Malicious Security, Continued

CS 598 DH

Today's objectives

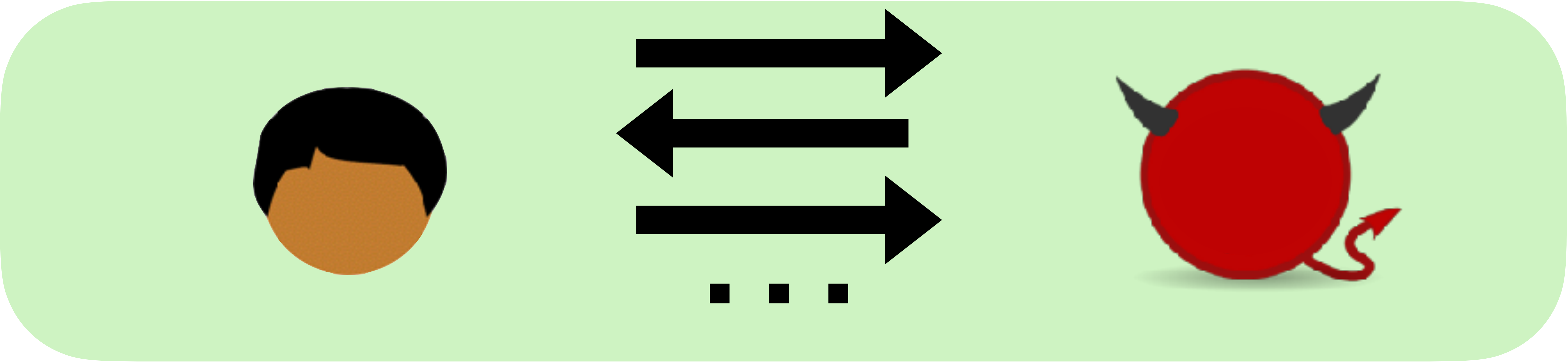
Review malicious security (with abort)

Discuss commitments

Understand "rewinding" in simulation proofs

See a proof for a (slightly) less contrived protocol

Malicious Security (with abort)



A protocol Π securely realizes a functionality f in the presence of a malicious (with abort) adversary if for ***every*** *real-world adversary $\mathcal{A}$ corrupting party i,* ***there exists*** *an ideal-world adversary $\mathcal{S}_i$ (a simulator) such that for all inputs x, y the following holds:*

$$\text{Real}^{\Pi}_{\mathcal{A}}(x, y) \approx \text{Ideal}^{f}_{\mathcal{S}_i}(x, y)$$

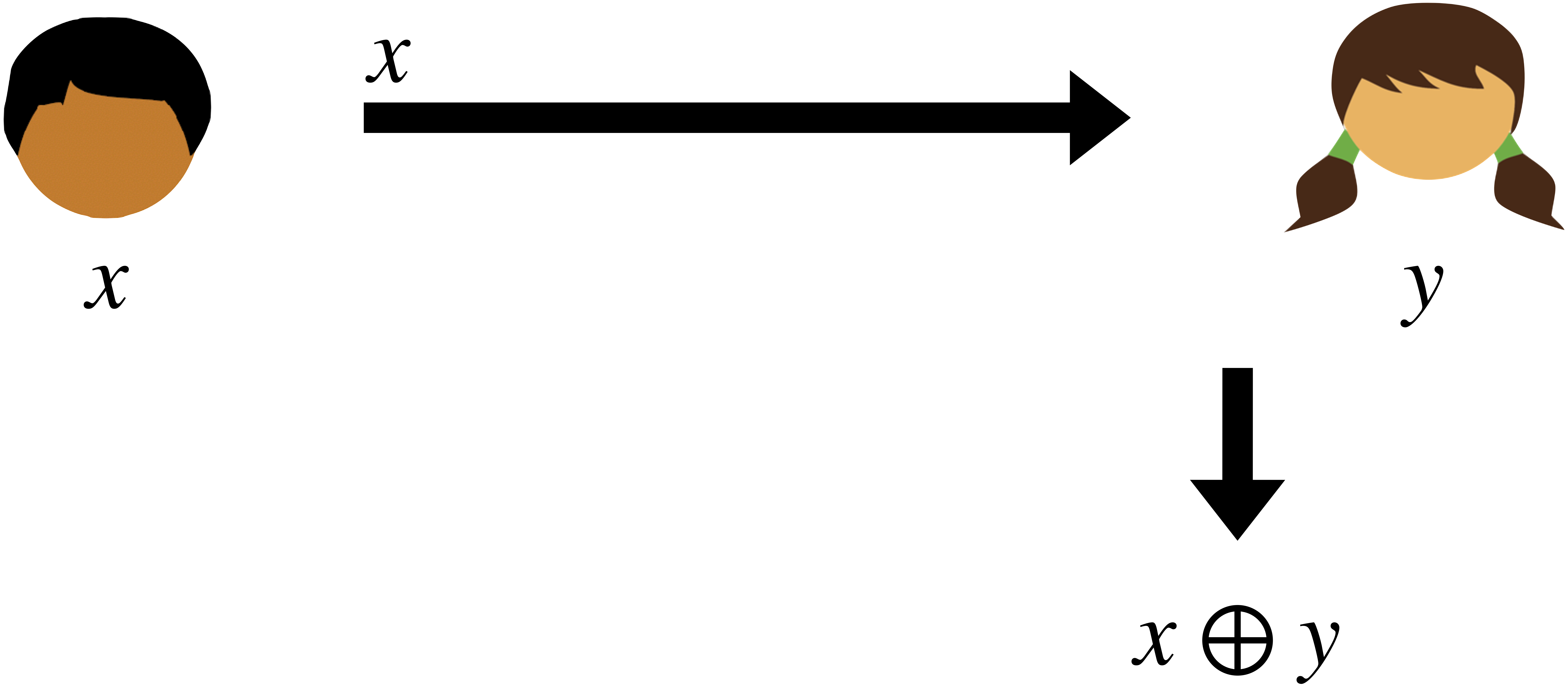
Ensemble of outputs of ***each*** party

Malicious security with abort ideal-world execution

honest party outputs
$f(x, y')$

adversary outputs… ?
whatever it wants

Real World Protocol



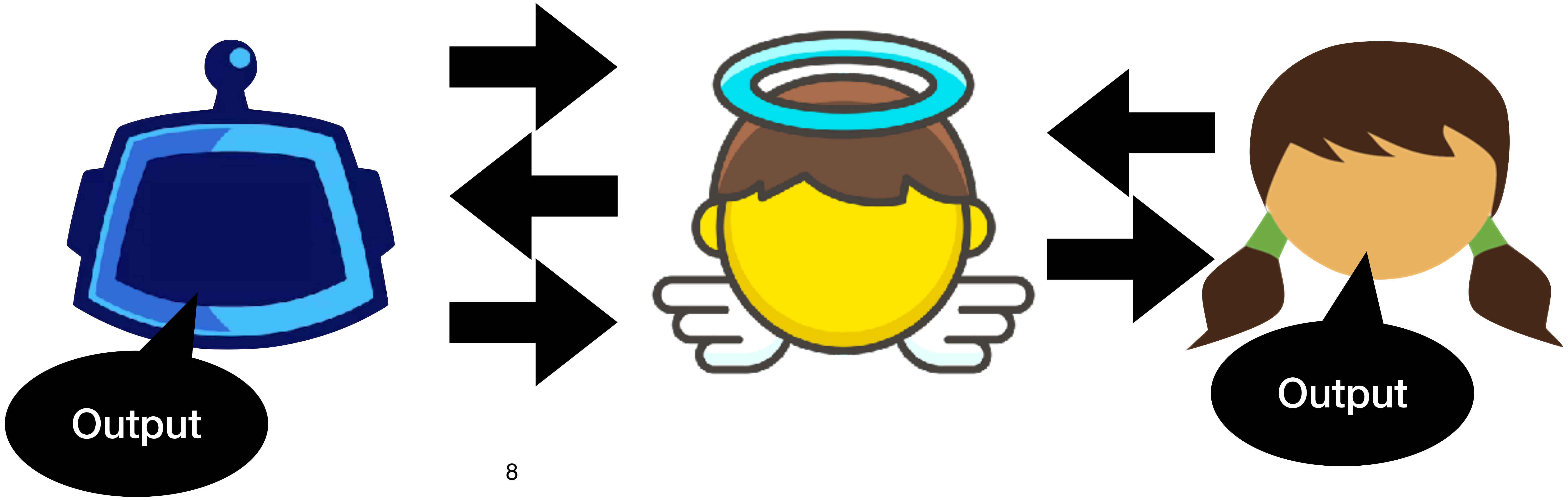
Real World Protocol

Real World Protocol

Real World Protocol

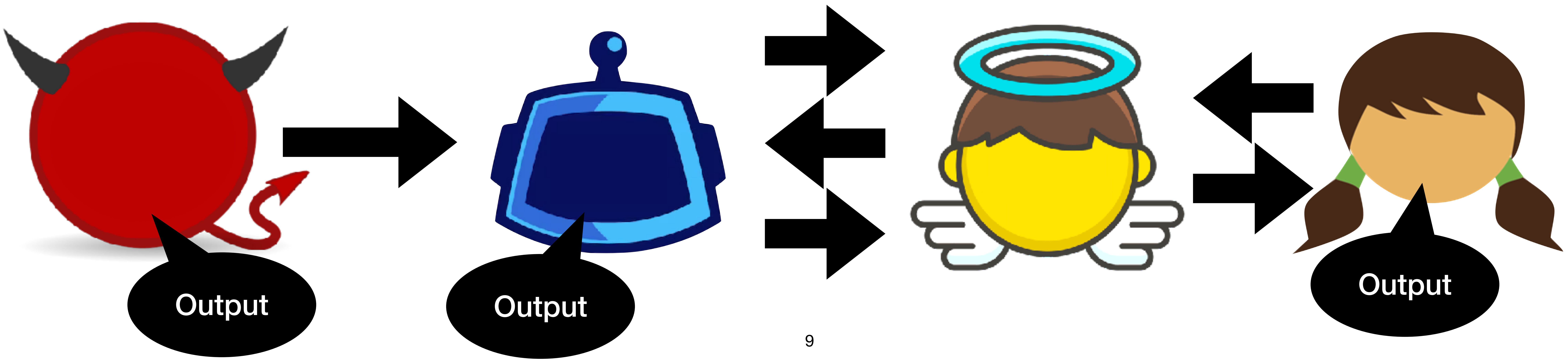
Ideal World Protocol

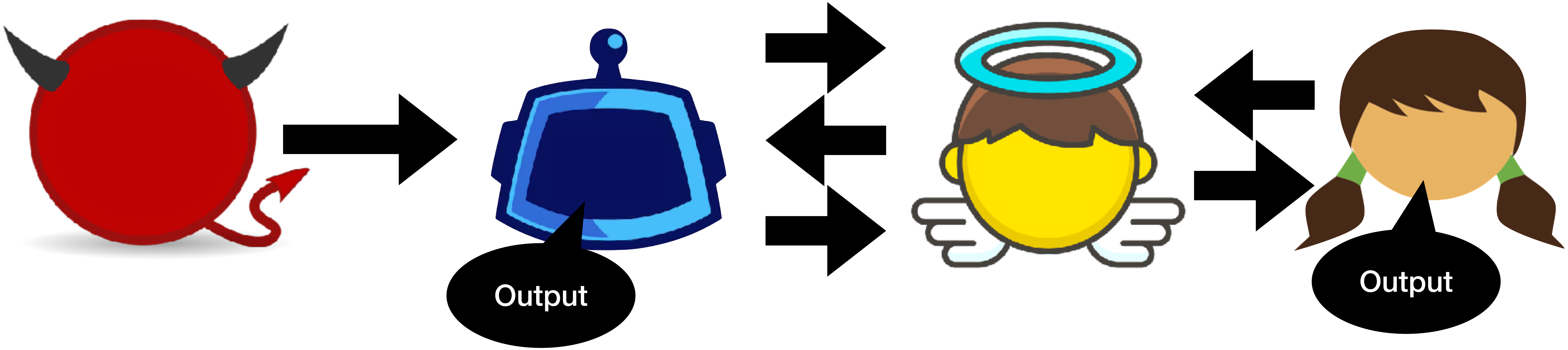
Security is defined by comparing the outputs in these two worlds

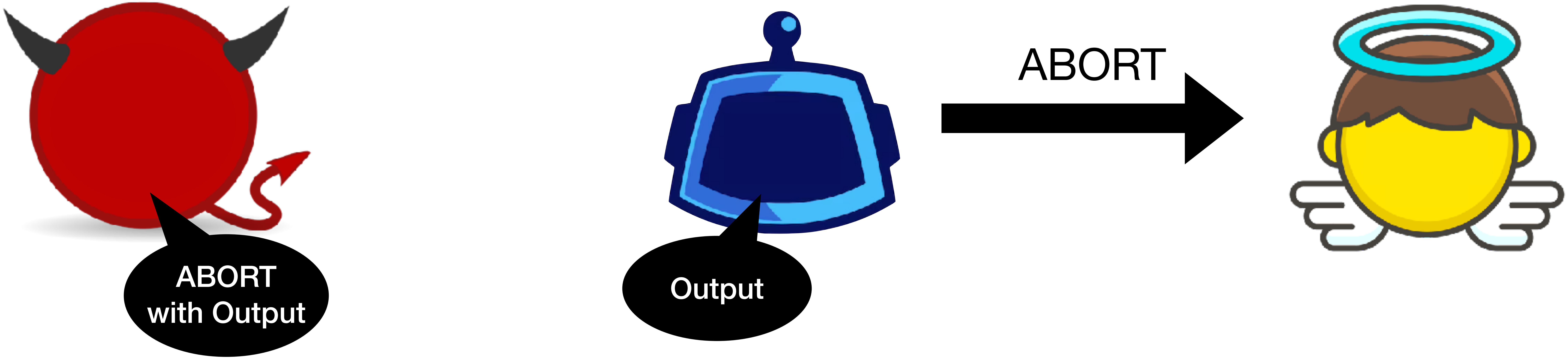


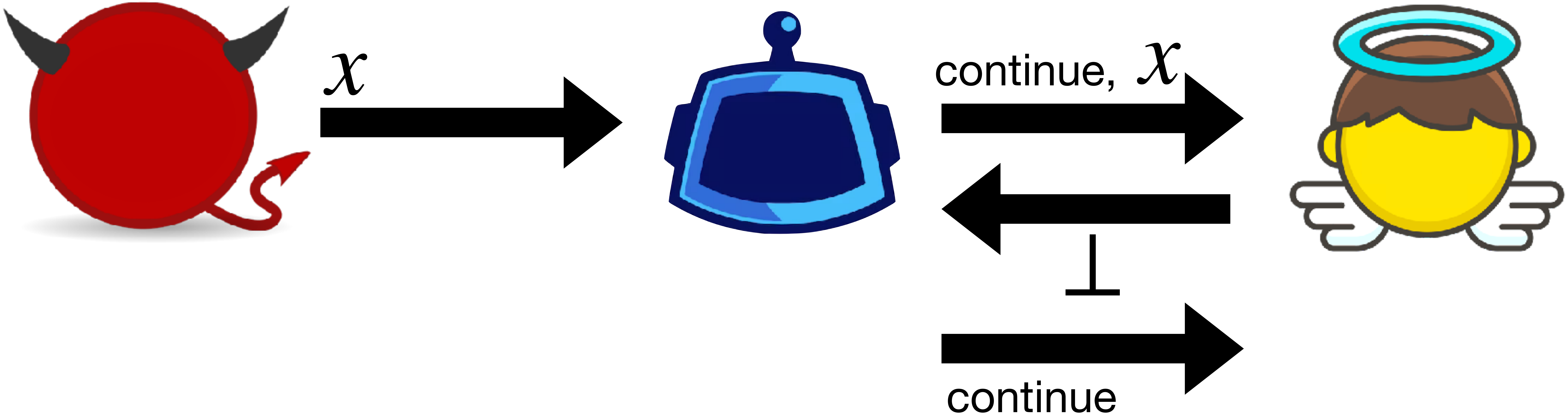
Real World Protocol

Ideal World Protocol

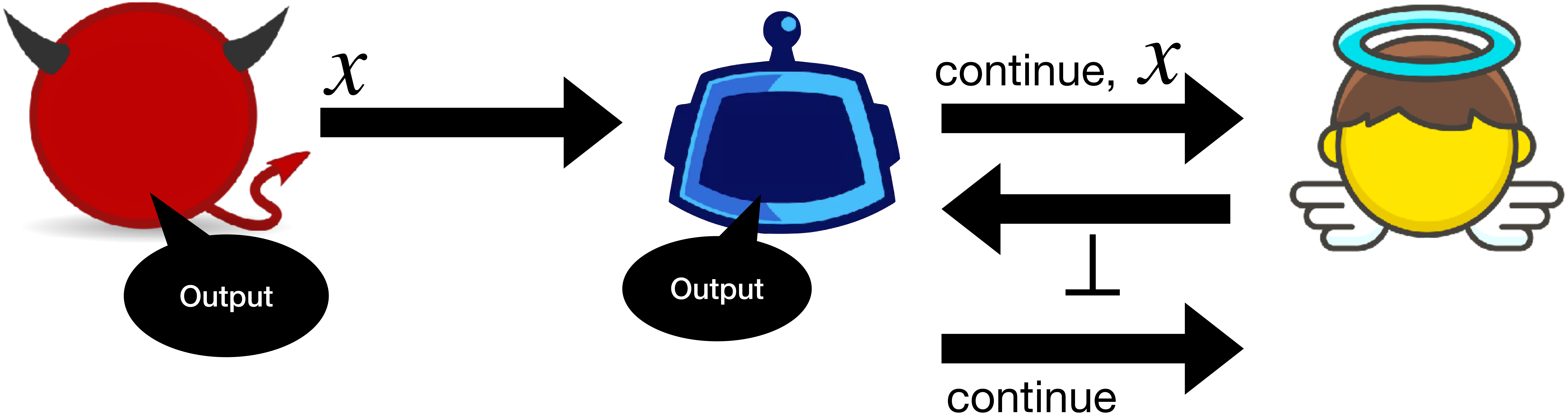



Output
Output


ABORT
with Output
Output
ABORT

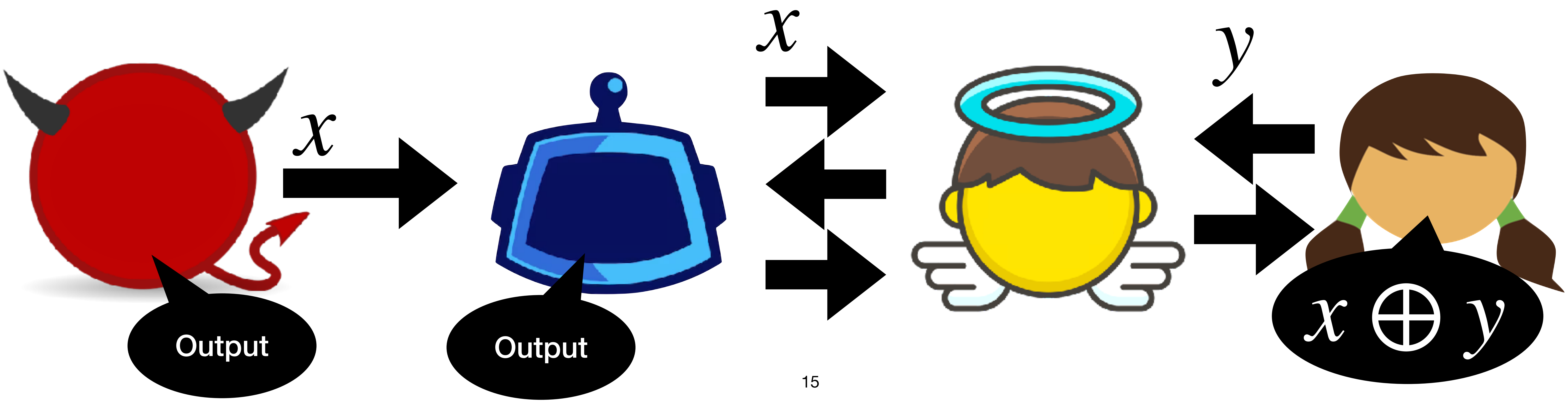


x
continue, x
$\perp$
continue

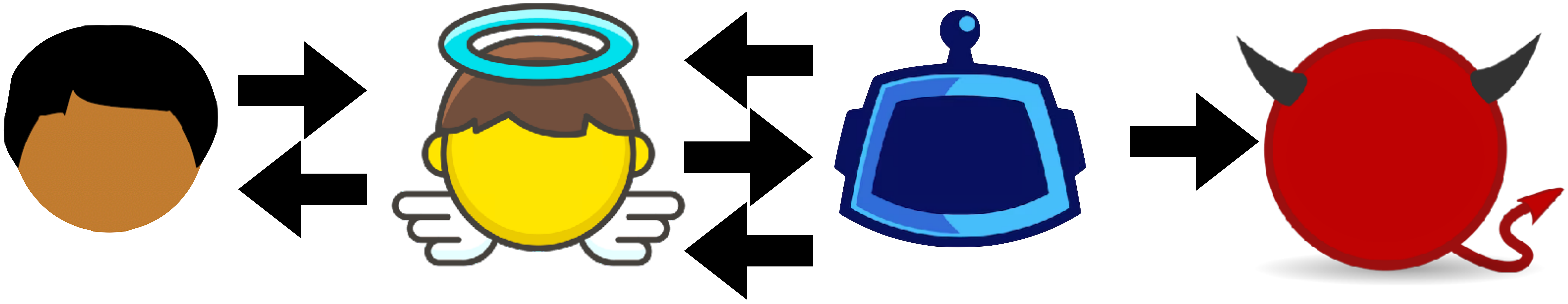

x
Output
Output
continue, x
$\perp$
continue

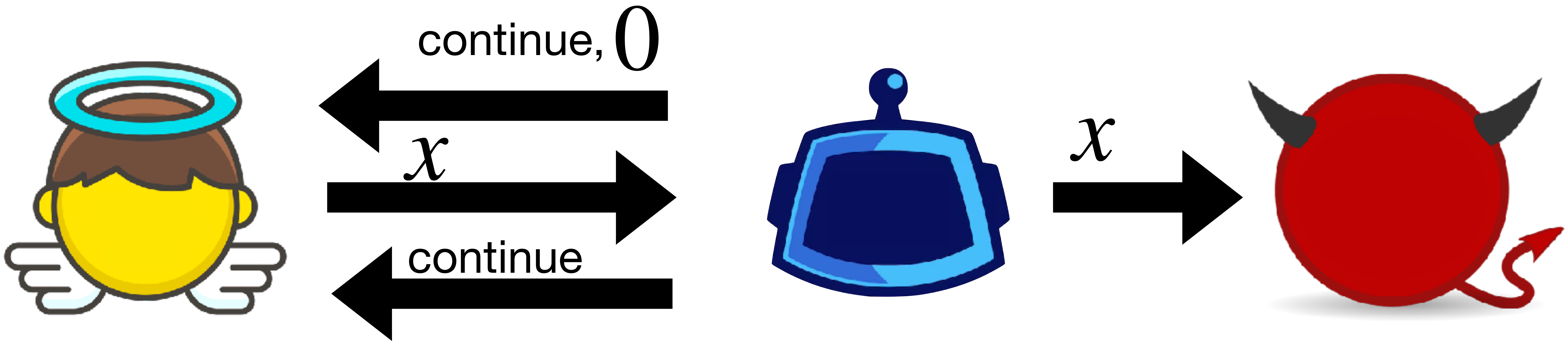
Real World Protocol

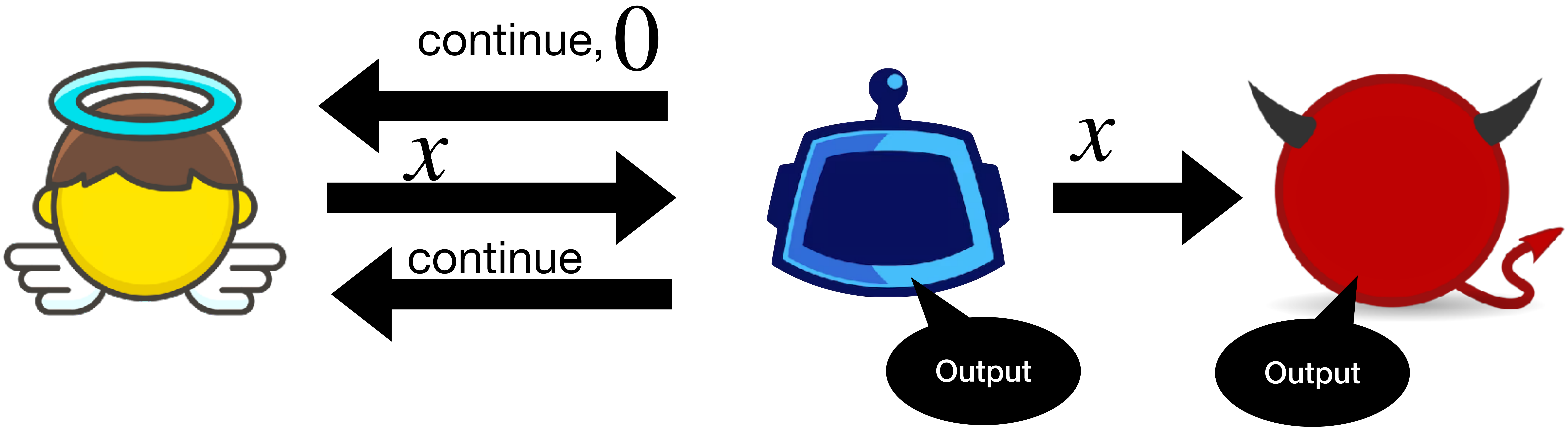
Ideal World Protocol



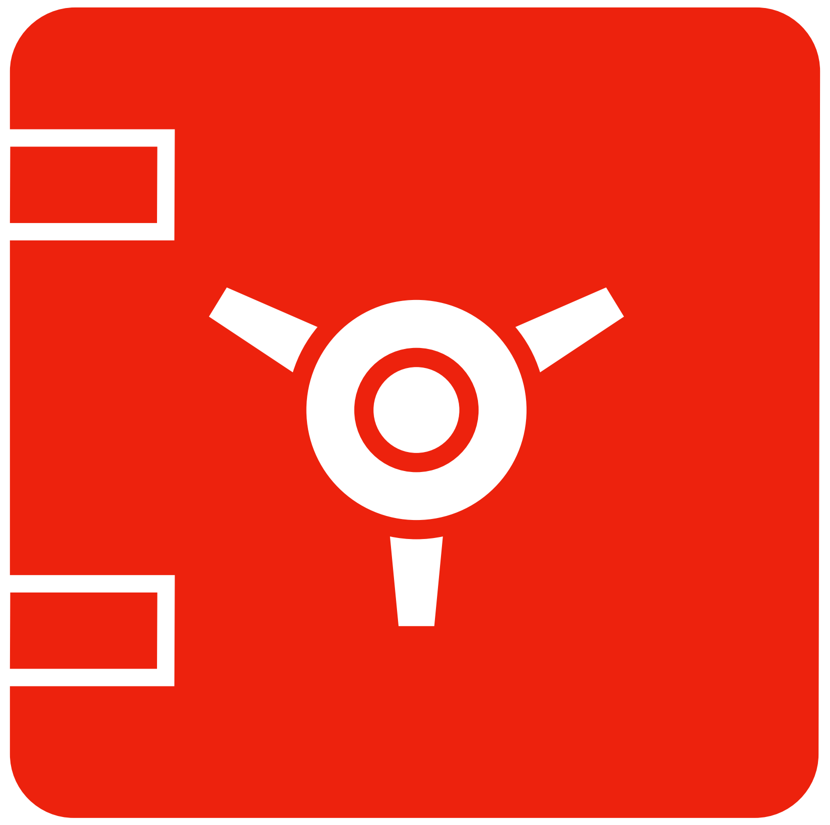
Real World Protocol




continue, 0
x
continue
x

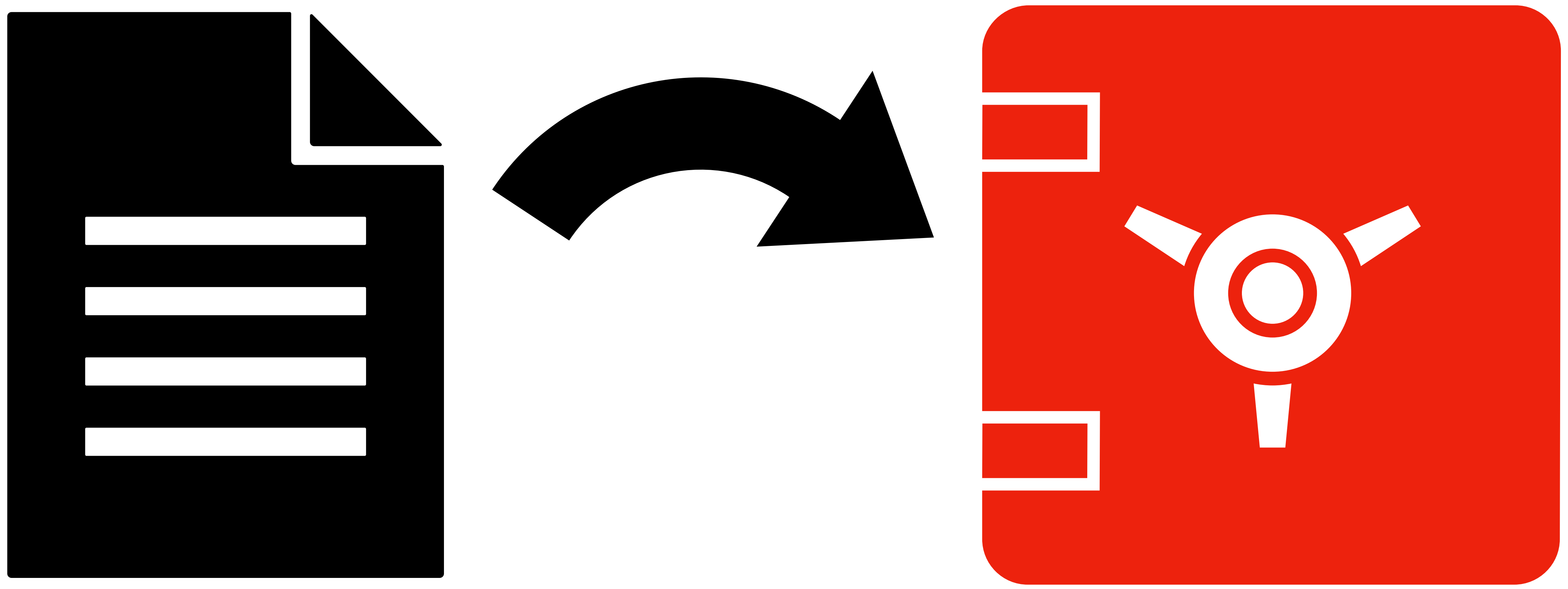

continue, 0
x
continue
x
Output
Output

Commitment Scheme



Commitments are digital analog of a lock box

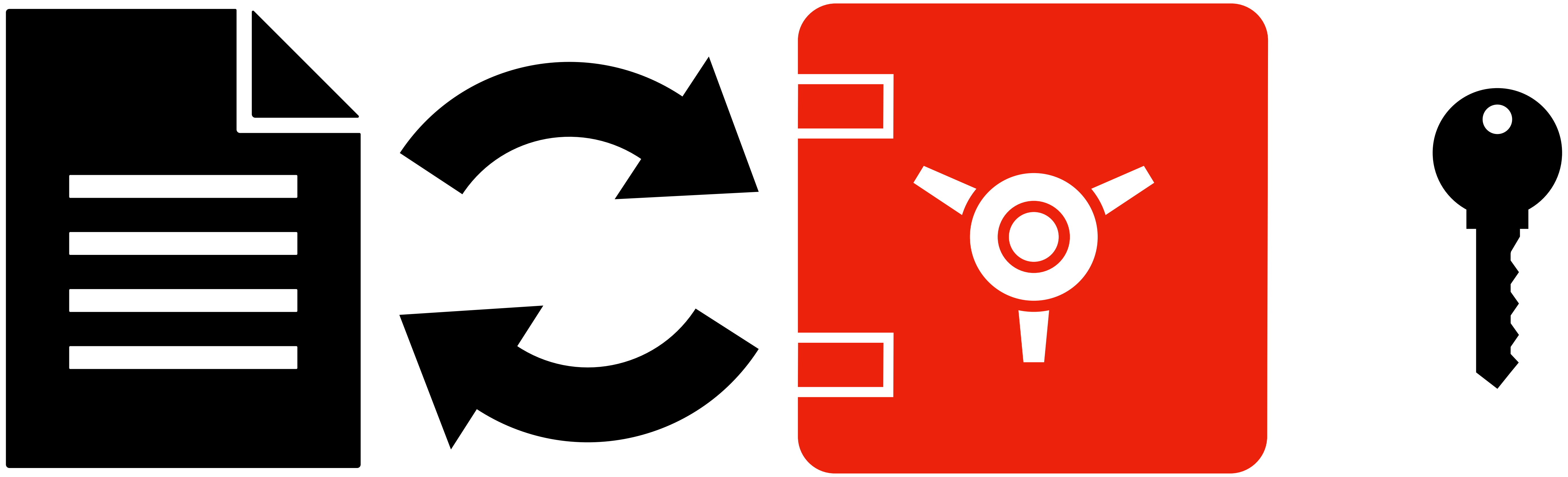
Commitment Scheme



Commitments are digital analog of a lock box

I can put a message in the lock box and then give it to you

Commitment Scheme

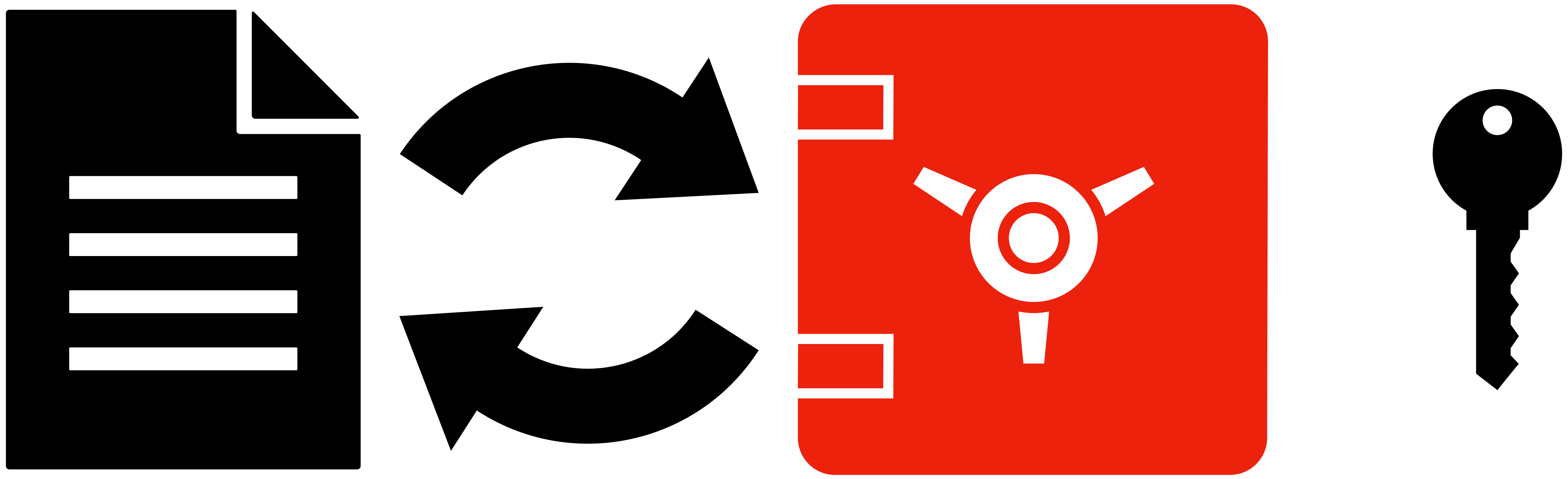


Commitments are digital analog of a lock box

I can put a message in the lock box and then give it to you

I can send you a key, allowing you to open the lock box

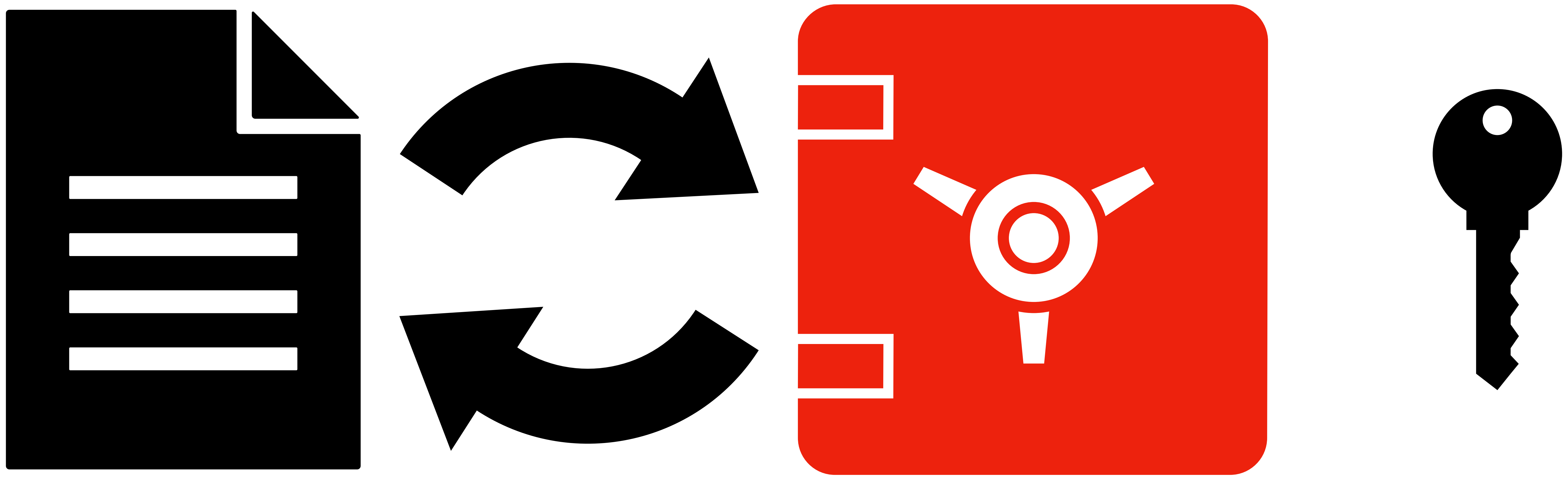
Commitment Scheme



Hiding

I am confident you cannot open the box without the key

Commitment Scheme



Hiding

I am confident you cannot open the box without the key

You are confident I cannot tamper with the content of the box

Binding

Commitment Scheme

$$\mathrm{com}(x; r)$$

Commitment to x with randomness $r \xleftarrow{\$} \{0,1\}^\lambda$

Commitment Scheme

$$\mathrm{com}(x; r)$$

Commitment to x with randomness $r \xleftarrow{\$} \{0,1\}^\lambda$

$$\mathrm{com}(x; r) \approx \mathrm{com}(y; r)$$

Computationally hiding

Commitment Scheme

$$\mathrm{com}(x; r)$$

Commitment to x with randomness $r \xleftarrow{\$} \{0,1\}^\lambda$

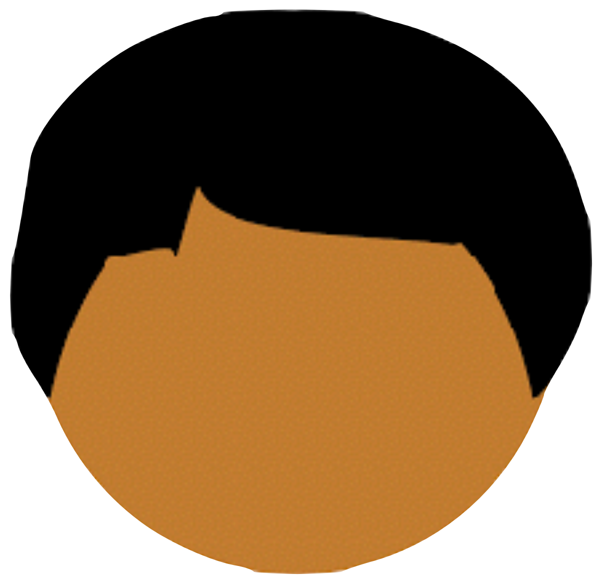
$$\mathrm{com}(x; r) \approx \mathrm{com}(y; r)$$

Computationally hiding

$x \neq y \implies \mathscr{A}$ cannot find $\mathrm{com}(x; r) = \mathrm{com}(y; r)$

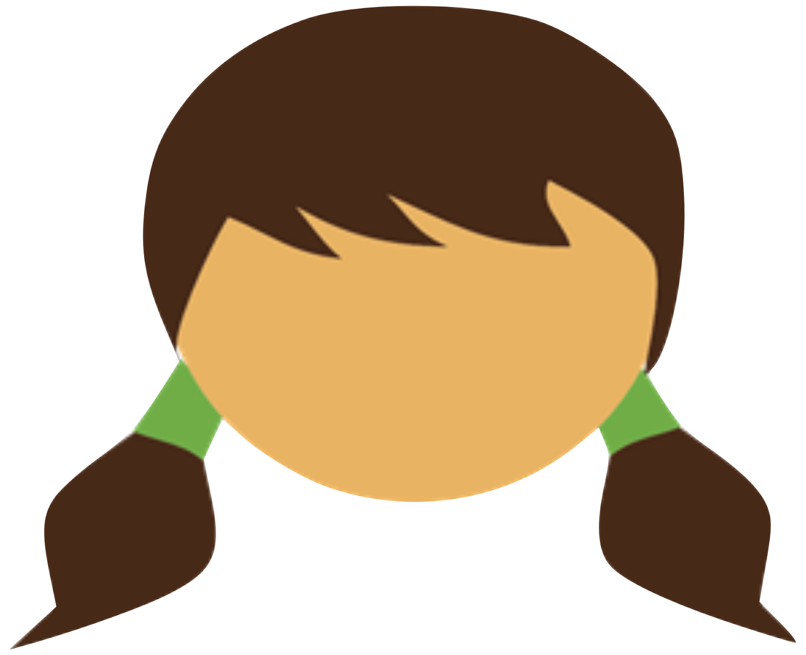
Perfectly Binding

Example Functionality



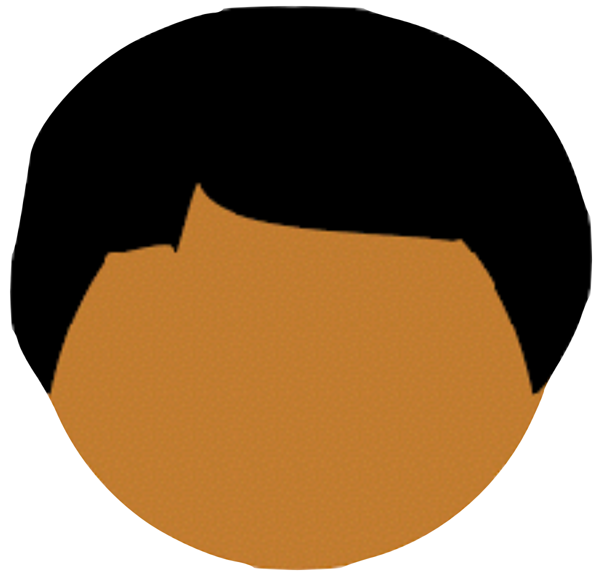
x

$$f(x, y) = x \oplus y$$

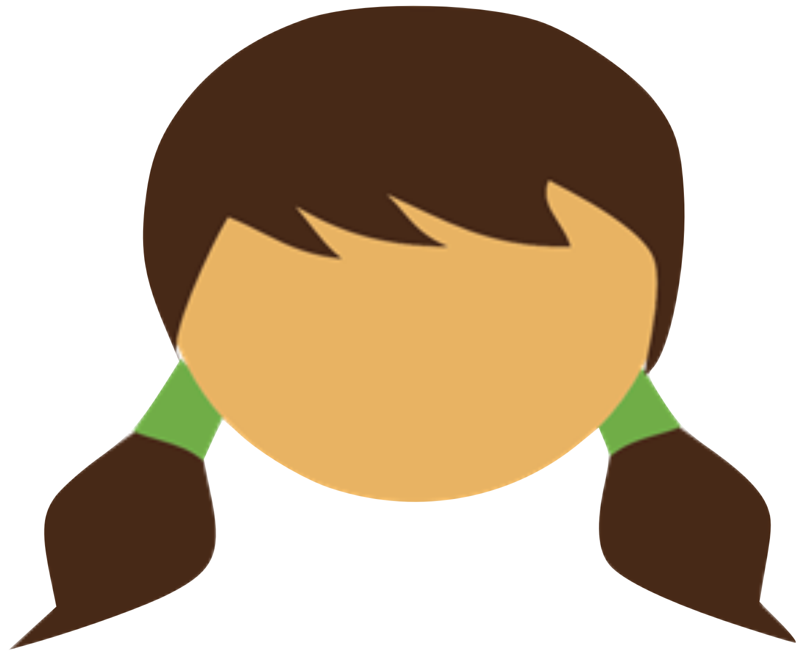


y

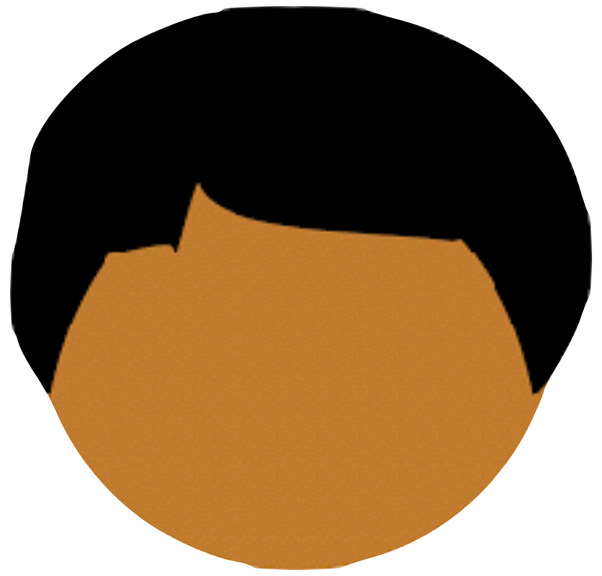
An even simpler functionality



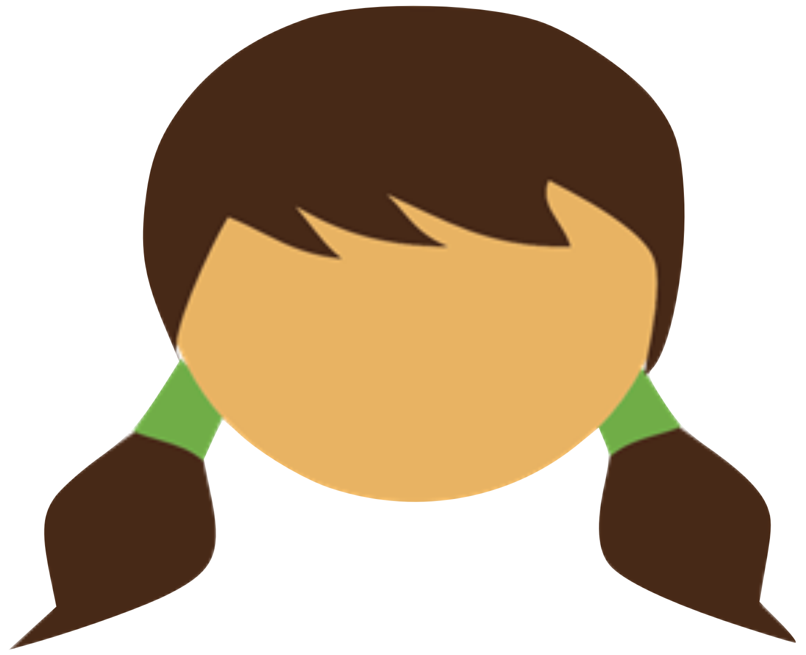
$$f(\cdot) = \{ \, r \mid r \xleftarrow{\$} \{0,1\} \, \}$$



An even simpler functionality



$$f(\cdot) = \{\ r \mid r \xleftarrow{\$} \{0,1\}\ \}$$



Attempt

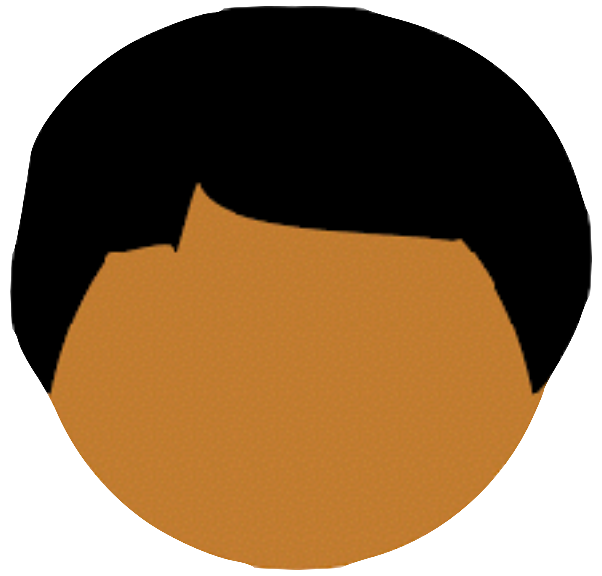
$b_0 \xleftarrow{\$} \{0,1\}$

$b_1 \xleftarrow{\$} \{0,1\}$

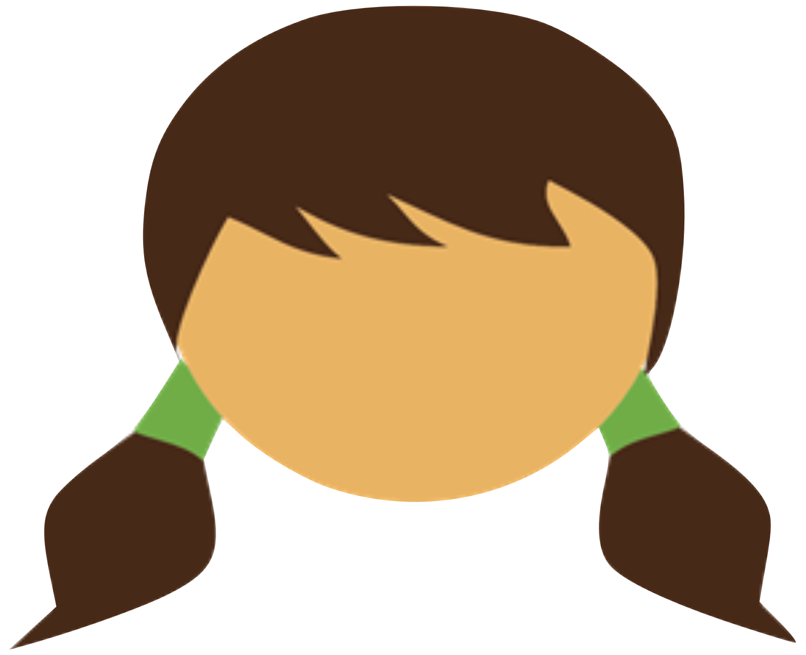
$$\xrightarrow{\quad b_0 \quad}$$

$$\xleftarrow{\quad b_1 \quad}$$

An even simpler functionality

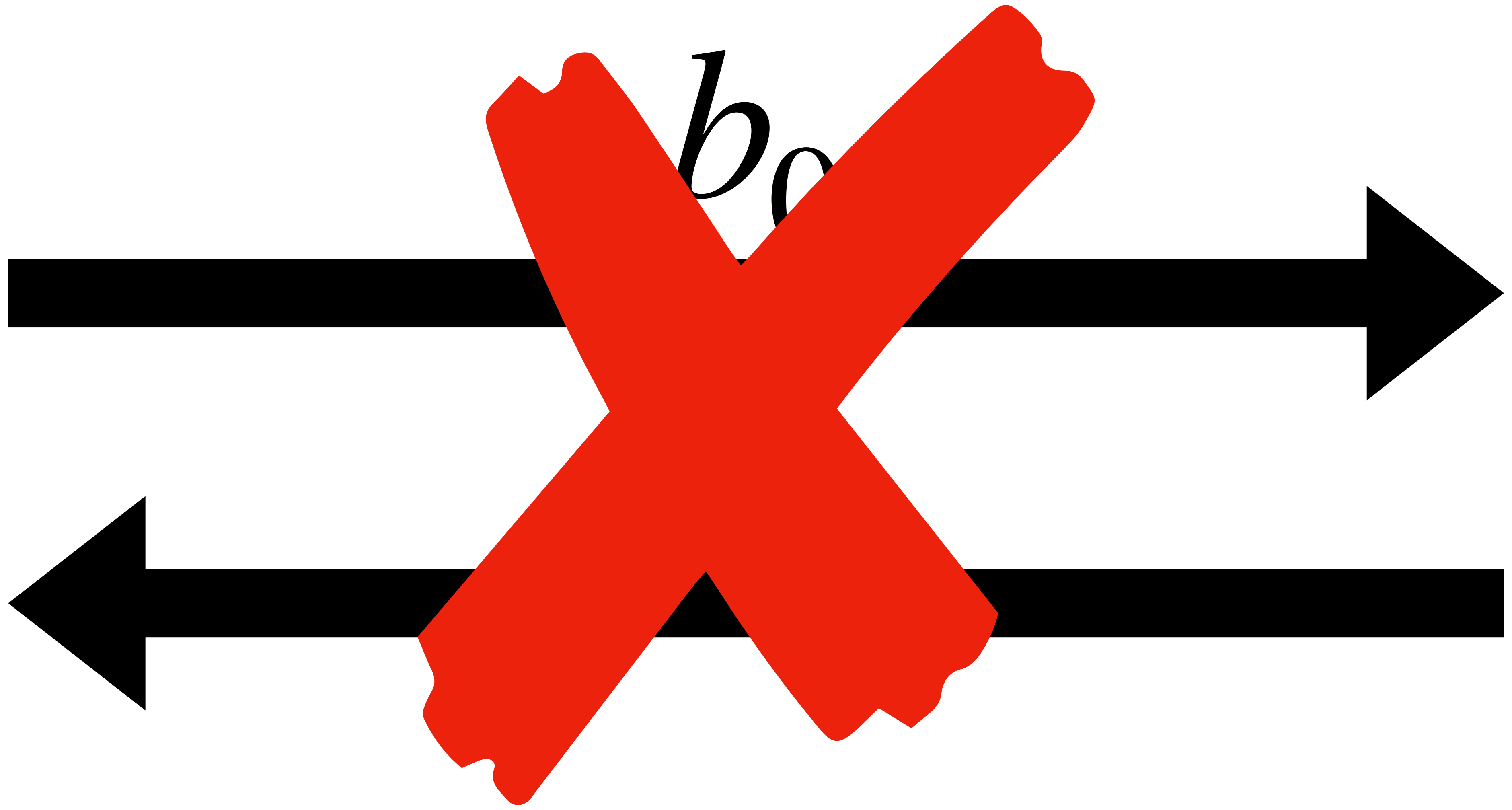


$$f(\cdot) = \{\ r \mid r \xleftarrow{\$} \{0,1\}\ \}$$



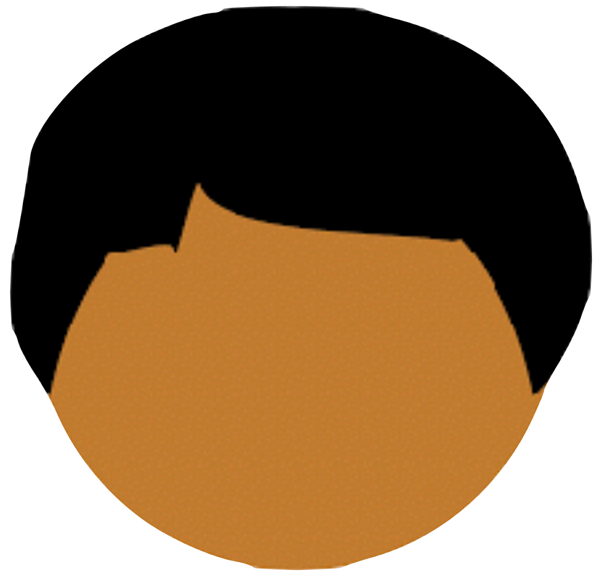
Attempt

$b_0 \xleftarrow{\$} \{0,1\}$

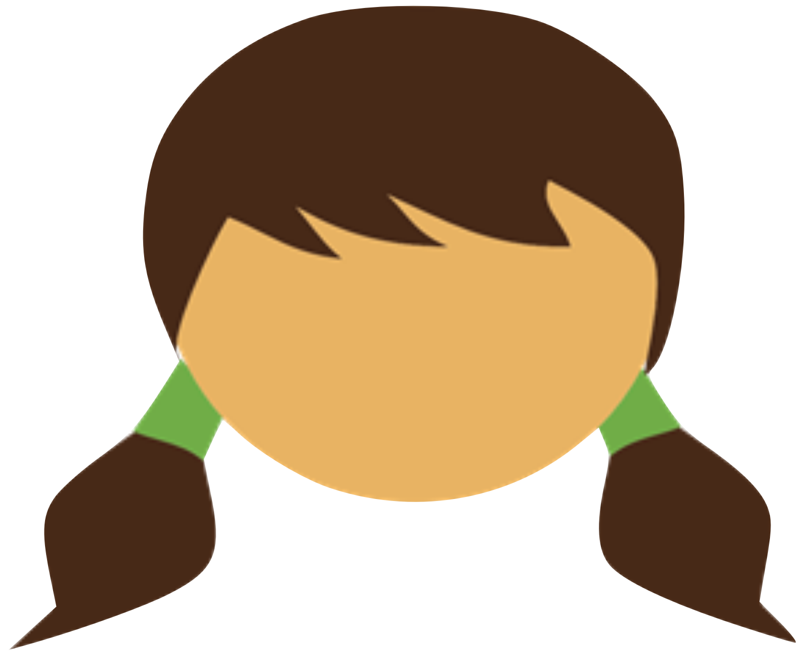


$b_1 \xleftarrow{\$} \{0,1\}$

An even simpler functionality



$$f(\cdot) = \{\ r \mid r \xleftarrow{\$} \{0,1\}\ \}$$



Attempt

$b_0 \xleftarrow{\$} \{0,1\}$

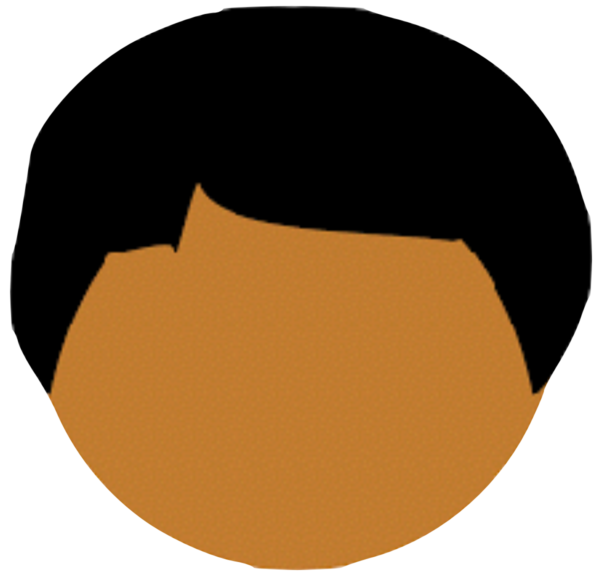
$$\cancel{\xrightarrow{b_0}}$$

$$\cancel{\xleftarrow{\quad}}$$

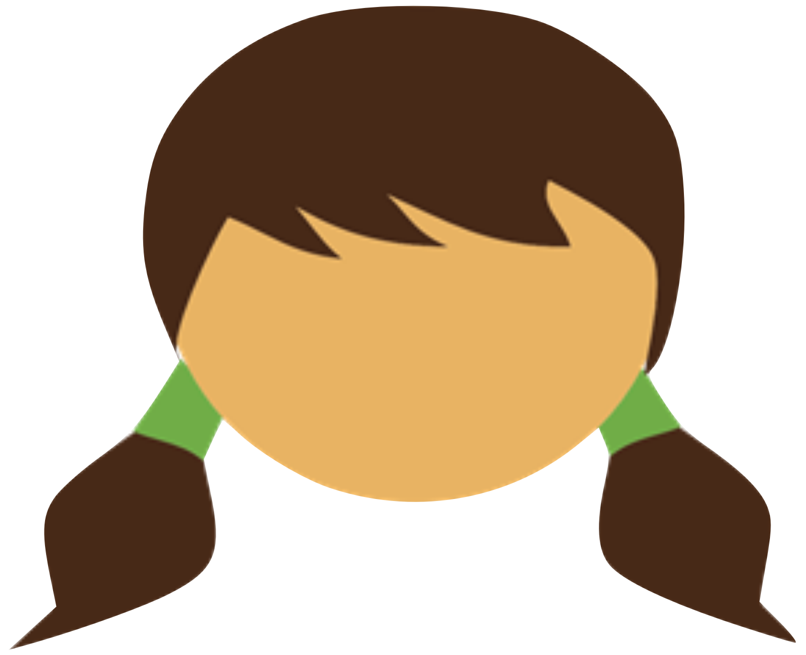
$b_1 \xleftarrow{\$} \{0,1\}$

Can choose b_1 based on b_0

An even simpler functionality



$$f(\cdot) = \{\ r \mid r \xleftarrow{\$} \{0,1\}\ \}$$



Attempt

$b_0 \xleftarrow{\$} \{0,1\}$

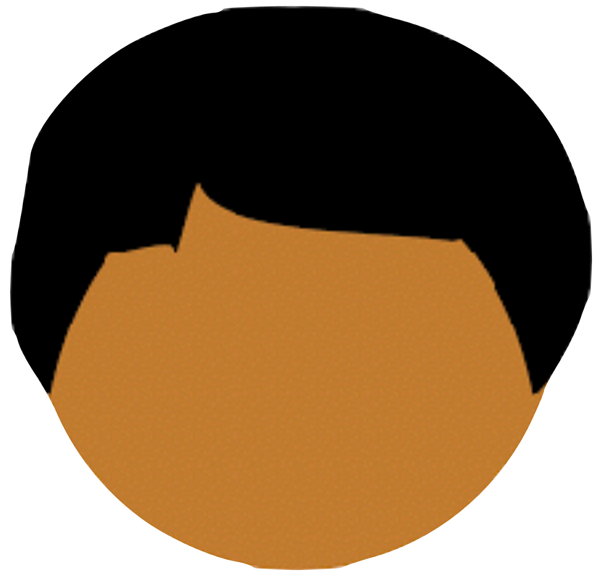
$b_1 \xleftarrow{\$} \{0,1\}$

$\cancel{b_0}$

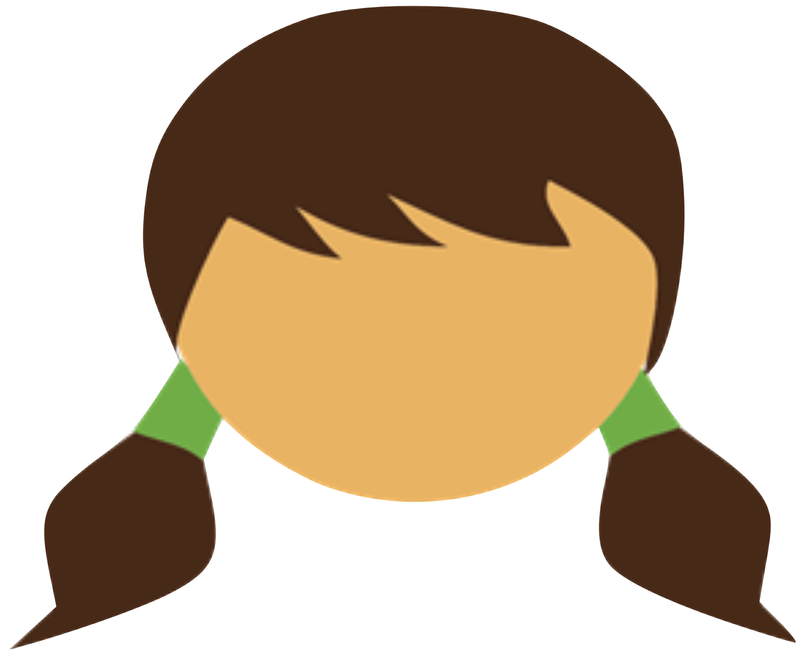
Can choose b_1 based on b_0

Could have Bob choose first, but this just lets Bob cheat

An even simpler functionality

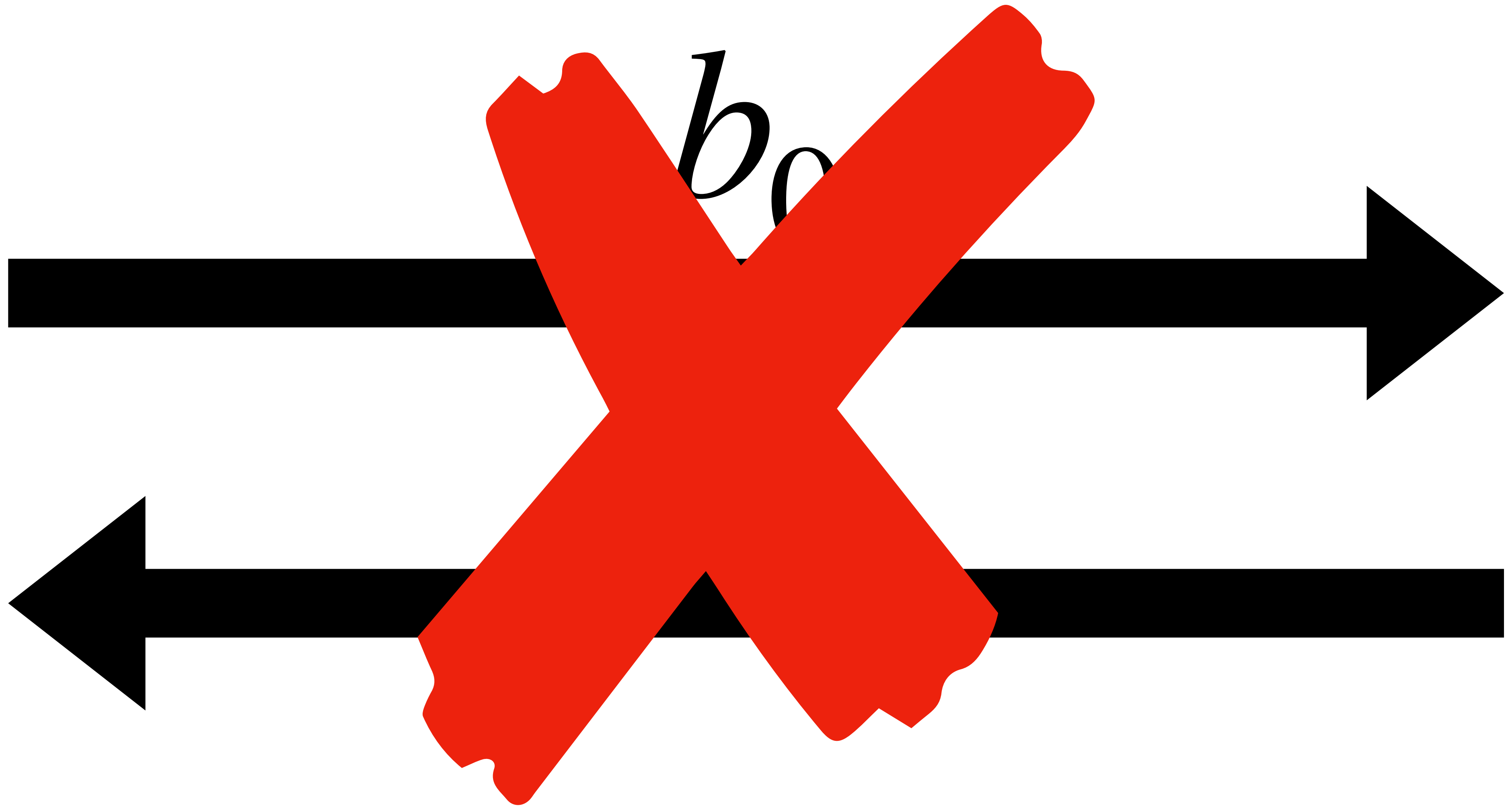


$$f(\cdot) = \{\ r \mid r \xleftarrow{\$} \{0,1\}\ \}$$



Attempt

$b_0 \xleftarrow{\$} \{0,1\}$



$b_1 \xleftarrow{\$} \{0,1\}$

Can choose b_1 based on b_0

Use a commitment!

How To Simulate It – A Tutorial on the Simulation Proof Technique*

Yehuda Lindell

Dept. of Computer Science
Bar-Ilan University, Israel
lindell@biu.ac.il

April 25, 2021

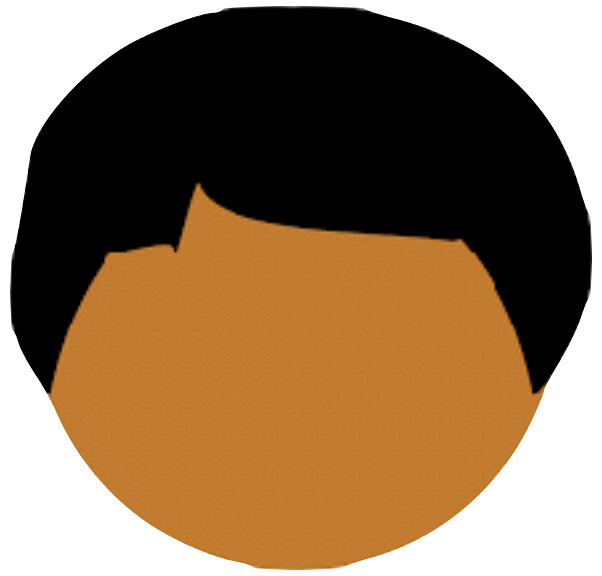
Abstract

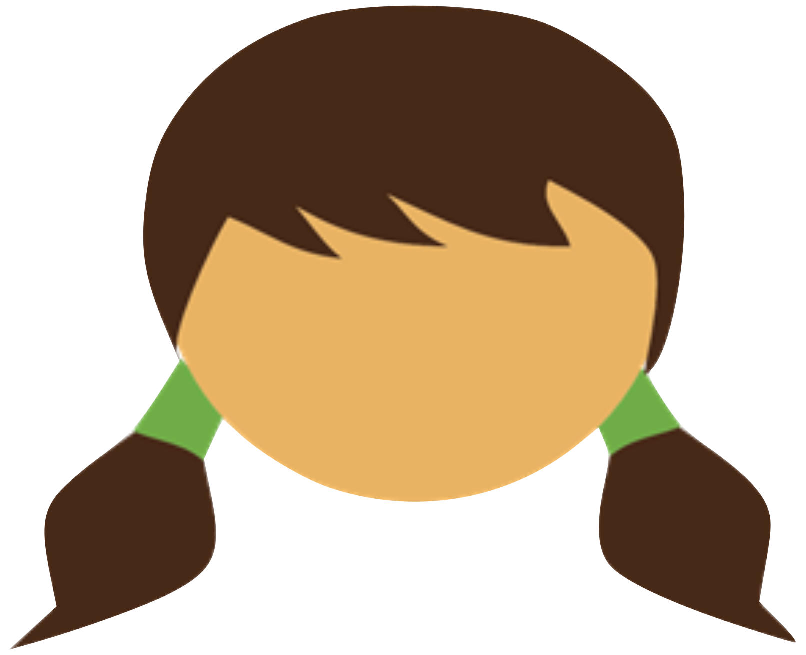
One of the most fundamental notions of cryptography is that of *simulation*. It stands behind the concepts of semantic security, zero knowledge, and security for multiparty computation. However, writing a simulator and proving security via the use of simulation is a non-trivial task, and one that many newcomers to the field often find difficult. In this tutorial, we provide a guide to how to write simulators and prove security via the simulation paradigm. Although we have tried to make this tutorial as stand-alone as possible, we assume some familiarity with the notions of secure encryption, zero-knowledge, and secure computation.

Keywords: secure computation, the simulation technique, tutorial

*This tutorial appeared in the book *Tutorials on the Foundations of Cryptography*, published in honor of Oded Goldreich's 60th birthday.

$$f(\cdot) = \{\ r \mid r \xleftarrow{\$} \{0,1\}\ \}$$





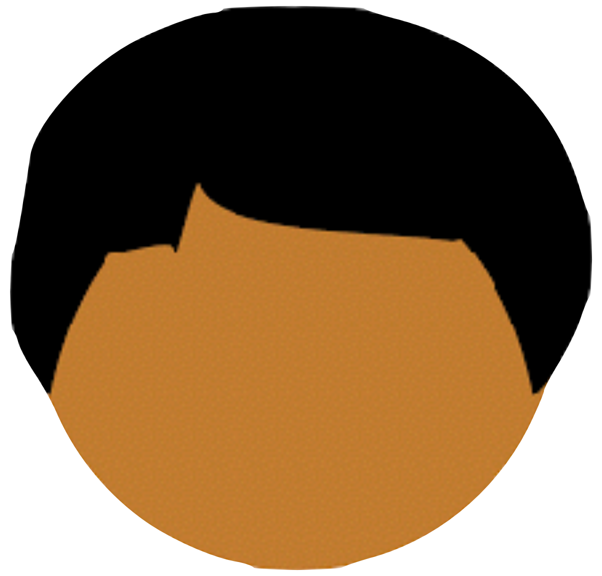
$b_0 \xleftarrow{\$} \{0,1\}$

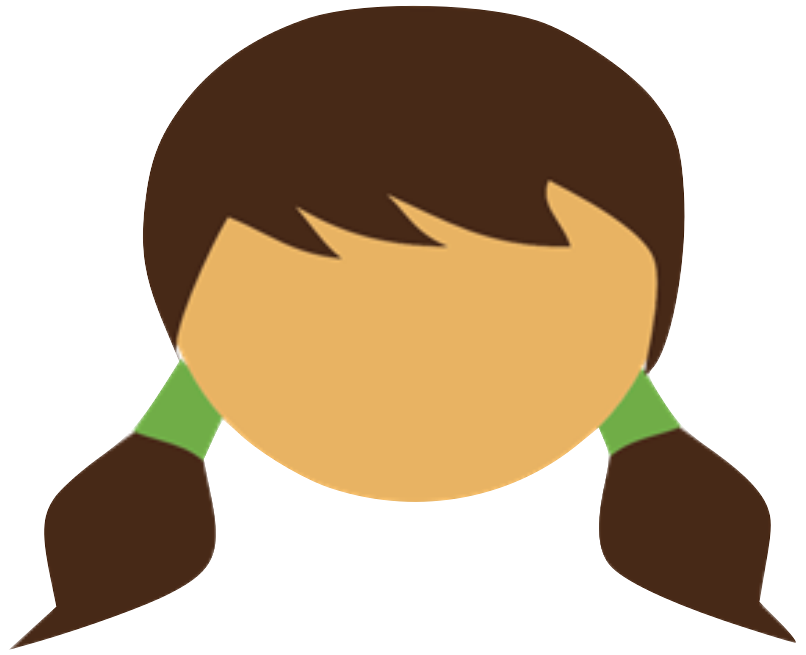
$r \xleftarrow{\$} \{0,1\}^\lambda$

$$\xrightarrow{\quad c = \mathrm{Com}(b_0; r) \quad}$$

$b_1 \xleftarrow{\$} \{0,1\}$

$$f(\cdot) = \{\ r \mid r \xleftarrow{\$} \{0,1\}\ \}$$





$b_0 \xleftarrow{\$} \{0,1\}$

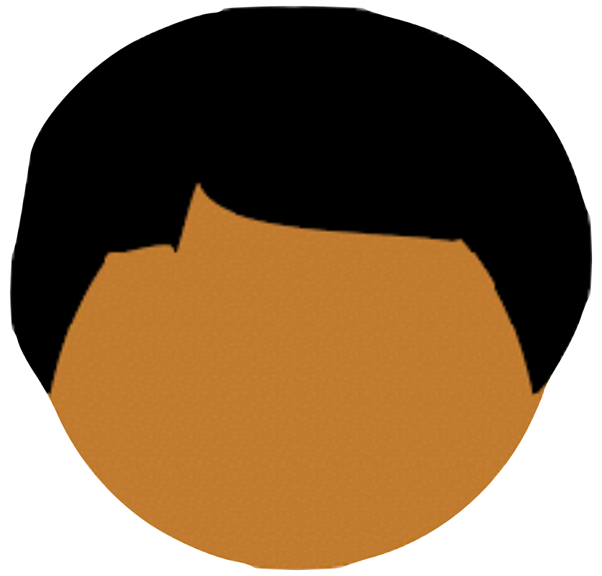
$r \xleftarrow{\$} \{0,1\}^\lambda$

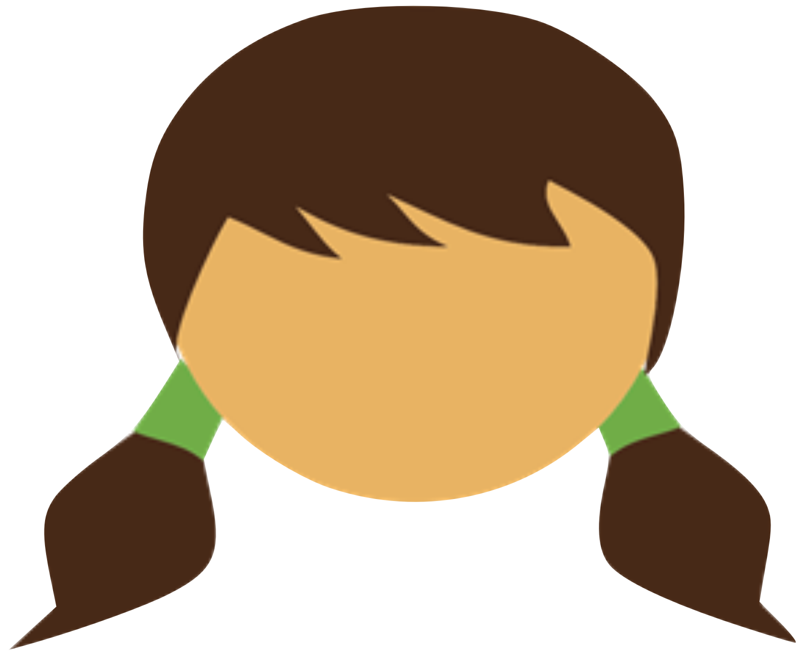
$b_1 \xleftarrow{\$} \{0,1\}$

$$\xrightarrow{c = \mathrm{Com}(b_0; r)}$$

$$\xleftarrow{b_1}$$

$$f(\cdot) = \{\ r \mid r \xleftarrow{\$} \{0,1\}\ \}$$





$b_0 \xleftarrow{\$} \{0,1\}$

$r \xleftarrow{\$} \{0,1\}^\lambda$

$b_1 \xleftarrow{\$} \{0,1\}$

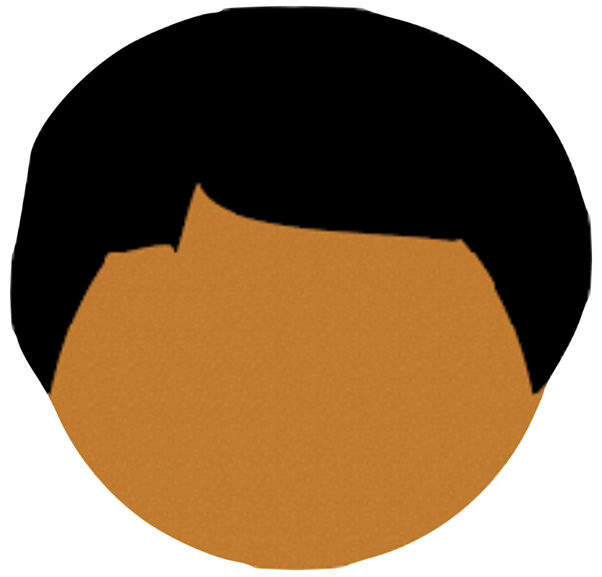
$$\xrightarrow{c = \mathrm{Com}(b_0; r)}$$

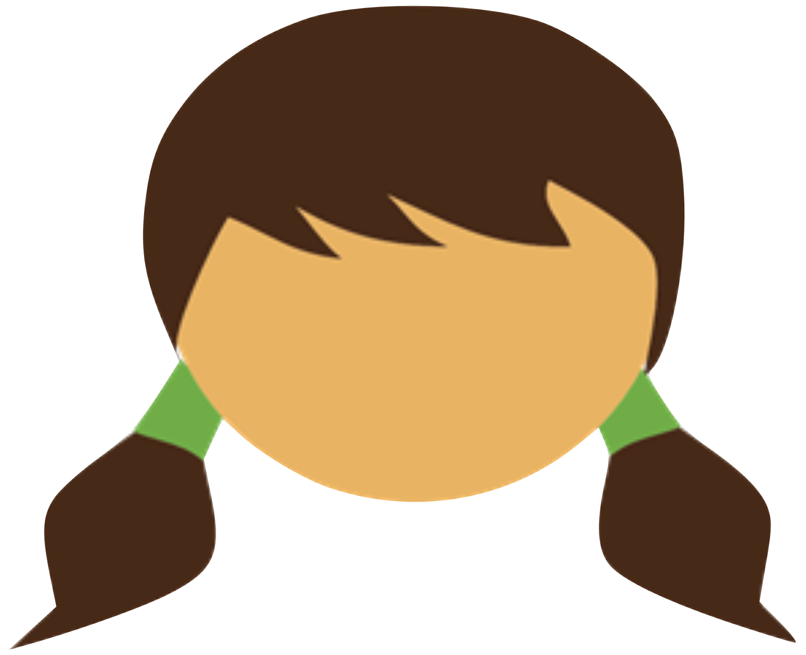
$$\xleftarrow{b_1}$$

$$\xrightarrow{b_0,\, r}$$

$c \stackrel{?}{=} \mathrm{Com}(b_0; r)$

$$f(\cdot) = \{\ r \mid r \xleftarrow{\$} \{0,1\}\ \}$$





$b_0 \xleftarrow{\$} \{0,1\}$

$r \xleftarrow{\$} \{0,1\}^\lambda$

$b_1 \xleftarrow{\$} \{0,1\}$

$c = \mathrm{Com}(b_0; r)$ →

← b_1

b_0, r →

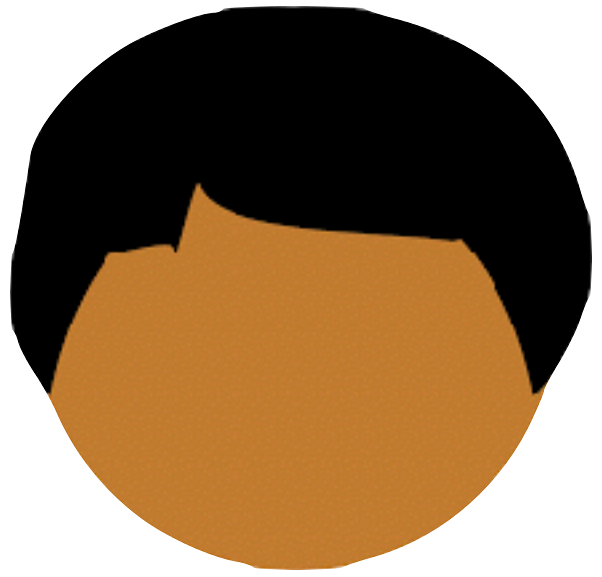
$b_0 \oplus b_1$

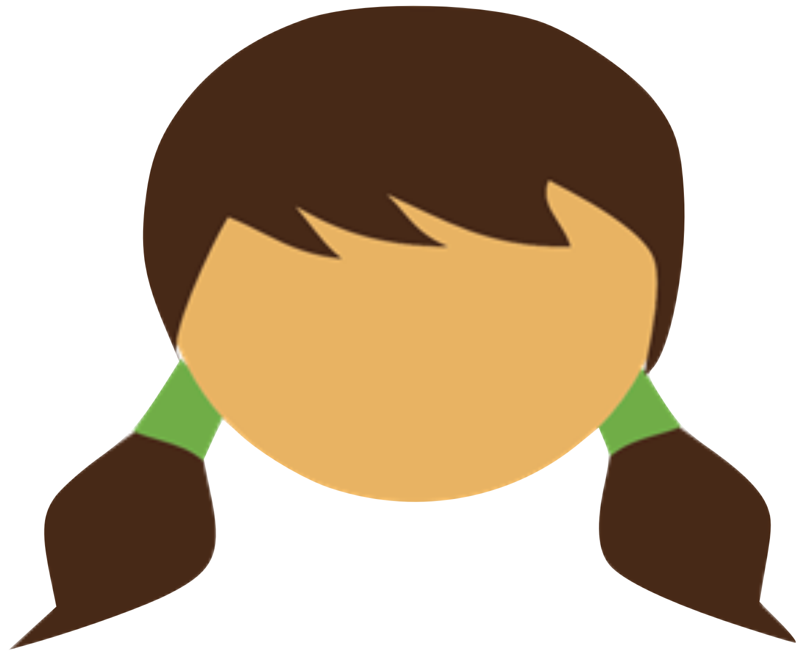
$c \stackrel{?}{=} \mathrm{Com}(b_0; r)$

abort

$b_0 \oplus b_1$

$$f(\cdot) = \{\ r \mid r \xleftarrow{\$} \{0,1\}\ \}$$





$b_0 \xleftarrow{\$} \{0,1\}$

$r \xleftarrow{\$} \{0,1\}^{\lambda}$

$\xrightarrow{\quad c = \mathrm{Com}(b_0; r) \quad}$

$b_1 \xleftarrow{\$} \{0,1\}$

$\xleftarrow{\quad b_1 \quad}$

$\xrightarrow{\quad b_0, r \quad}$

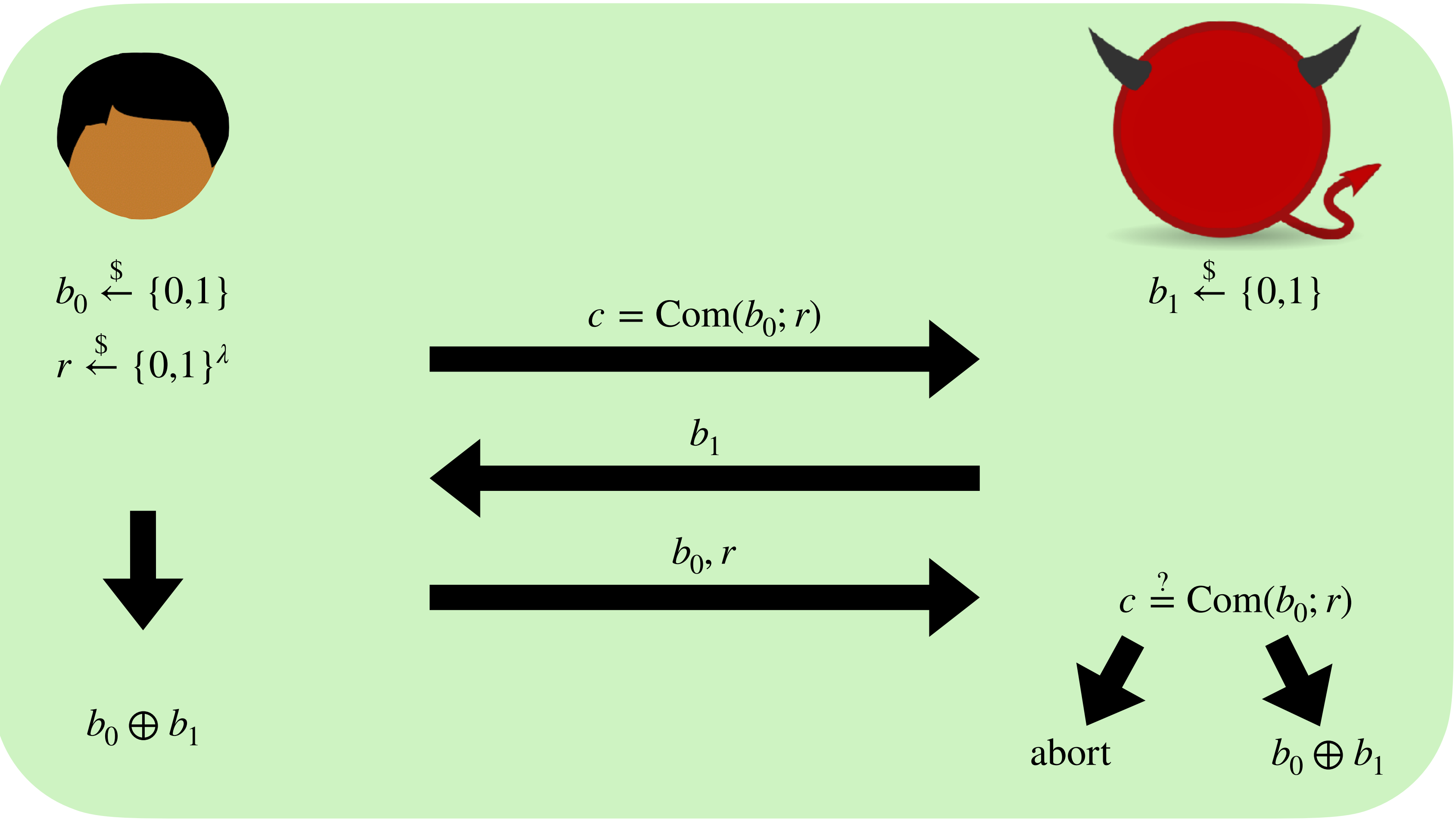
$c \stackrel{?}{=} \mathrm{Com}(b_0; r)$

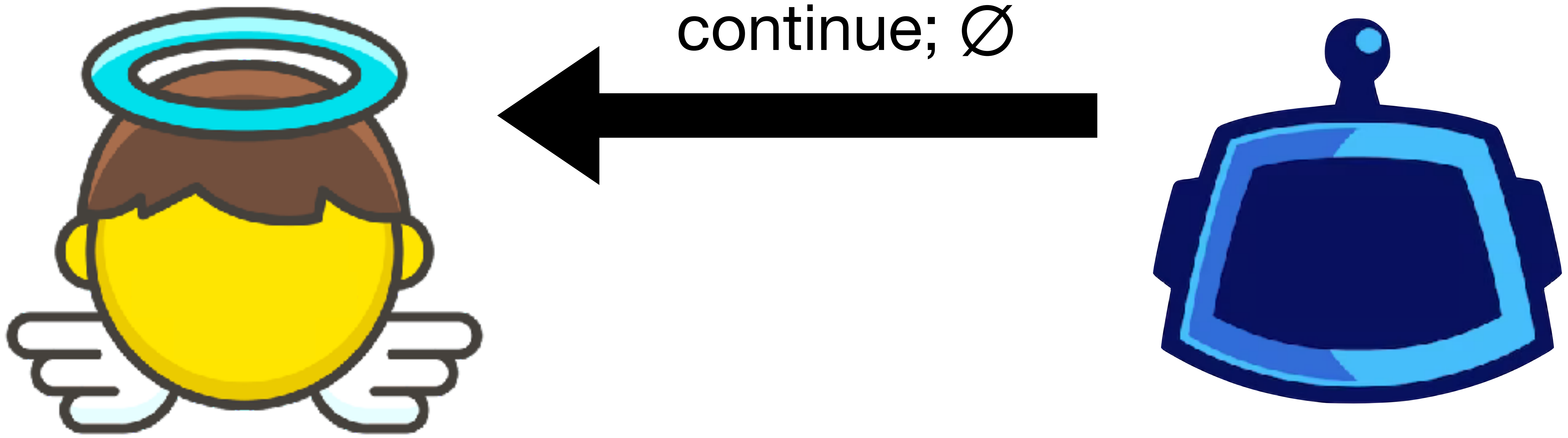
$b_0 \oplus b_1$

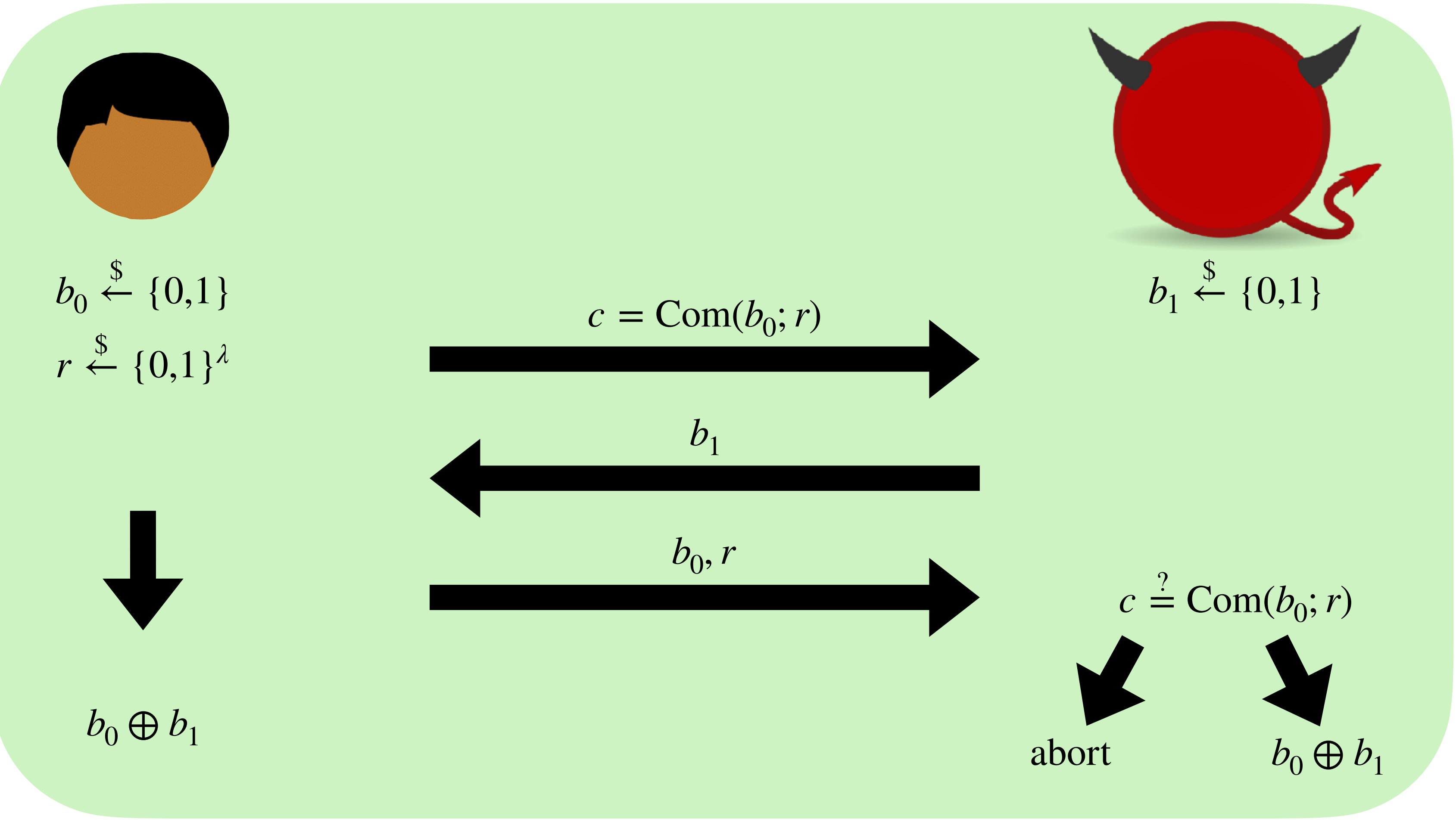
($b_1 = 0$ if Alice aborts)

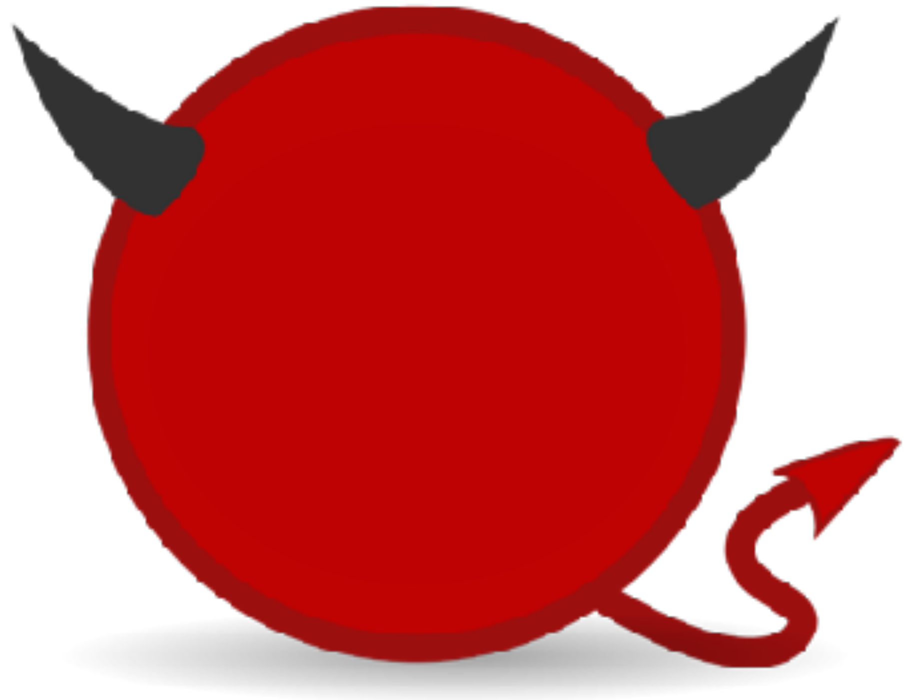
abort

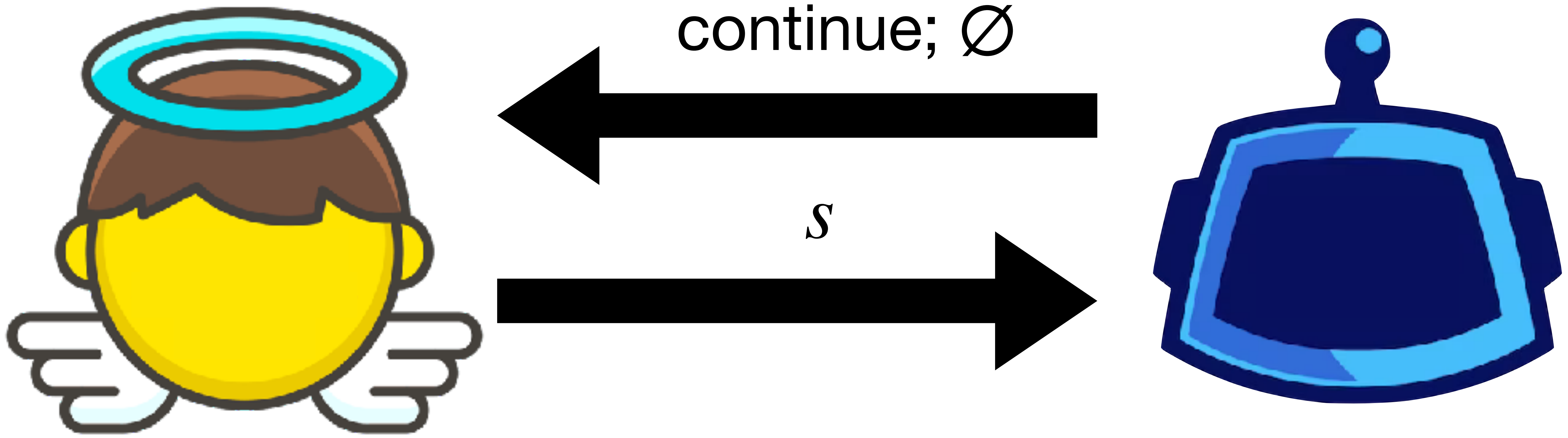
$b_0 \oplus b_1$


$b_0 \xleftarrow{\$} \{0,1\}$
$r \xleftarrow{\$} \{0,1\}^\lambda$
$c = \mathrm{Com}(b_0; r)$
$b_1 \xleftarrow{\$} \{0,1\}$
b_1
b_0, r
$c \stackrel{?}{=} \mathrm{Com}(b_0; r)$
abort
$b_0 \oplus b_1$
$b_0 \oplus b_1$

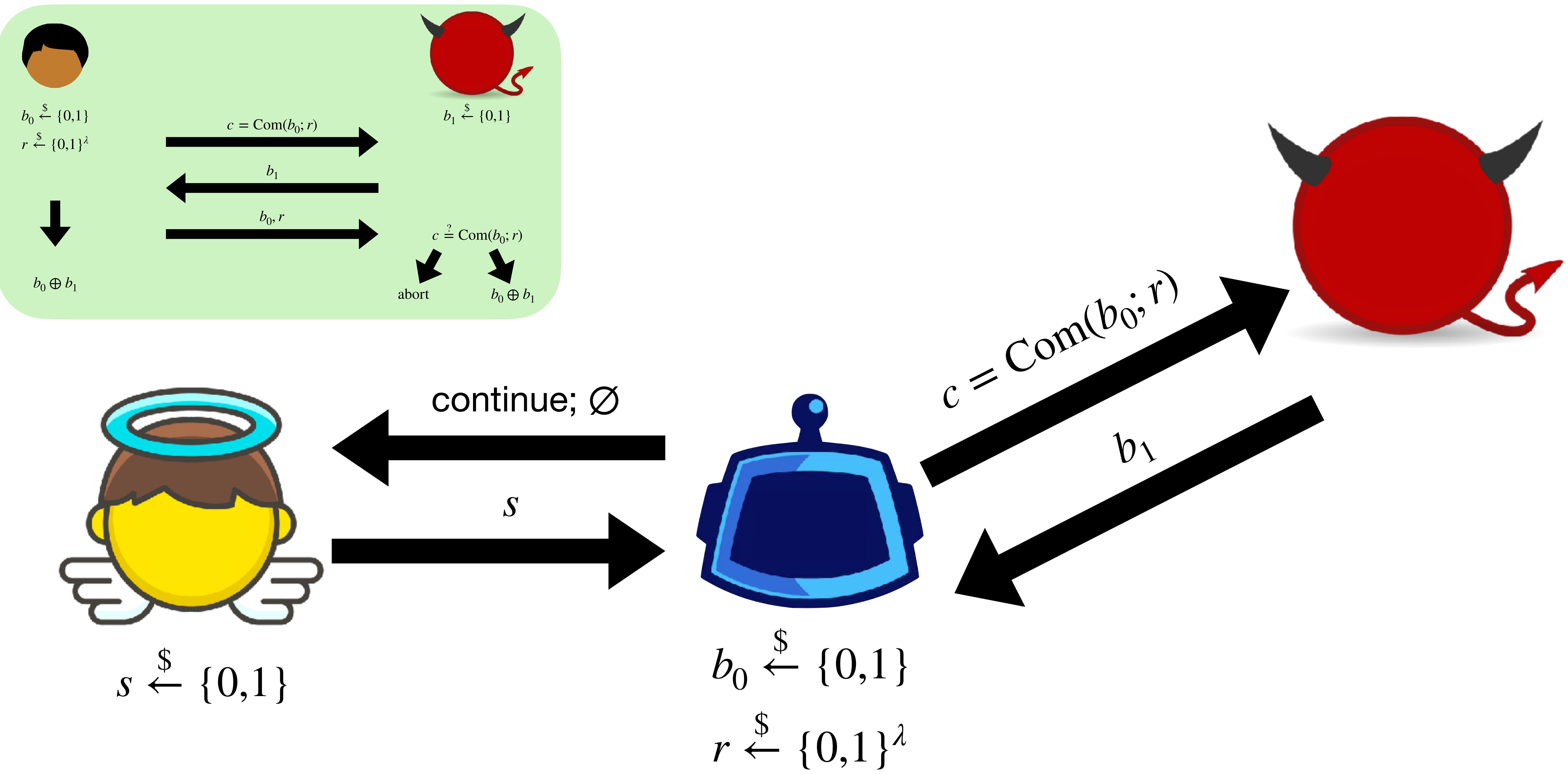

continue; $\varnothing$



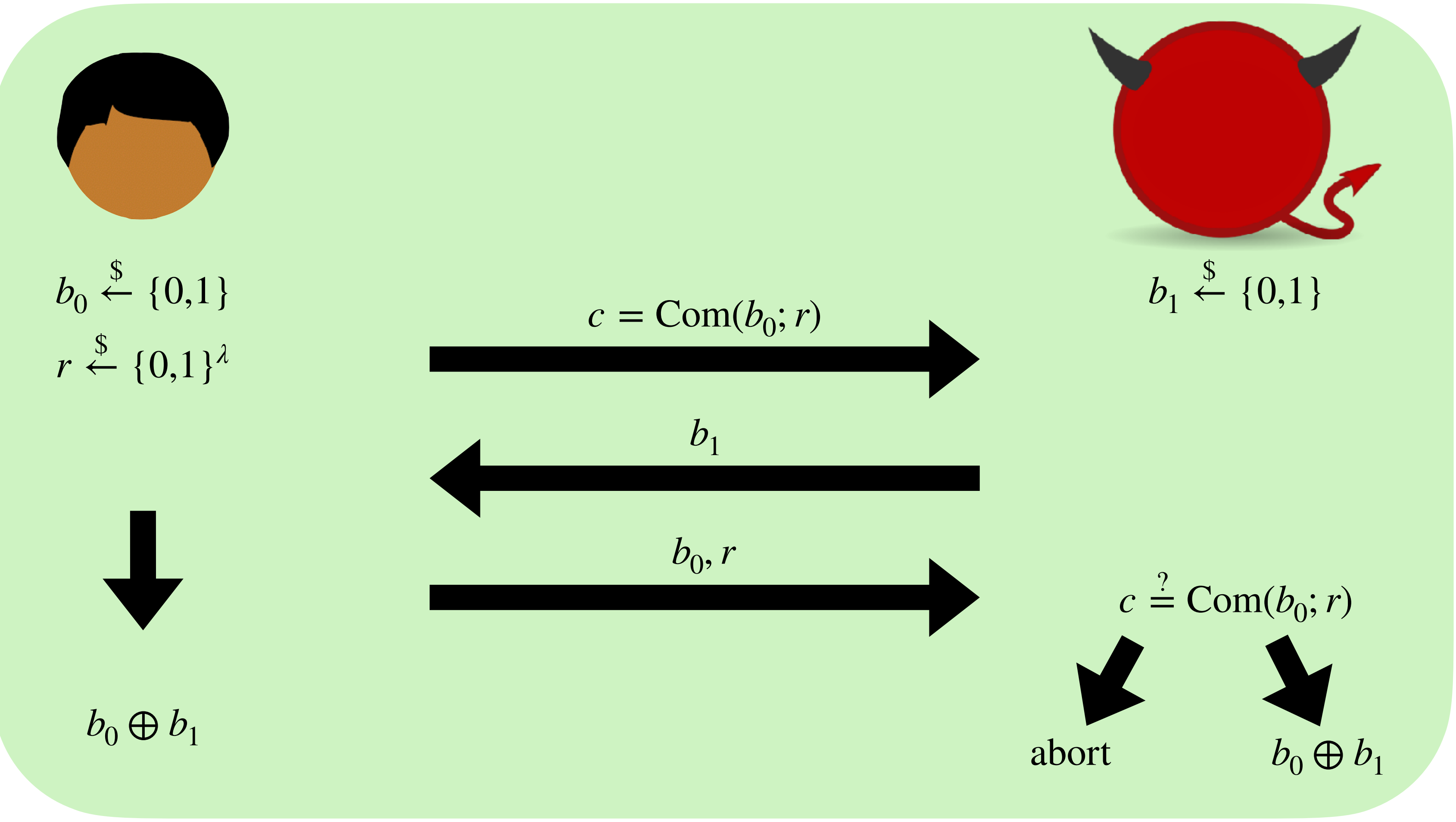




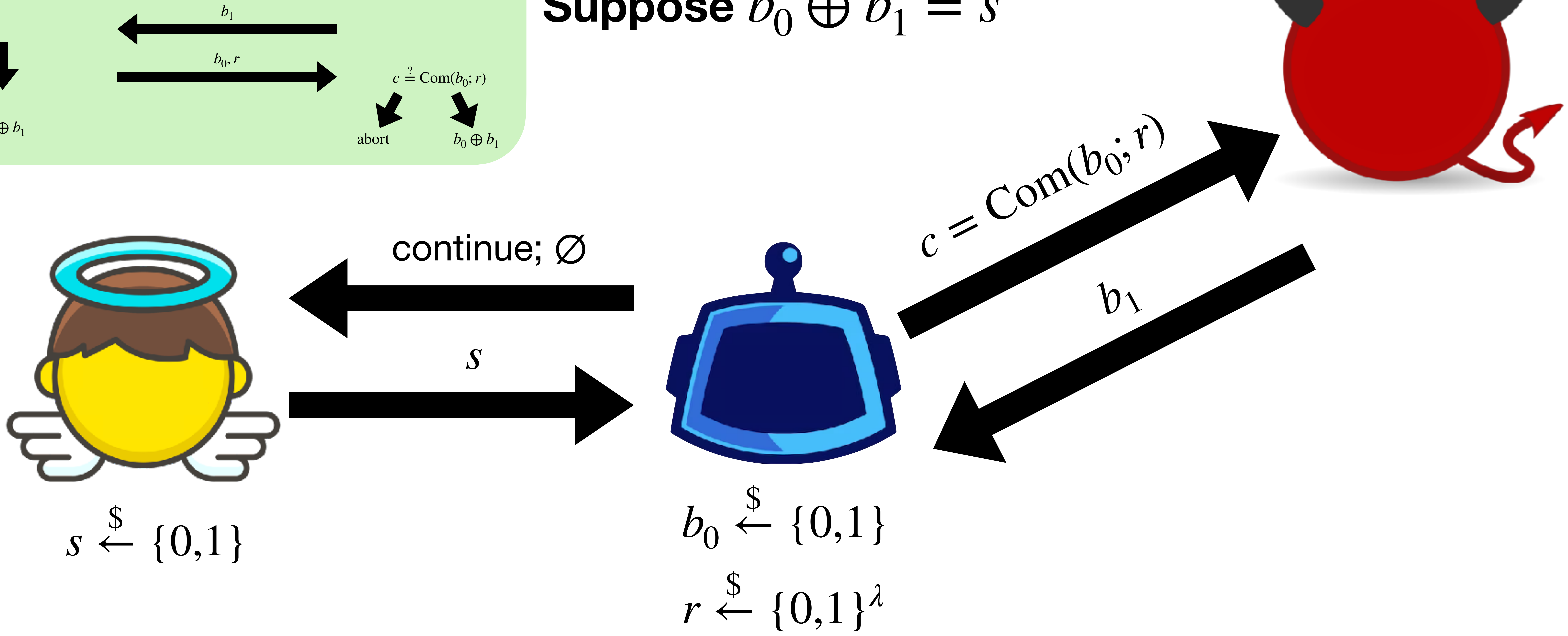
$s \xleftarrow{\$} \{0,1\}$



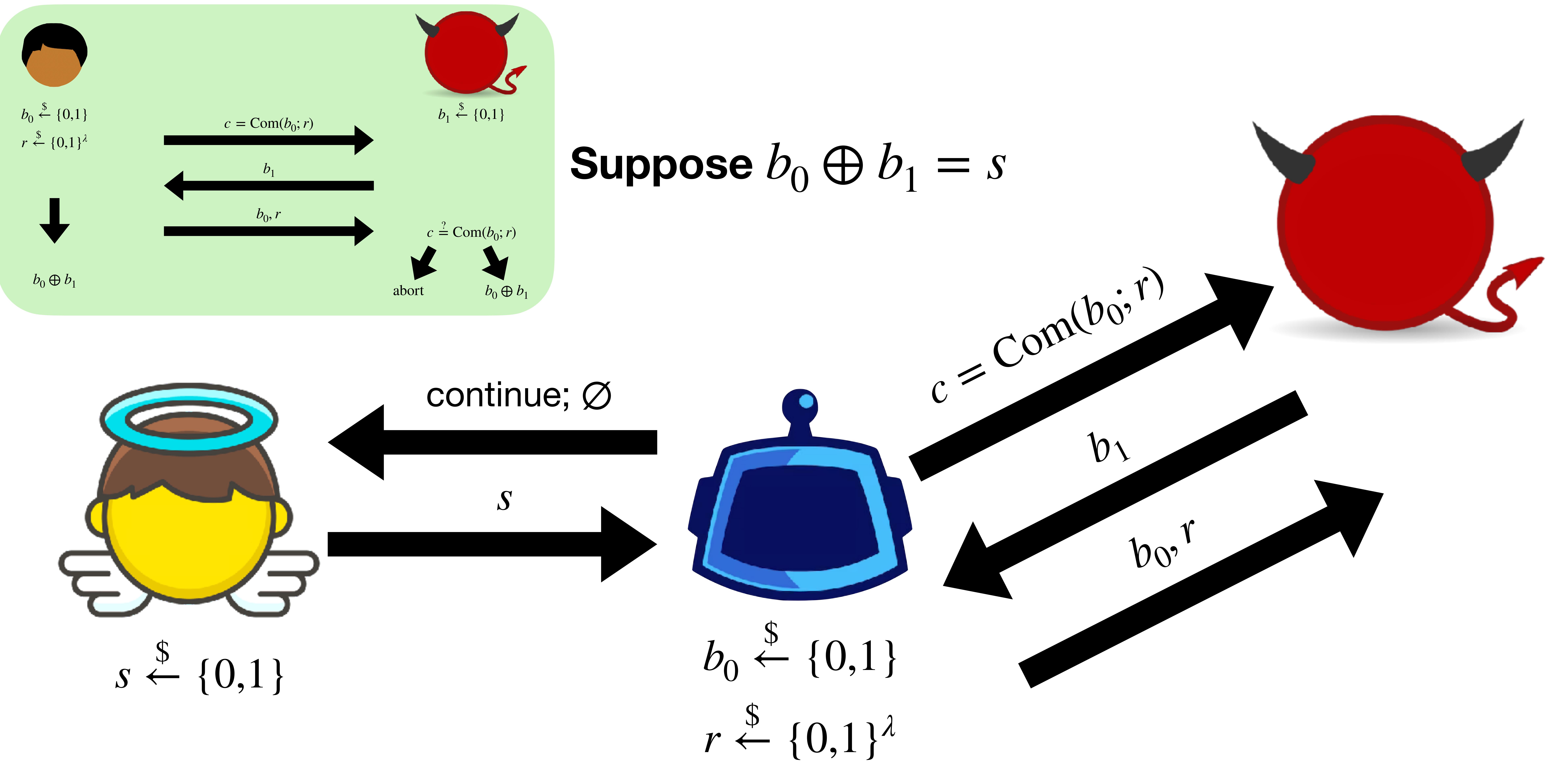
$b_0 \xleftarrow{\$} \{0,1\}$
$r \xleftarrow{\$} \{0,1\}^\lambda$
$c = \mathrm{Com}(b_0; r)$
$b_1 \xleftarrow{\$} \{0,1\}$
b_1
b_0, r
$c \stackrel{?}{=} \mathrm{Com}(b_0; r)$
$b_0 \oplus b_1$
abort
$b_0 \oplus b_1$
continue; ∅
s
$c = \mathrm{Com}(b_0; r)$
b_1
$s \xleftarrow{\$} \{0,1\}$
$b_0 \xleftarrow{\$} \{0,1\}$
$r \xleftarrow{\$} \{0,1\}^\lambda$

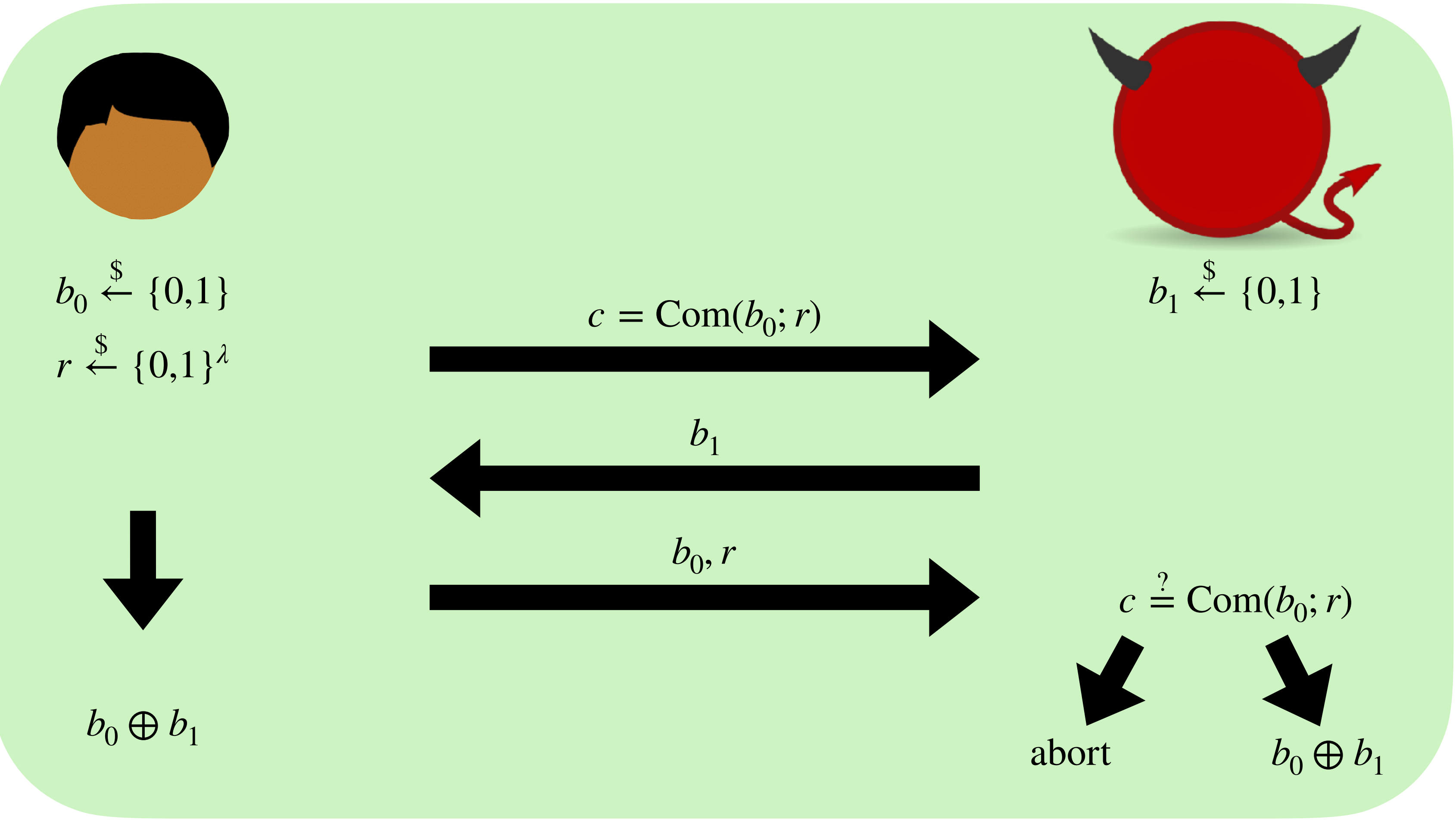


Suppose $b_0 \oplus b_1 = s$

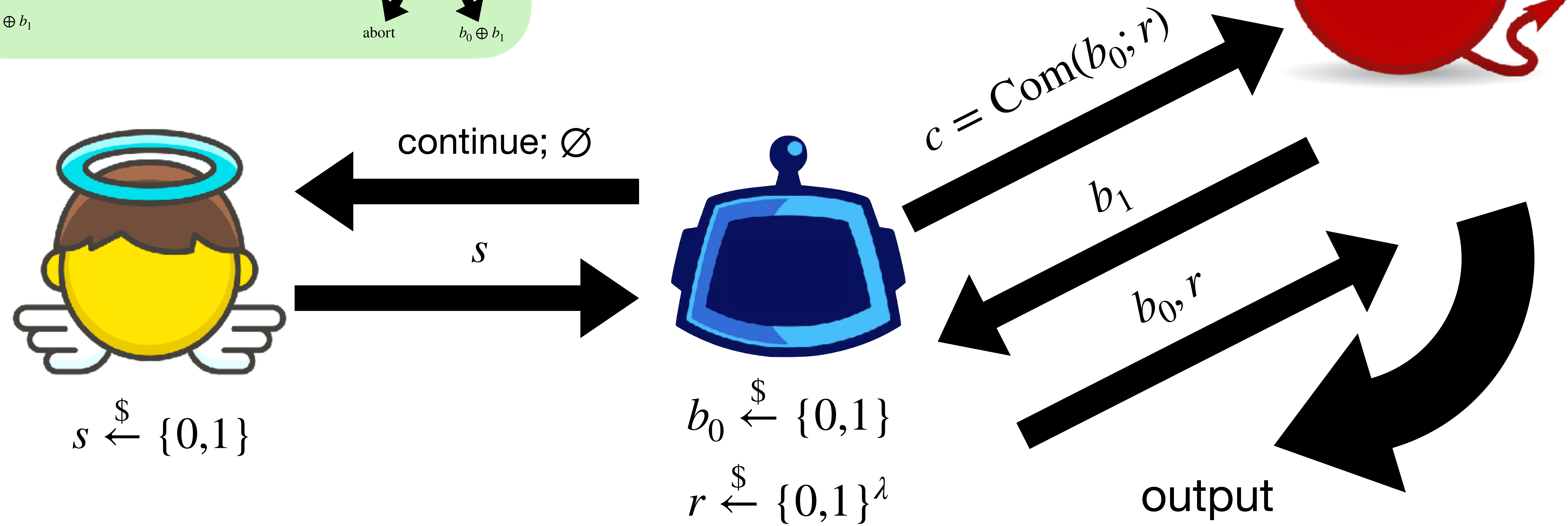


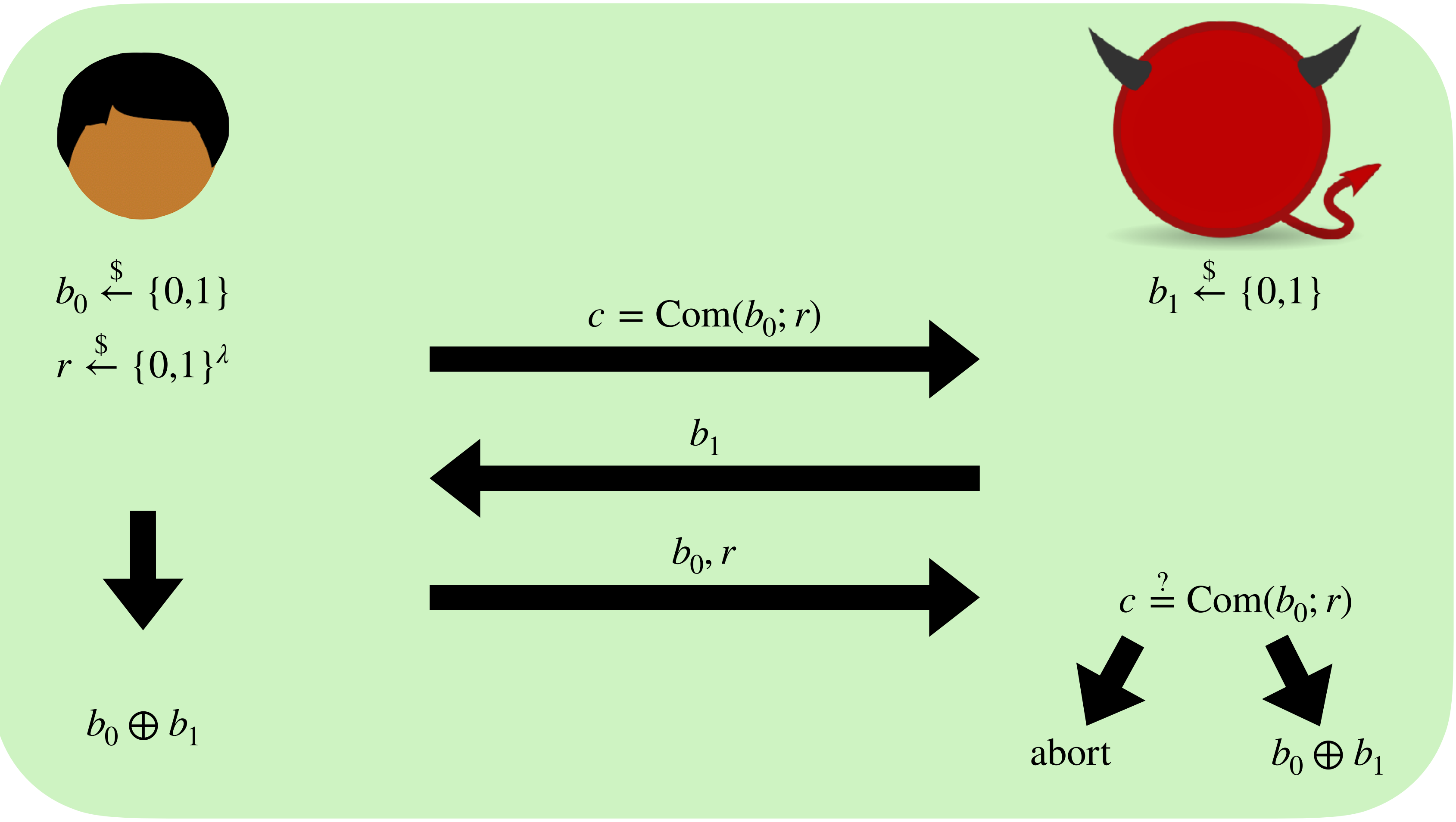
Suppose $b_0 \oplus b_1 = s$



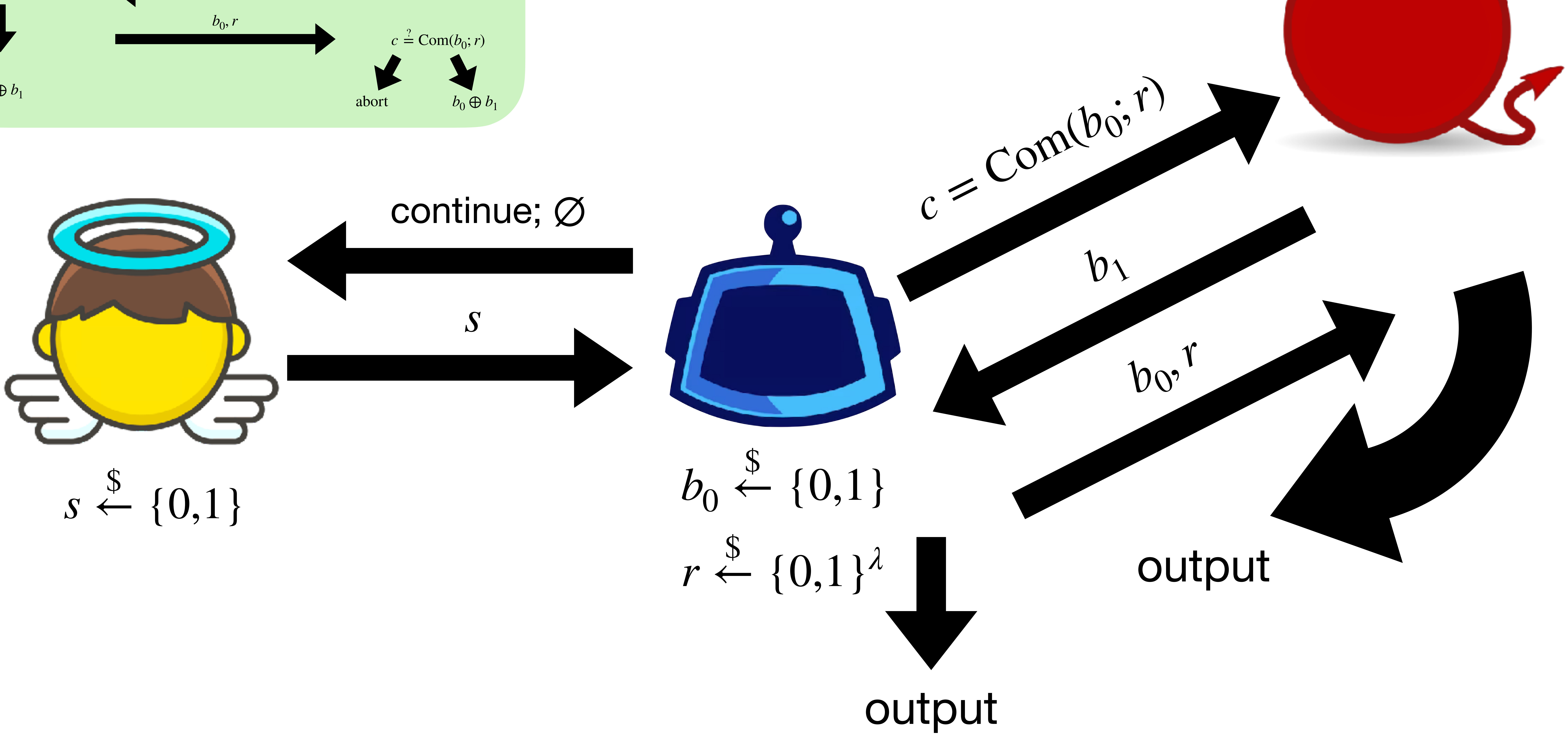


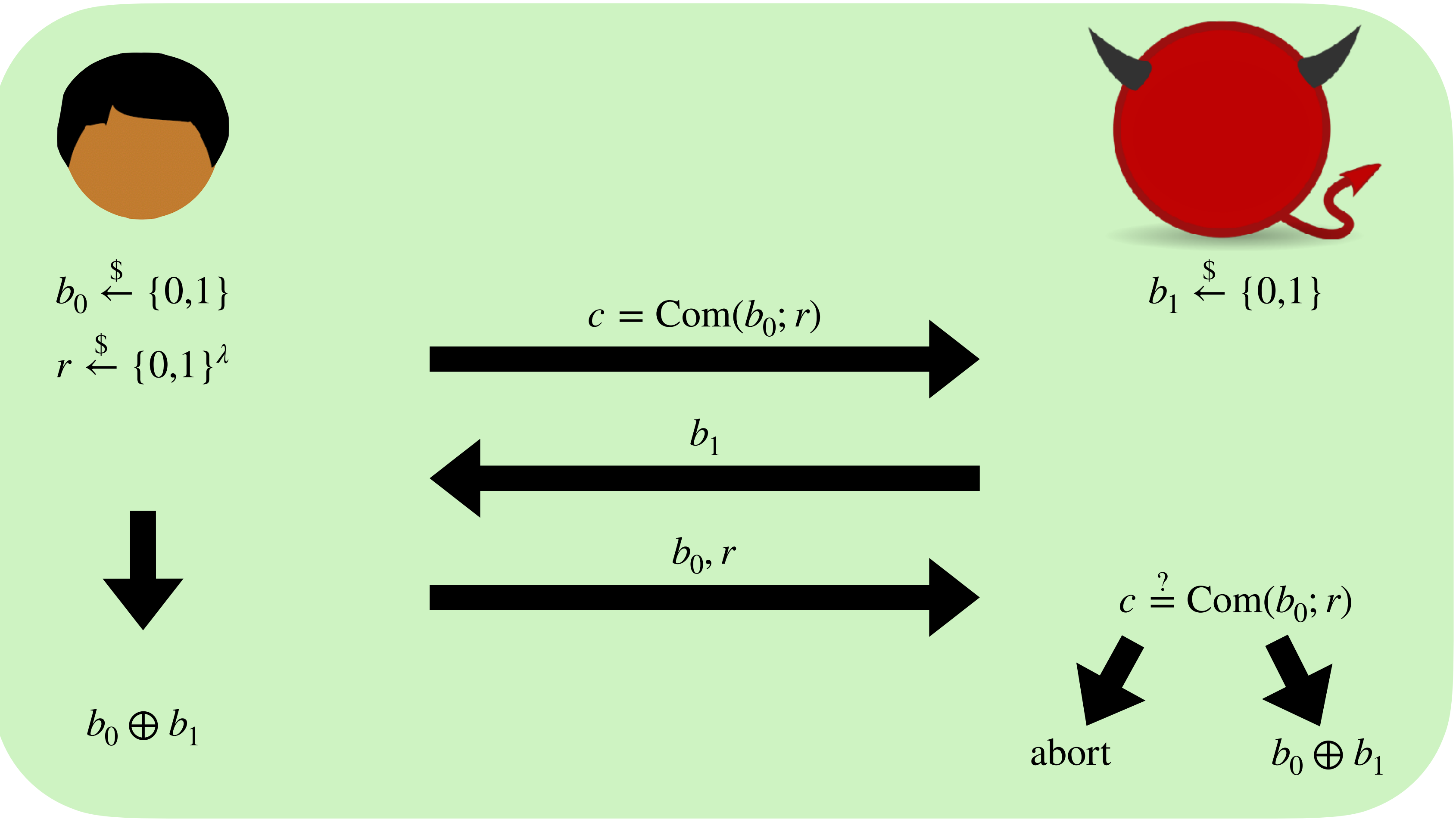
Suppose $b_0 \oplus b_1 = s$



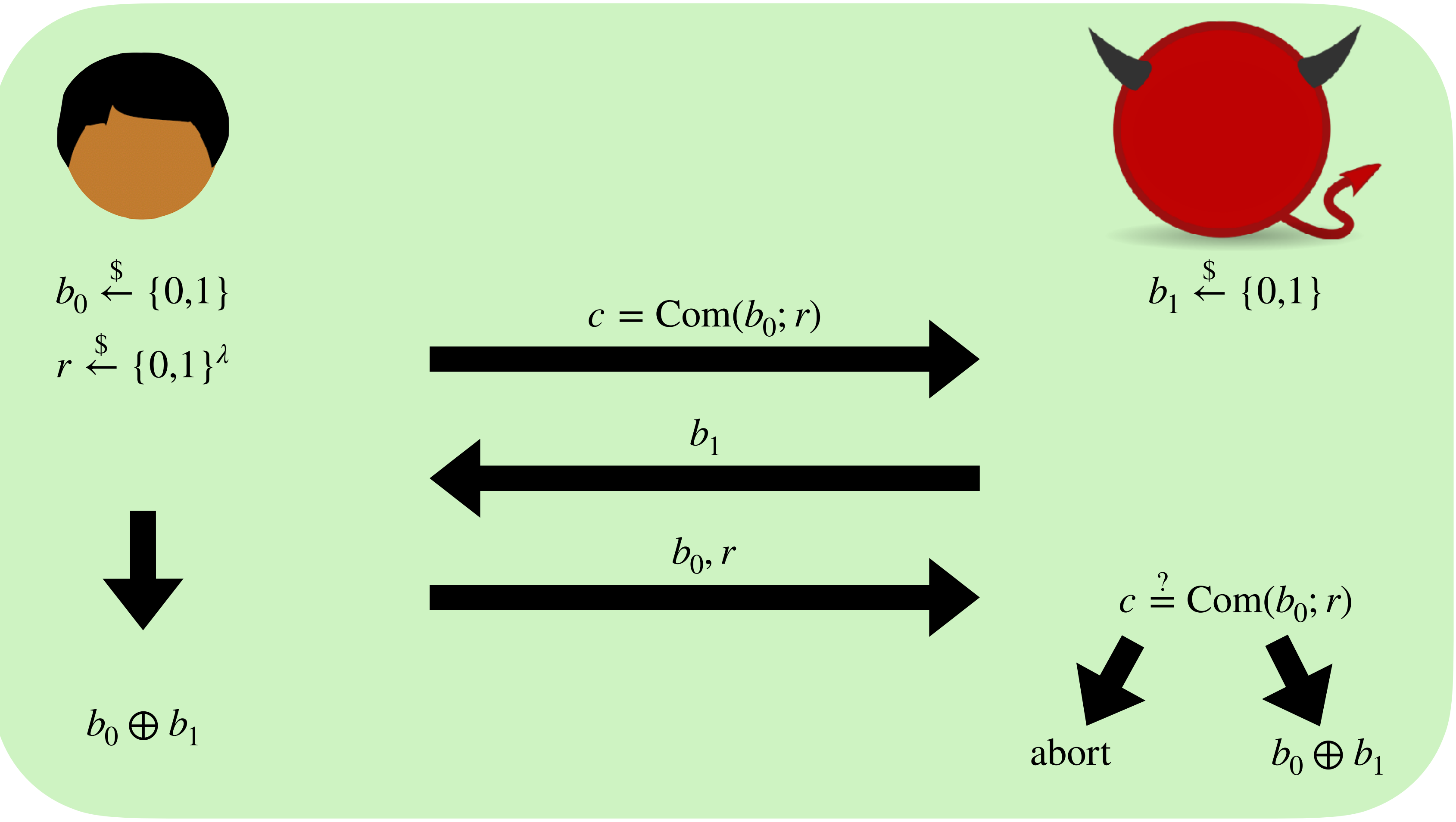


Suppose $b_0 \oplus b_1 = s$

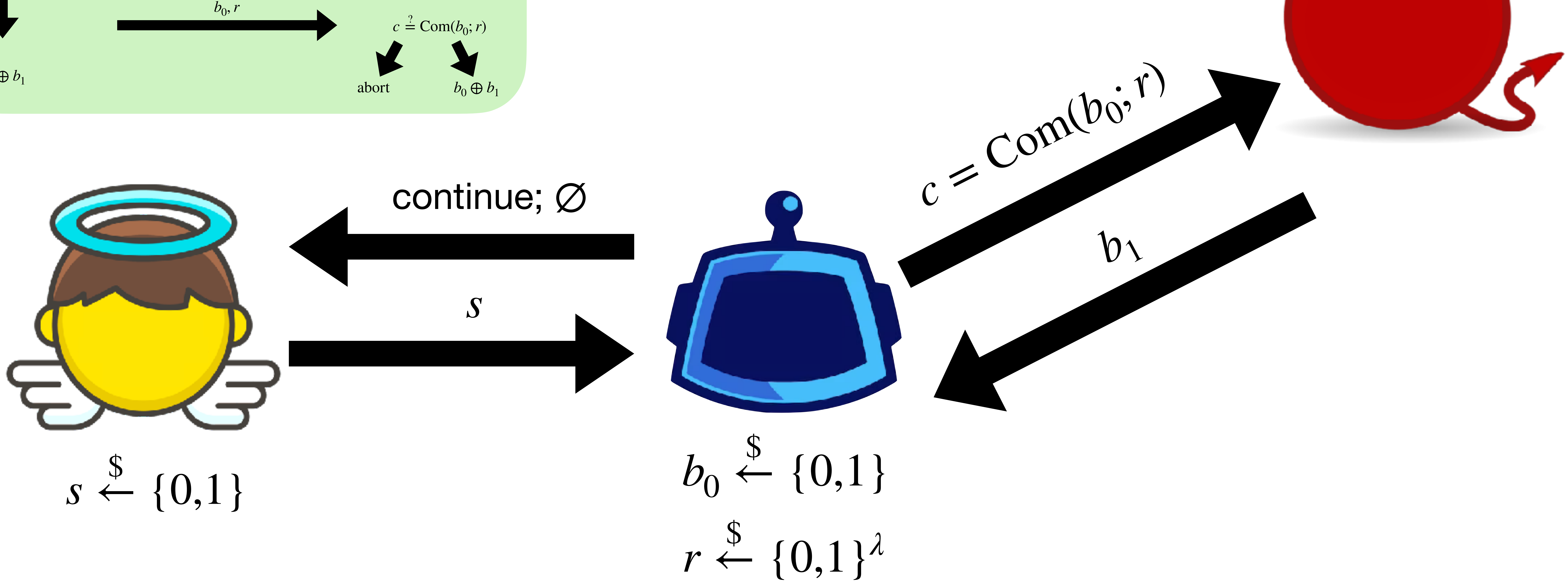


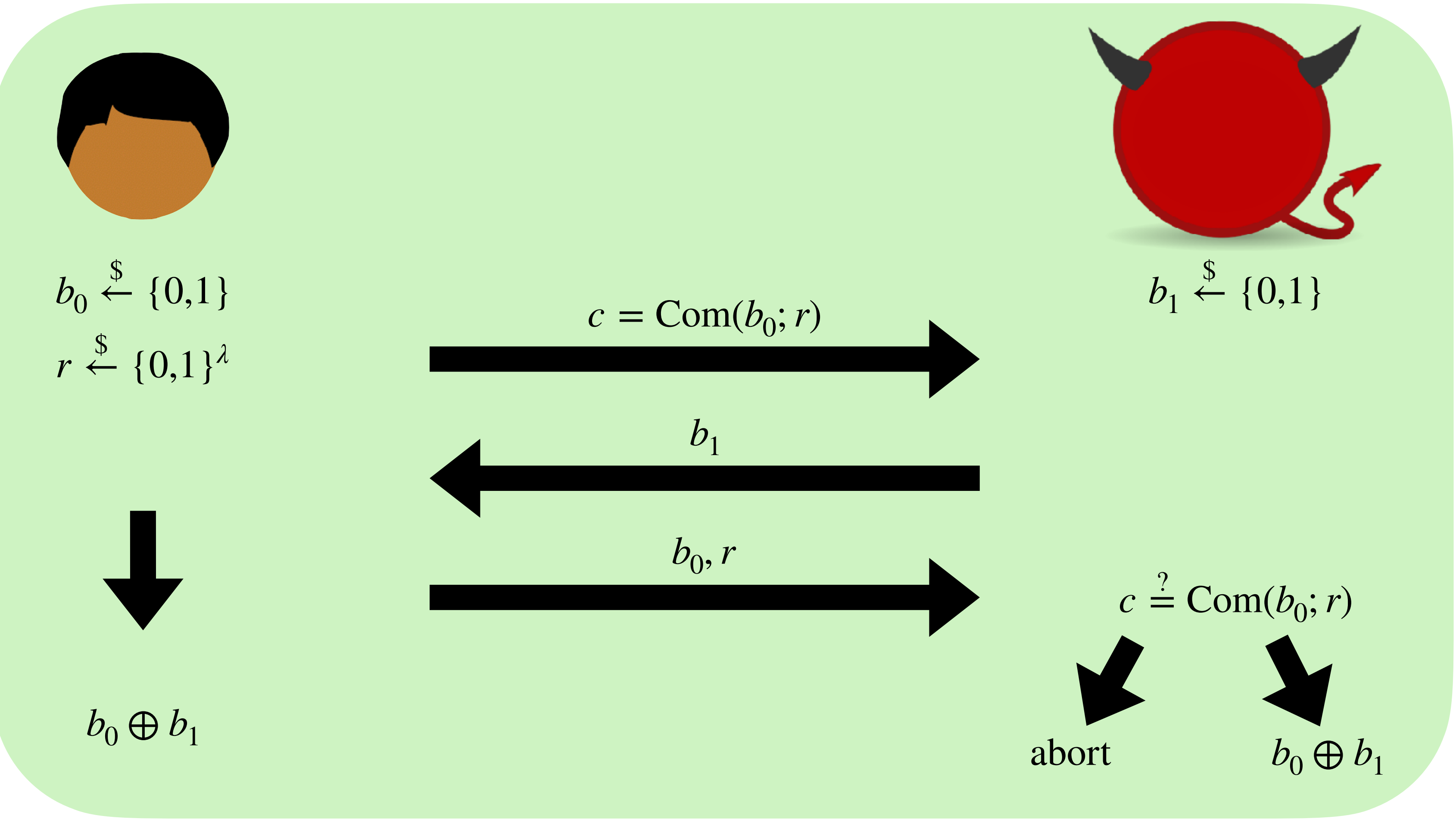


Suppose $b_0 \oplus b_1 = s$

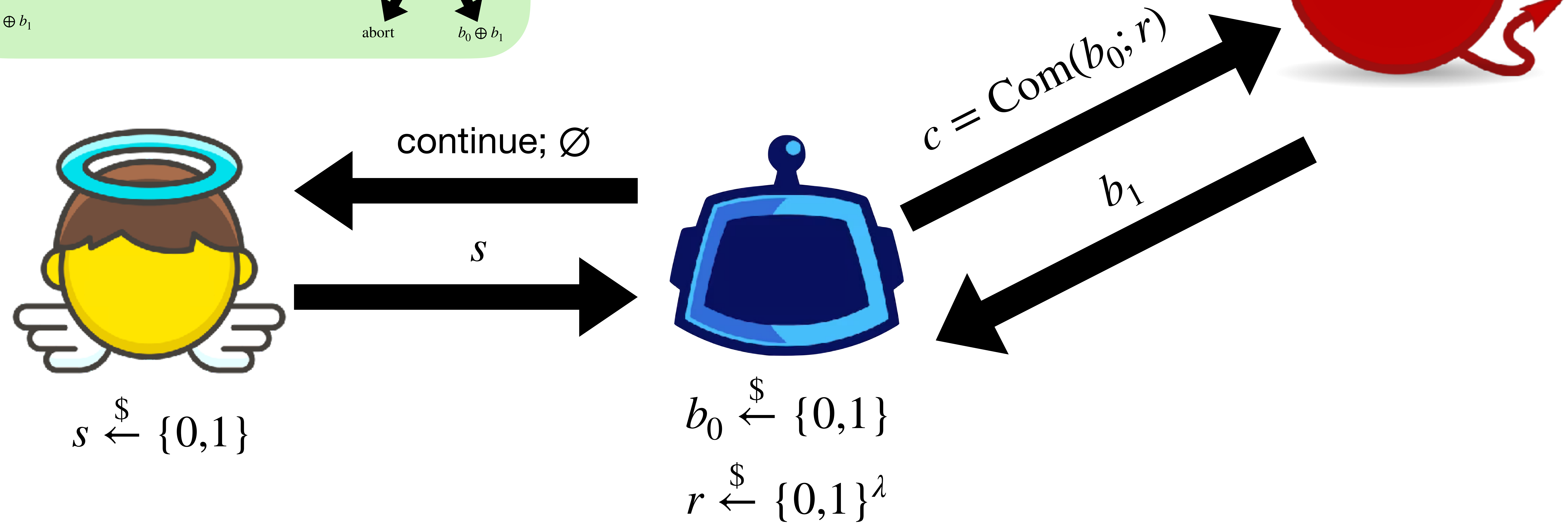


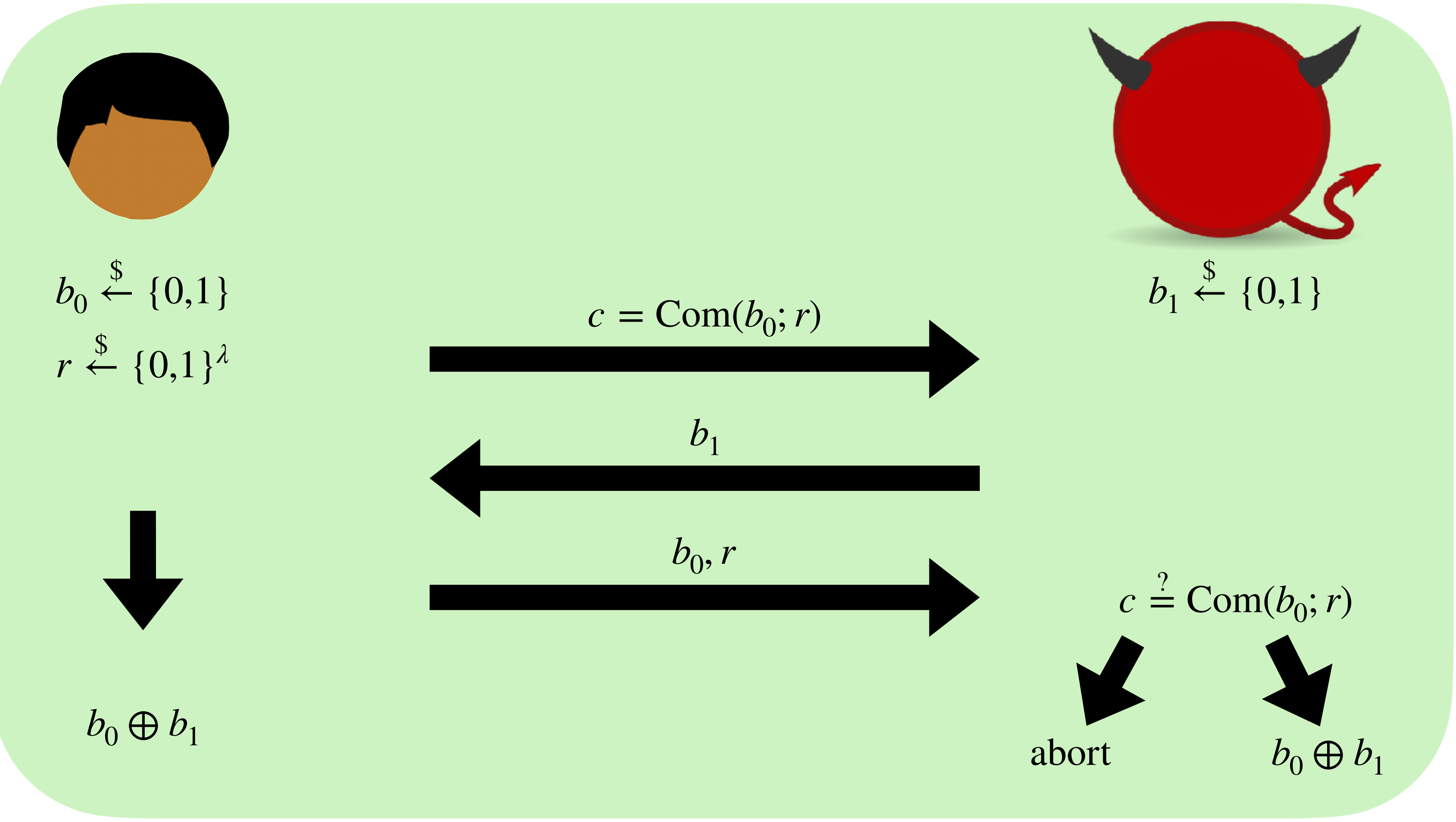
What if $b_0 \oplus b_1 \neq s$?





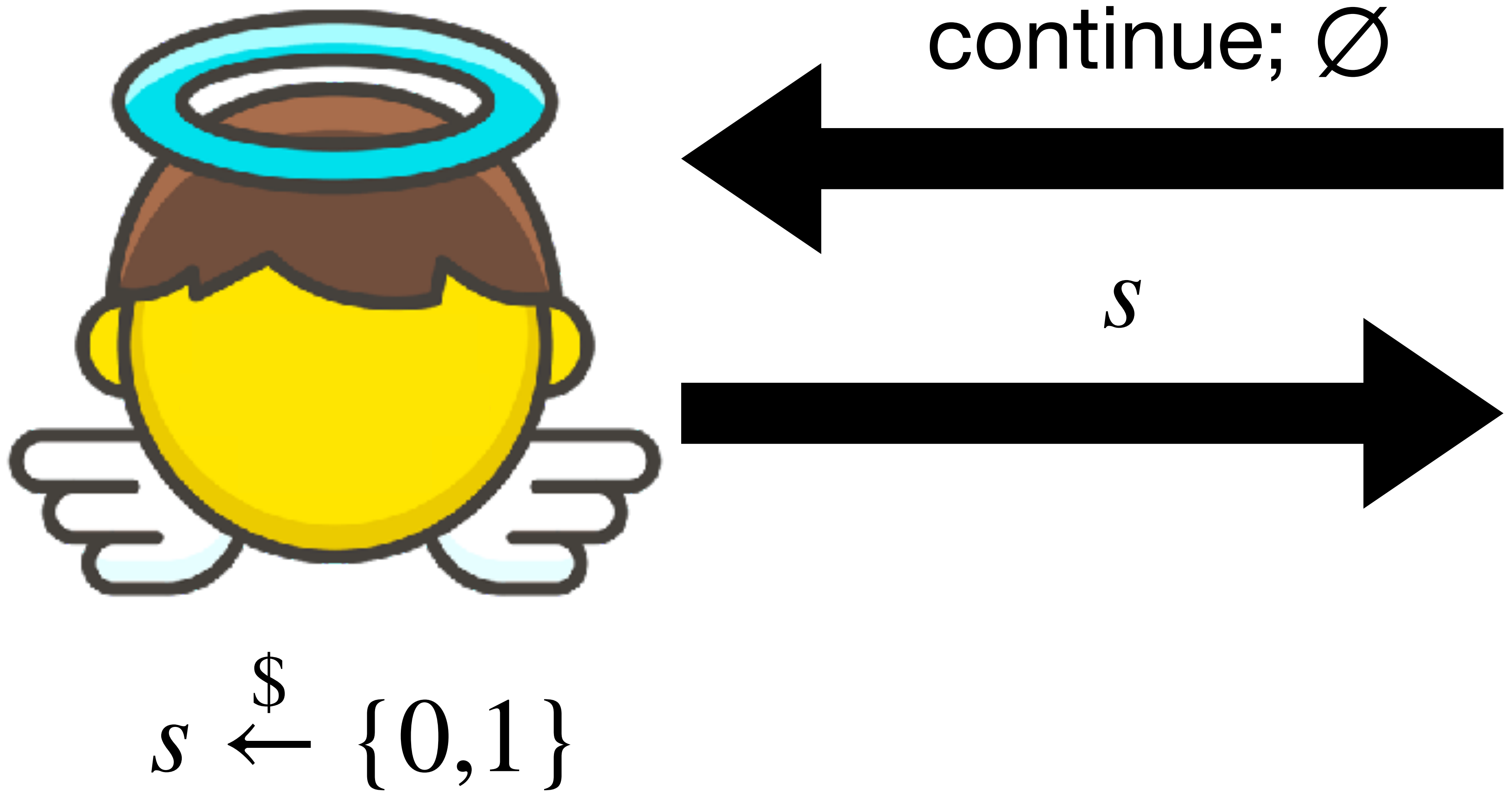
What if $b_0 \oplus b_1 \neq s$?

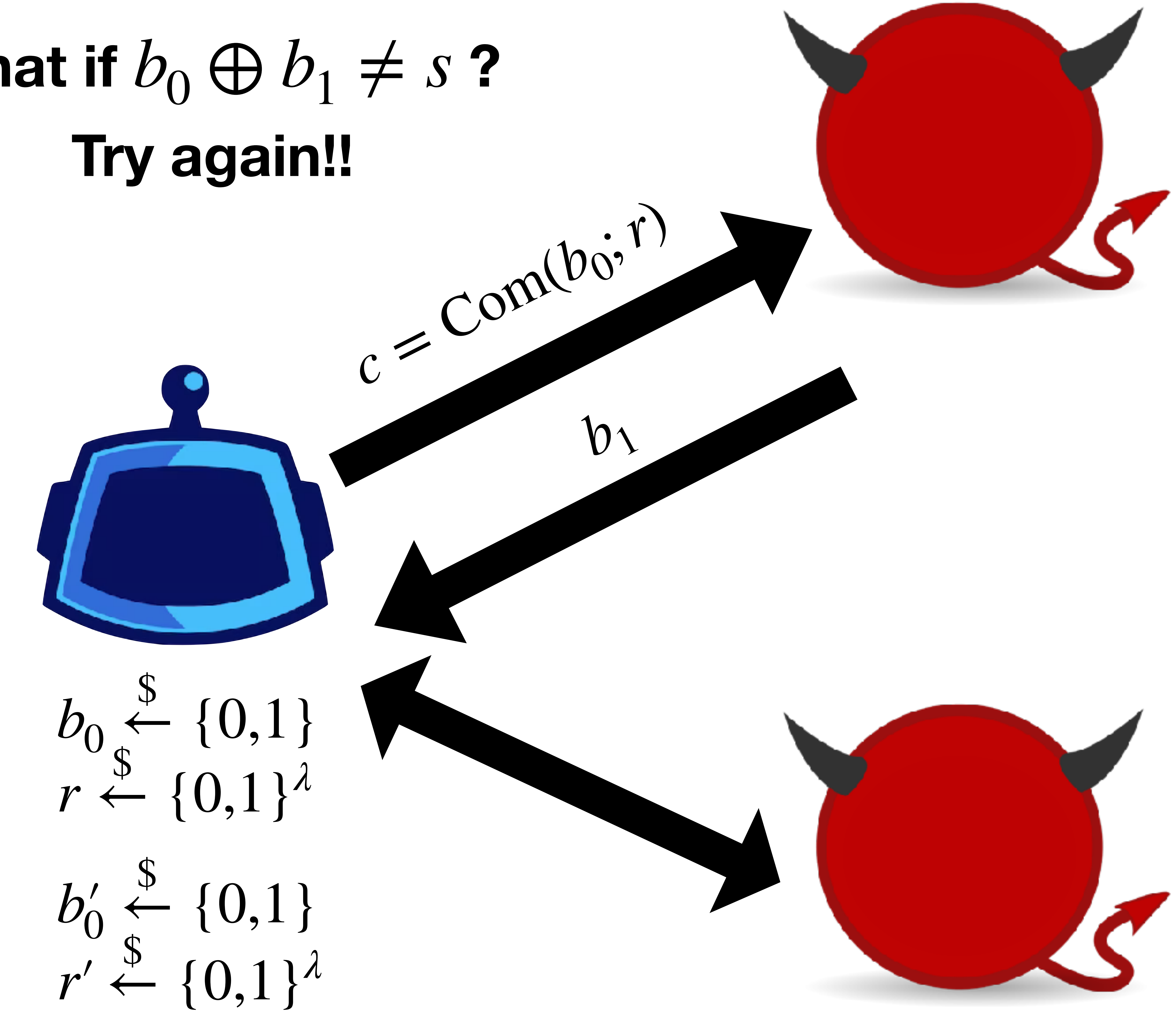




What if $b_0 \oplus b_1 \neq s$?

Try again!!





Today's objectives

Review malicious security (with abort)

Discuss commitments

Understand "rewinding" in simulation proofs

See a proof for a (slightly) less contrived protocol