Adaptive Distributional Security for Garbling Schemes with O(|x|) Online Complexity

Estuardo Alpirez Bock, Chris Brzuska, Pihla Karanko, Sabine Oechsner, Kirthivaasan Puniamurthy

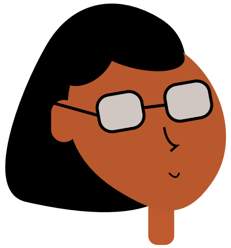
CIPHERA

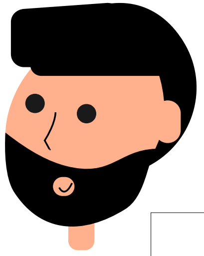
Aalto University

THE UNIVERSITY of EDINBURGH

Preprocessing Multiparty Computation: Offline Garbling

Evaluator



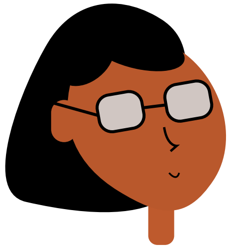


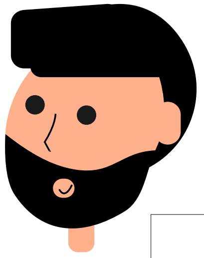
Garbler

$C(\cdot, x_G)$

Preprocessing Multiparty Computation: Offline Garbling

Evaluator



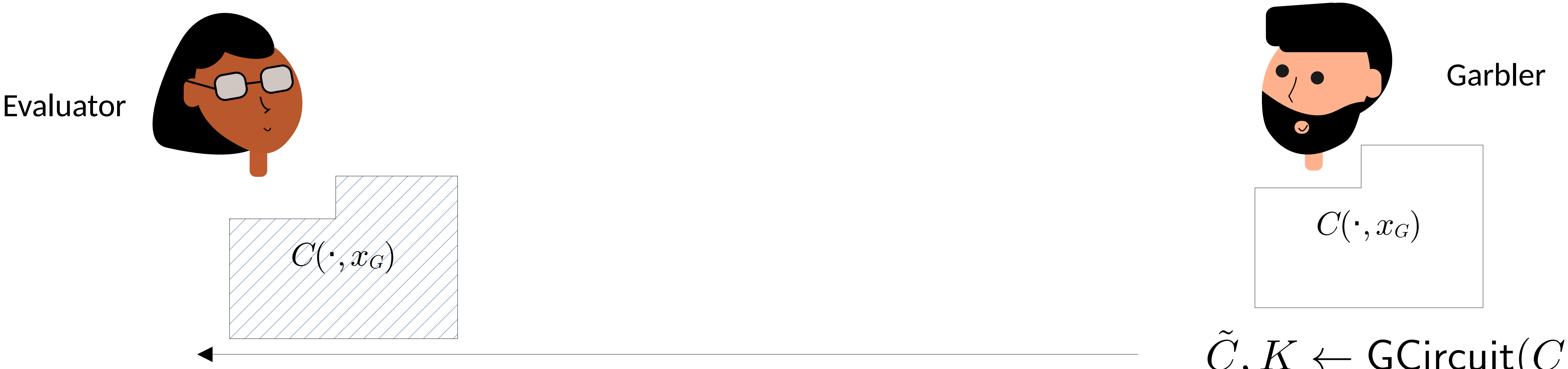


Garbler

$C(\cdot, x_G)$

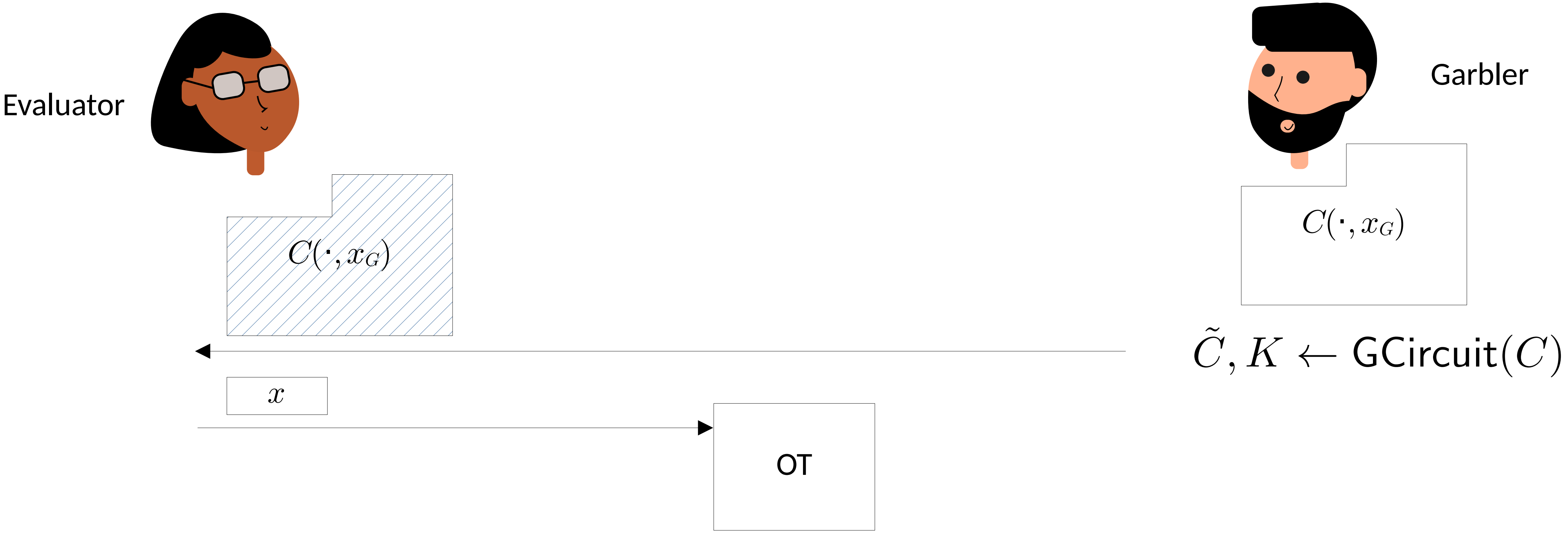
$$\tilde{C}, K \leftarrow \mathsf{GCircuit}(C)$$

Preprocessing Multiparty Computation: Offline Garbling

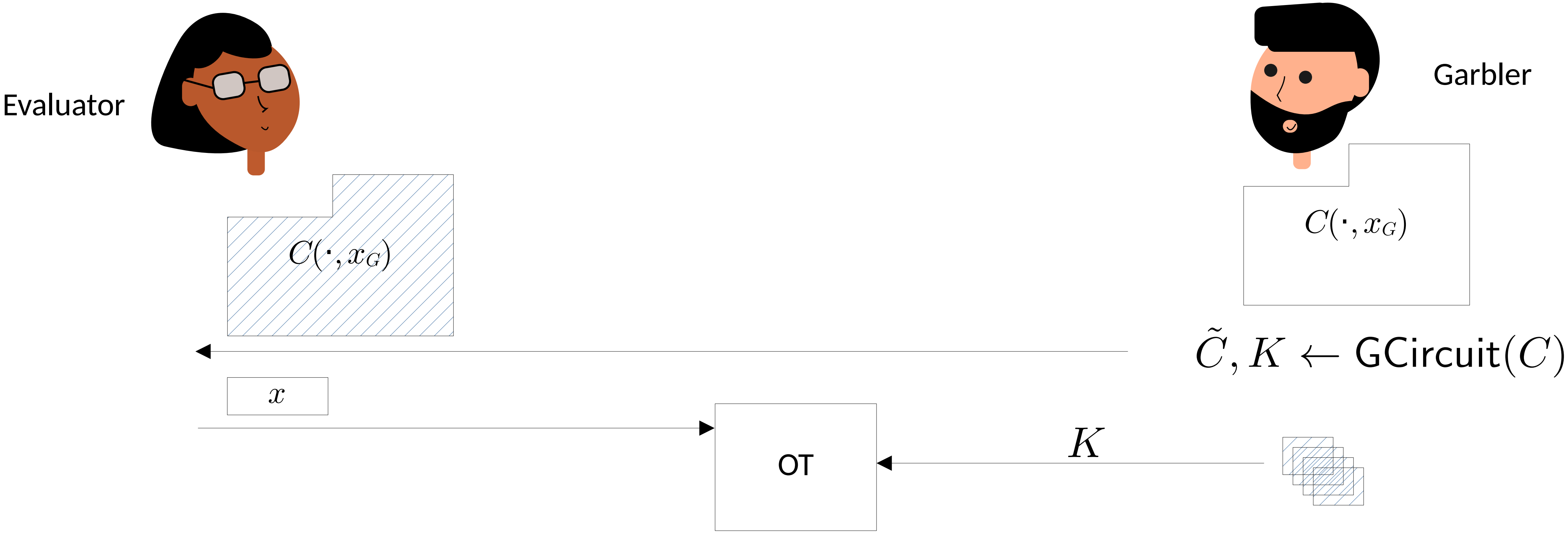


$$\tilde{C}, K \leftarrow \mathsf{GCircuit}(C)$$

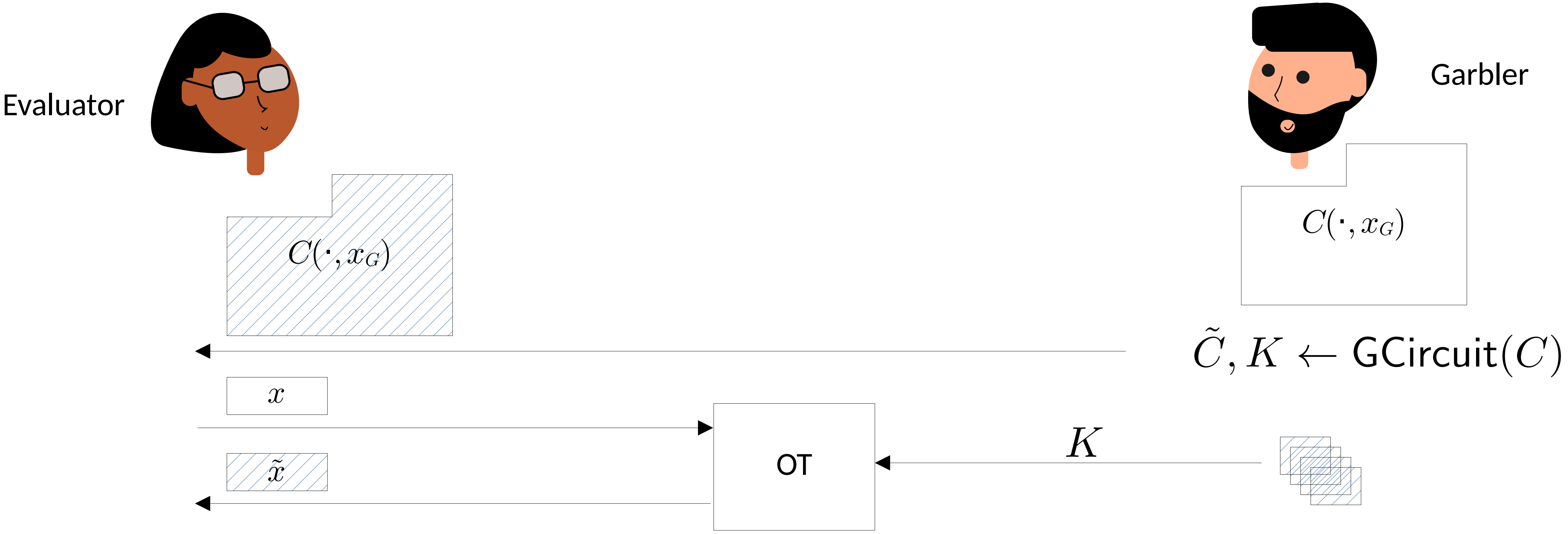
Preprocessing Multiparty Computation: Offline Garbling



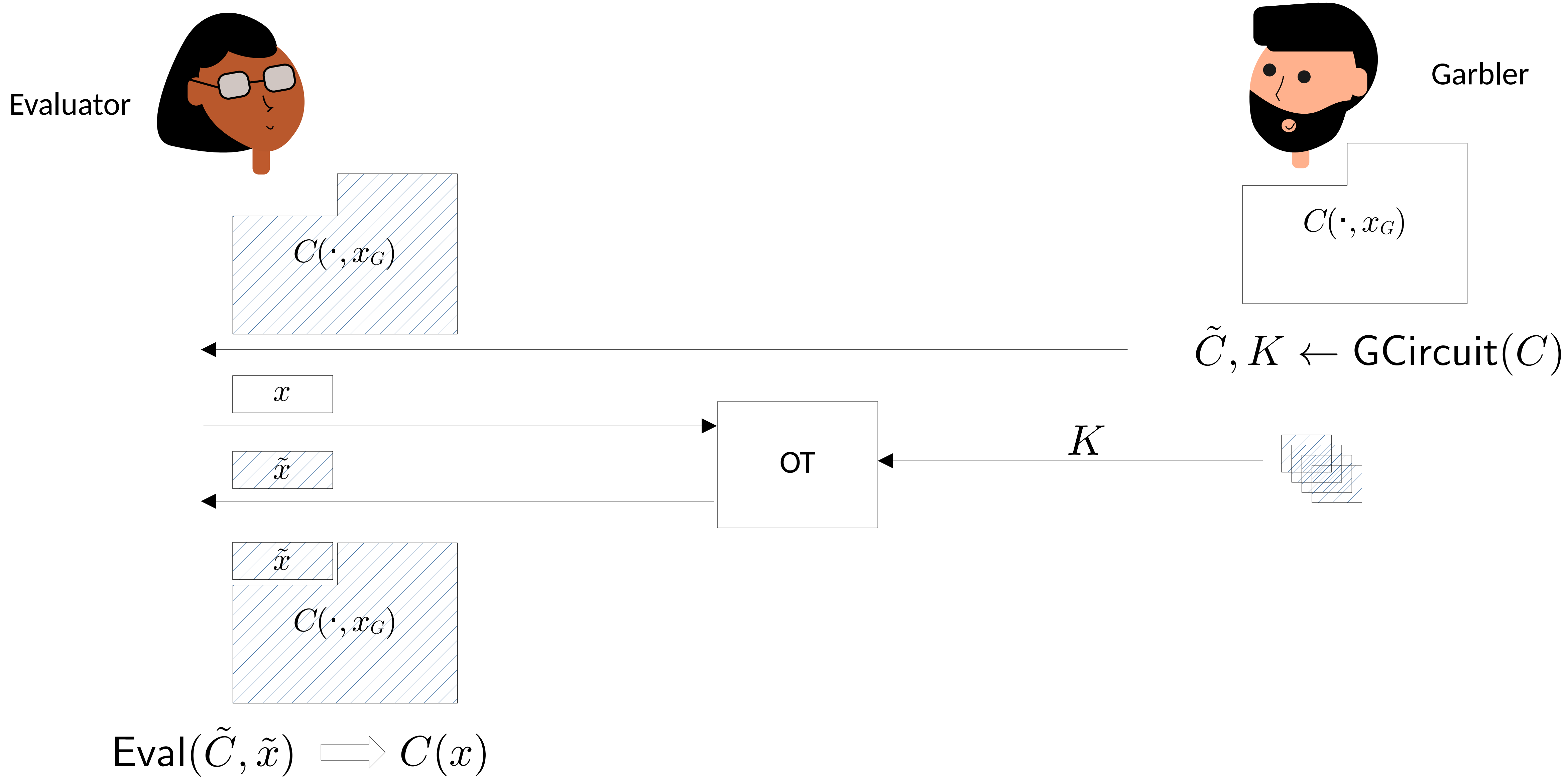
Preprocessing Multiparty Computation: Offline Garbling



Preprocessing Multiparty Computation: Offline Garbling

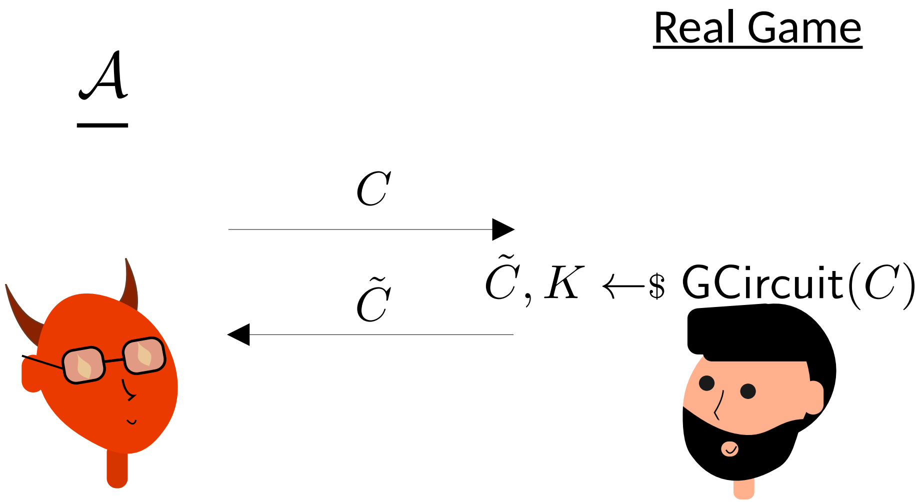


Preprocessing Multiparty Computation: Offline Garbling



Adaptive Simulation Security (SIM) [BHR'12]

(Defined by Bellare, Hoang and Rogaway **[BHR'12]**)



Adaptive Simulation Security (SIM) [BHR'12]

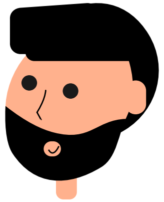
<u>Real Game</u>

$\underline{\mathcal{A}}$

$$\xrightarrow{\quad C \quad}$$

$$\xleftarrow{\quad \tilde{C} \quad}$$

$\tilde{C}, K \leftarrow_{\$} \mathsf{GCircuit}(C)$



Adaptive Simulation Security (SIM) [BHR'12]

Real Game

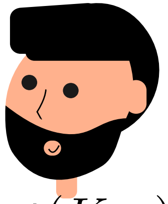
$\underline{\mathcal{A}}$

$\xrightarrow{\quad C \quad}$

$\xleftarrow{\quad \tilde{C} \quad}$ $\tilde{C}, K \leftarrow\$ \ \mathsf{GCircuit}(C)$

$\xrightarrow{\quad x \quad}$

$\xleftarrow{\quad \tilde{x} \quad}$ $\tilde{x} \leftarrow \mathsf{GInput}(K, x)$



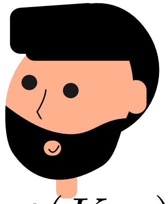
Adaptive Simulation Security (SIM) [BHR'12]

<u>Real Game</u>

$\underline{\mathcal{A}}$

$\xrightarrow{C}$

$\xleftarrow{\tilde{C}}$ $\tilde{C}, K \leftarrow_{\$} \mathsf{GCircuit}(C)$



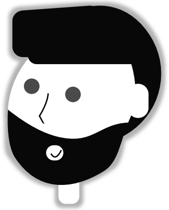
$\xrightarrow{x}$

$\xleftarrow{\tilde{x}}$ $\tilde{x} \leftarrow \mathsf{GInput}(K, x)$

<u>Ideal Game</u> (with ***Simulator***)

$\xrightarrow{C}$

$\underline{\mathcal{S}}$



Adaptive Simulation Security (SIM) [BHR'12]

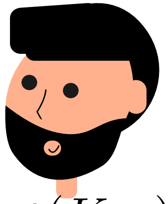
<u>Real Game</u>

$\underline{\mathcal{A}}$

$\xrightarrow{C}$

$\tilde{C}, K \leftarrow_{\$} \mathsf{GCircuit}(C)$

$\xleftarrow{\tilde{C}}$



$\xrightarrow{x}$

$\tilde{x} \leftarrow \mathsf{GInput}(K, x)$

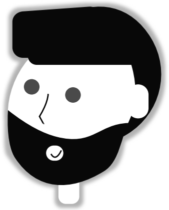
$\xleftarrow{\tilde{x}}$

<u>Ideal Game</u> (with ***Simulator***)

$\xrightarrow{C}$

$\xrightarrow{\Phi(C)}$

$\underline{\mathcal{S}}$



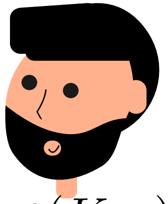
Adaptive Simulation Security (SIM) [BHR'12]

<u>Real Game</u>

$\underline{\mathcal{A}}$

C →

← $\tilde{C}$ $\quad$ $\tilde{C}, K \leftarrow_{\$} \mathsf{GCircuit}(C)$

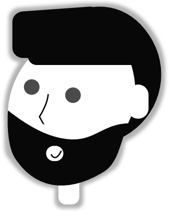


x →

← $\tilde{x}$ $\quad$ $\tilde{x} \leftarrow \mathsf{GInput}(K, x)$

<u>Ideal Game</u> (with ***Simulator***)

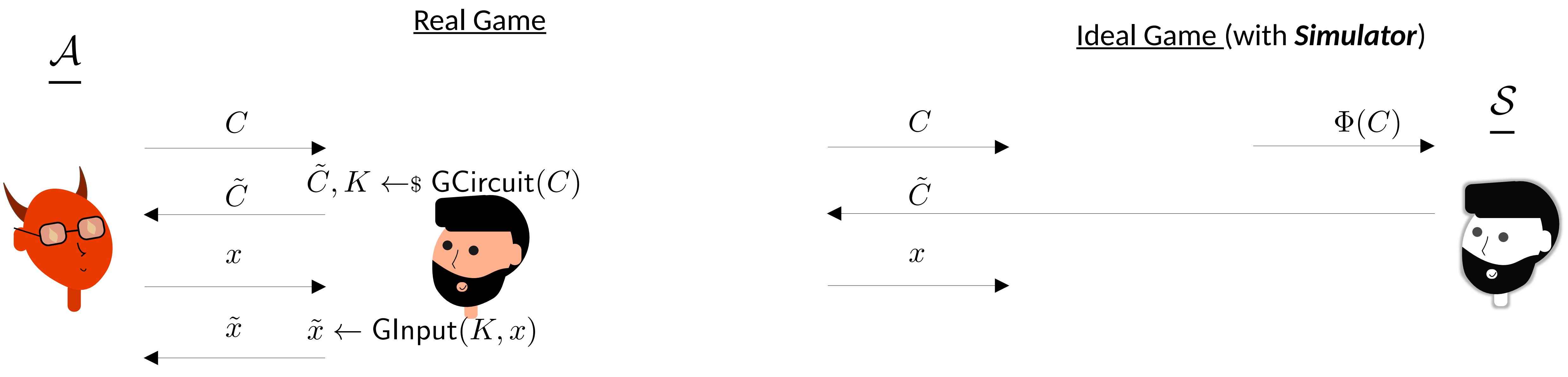
$\underline{\mathcal{S}}$



C → $\qquad$ $\Phi(C)$ →

← $\tilde{C}$

Adaptive Simulation Security (SIM) [BHR’12]



Adaptive Simulation Security (SIM) [BHR'12]

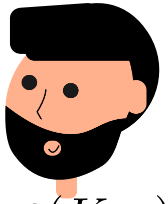
Real Game

$\underline{\mathcal{A}}$

C

$\tilde{C}$

$\tilde{C}, K \leftarrow_{\$} \mathsf{GCircuit}(C)$



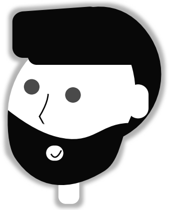
x

$\tilde{x}$

$\tilde{x} \leftarrow \mathsf{GInput}(K, x)$

Ideal Game (with ***Simulator***)

$\underline{\mathcal{S}}$



C

$\Phi(C)$

$\tilde{C}$

x

$C(x)$

Adaptive Simulation Security (SIM) [BHR'12]

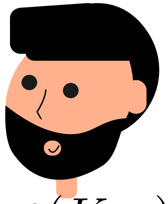
Real Game

$\mathcal{A}$

$C \longrightarrow$

$\tilde{C}, K \leftarrow_{\$} \mathsf{GCircuit}(C)$

$\longleftarrow \tilde{C}$



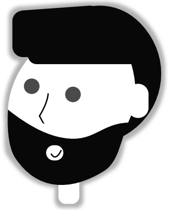
$x \longrightarrow$

$\tilde{x} \leftarrow \mathsf{GInput}(K, x)$

$\longleftarrow \tilde{x}$

Ideal Game (with ***Simulator***)

$\mathcal{S}$



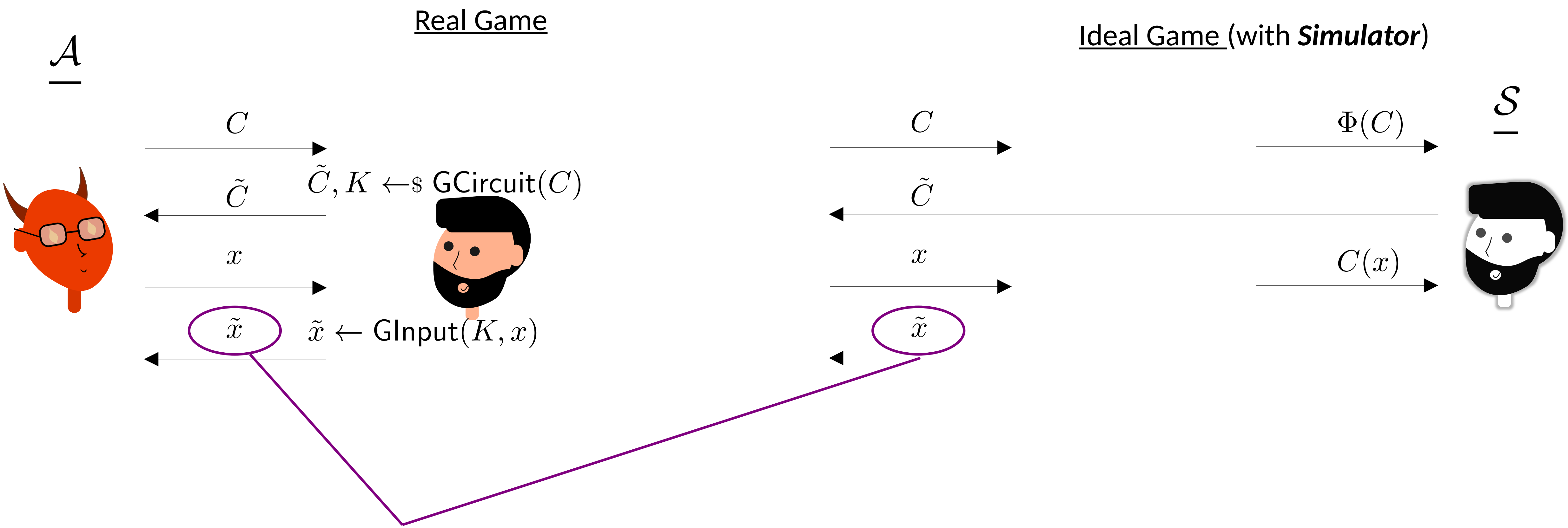
$C \longrightarrow$ $\quad \Phi(C) \longrightarrow$

$\longleftarrow \tilde{C}$

$x \longrightarrow$ $\quad C(x) \longrightarrow$

$\longleftarrow \tilde{x}$

Adaptive Simulation Security (SIM) [BHR'12]



The size of $\tilde{x}$, $|\tilde{x}|$ is known as the ***online complexity***

Adaptive Simulation Security (SIM) [BHR'12]

Real Game

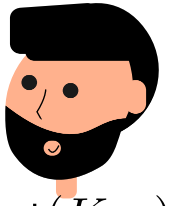
$\underline{\mathcal{A}}$

C $\longrightarrow$

$\tilde{C}, K \leftarrow_{\$} \mathsf{GCircuit}(C)$

$\longleftarrow$ $\tilde{C}$

x $\longrightarrow$



$\tilde{x} \leftarrow \mathsf{GInput}(K, x)$

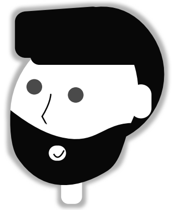
$\longleftarrow$ $\tilde{x}$

Ideal Game (with ***Simulator***)

C $\longrightarrow$

$\Phi(C)$ $\longrightarrow$

$\underline{\mathcal{S}}$



$\longleftarrow$ $\tilde{C}$

x $\longrightarrow$

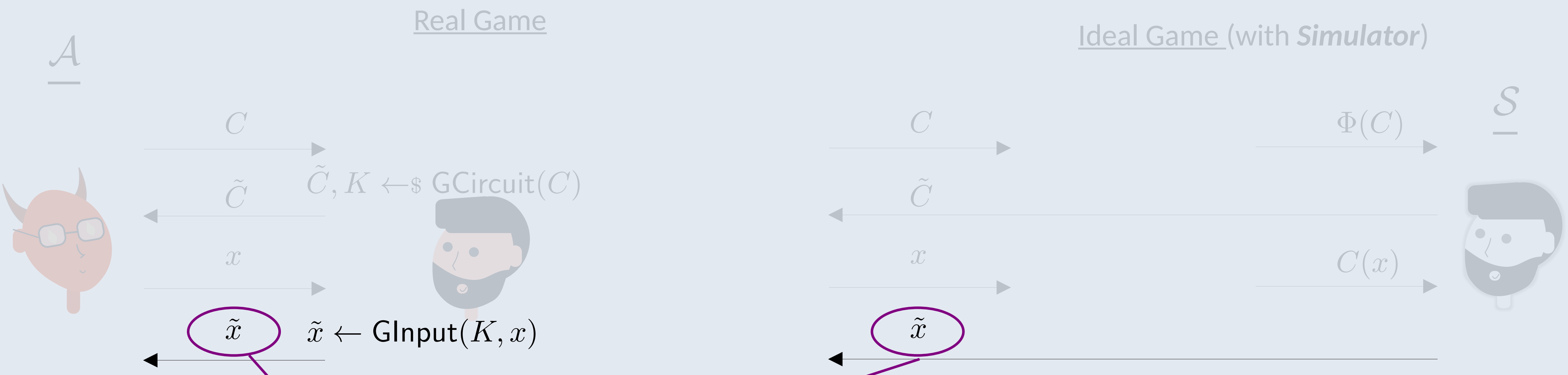
$C(x)$ $\longrightarrow$

$\longleftarrow$ $\tilde{x}$

The size of $\tilde{x}$, $|\tilde{x}|$ is known as the ***online complexity***

Ideally, $|\tilde{x}| = O(|x|)$

Adaptive Simulation Security (SIM) [BHR'12]

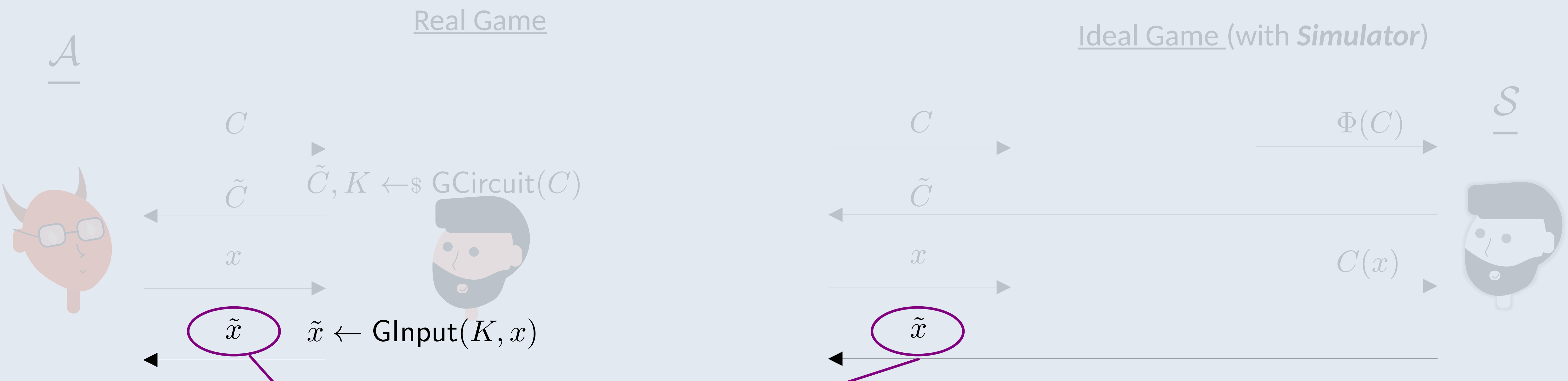


Unfortunately: Applebaum, Ishai, Kushilevitz, Waters [**AIKW'13**] show in general SIM requires $|\tilde{x}| = O(|x| + |y|)$.

The size of $\tilde{x}$, $|\tilde{x}|$ is known as the ***online complexity***

Ideally, $|\tilde{x}| = O(|x|)$

Adaptive Simulation Security (SIM) [BHR'12]



The size of $\tilde{x}$, $|\tilde{x}|$ is known as the ***online complexity***

Ideally, $|\tilde{x}| = O(|x|)$

Unfortunately: Applebaum, Ishai, Kushilevitz, Waters [**AIKW'13**] show in general SIM requires $|\tilde{x}| = O(|x| + |y|)$.

In general, *pseudorandom* circuits such as PRGs, PRFs, etc have this requirement for simulation based MPC [**Hubacek-Wichs'14**]*.

*We give a slightly better lower bound for garbling schemes.

Adaptive Indistinguishability Security (IND) [BHR'12]

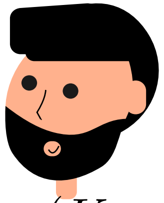
$\underline{\mathcal{A}}$

$\xrightarrow{C_0, C_1}$

$\xleftarrow{\tilde{C}}$ $\tilde{C}, K \leftarrow_{\$} \mathsf{GCircuit}(C_b)$

$\xrightarrow{x_0, x_1}$

$\xleftarrow{\tilde{x}}$ $\tilde{x} \leftarrow \mathsf{GInput}(K, x_b)$



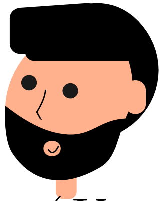
Requirement: $C_0(x_0) = C_1(x_1)$

Adaptive Indistinguishability Security (IND) [BHR'12]

$\underline{\mathcal{A}}$

$\xrightarrow{C_0, C_1}$

$\xleftarrow{\tilde{C}}$ $\quad \tilde{C}, K \leftarrow_{\$} \mathsf{GCircuit}(C_b)$



$\xrightarrow{x_0, x_1}$

$\xleftarrow{\tilde{x}}$ $\quad \tilde{x} \leftarrow \mathsf{GInput}(K, x_b)$

Requirement: $C_0(x_0) = C_1(x_1)$

Seems ***too*** strong for *cryptographic circuits*
i.e. only puncturable primitives.

Adaptive Indistinguishability Security (IND) [BHR'12]

Can we define a notion that
- Does not suffer from SIM lower bounds, and IND weakness?

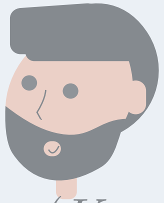
- Garbling cryptographic circuits with long outputs?

$\underline{\mathcal{A}}$

C_0, C_1

$\tilde{C}, K \leftarrow\$ \ \mathsf{GCircuit}(C_b)$

$\tilde{C}$



x_0, x_1

$\tilde{x} \leftarrow \mathsf{GInput}(K, x_b)$

$\tilde{x}$

Adaptive Indistinguishability Security (IND) [BHR'12]

Can we define a notion that
- Does not suffer from SIM lower bounds, and IND weakness?

- Garbling cryptographic circuits with long outputs?

$\mathcal{A}$

In fact, we want a *distributional* notion (*Idea*: Limit adversarial knowledge).

This has been a successful approach for defining things like:

Adaptive Indistinguishability Security (IND) [BHR'12]

Can we define a notion that
- Does not suffer from SIM lower bounds, and IND weakness?

- Garbling cryptographic circuits with long outputs?

$\mathcal{A}$

In fact, we want a *distributional* notion (*Idea*: Limit adversarial knowledge).

This has been a successful approach for defining things like:

1. Deterministic Public Key Encryption (Bellare, Boldyreva, O'Neill [**BBO'07**]):
   (Cannot allow adversary to pick the whole message)

Adaptive Indistinguishability Security (IND) [BHR'12]

Can we define a notion that
- Does not suffer from SIM lower bounds, and IND weakness?

- Garbling cryptographic circuits with long outputs?

In fact, we want a *distributional* notion (*Idea*: Limit adversarial knowledge).

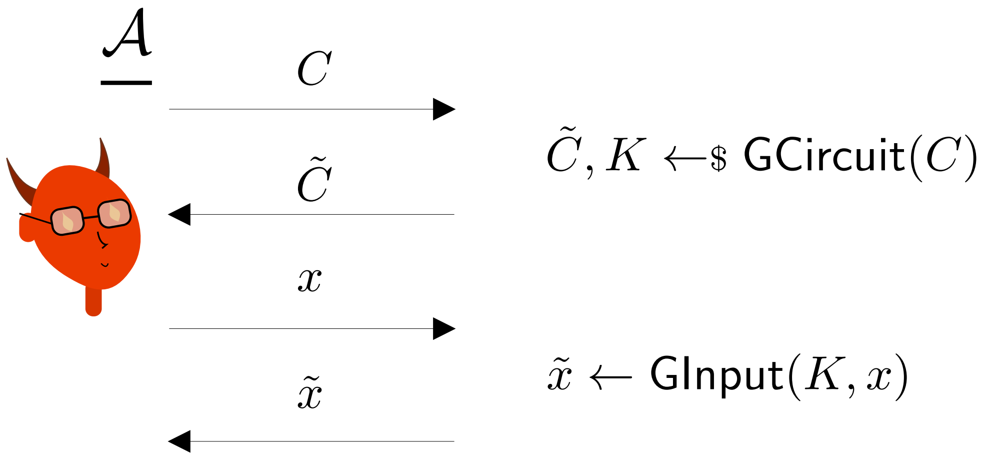
This has been a successful approach for defining things like:

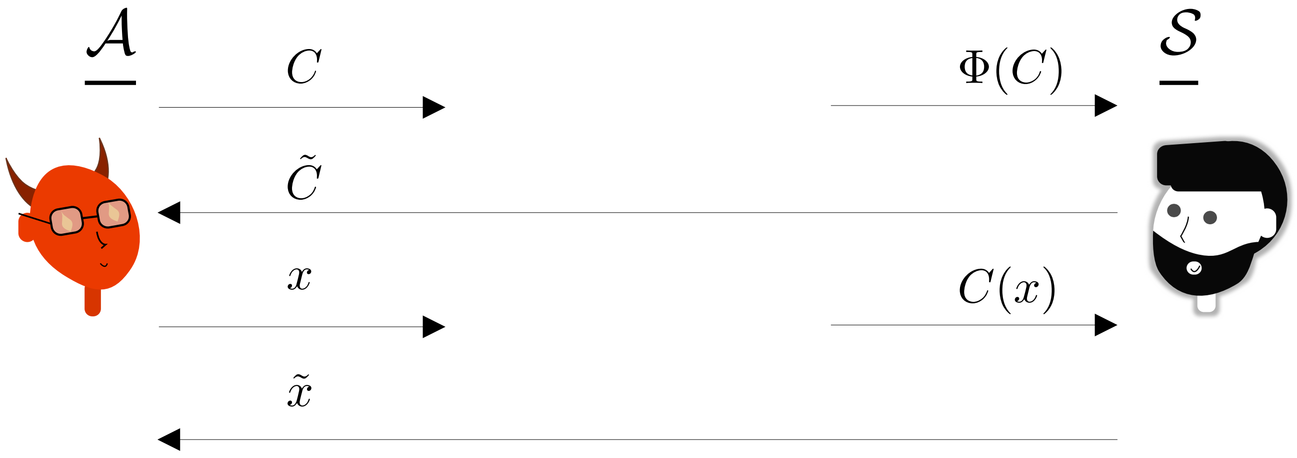
1. Deterministic Public Key Encryption (Bellare, Boldyreva, O'Neill [**BBO'07**]):
   (Cannot allow adversary to pick the whole message)

2. Distributional Zero Knowledge [**Goldreich'93, DNRS'99, JKKR'17, Khurana'21**]
   - Random statement instead of *any* statement.

Adaptive Distributional Simulation Security (DSIM)

Recall SIM notion:





Adaptive Distributional Simulation Security (DSIM)

Our DSIM notion:

$\underline{\mathcal{A}}$

$\xrightarrow{C}$

$\xleftarrow{\tilde{C}}$

$\tilde{C}, K \leftarrow_{\$} \mathsf{GCircuit}(C)$

$\xrightarrow{x}$

$\xleftarrow{\tilde{x}}$

$\tilde{x} \leftarrow \mathsf{GInput}(K, x)$

$\underline{\mathcal{A}}$

$\xrightarrow{C}$

$\xleftarrow{\tilde{C}}$

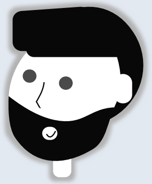
$\xrightarrow{x}$

$\xleftarrow{\tilde{x}}$

$\xrightarrow{\Phi(C)}$

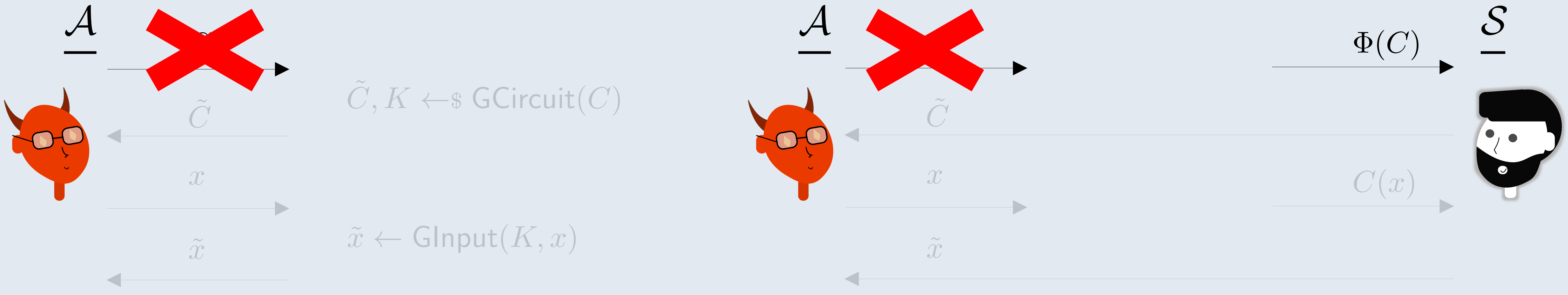
$\xrightarrow{C(x)}$

$\underline{\mathcal{S}}$



Adaptive Distributional Simulation Security (DSIM)

Our DSIM notion:



Adversary no longer chooses the circuit C

Adaptive Distributional Simulation Security (DSIM)

Our DSIM notion:

$\underline{\mathcal{A}}$

$\tilde{C}$

x

$\tilde{x}$

$\tilde{C}, K \leftarrow_{\$} \mathsf{GCircuit}(C)$

$\tilde{x} \leftarrow \mathsf{GInput}(K, x)$

$\underline{\mathcal{A}}$

$\tilde{C}$

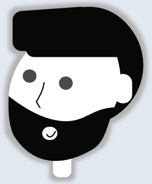
x

$\tilde{x}$

$\Phi(C)$

$C(x)$

$\underline{\mathcal{S}}$



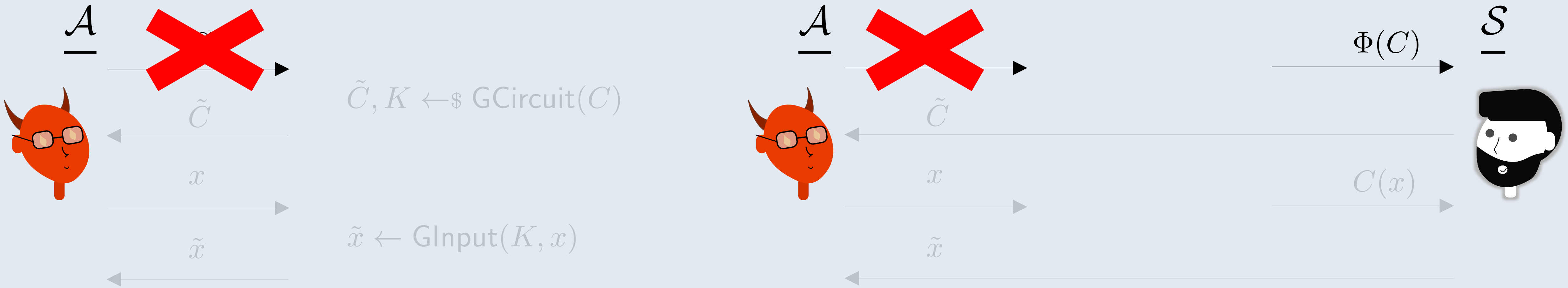
Adversary no longer chooses the circuit C

Instead, C comes from a sampler:

$(C,\ \Phi(C),\ \mathcal{O}) \leftarrow_{\$} \mathsf{Sam}(1^{\lambda})$

Adaptive Distributional Simulation Security (DSIM)

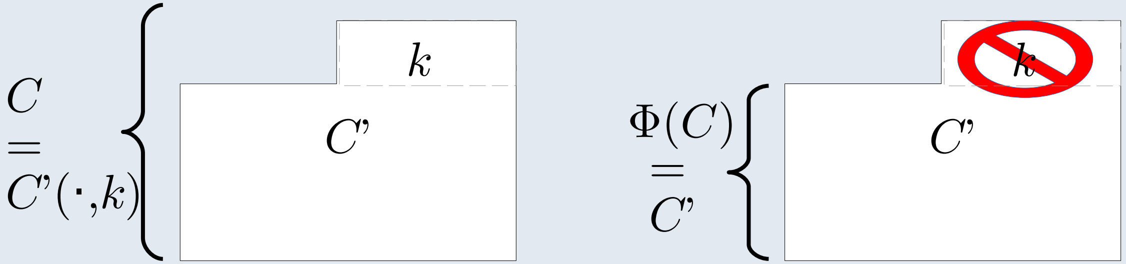
Our DSIM notion:



Adversary no longer chooses the circuit C

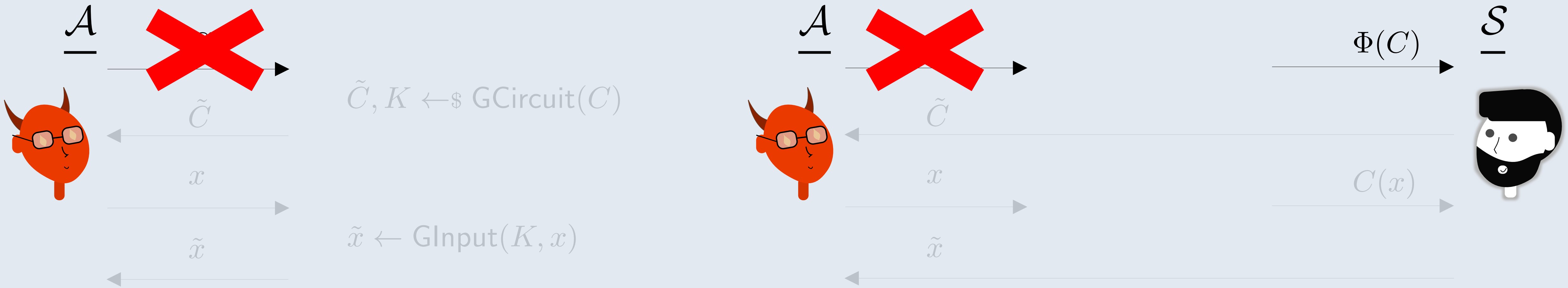
Instead, C comes from a sampler:

$(C,\ \Phi(C),\ \mathcal{O}) \leftarrow\$ \mathsf{Sam}(1^\lambda)$



Adaptive Distributional Simulation Security (DSIM)

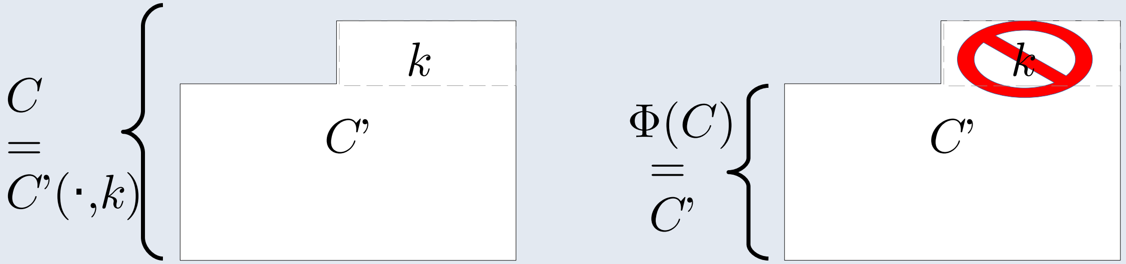
Our DSIM notion:



Adversary no longer chooses the circuit C

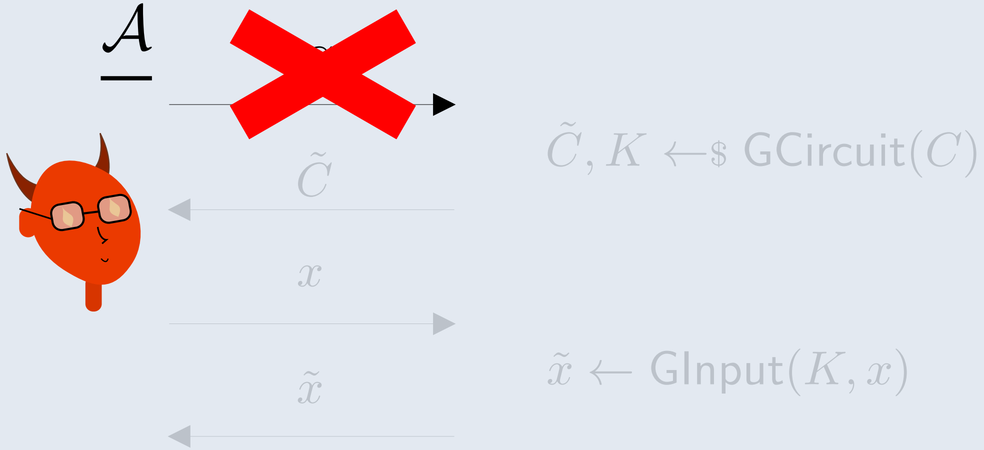
Instead, C comes from a sampler:
$(C,\ \Phi(C),\ \mathcal{O}) \leftarrow\$ \mathsf{Sam}(1^\lambda)$

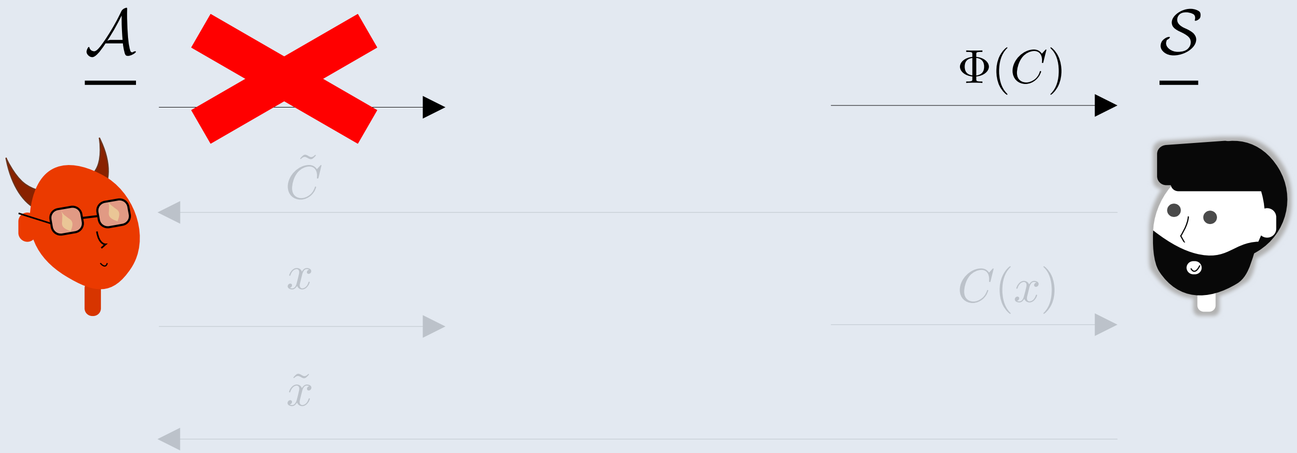
Think of $C = C'(\cdot, k)$ with public part $\Phi(C) = C'$ and secret k.



Adaptive Distributional Simulation Security (DSIM)

Our DSIM notion:





Adversary no longer chooses the circuit C

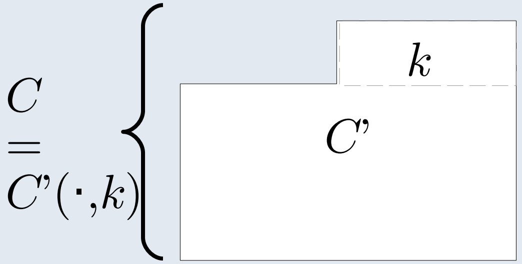
Instead, C comes from a sampler:

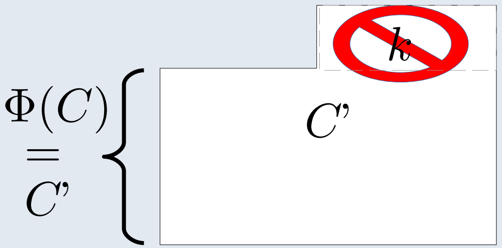
$(C,\ \Phi(C),\ \mathcal{O}) \leftarrow_\$ \mathsf{Sam}(1^\lambda)$

Think of $C = C'(\cdot, k)$ with public part $\Phi(C) = C'$ and secret k.

C is "cryptographic":

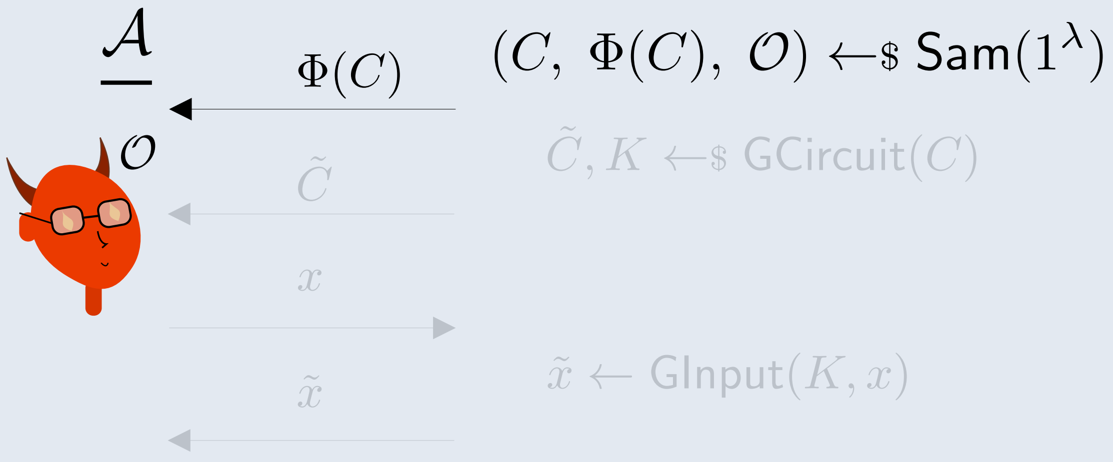
If k unknown, $C(x,k) \approx C(0^{|x|},k)$

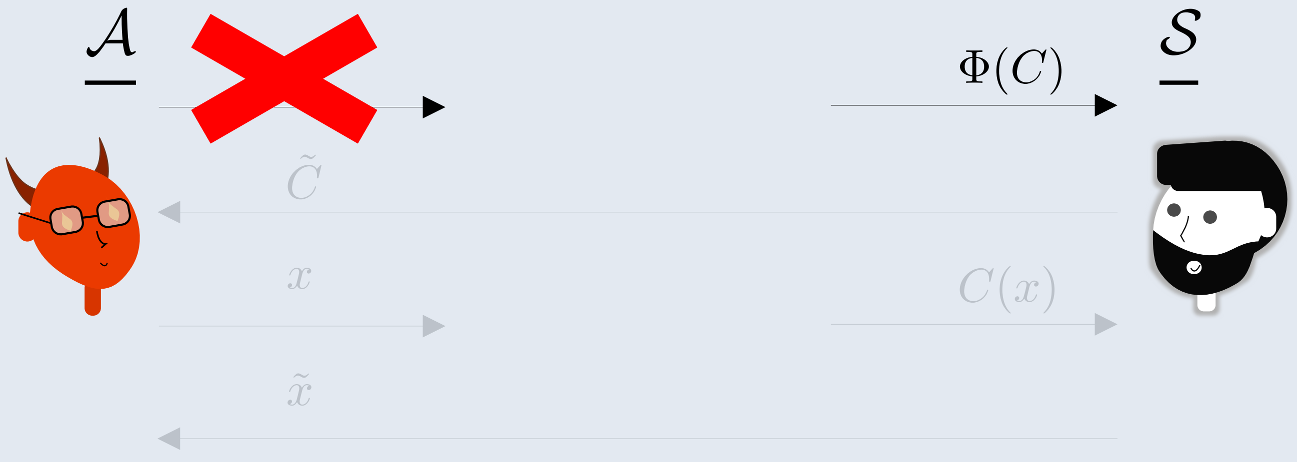




Adaptive Distributional Simulation Security (DSIM)

Our DSIM notion:





Instead, C comes from a sampler:

$(C,\ \Phi(C),\ \mathcal{O}) \leftarrow\!\$\ \mathsf{Sam}(1^\lambda)$

Adaptive Distributional Simulation Security (DSIM)

Our DSIM notion:

$\underline{\mathcal{A}}$

$\Phi(C)$

$(C,\ \Phi(C),\ \mathcal{O}) \leftarrow_{\$} \mathsf{Sam}(1^\lambda)$

$\mathcal{O}$

$\tilde{C}$

$\tilde{C}, K \leftarrow_{\$} \mathsf{GCircuit}(C)$

x

$\tilde{x}$

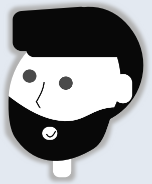
$\tilde{x} \leftarrow \mathsf{GInput}(K, x)$

$(C,\ \Phi(C),\ \mathcal{O}) \leftarrow_{\$} \mathsf{Sam}(1^\lambda)$

$\underline{\mathcal{A}}$ $\qquad$ $\underline{\mathcal{S}}$

$\Phi(C)$ $\qquad$ $\Phi(C)$

$\mathcal{O}$ $\qquad$ $\mathcal{O}$



$\tilde{C}$

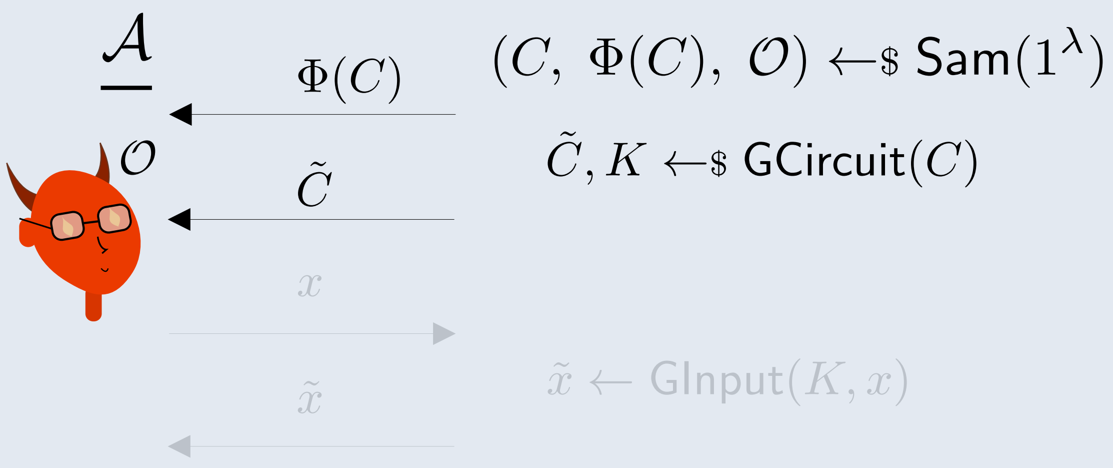
x $\qquad$ $C(x)$

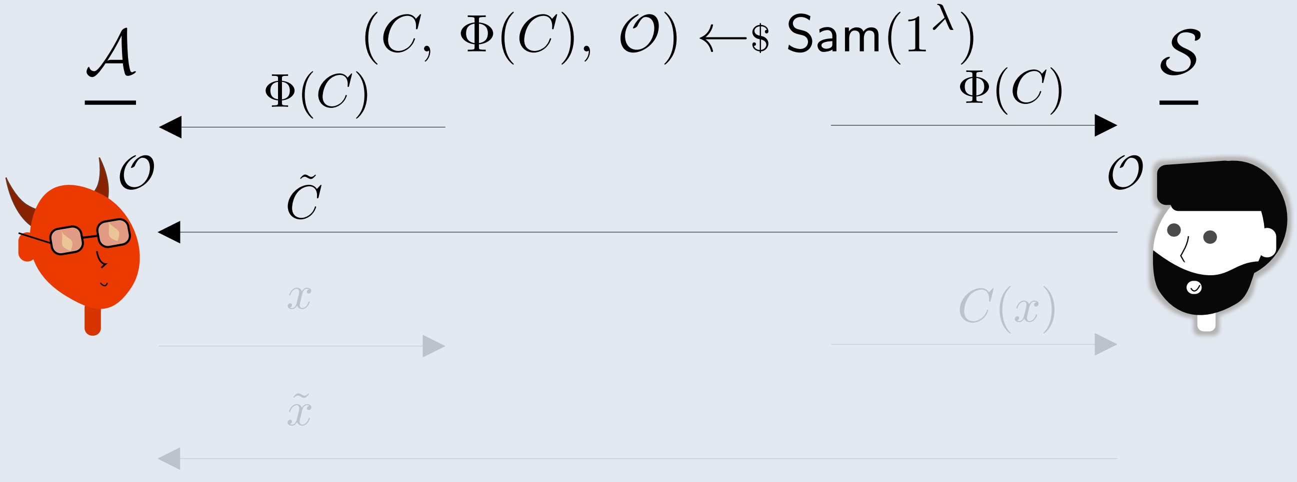
$\tilde{x}$

Instead, C comes from a sampler:
$(C,\ \Phi(C),\ \mathcal{O}) \leftarrow_{\$} \mathsf{Sam}(1^\lambda)$

Adaptive Distributional Simulation Security (DSIM)

Our DSIM notion:



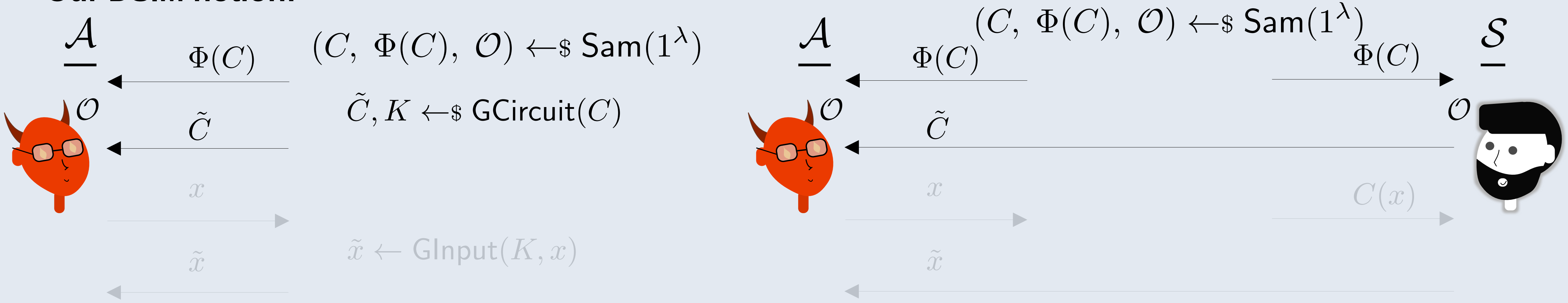


Instead, C comes from a sampler:

$(C,\ \Phi(C),\ \mathcal{O}) \leftarrow_{\$} \mathsf{Sam}(1^\lambda)$

Adaptive Distributional Simulation Security (DSIM)

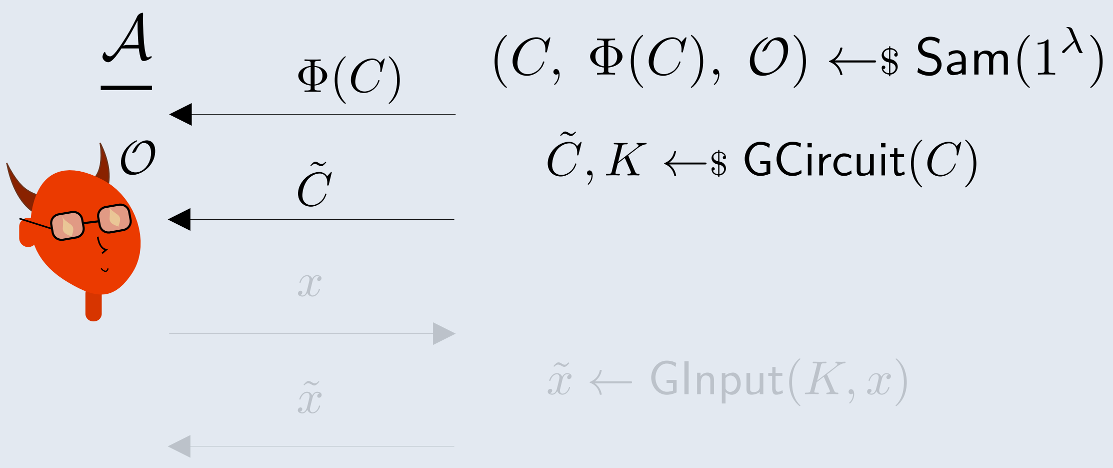
Our DSIM notion:

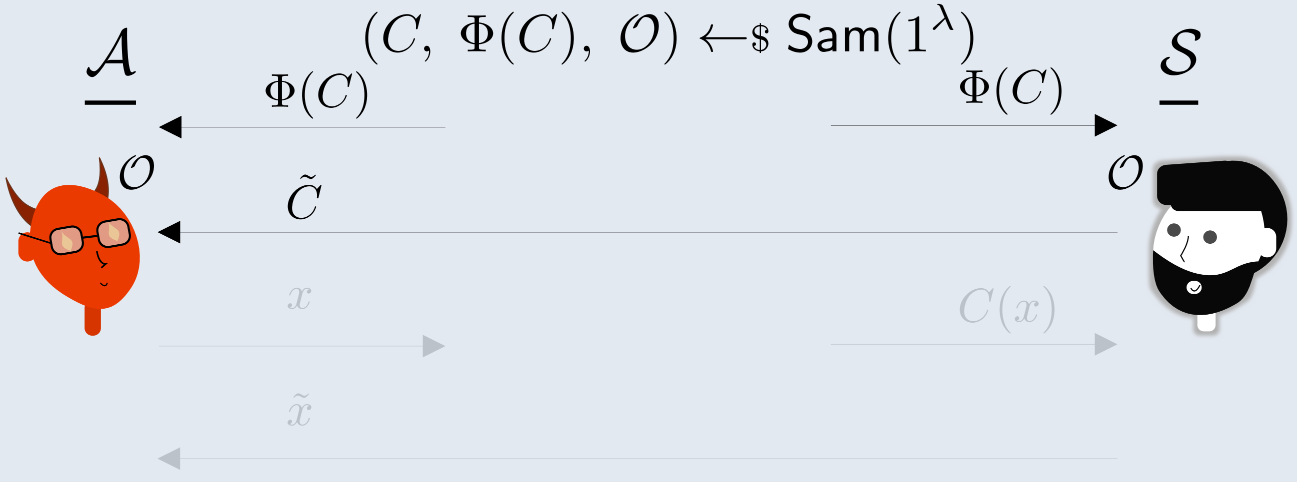


Next we make the simulator's *consistency* requirement flexible: (via leakage function $\Lambda(C,x)$)

Adaptive Distributional Simulation Security (DSIM)

Our DSIM notion:





Next we make the simulator's *consistency* requirement flexible: (via leakage function $\Lambda(C,x)$)

In the extreme case (SIM) the simulator is given $\Lambda(C,x) = C(x)$ since it needs to produce the exact same output.

Adaptive Distributional Simulation Security (DSIM)

Our DSIM notion:

$\underline{\mathcal{A}}$

$\mathcal{O}$

$\Phi(C)$

$(C,\ \Phi(C),\ \mathcal{O}) \leftarrow\!\$\ \mathsf{Sam}(1^\lambda)$

$\tilde{C}$

$\tilde{C}, K \leftarrow\!\$\ \mathsf{GCircuit}(C)$

x

$\tilde{x}$

$\tilde{x} \leftarrow \mathsf{GInput}(K, x)$

$\underline{\mathcal{A}}$

$\mathcal{O}$

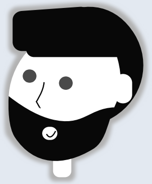
$(C,\ \Phi(C),\ \mathcal{O}) \leftarrow\!\$\ \mathsf{Sam}(1^\lambda)$

$\Phi(C)$

$\Phi(C)$

$\underline{\mathcal{S}}$

$\mathcal{O}$



$\tilde{C}$

x

$C(x)$

$\tilde{x}$

Next we make the simulator's *consistency* requirement flexible: (via leakage function $\Lambda(C,x)$)

In the extreme case (SIM) the simulator is given $\Lambda(C,x) = C(x)$ since it needs to produce the exact same output.

In the other extreme we can make $\Lambda(C,x) = \emptyset$ for *pseudorandom outputs*

Adaptive Distributional Simulation Security (DSIM)

Our DSIM notion:

$\underline{\mathcal{A}}$

$\mathcal{O}$

$(C,\ \Phi(C),\ \mathcal{O}) \leftarrow_{\$} \mathsf{Sam}(1^\lambda)$

$\xleftarrow{\Phi(C)}$

$\tilde{C}, K \leftarrow_{\$} \mathsf{GCircuit}(C)$

$\xleftarrow{\tilde{C}}$

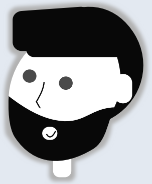
$\xrightarrow{x}$

$\tilde{x} \leftarrow \mathsf{GInput}(K, x)$

$\xleftarrow{\tilde{x}}$

$\underline{\mathcal{A}}$ $\qquad (C,\ \Phi(C),\ \mathcal{O}) \leftarrow_{\$} \mathsf{Sam}(1^\lambda)$ $\qquad \underline{\mathcal{S}}$

$\mathcal{O}$



$\mathcal{O}$

$\xleftarrow{\Phi(C)}$ $\qquad \xrightarrow{\Phi(C)}$

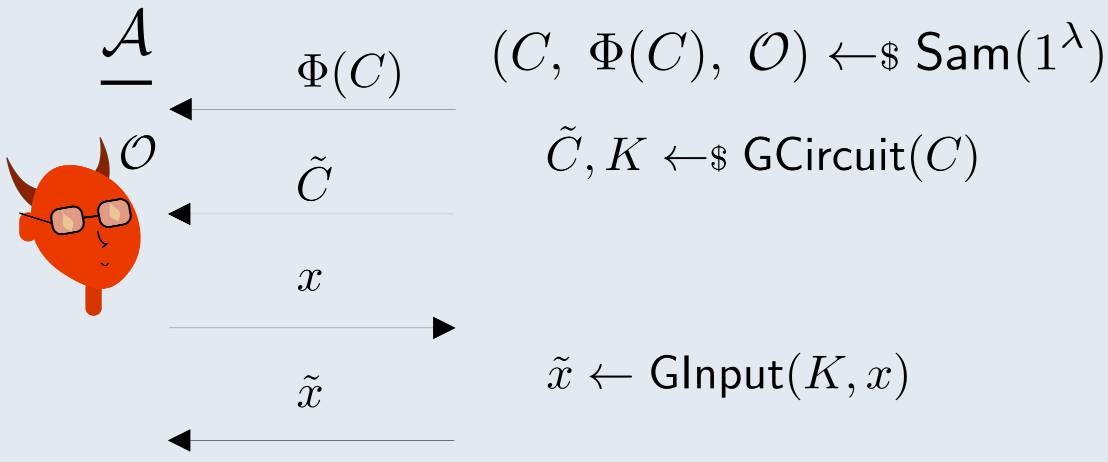
$\xleftarrow{\tilde{C}}$

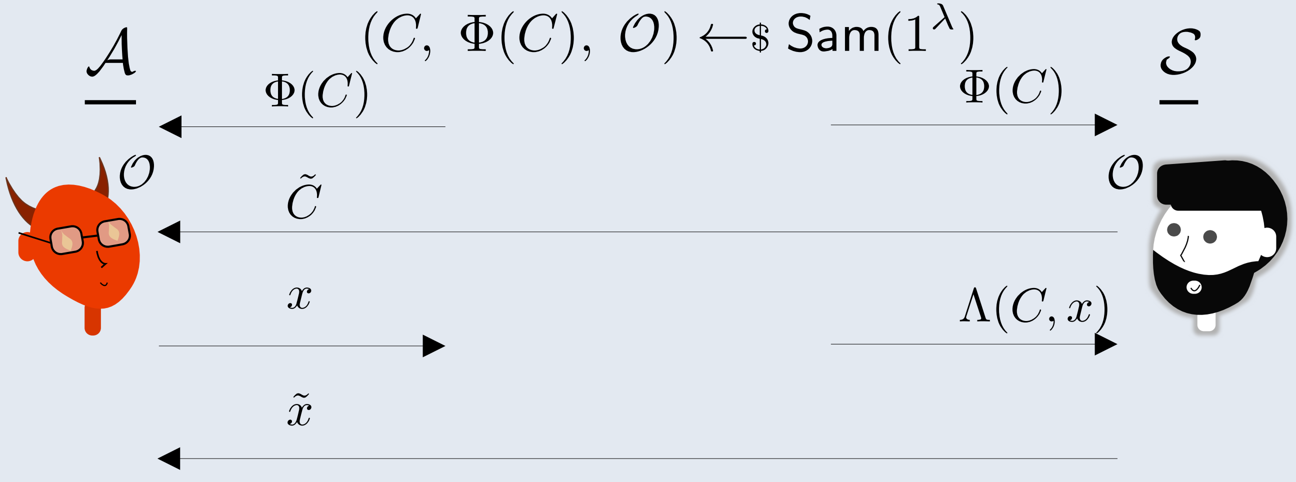
$\xrightarrow{x}$ $\qquad \xrightarrow{\Lambda(C, x)}$

$\xleftarrow{\tilde{x}}$

Adaptive Distributional Simulation Security (DSIM)

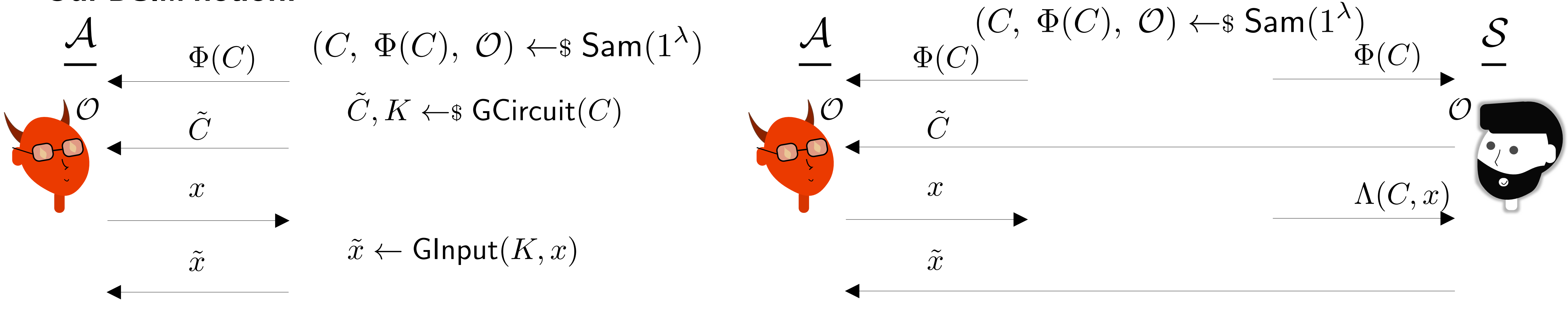
Our DSIM notion:





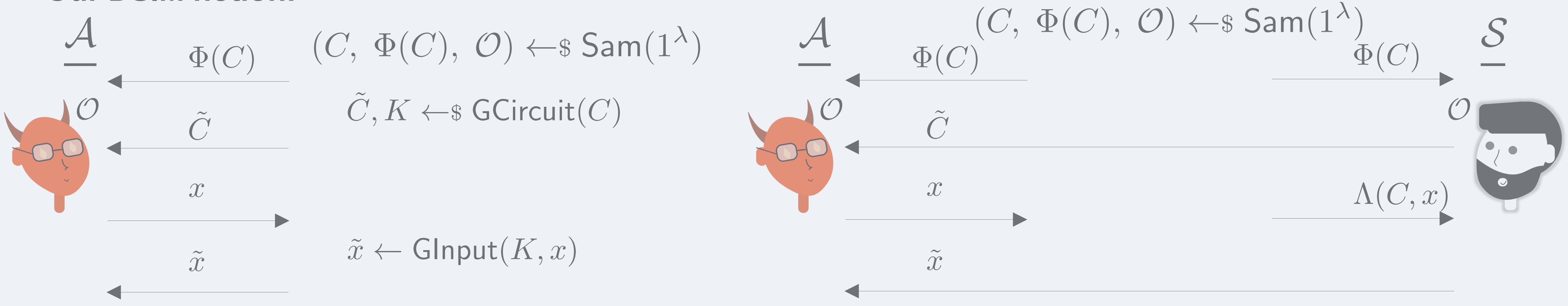
Adaptive Distributional Simulation Security (DSIM)

Our DSIM notion:



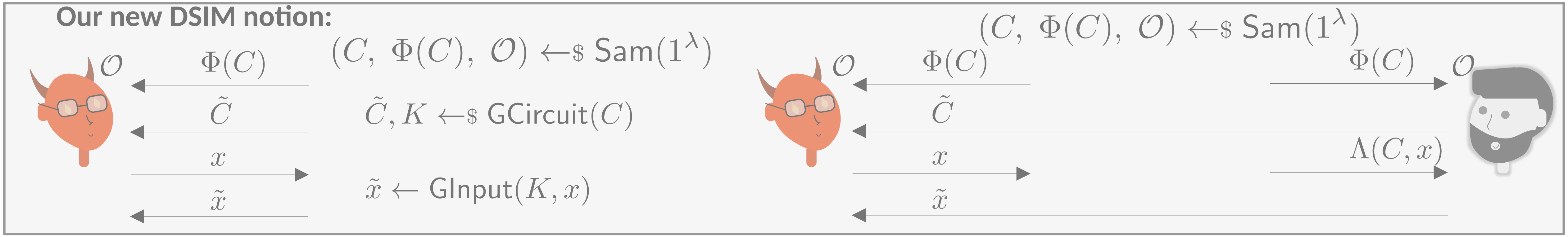
Adaptive Distributional Simulation Security (DSIM)

Our DSIM notion:

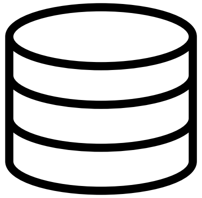


Let's look at an example application of DSIM, where we want to garble a length expanding cryptographic (encryption) circuit.

Example: Distributed Symmetric Encryption (DSE) from DSIM



Example: Distributed Symmetric Encryption (DSE) from DSIM



We want to store some encrypted data $enc(k, m)$ in some external database.

Our new DSIM notion:

$\mathcal{O}$

$\xleftarrow{\Phi(C)}$ $(C,\ \Phi(C),\ \mathcal{O}) \leftarrow\$\ \mathsf{Sam}(1^\lambda)$

$\xleftarrow{\tilde{C}}$ $\tilde{C}, K \leftarrow\$\ \mathsf{GCircuit}(C)$

$\xrightarrow{x}$

$\xleftarrow{\tilde{x}}$ $\tilde{x} \leftarrow \mathsf{GInput}(K, x)$

$(C,\ \Phi(C),\ \mathcal{O}) \leftarrow\$\ \mathsf{Sam}(1^\lambda)$

$\mathcal{O}$

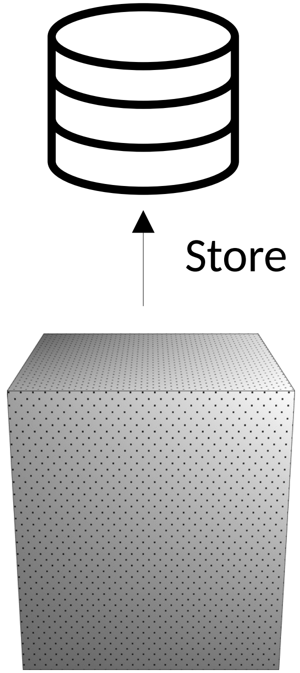
$\xleftarrow{\Phi(C)}$ $\xrightarrow{\Phi(C)}$ $\mathcal{O}$

$\xleftarrow{\tilde{C}}$

$\xrightarrow{x}$ $\xrightarrow{\Lambda(C, x)}$

$\xleftarrow{\tilde{x}}$

Example: Distributed Symmetric Encryption (DSE) from DSIM



We want to store some encrypted data $enc(k, m)$ in some external database.

Store $enc(k, m)$.

Server keeps k.

Our new DSIM notion:

$(C,\ \Phi(C),\ \mathcal{O}) \leftarrow_\$ \mathsf{Sam}(1^\lambda)$

$\mathcal{O}$ $\xleftarrow{\Phi(C)}$

$\xleftarrow{\tilde{C}}$ $\qquad \tilde{C}, K \leftarrow_\$ \mathsf{GCircuit}(C)$

$\xrightarrow{x}$

$\xleftarrow{\tilde{x}}$ $\qquad \tilde{x} \leftarrow \mathsf{GInput}(K, x)$

$(C,\ \Phi(C),\ \mathcal{O}) \leftarrow_\$ \mathsf{Sam}(1^\lambda)$

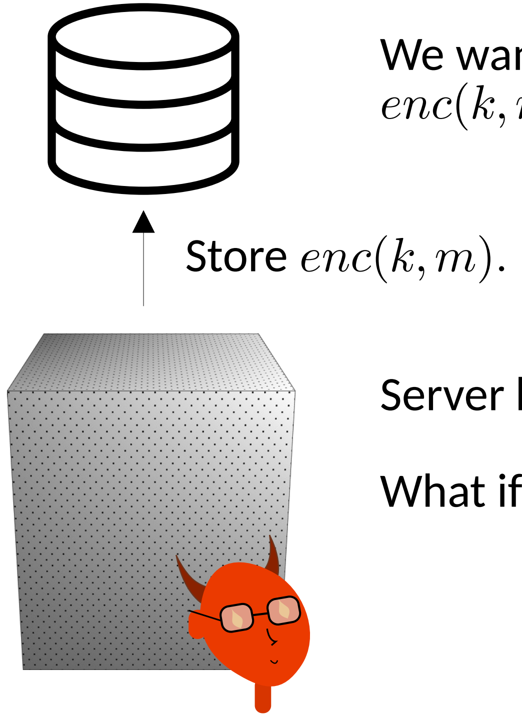
$\mathcal{O}$ $\xleftarrow{\Phi(C)}$ $\qquad \xrightarrow{\Phi(C)}$ $\mathcal{O}$

$\xleftarrow{\tilde{C}}$

$\xrightarrow{x}$ $\qquad \xrightarrow{\Lambda(C, x)}$

$\xleftarrow{\tilde{x}}$

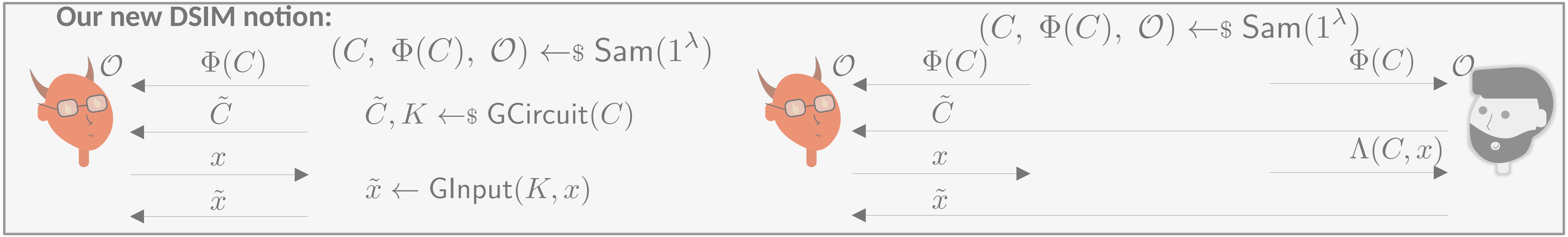
Example: Distributed Symmetric Encryption (DSE) from DSIM



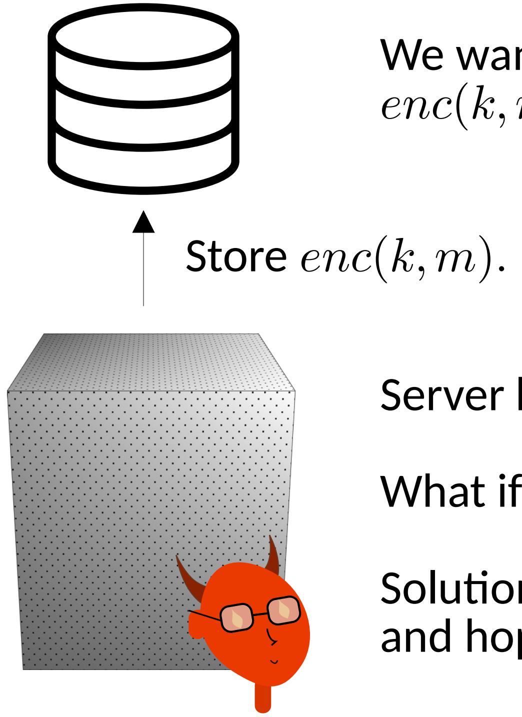
We want to store some encrypted data $enc(k, m)$ in some external database.

Server keeps k.

What if it gets corrupted?



Example: Distributed Symmetric Encryption (DSE) from DSIM



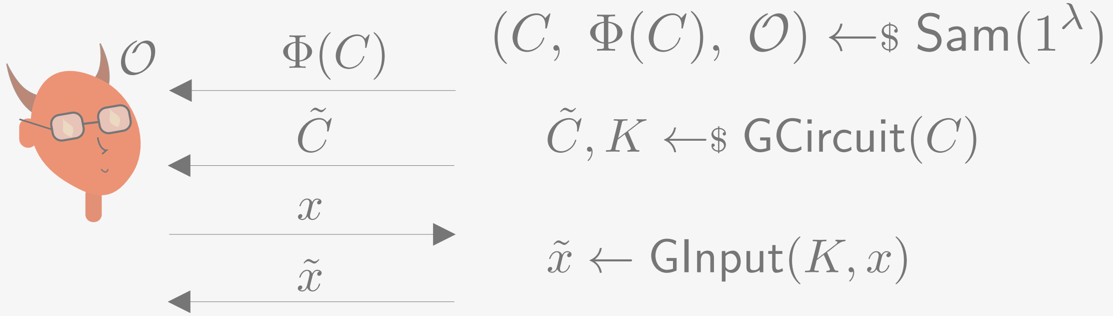
We want to store some encrypted data $enc(k, m)$ in some external database.

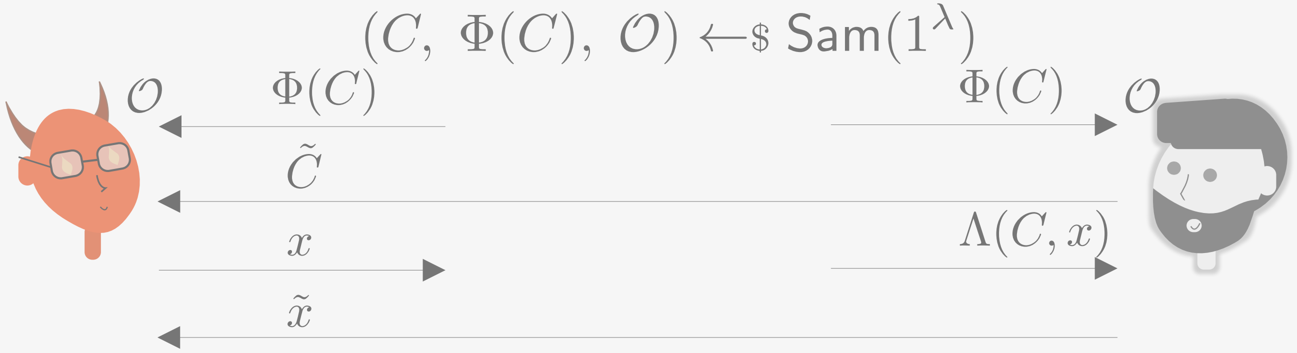
Server keeps k.

What if it gets corrupted?

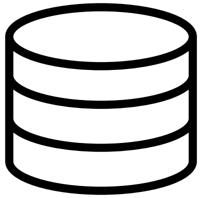
Solution: ***Secret share key*** k to many servers and hope that not all of them get corrupted.

Our new DSIM notion:





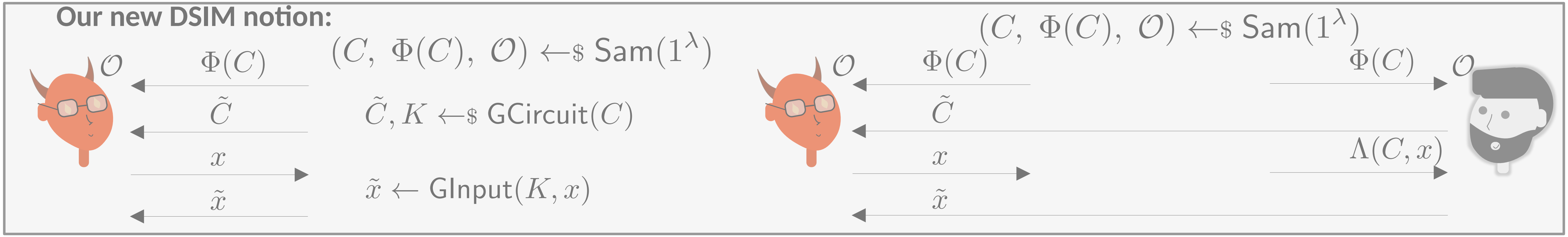
2-Party DSE Protocol



Key share k_1

($k = k_1 \oplus k_2$ secret shared in a setup phase)

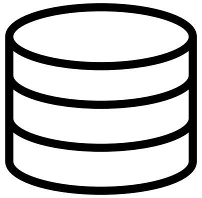
Key share k_2

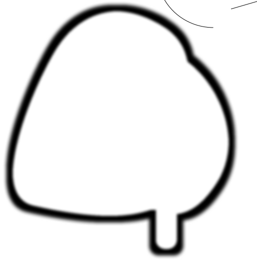


2-Party DSE Protocol

*Randomness made implicit to avoid clutter

Store $enc(k_1 \oplus k_2, m)$.





Key share k_1

($k = k_1 \oplus k_2$ secret shared in a setup phase)

Key share k_2

Our new DSIM notion:

$\mathcal{O}$

$\Phi(C)$

$(C,\ \Phi(C),\ \mathcal{O}) \leftarrow_\$ \mathsf{Sam}(1^\lambda)$

$\tilde{C}$

$\tilde{C}, K \leftarrow_\$ \mathsf{GCircuit}(C)$

x

$\tilde{x} \leftarrow \mathsf{GInput}(K, x)$

$\tilde{x}$

$(C,\ \Phi(C),\ \mathcal{O}) \leftarrow_\$ \mathsf{Sam}(1^\lambda)$

$\mathcal{O}$

$\Phi(C)$

$\Phi(C)$

$\mathcal{O}$

$\tilde{C}$

x

$\Lambda(C, x)$

$\tilde{x}$

Goal: Store $enc(k_1 \oplus k_2, m)$.

2-Party DSE Protocol

Key share k_2

Key share k_1

Our new DSIM notion:

$\mathcal{O}$

$\Phi(C)$

$\tilde{C}$

x

$\tilde{x}$

$(C,\ \Phi(C),\ \mathcal{O}) \leftarrow_{\$} \mathsf{Sam}(1^\lambda)$

$\tilde{C}, K \leftarrow_{\$} \mathsf{GCircuit}(C)$

$\tilde{x} \leftarrow \mathsf{GInput}(K, x)$

$(C,\ \Phi(C),\ \mathcal{O}) \leftarrow_{\$} \mathsf{Sam}(1^\lambda)$

$\mathcal{O}$

$\Phi(C)$

$\tilde{C}$

x

$\tilde{x}$

$\Phi(C)$

$\Lambda(C, x)$

$\mathcal{O}$

Goal: Store $enc(k_1 \oplus k_2, m)$.

2-Party DSE Protocol

Key share k_2

Key share k_1

$$\underbrace{enc(k_1 \oplus \cdot, \cdot)}_{C}$$

Our new DSIM notion:

$\mathcal{O}$

$(C,\ \Phi(C),\ \mathcal{O}) \leftarrow\$ \ \mathsf{Sam}(1^\lambda)$

$\xleftarrow{\Phi(C)}$

$\tilde{C}, K \leftarrow\$ \ \mathsf{GCircuit}(C)$

$\xleftarrow{\tilde{C}}$

$\xrightarrow{x}$

$\tilde{x} \leftarrow \mathsf{GInput}(K, x)$

$\xleftarrow{\tilde{x}}$

$\mathcal{O}$

$(C,\ \Phi(C),\ \mathcal{O}) \leftarrow\$ \ \mathsf{Sam}(1^\lambda)$

$\xleftarrow{\Phi(C)}$ $\xrightarrow{\Phi(C)}$

$\xleftarrow{\tilde{C}}$

$\xrightarrow{x}$ $\xrightarrow{\Lambda(C, x)}$

$\xleftarrow{\tilde{x}}$

$\mathcal{O}$

Goal: Store $enc(k_1 \oplus k_2, m)$.

2-Party DSE Protocol

Key share k_2

Key share k_1

$$\tilde{C}, K \leftarrow\!\$\ \mathsf{GCircuit}(\underbrace{enc(k_1 \oplus \cdot, \cdot)}_{C})$$

Our new DSIM notion:

$\mathcal{O}$

$(C,\ \Phi(C),\ \mathcal{O}) \leftarrow\!\$\ \mathsf{Sam}(1^\lambda)$

$\xleftarrow{\Phi(C)}$

$\tilde{C}, K \leftarrow\!\$\ \mathsf{GCircuit}(C)$

$\xleftarrow{\tilde{C}}$

$\xrightarrow{x}$

$\tilde{x} \leftarrow \mathsf{GInput}(K, x)$

$\xleftarrow{\tilde{x}}$

$\mathcal{O}$

$(C,\ \Phi(C),\ \mathcal{O}) \leftarrow\!\$\ \mathsf{Sam}(1^\lambda)$

$\xleftarrow{\Phi(C)}$ $\xrightarrow{\Phi(C)}$

$\xleftarrow{\tilde{C}}$

$\xrightarrow{x}$ $\xrightarrow{\Lambda(C, x)}$

$\xleftarrow{\tilde{x}}$

$\mathcal{O}$

Goal: Store $enc(k_1 \oplus k_2, m)$.

2-Party DSE Protocol

Key share k_2

Key share k_1

$\tilde{C}$

$\tilde{C}, K \leftarrow\$ \ \mathsf{GCircuit}(\underbrace{enc(k_1 \oplus \cdot, \cdot)}_{C})$

Our new DSIM notion:

$\mathcal{O}$

$(C,\ \Phi(C),\ \mathcal{O}) \leftarrow\$ \ \mathsf{Sam}(1^\lambda)$

$\Phi(C)$

$\tilde{C}$

$\tilde{C}, K \leftarrow\$ \ \mathsf{GCircuit}(C)$

x

$\tilde{x}$

$\tilde{x} \leftarrow \mathsf{GInput}(K, x)$

$\mathcal{O}$

$(C,\ \Phi(C),\ \mathcal{O}) \leftarrow\$ \ \mathsf{Sam}(1^\lambda)$

$\Phi(C)$

$\Phi(C)$

$\mathcal{O}$

$\tilde{C}$

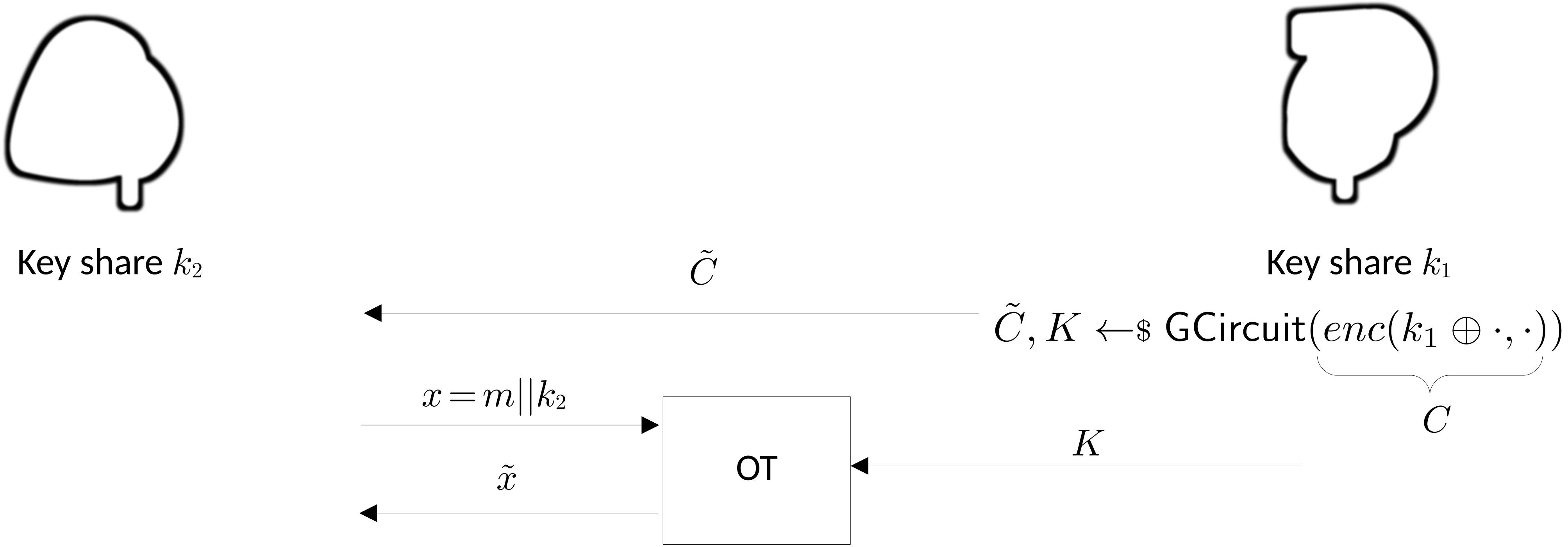
x

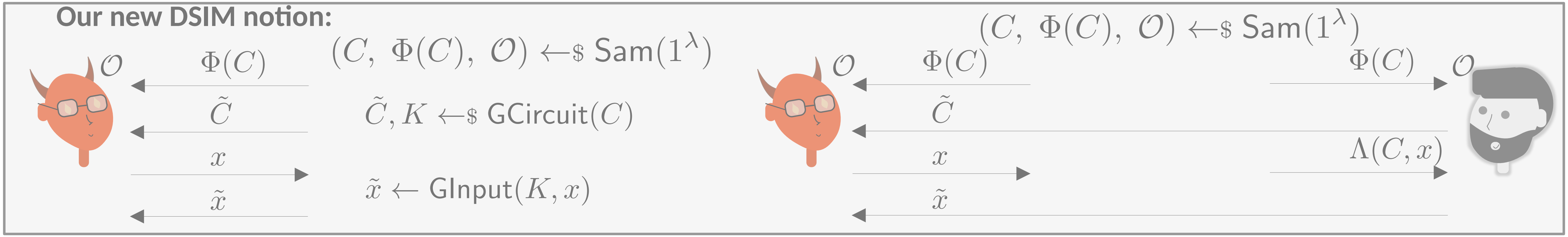
$\Lambda(C, x)$

$\tilde{x}$

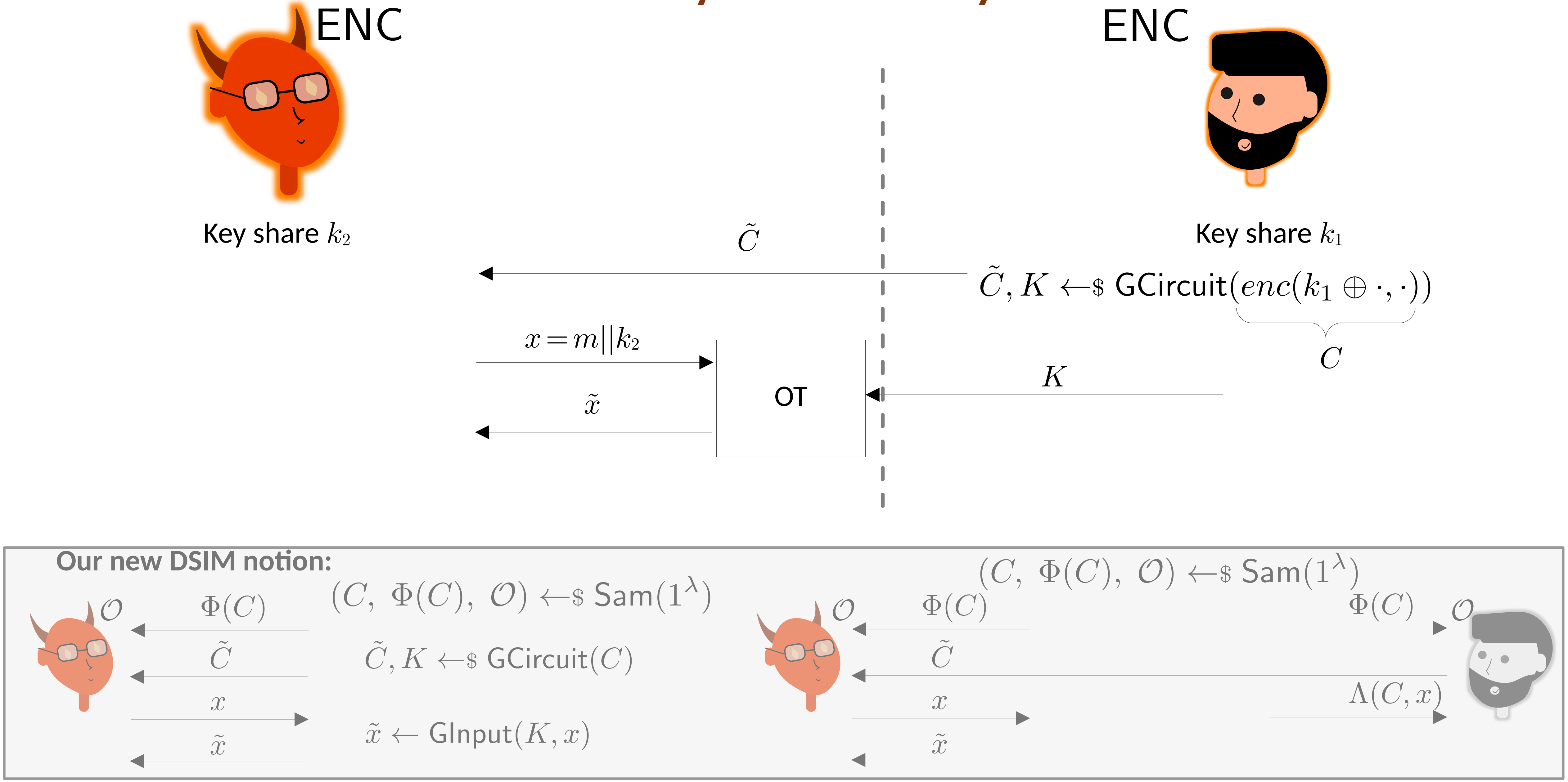
Goal: Store $enc(k_1 \oplus k_2, m)$.

2-Party DSE Protocol

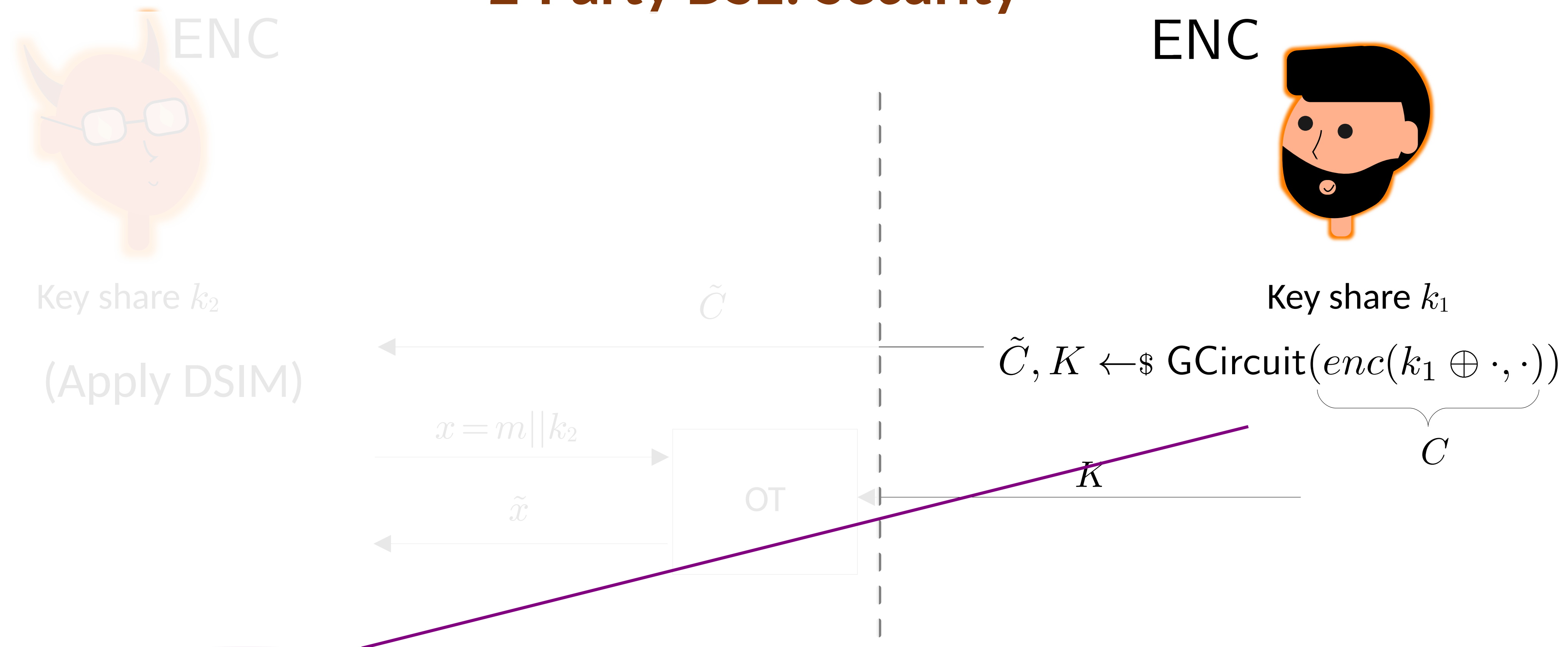




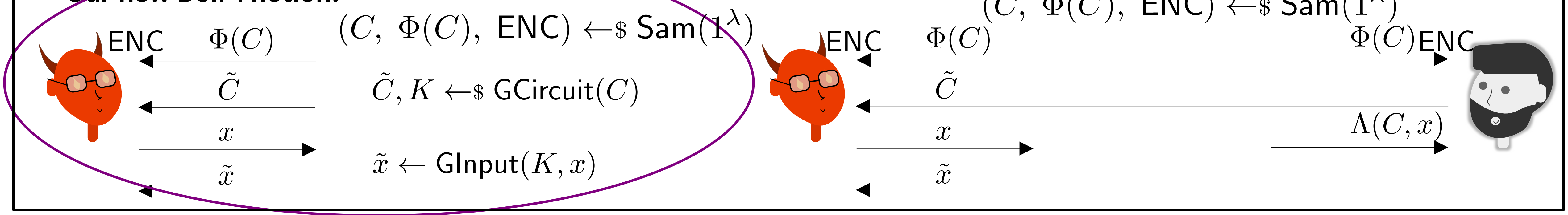
2-Party DSE: Security



2-Party DSE: Security



Our new DSIM notion:



2-Party DSE: Security

ENC

Key share k_2

(Apply DSIM)

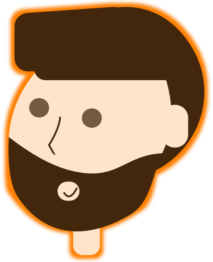
$\tilde{C}$

$x = m || k_2$

$\tilde{x}$

OT

ENC



Key share k_1

$\tilde{C}, K \leftarrow_\$ \mathcal{S}^{\mathsf{ENC}}(\Phi(C))$

K

Our new DSIM notion:

ENC

$\Phi(C)$

$\tilde{C}$

x

$\tilde{x}$

$(C,\ \Phi(C),\ \mathsf{ENC}) \leftarrow_\$ \mathsf{Sam}(1^\lambda)$

$\tilde{C}, K \leftarrow_\$ \mathsf{GCircuit}(C)$

$\tilde{x} \leftarrow \mathsf{GInput}(K, x)$

$(C,\ \Phi(C),\ \mathsf{ENC}) \leftarrow_\$ \mathsf{Sam}(1^\lambda)$

ENC

$\Phi(C)$

$\tilde{C}$

x

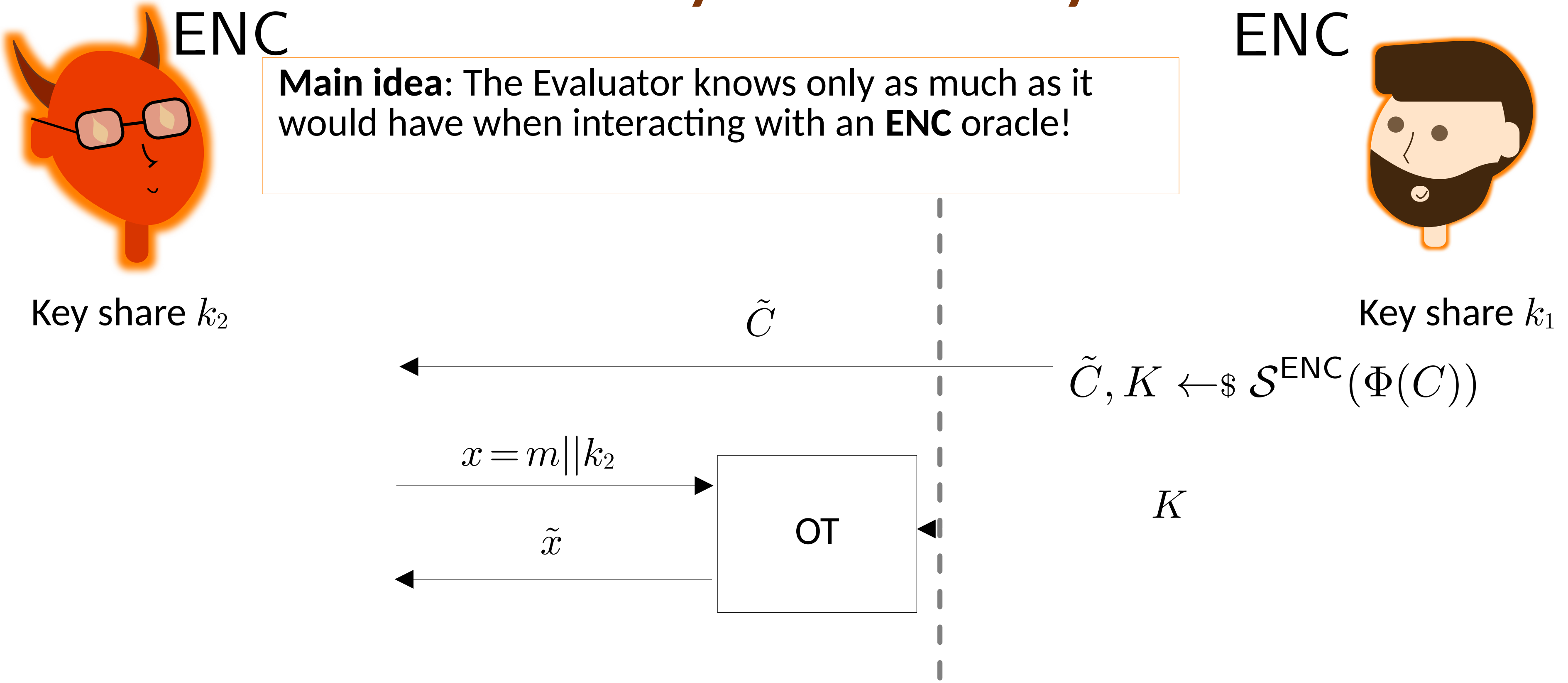
$\tilde{x}$

$\Phi(C)$ ENC

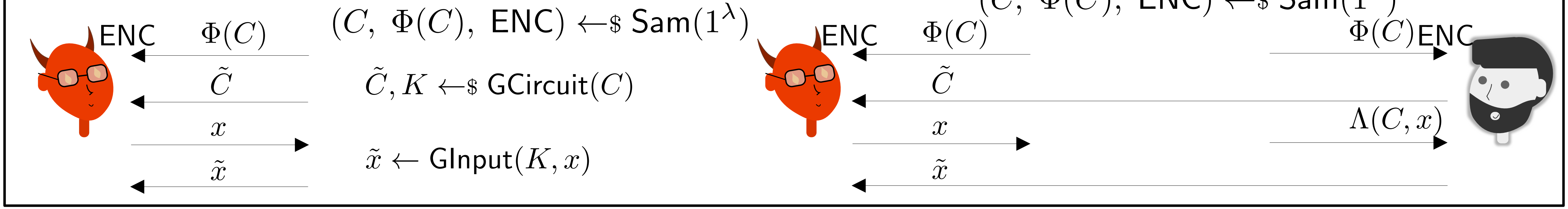
$\Lambda(C, x)$

2-Party DSE: Security

Main idea: The Evaluator knows only as much as it would have when interacting with an **ENC** oracle!



Our new DSIM notion:



Summary of Results (Check out our paper! 😎)

1. DSIM Definition

2. Application of DSIM to DSE

Summary of Results (Check out our paper! 😎)

1. DSIM Definition

2. Application of DSIM to DSE

3. Bootstrapping: We show that DSIM[**NC0**] → DSIM[**P/Poly**]!
Idea: We can represent a circuit as a randomized encoding (REs), which itself is of constant depth.

Summary of Results (Check out our paper! 😎)

1. DSIM Definition

2. Application of DSIM to DSE

3. Bootstrapping: We show that DSIM[**NC0**] $\rightarrow$ DSIM[**P/Poly**]!
Idea: We can represent a circuit as a randomized encoding (REs), which itself is of constant depth.

4. We observe that the *online complexity* of the Jafargholi-Scafuro-Wichs [**JSW'17**] construction for IND can be improved: $O(|x|)$ instead of $O(|x|+d)$.

Summary of Results (Check out our paper! 😎)

1. DSIM Definition

2. Application of DSIM to DSE

3. Bootstrapping: We show that DSIM[**NC0**] $\rightarrow$ DSIM[**P/Poly**]!
Idea: We can represent a circuit as a randomized encoding (REs), which itself is of constant depth.

4. We observe that the *online complexity* of the Jafargholi-Scafuro-Wichs [**JSW'17**] construction for IND can be improved: $O(|x|)$ instead of $O(|x|+d)$.

5. Tighter online complexity lower bounds for SIM but for garbling schemes (improvement of the Hubacek-Wichs [**HW'14**] bound for MPC) via pseudo-entropy.

Open Questions

- Can we construct DSIM (from reasonable, standard assumptions)?
  In particular can we construct DSIM[**NC0**]?

- Would DSIM have applications beyond MPC?