathcal{O}$ hides $\mathcal{A}(C)$

Q. Use $i\mathcal{O}$ in place of ideal obfuscation in applications? (**A.** Not clear.)

$$\Pi^{\text{Obf}} \text{ secure } \overset{?}{\implies} \Pi^{i\mathcal{O}(\text{Obf}(\cdot))} \text{ secure } \implies \Pi^{i\mathcal{O}(\text{padded}(\cdot))} \text{ secure}$$