Black-Box Constructions of Concurrently Secure Protocols

Huijia (Rachel) Lin
MIT & BU

Rafael Pass
Cornell

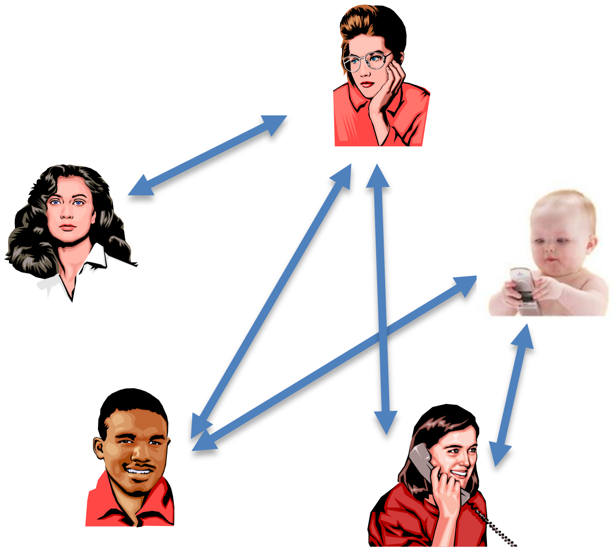
Secure MPC

Secure MPC

Goal: Allow a set of distrustful parties to compute ANY function f on their own

Secure MPC

Goal: Allow a set of distrustful parties to compute ANY function *f* on their own



Secure MPC

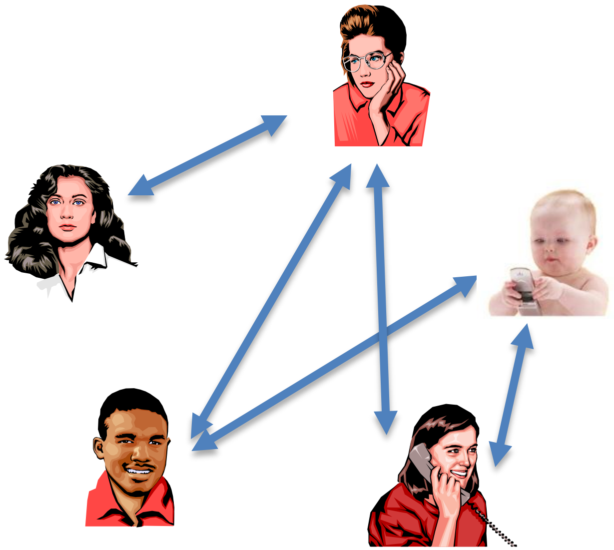
Goal: Allow a set of distrustful parties to compute ANY function *f* on their own

Correctness

What to get---the outputs

Privacy

What to hide---the private inputs



Secure MPC

Goal: Allow a set of distrustful parties to compute ANY function *f* on their own

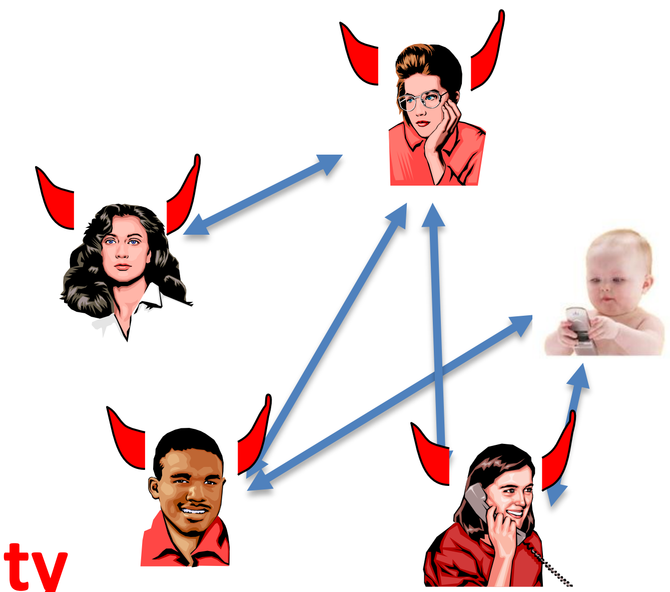
Correctness

What to get---the outputs

Privacy

What to hide---the private inputs

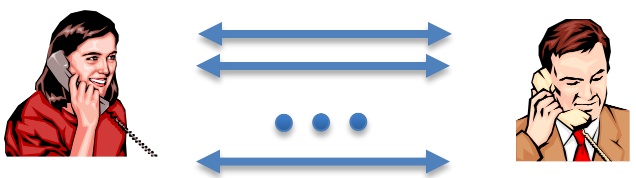
Even when no honest majority

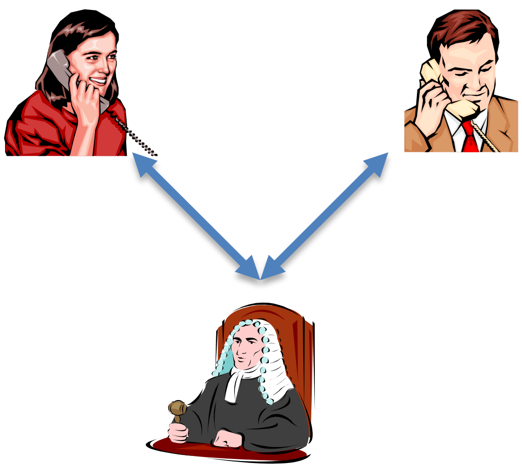


Simulation Paradigm

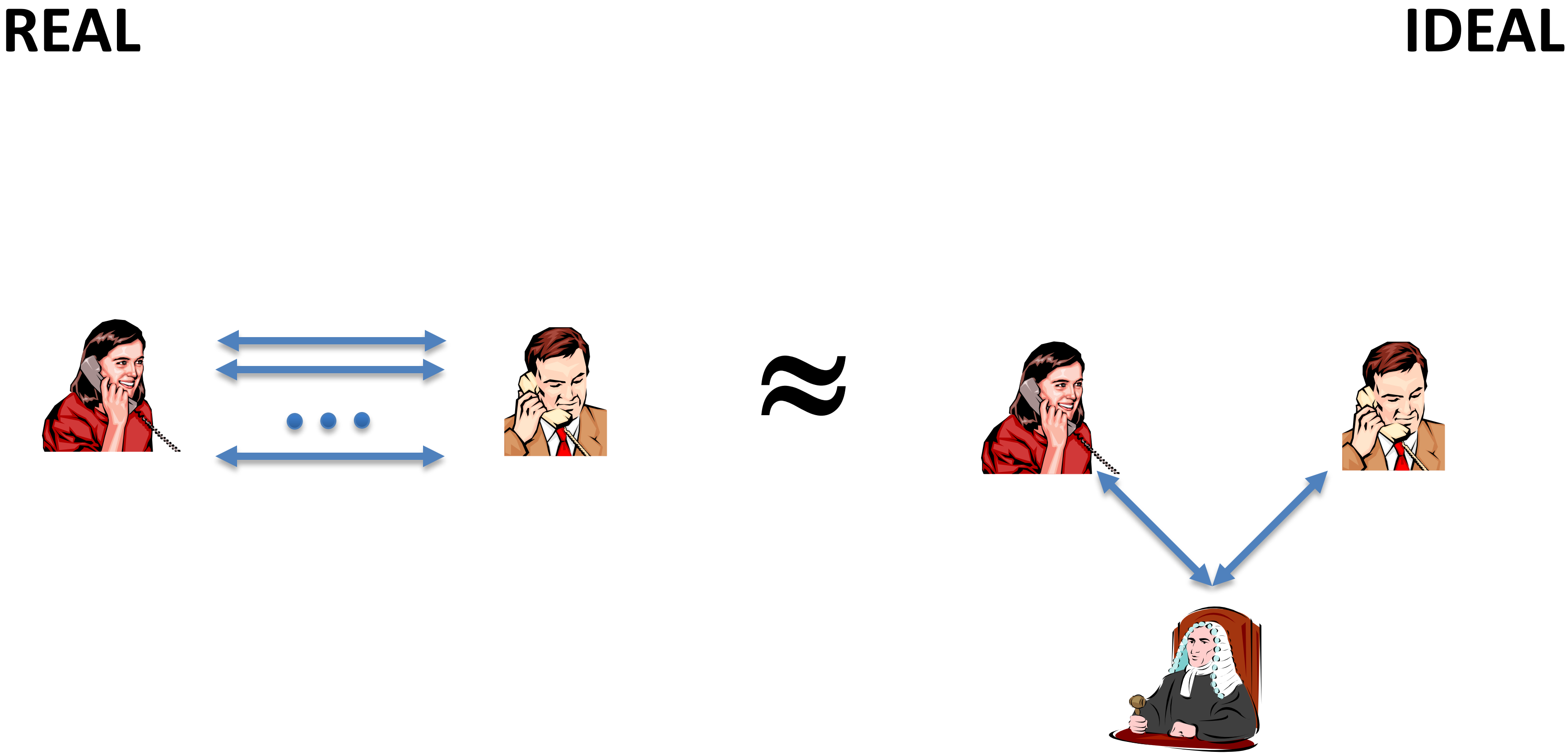
REAL

IDEAL





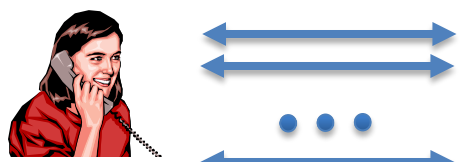
Simulation Paradigm



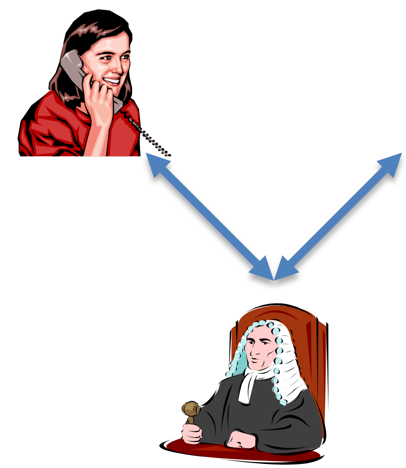
“as correct & private as”

Simulation Paradigm

REAL **IDEAL**



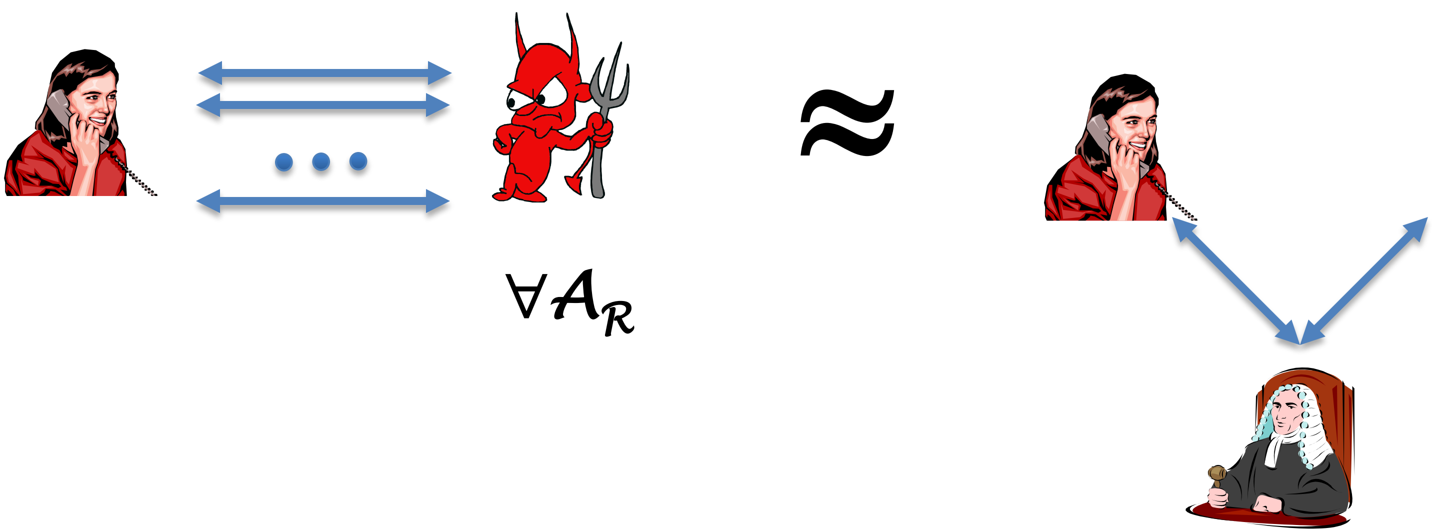
$\approx$



“as correct & private as”

Simulation Paradigm

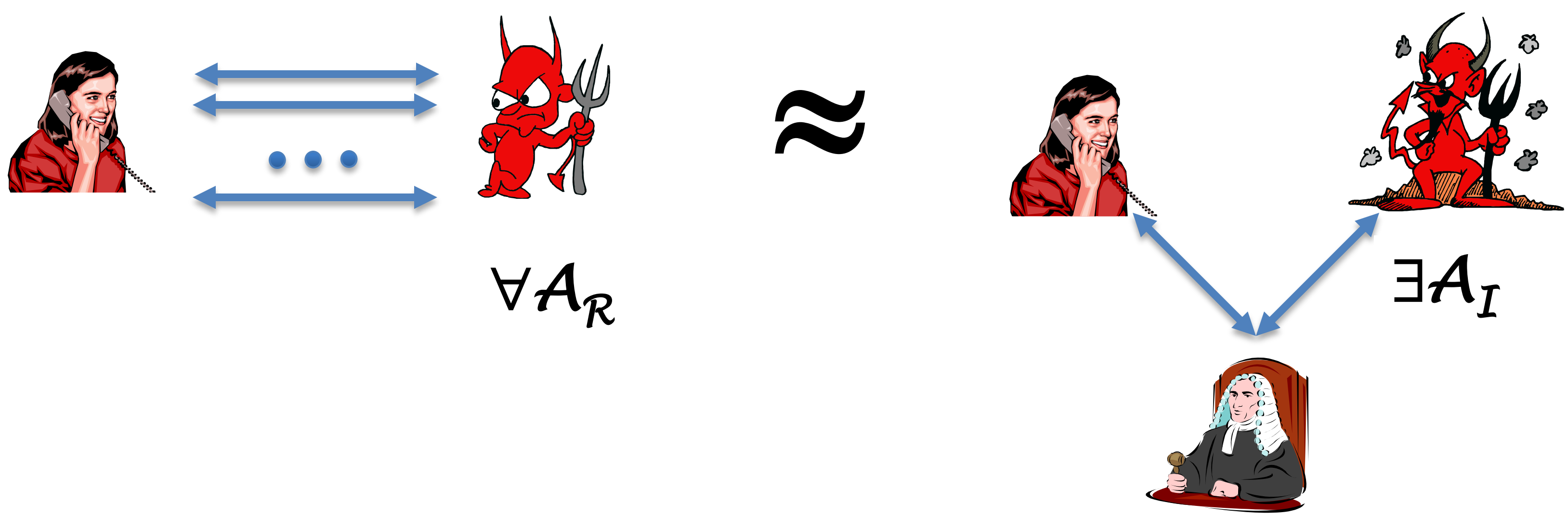
REAL **IDEAL**



“as correct & private as”

Simulation Paradigm

REAL **IDEAL**



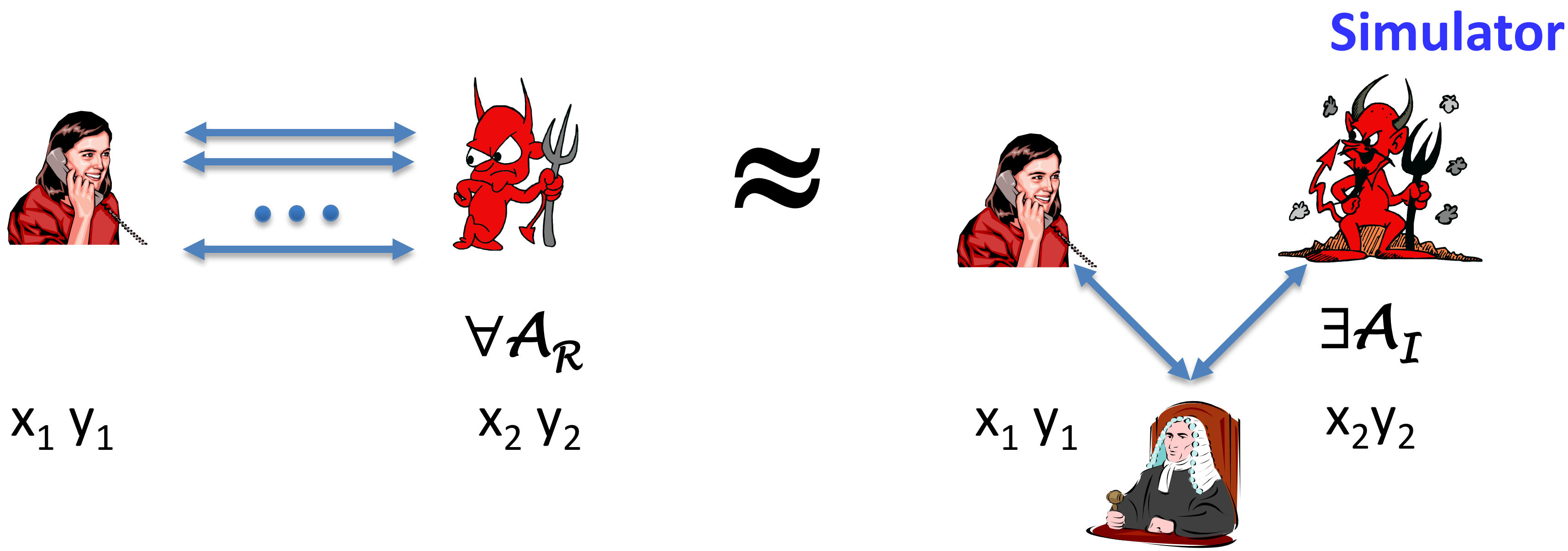
"as correct & private as"

Simulation Paradigm

"as correct & private as"

Simulation Paradigm

REAL **IDEAL**

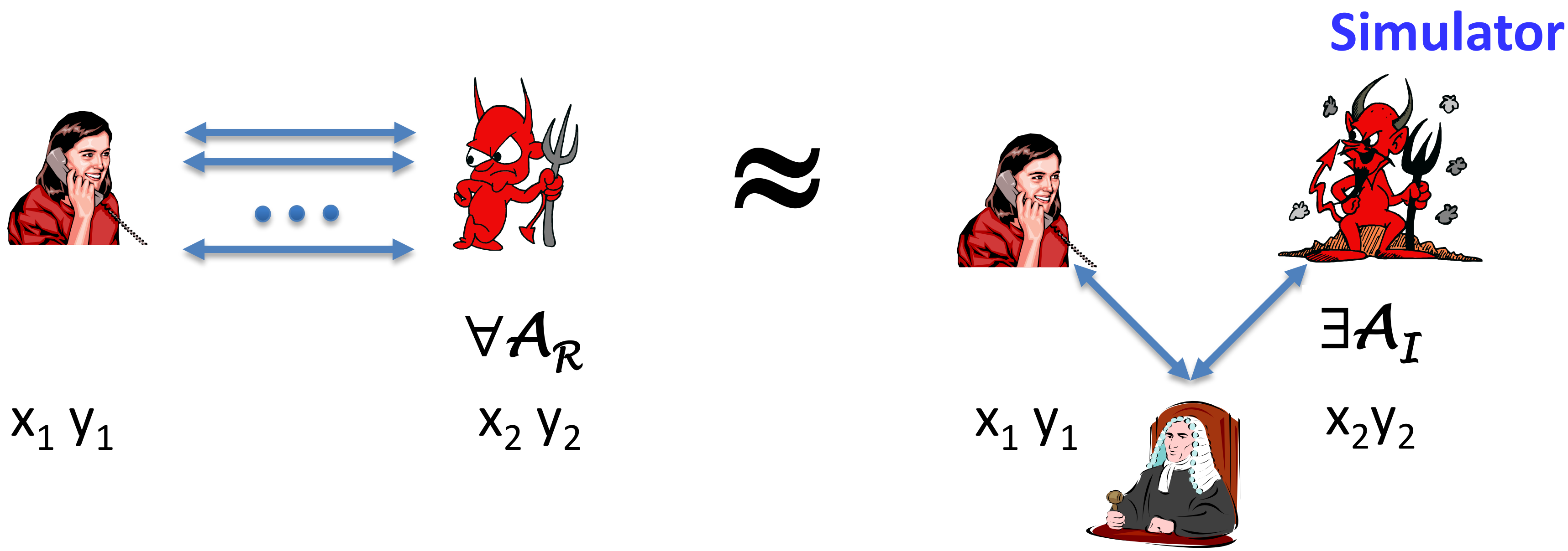


“as correct & private as”

Correctness: The output of every player in ideal is the same as in real

Simulation Paradigm

REAL **IDEAL**

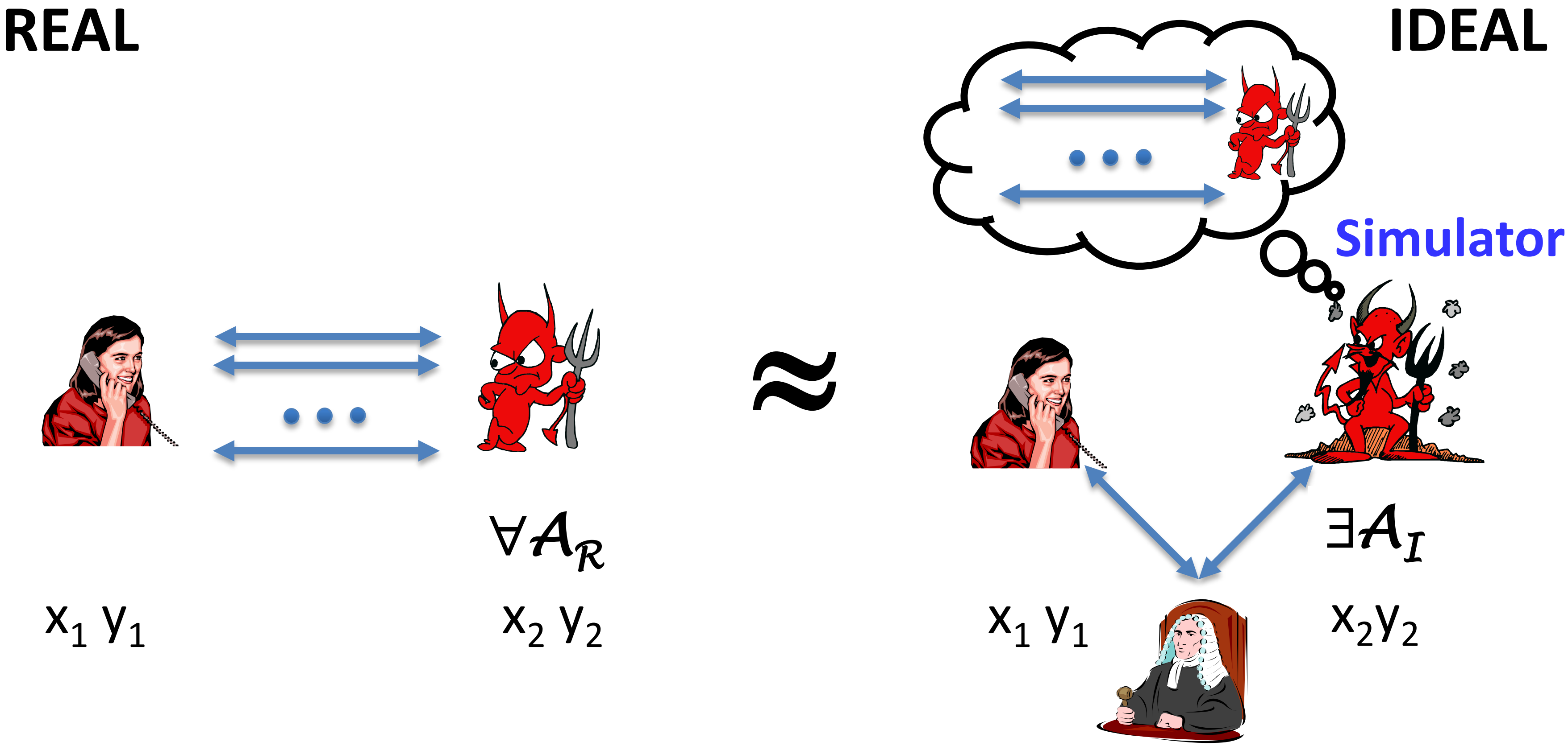


"as correct & private as"

Correctness: The output of every player in ideal is the same as in real

Privacy: The simulator can learn whatever the adv learns

Simulation Paradigm

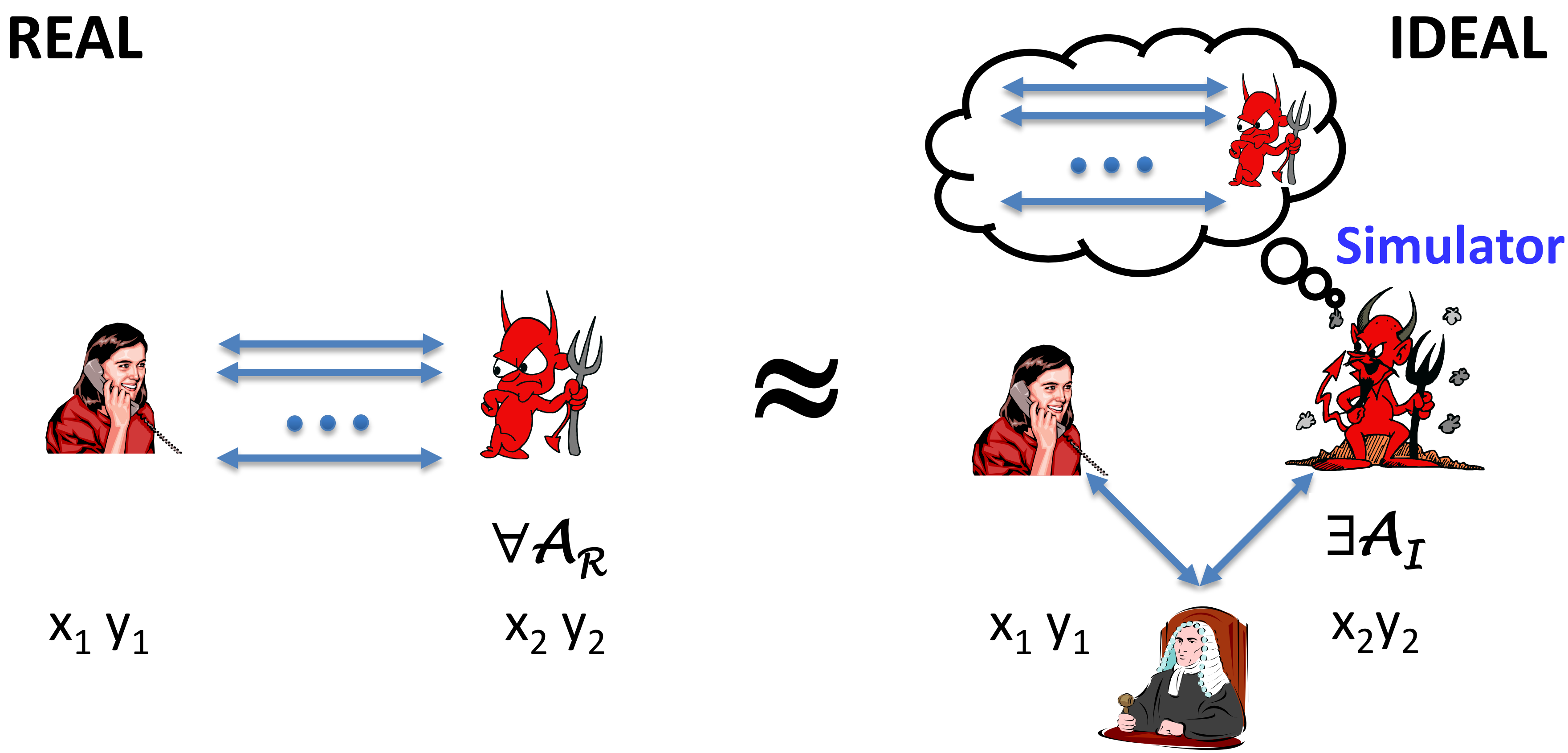


"as correct & private as"

Correctness: The output of every player in ideal is the same as in real

Privacy: The simulator can learn whatever the adv learns

Simulation Paradigm



"as correct & private as"

Correctness: The output of every player in ideal is the same as in real

Privacy: The simulator can learn whatever the adv learns

In this talk, we focus on static malicious corruption

The Concurrent Model

The Concurrent Model

MANY sets of players executing
MANY different protocols all at once
[DDN, DNS, GK, Fe, KPR, RK, CKPR, KP, PRS, C...and many others]

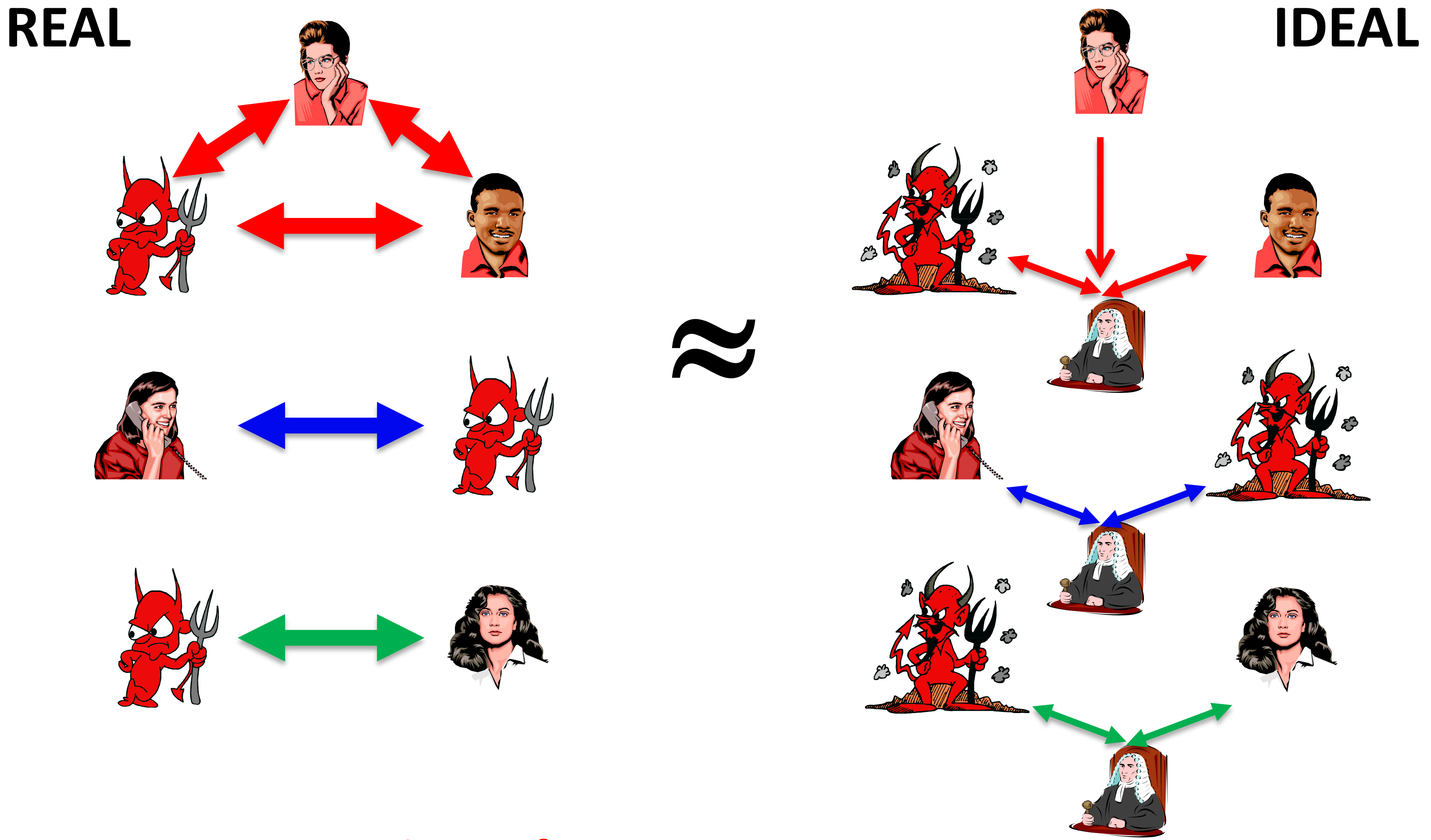
The Concurrent Model

MANY sets of players executing

MANY different protocols all at once

[DDN, DNS, GK, Fe, KPR, RK, CKPR, KP, PRS, C...and many others]

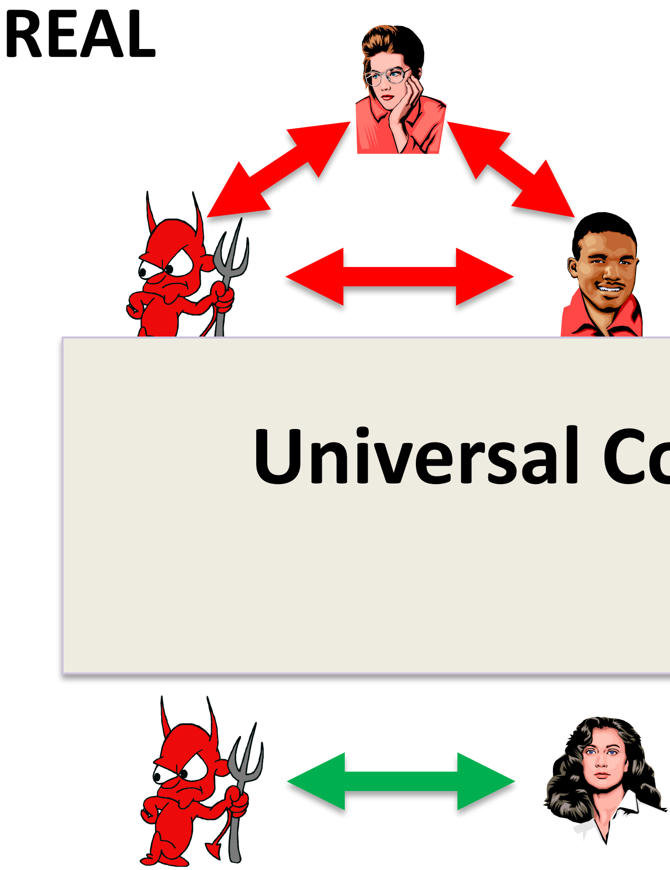
Concurrent Security (informally)

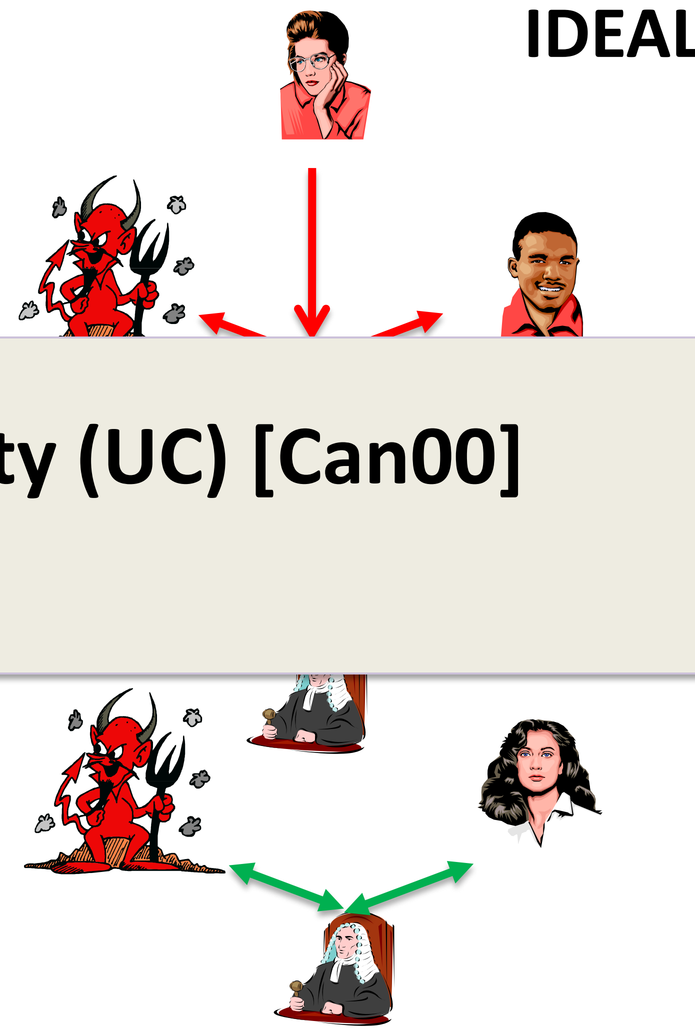


Many executions of different protocols

Many executions with INDEPENDENT trusted parties

Concurrent Security (informally)





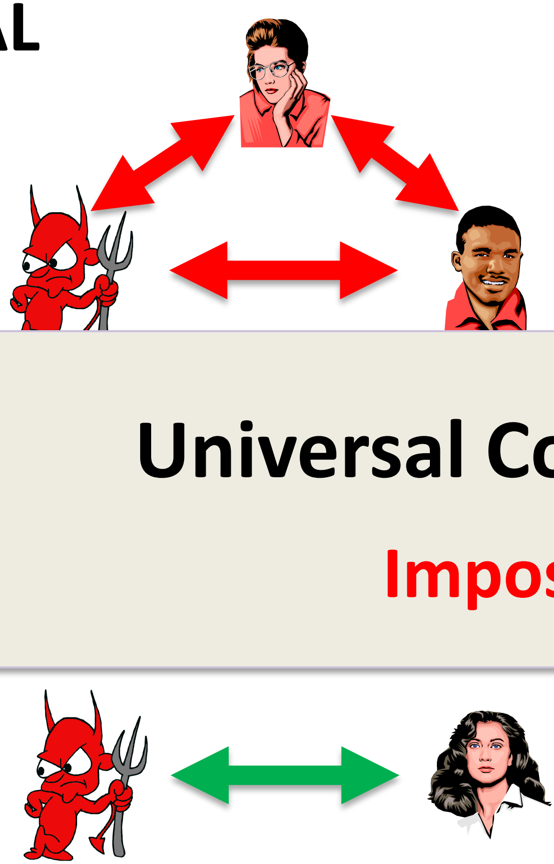
Universal Composibility (UC) [Can00]

Many executions of different protocols

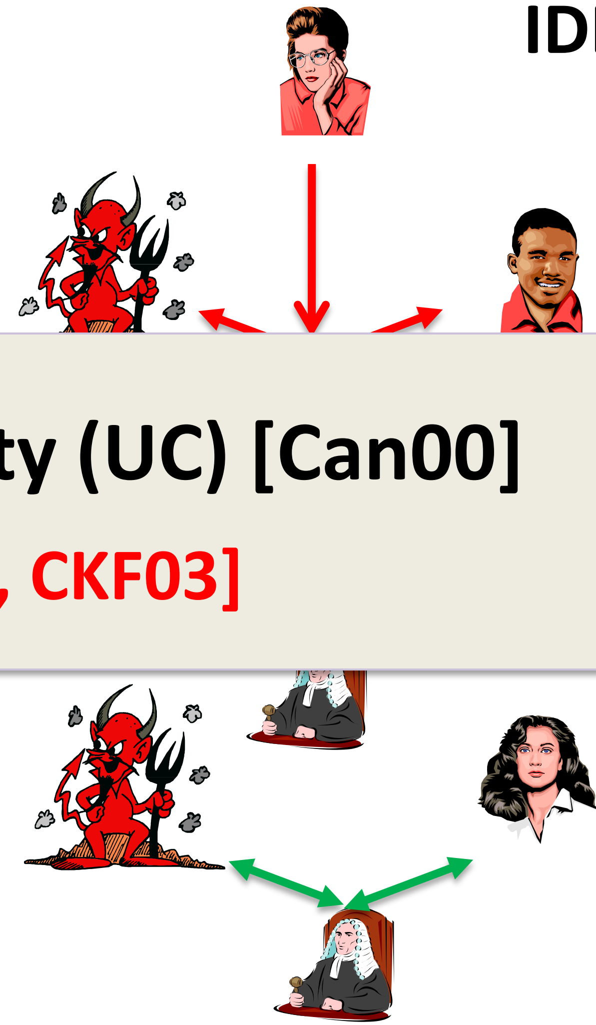
Many executions with INDEPENDENT trusted parties

Concurrent Security (informally)

REAL



IDEAL



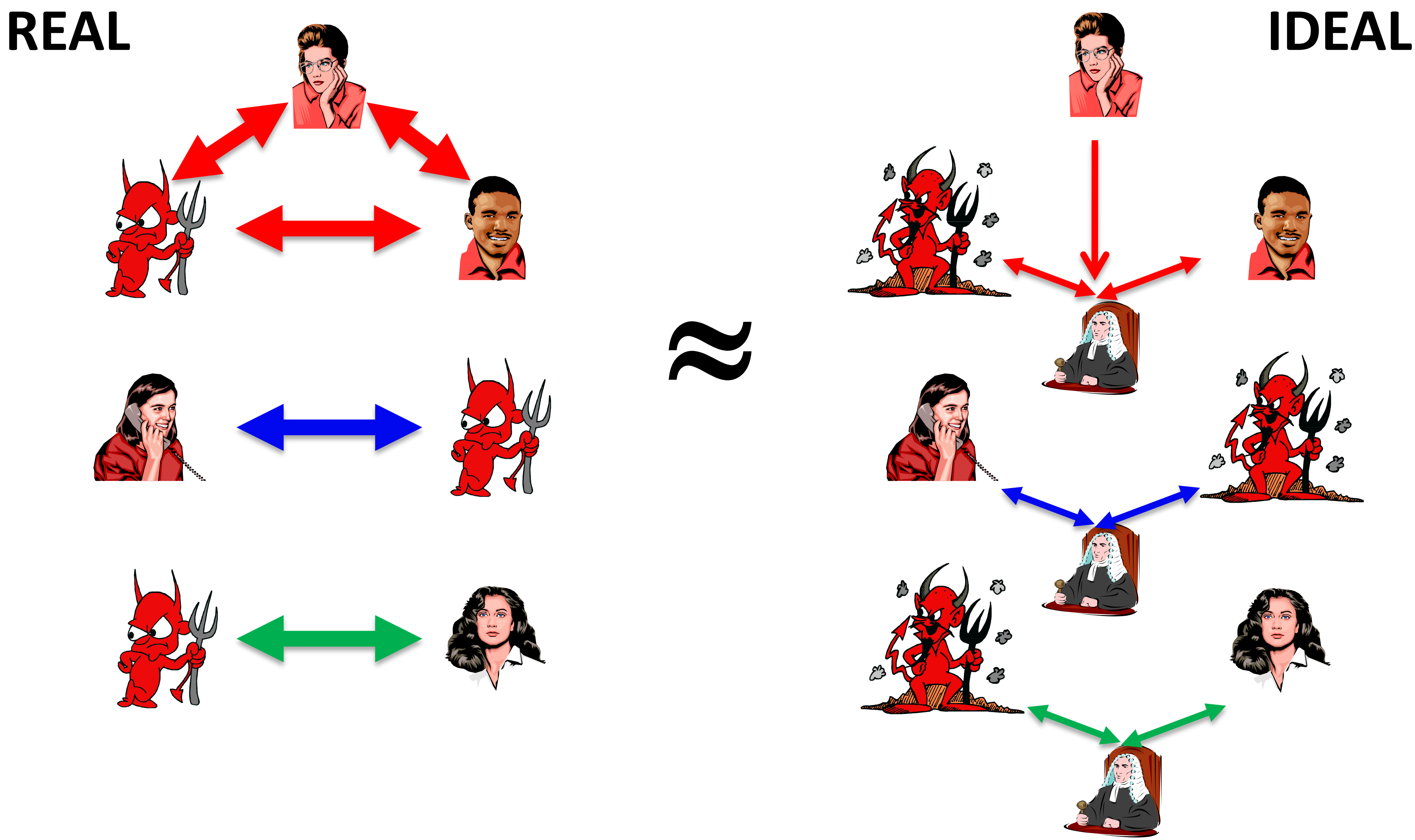
Universal Composibility (UC) [Can00]

Impossible [CF01, CKF03]

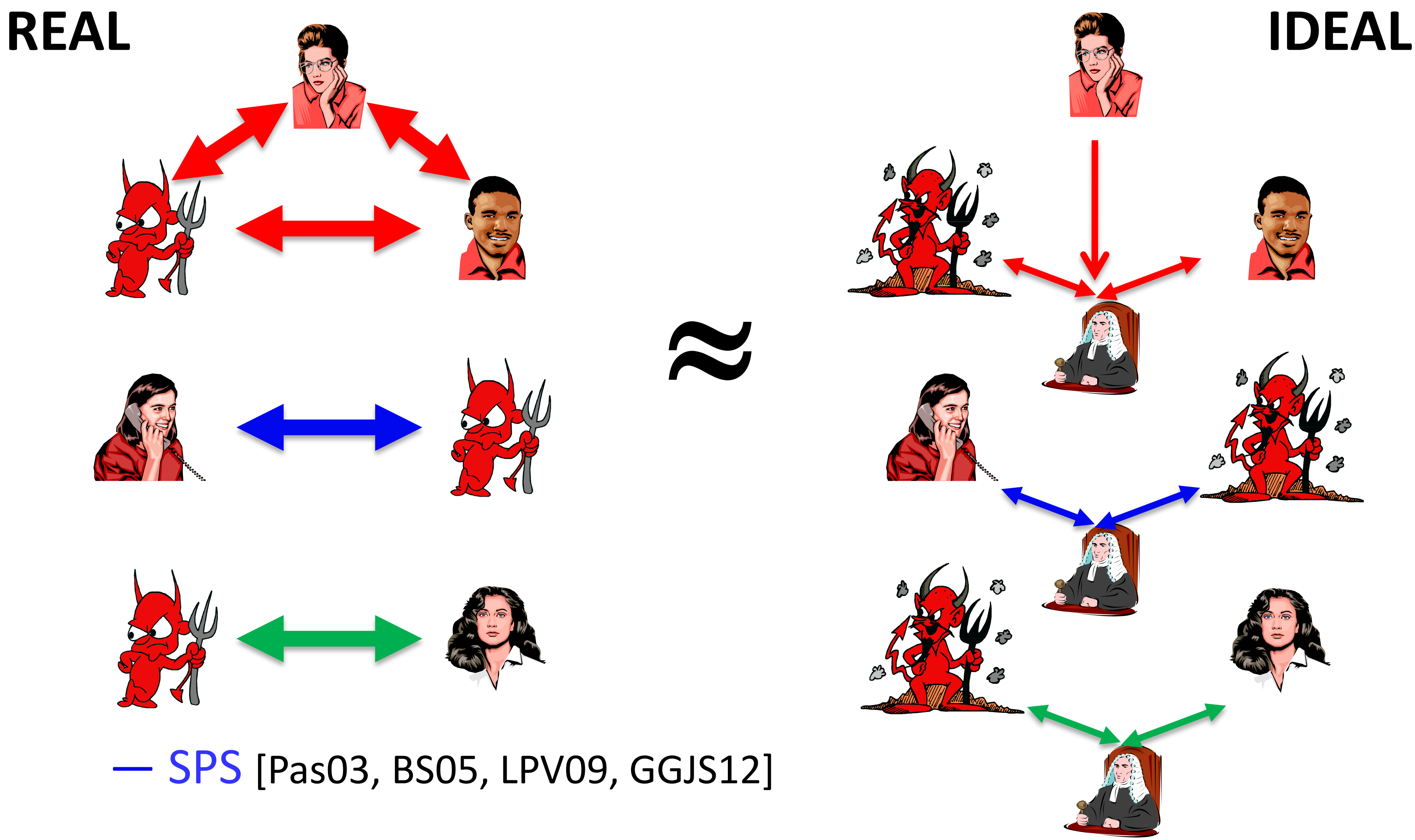
Many executions of different protocols

Many executions with INDEPENDENT trusted parties

Super Polynomial Time Simulation (**SPS**)

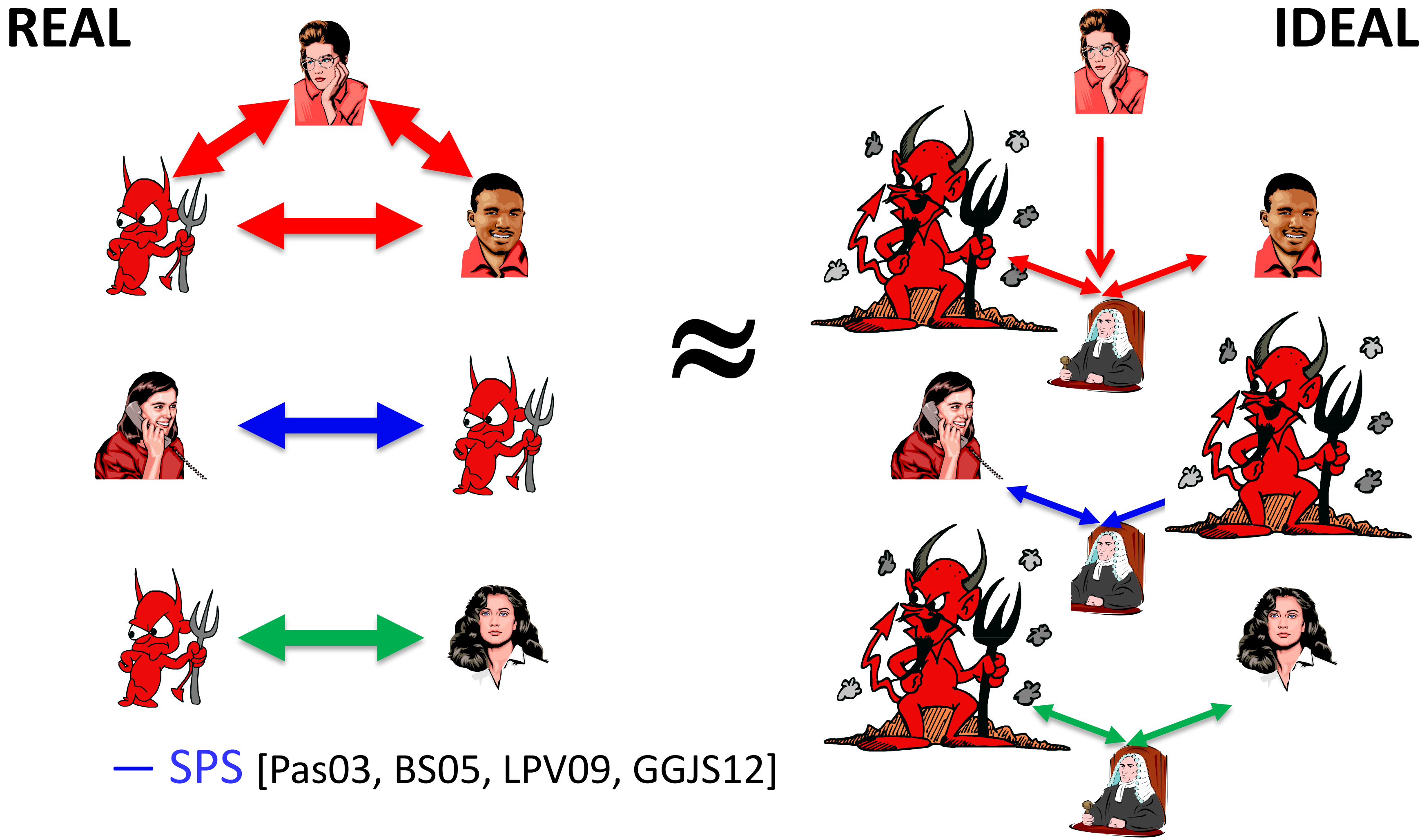


Super Polynomial Time Simulation (**SPS**)



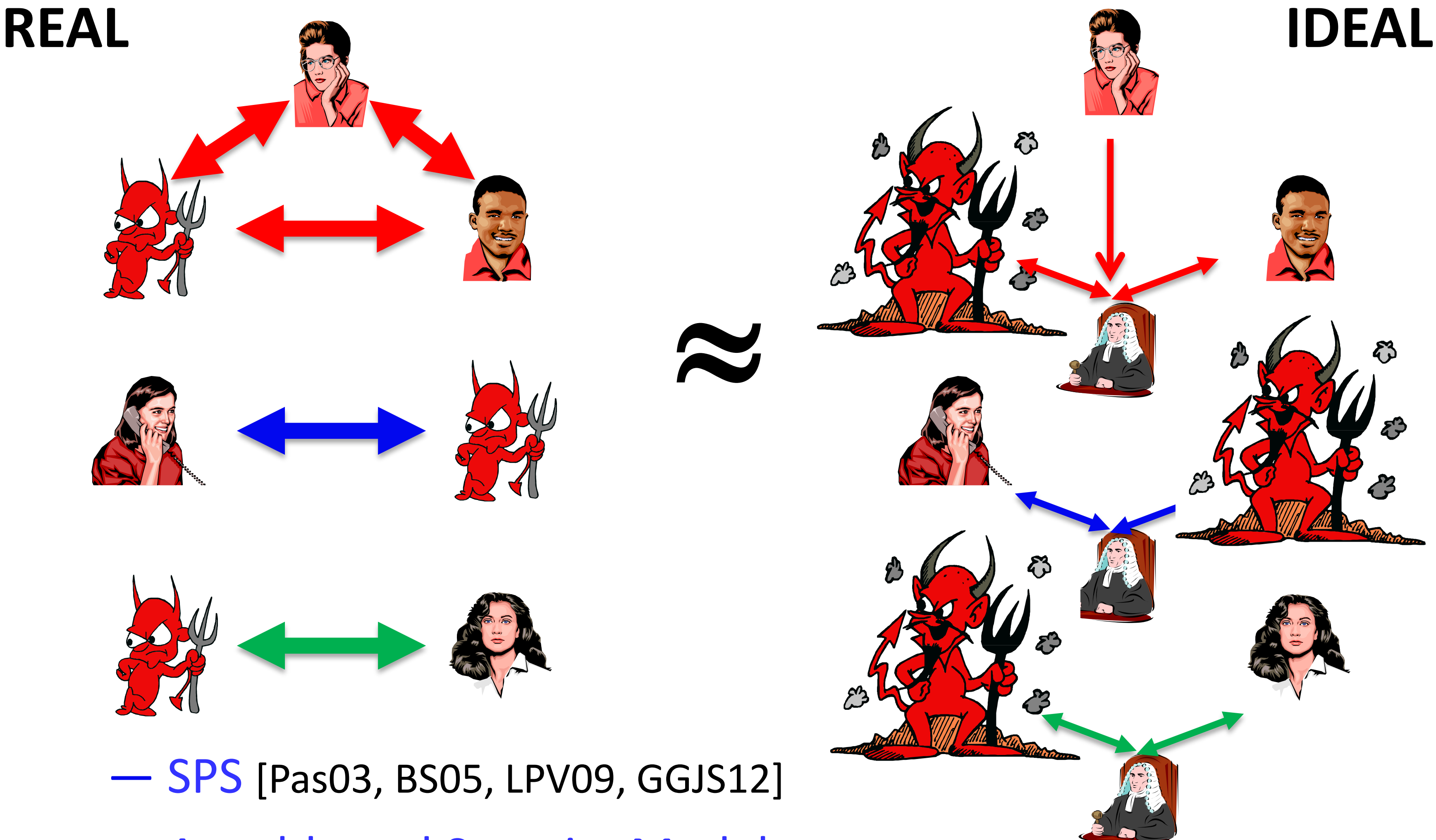
— SPS [Pas03, BS05, LPV09, GGJS12]

Super Polynomial Time Simulation (**SPS**)



— SPS [Pas03, BS05, LPV09, GGJS12]

Super Polynomial Time Simulation (**SPS**)



- SPS [Pas03, BS05, LPV09, GGJS12]
- Angel-based Security Model [PS04, MMY06]
- UC with super-poly helpers [CLP10]

Super Polynomial Time Simulation (**SPS**)

Feasibility Results Only

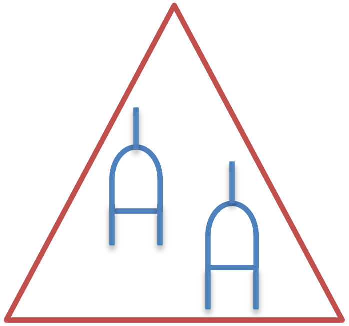
- SPS [Pas03, BS05, LPV09, GGJS12]
- Angel-based Security Model [PS04, MMY06]
- UC with super-poly helpers [CLP10]

Super Polynomial time (SPS) Security

Feasibility Results Only

Due to the Non-Black-Box constructions

(**Lots** of Karp reductions)



Super Polynomial time (SPS) Security

Feasibility Results Only

Naturally,

Solution: Black-box Constructions

(**No** Karp reductions)

Super Polynomial time (SPS) Security

Feasibility Results Only

Naturally,

Solution: Black-box Constructions

(**No** Karp reductions)

Efficient Protocols

BB MPC Protocols

BB MPC Protocols

In the stand alone setting---Solved!

O(1) round BB MPC, f/ minimal assumption semi-honest OT

[Kil88,IPS08,IKLP06,Hai08,Wee10,Goy11]

BB MPC Protocols

In the stand alone setting---Solved!

O(1) round BB MPC, f/ minimal assumption semi-honest OT [Kil88,IPS08,IKLP06,Hai08,Wee10,Goy11]

In the concurrent setting

Only **unconditionally secure** UC protocols f/ strong set-ups e.g. Ideal OT [Kil88,IPS08], hardware tokens [GISVW10]

BB MPC Protocols

In the stand alone setting---Solved!

O(1) round BB MPC, f/ minimal assumption semi-honest OT [Kil88,IPS08,IKLP06,Hai08,Wee10,Goy11]

In the concurrent setting

Only **unconditionally secure** UC protocols f/ strong set-ups e.g. Ideal OT [Kil88,IPS08], hardware tokens [GISVW10]

Can we have
BB concurrently secure protocols
in the **plain** model?

Yes!

Our Result (informal) :

BB construction of concurrently secure MPC protocols

- In the plain model
- Based on minimal assumption Semi-Honest OT
- Security in the UC with super-poly helper model
  - Implies super-polynomial time simulation security
  - Closed under universal composition

Yes!

Our Result (informal) :

BB construction of concurrently secure MPC protocols

- In the plain model
- Based on minimal assumption Semi-Honest OT
- Security in the UC with super-poly helper model
    - Implies super-polynomial time simulation security
    - Closed under universal composition

How?

Any Functionality

[Kil88,IPS08,GMW87,BGW88]:

Unconditional UC-security

Ideal Oblivious Transfer Box F_{OT}

Any Functionality

[Kil88,IPS08,GMW87,BGW88]:

Unconditional UC-security

Ideal Oblivious Transfer Box F_{OT}

BB

Stand-alone Semi-honest OT SH-OT

Any Functionality

[Kil88,IPS08,GMW87,BGW88]:

Unconditional UC-security

Ideal Oblivious Transfer Box F_{OT}

[IKLP06,Hai08,Wee10,Goy11]

Stand-Alone Security

BB

Stand-alone Semi-honest OT SH-OT

Any Functionality

[Kil88,IPS08,GMW87,BGW88]:

Unconditional UC-security

Ideal Oblivious Transfer Box F_{OT}

[IKLP06,Hai08,Wee10,Goy11]

Stand-Alone Security

BB

This work

UC with Super-Poly Helper

Stand-alone Semi-honest OT SH-OT

Any Functionality

[Kil88,IPS08,GMW87,BGW88]:

Unconditional UC-security

Ideal Oblivious Transfer Box F_{OT}

[IKLP06,Hai08,Wee10,Goy11]

Stand-Alone Security

BB

This work

UC with Super-Poly Helper

Stand-alone Semi-honest OT SH-OT

The main tool: BB CCA-Secure Commitments [CLP10]

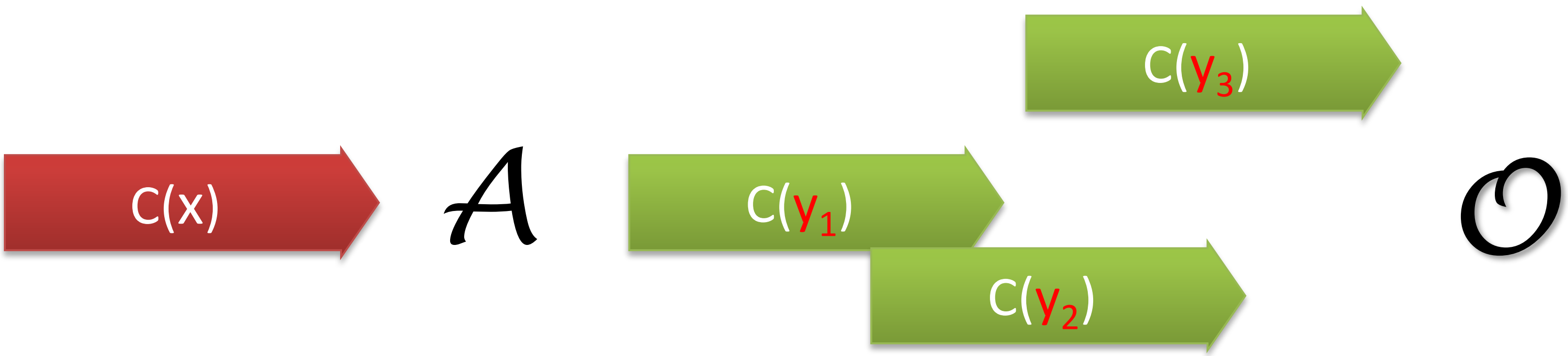
CCA-Secure Commitments

CCA-Secure Commitments

The commitment analogue of CCA2 encryption.

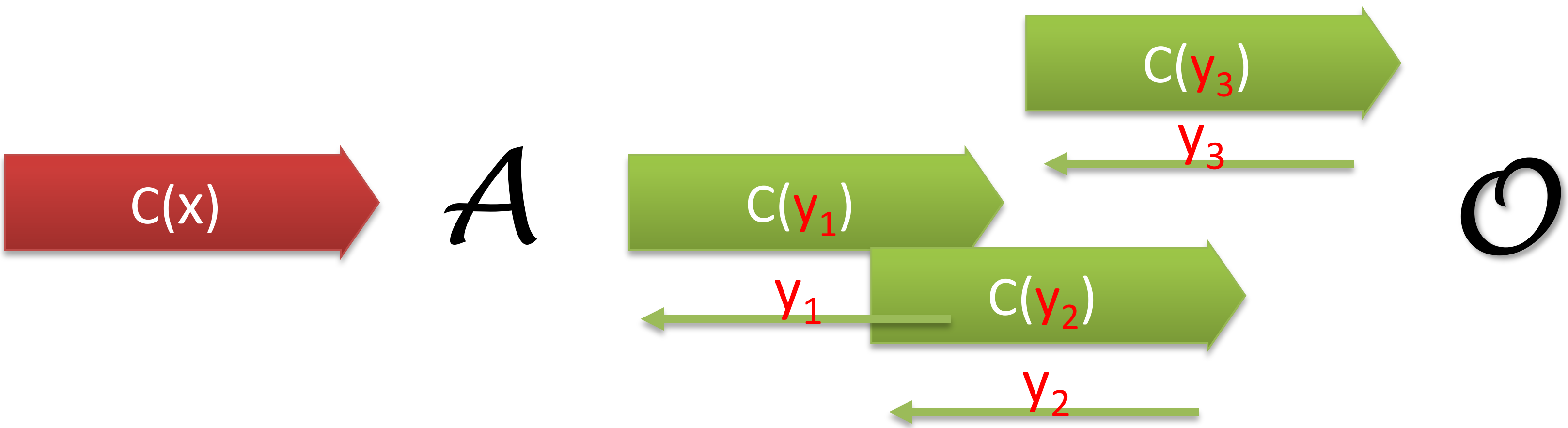
CCA-Secure Commitments

The commitment analogue of CCA2 encryption.



CCA-Secure Commitments

The commitment analogue of CCA2 encryption.



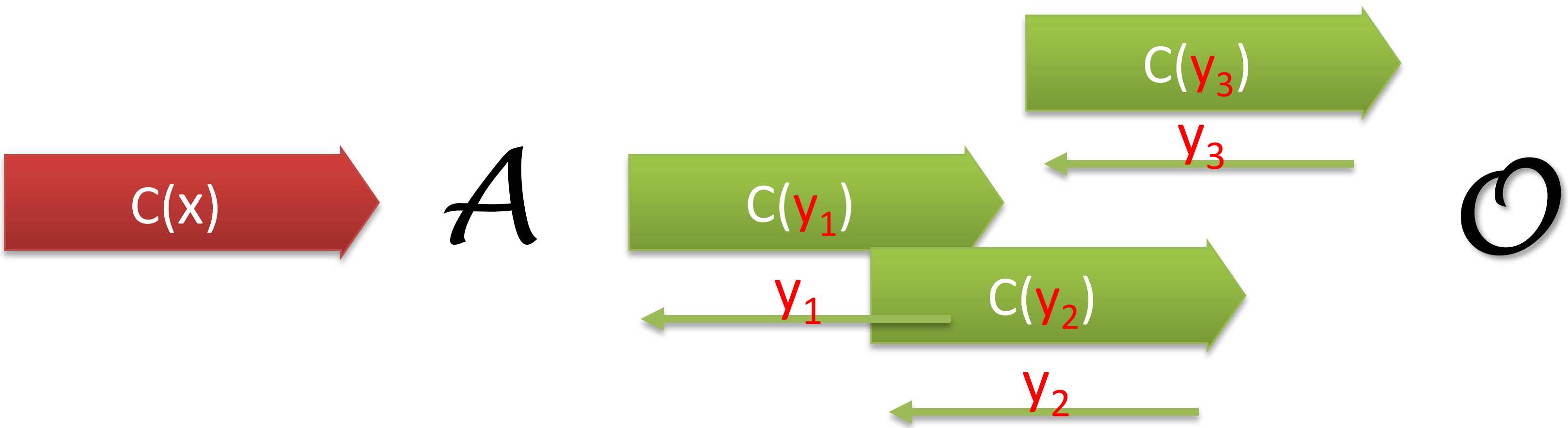
O is a committed-value oracle

If valid com, y = the committed value

Else if invalid com, y = bot

CCA-Secure Commitments

The commitment analogue of CCA2 encryption.



O is a committed-value oracle

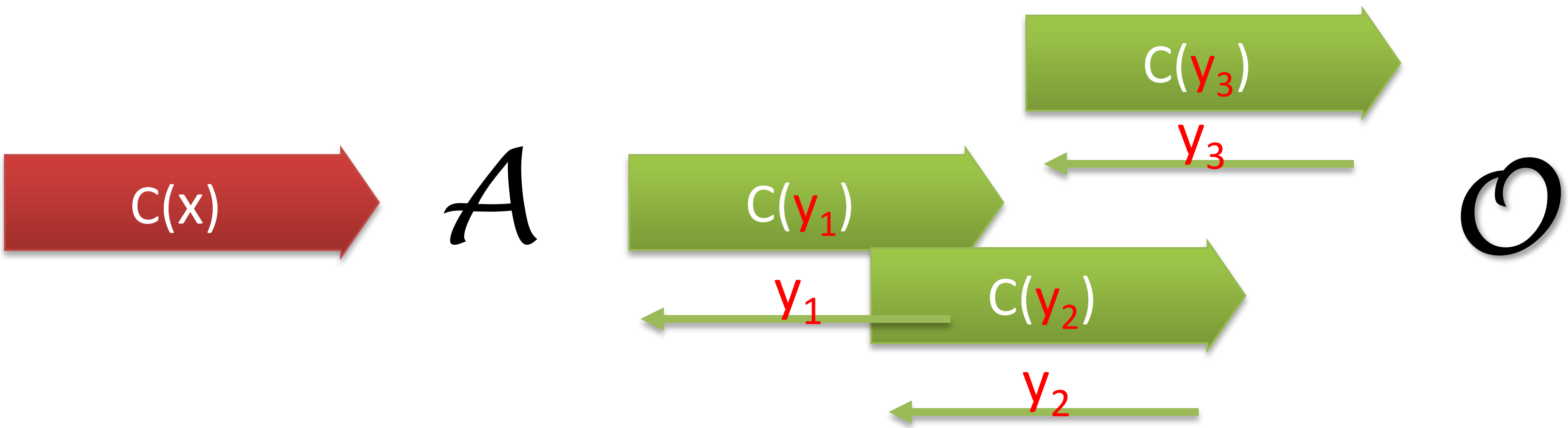
If valid com, y = the committed value

Else if invalid com, y = bot

Note: Original definition in [CLP10] considers a decommitment oracle.
(with black-box construction, we can only achieve the weaker notion.)

CCA-Secure Commitments

The commitment analogue of CCA2 encryption.

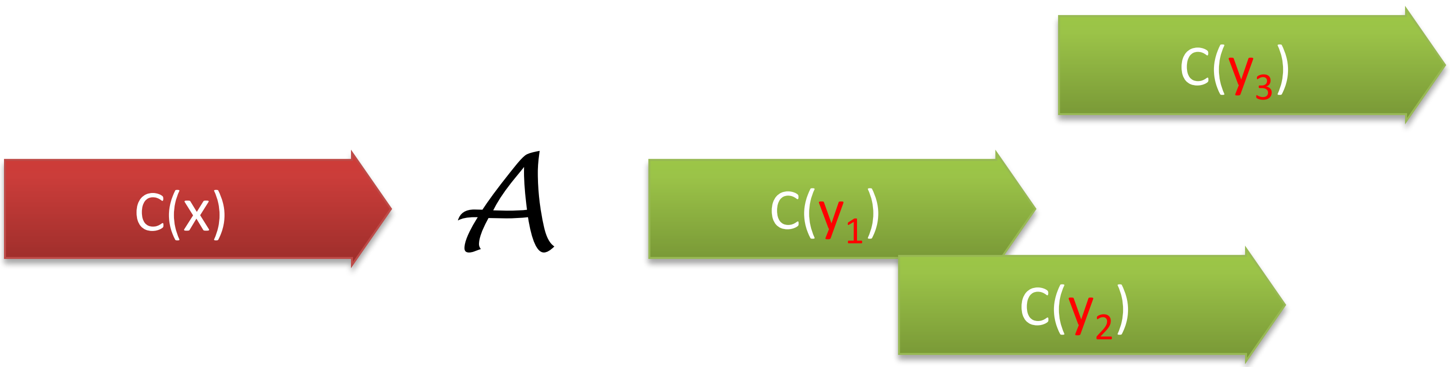


Chosen-Commitment-Attack (CCA) security:

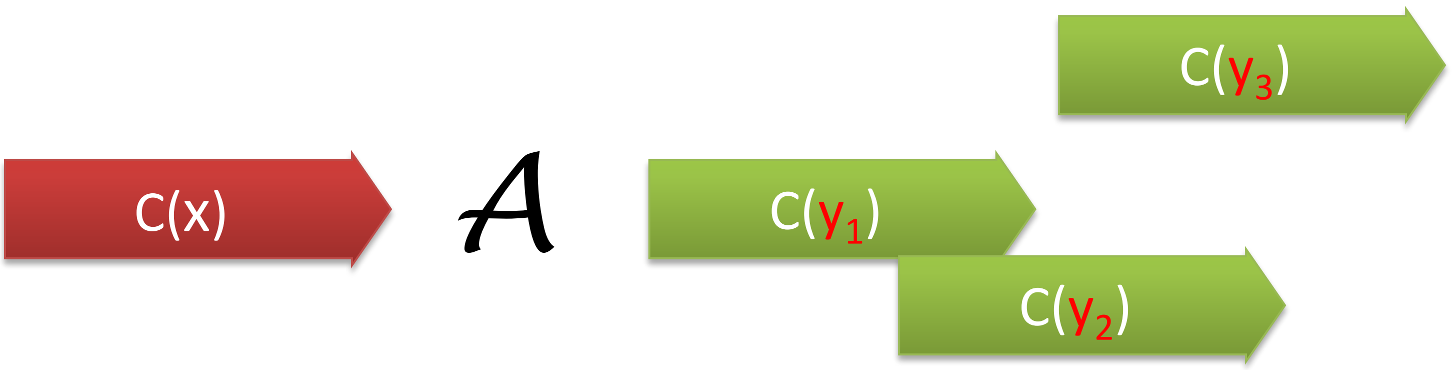
Either $\mathcal{A}$ forwards the left commitment to the right

Or LHS is hiding --- view of $\mathcal{A}$ indistinguishable

Concurrent Non-Malleable Commitments



Concurrent Non-Malleable Commitments



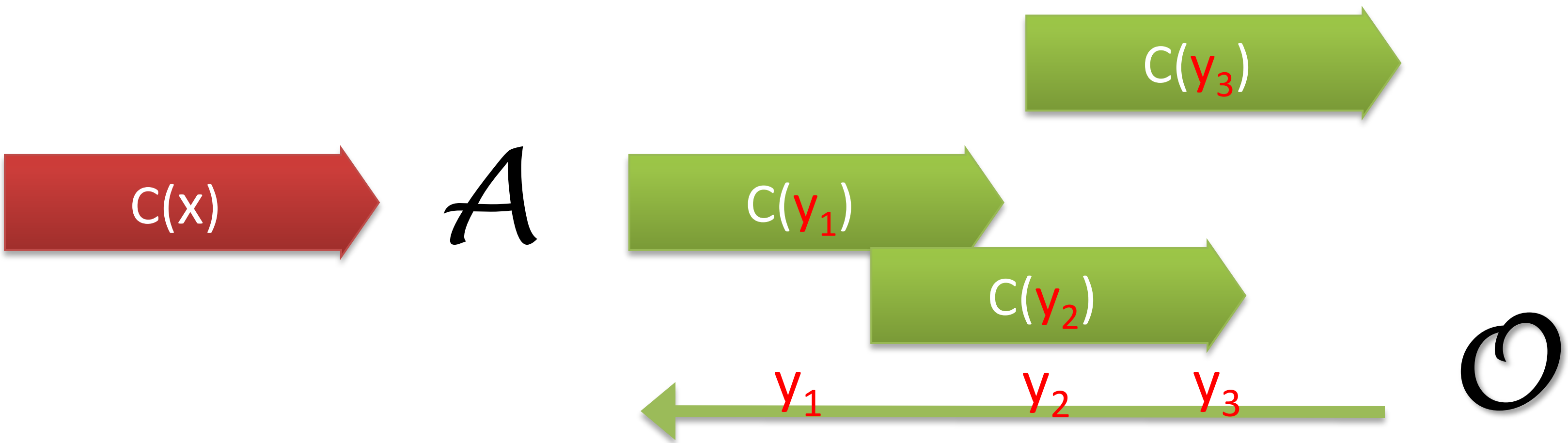
Non-Malleability

Either $\mathcal{A}$ copies the left commitment to the right

Or x and (y_1, y_2, y_3) independent

--- view of $\mathcal{A}$ + (y_1, y_2, y_3) indistinguishable

Concurrent Non-Malleable Commitments



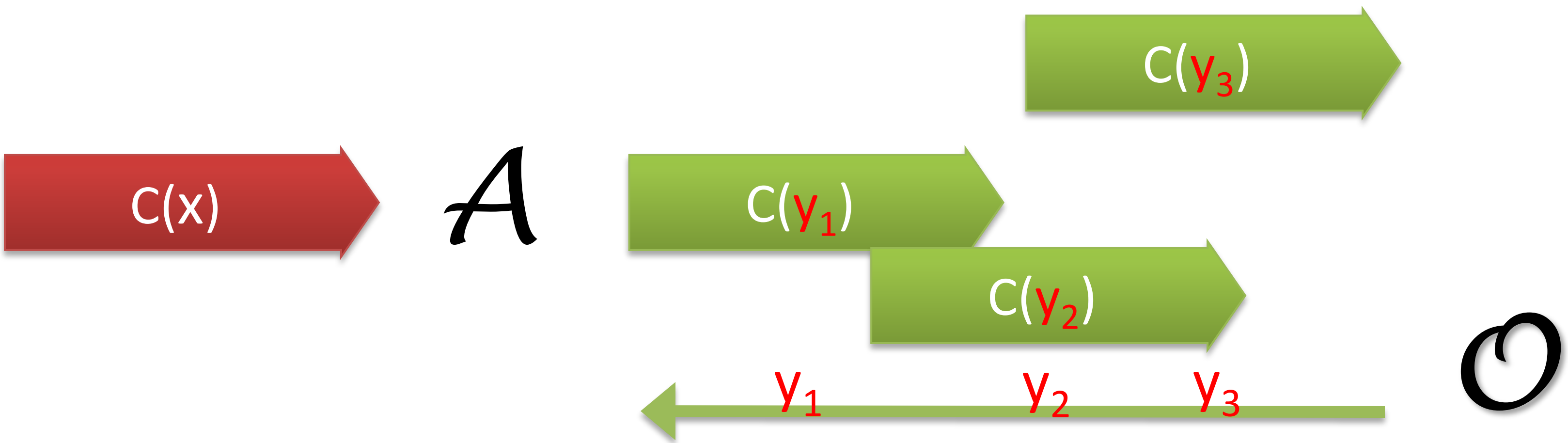
Non-Malleability

Either $\mathcal{A}$ copies the left commitment to the right

Or x and (y_1, y_2, y_3) independent

--- view of $\mathcal{A}$ + (y_1, y_2, y_3) indistinguishable

Concurrent Non-Malleable Commitments



Non-Malleability

Either $\mathcal{A}$ copies the left commitment to the right

Or x and (y_1, y_2, y_3) independent

--- view of $\mathcal{A}$ + (y_1, y_2, y_3) indistinguishable

CCA security → Non-Malleability

Theorem 1: OWF → BB construction of CCA commitments

Theorem 1: OWF → BB construction of CCA commitments

Theorem 2: CCA commitments + SH-OT

→ BB implementation of F_{OT}

Theorem 1: **OWF ➔ BB construction of CCA commitments**

Proof: [CLP10]---Non-BB CCA commitments

\+ [PW08]---BB trapdoor commitments

\+ [CDMW08,09]---Cut & choose for consistency

Theorem 2: **CCA commitments + SH-OT**

➔ BB implementation of F_{OT}

Theorem 1: **OWF ➔ BB construction of CCA commitments**

Proof: [CLP10]---Non-BB CCA commitments

\+ [PW08]---BB trapdoor commitments

\+ [CDMW08,09]---Cut & choose for consistency

Theorem 2: **CCA commitments + SH-OT**

➔ BB implementation of F_{OT}

Theorem 1: **OWF → BB construction of CCA commitments**

Proof: [CLP10]---Non-BB CCA commitments

+ [PW08]---BB trapdoor commitments

+ [CDMW08,09]---Cut & choose for consistency

Theorem 2: **CCA commitments + SH-OT**

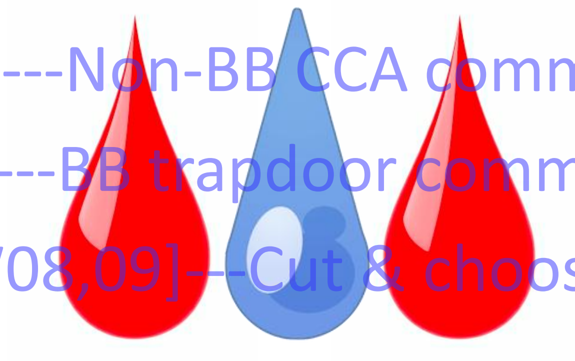
→ BB implementation of F_{OT}

Theorem 1: **OWF ➔ BB construction of CCA commitments**

Proof: [CLP10]---Non-BB CCA commitments

+ [PW08]---BB trapdoor commitments

+ [CDMW08,09]---Cut & choose for consistency



Theorem 2: **CCA commitments + SH-OT**

➔ BB implementation of F_{OT}

1. CCA is the right notion for BB concurrent MPC protocols

Theorem 1: OWF → BB construction of CCA commitments

Proof: [CLP10]---Non-BB CCA commitments

+ [PW08]---BB trapdoor commitments

+ [CDMW08,09]---Cut & choose for consistency

Theorem 2: CCA commitments + SH-OT

→ BB implementation of F_{OT}

1. CCA is the right notion for BB concurrent MPC protocols
2. Assuming **"AES"** is a CCA commitment

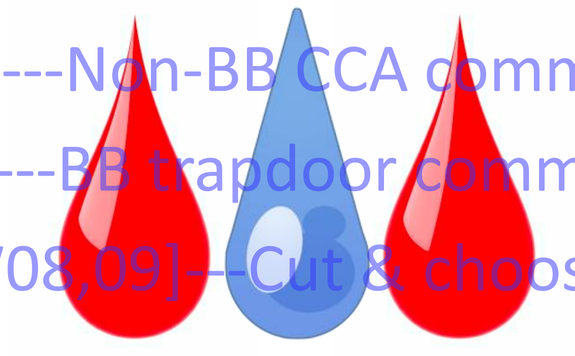
→ **Efficient Constant-round** BB concurrent MPC protocols

Theorem 1: OWF → BB construction of CCA commitments

Proof: [CLP10]---Non-BB CCA commitments

\+ [PW08]---BB trapdoor commitments

\+ [CDMW08,09]---Cut & choose for consistency



Theorem 2: CCA commitments + SH-OT

→ BB implementation of F_{OT}

1. CCA is the right notion for BB concurrent MPC protocols
2. Assuming **"AES"** is a CCA commitment

→ **Efficient Constant-round** BB concurrent MPC protocols

Theorem 2: CCA + SH-OT → BB implementation of F_{OT},

Theorem 2: CCA + mS-OT ➔ BB implementation of F_{OT}

Malicious Sender OT (ms-OT)---OT secure for malicious sender & SH receiver

Theorem 2: CCA + mS-OT ➔ BB implementation of F_{OT}

Malicious Sender OT (ms-OT)---OT secure for malicious sender & SH receiver

$\mathcal{S}$ $(m_0 m_1)$

$\mathcal{R}$(b)

Theorem 2: CCA + mS-OT ➔ BB implementation of F_{OT}

Malicious Sender OT (ms-OT)---OT secure for malicious sender & SH receiver

$\mathcal{S}$ $(m_0 m_1)$

$\mathcal{R}$$(b)$

2n ms-OT executions with random inputs

OT_1 ·· OT_k ·· OT_{2n}

Theorem 2: CCA + mS-OT → BB implementation of F_{OT}

Malicious Sender OT (ms-OT)---OT secure for malicious sender & SH receiver

$\mathcal{S}$ $(m_0 m_1)$

$\mathcal{R}(b)$

2n ms-OT executions with random inputs

OT_1 ·· OT_k ·· OT_{2n}

Want: Enforce R behave honestly in OTs

Theorem 2: CCA + mS-OT → BB implementation of F_{OT}

Malicious Sender OT (ms-OT)---OT secure for malicious sender & SH receiver

$\mathcal{S}$ $(m_0 m_1)$

$\mathcal{R}$$(b)$

2n ms-OT executions with random inputs

OT_1 ·· OT_k ·· OT_{2n}

Non-BB Solution

ZK proof R acts honestly

Want: Enforce R behave honestly in OTs

Theorem 2: CCA + mS-OT ➔ BB implementation of F_{OT}

Malicious Sender OT (ms-OT)---OT secure for malicious sender & SH receiver

$\mathcal{S}$ $(m_0 m_1)$

$\mathcal{R}(b)$

2n ms-OT executions with random inputs

OT_1 ·· OT_k ·· OT_{2n}

BB Solution:
Cut & Choose

Want: Enforce R behave honestly in OTs

Theorem 2: CCA + mS-OT ➔ BB implementation of F_{OT}

Malicious Sender OT (ms-OT)---OT secure for malicious sender & SH receiver

$\mathcal{S}$ $(m_0 m_1)$ $\mathcal{R}(b)$

2n ms-OT executions with random inputs

OT_1 ·· OT_k ·· OT_{2n}

BB Solution: Cut & Choose

T $\subset$[2n], |T| = n $\longrightarrow$

Want: Enforce R behave honestly in OTs

Theorem 2: CCA + mS-OT ➔ BB implementation of F_{OT}

Malicious Sender OT (ms-OT)---OT secure for malicious sender & SH receiver

$\mathcal{S}$ $(m_0 m_1)$

$\mathcal{R}$ (b)

2n ms-OT executions with random inputs

OT_1 ·· OT_k ·· OT_{2n}

BB Solution: Cut & Choose

$\xrightarrow{T \subset [2n],\ |T| = n}$

$\xleftarrow{\text{Open Randomness in } OT_k \text{ for } k \in T}$

Want: Enforce R behave honestly in OTs

Theorem 2: CCA + mS-OT ➔ BB implementation of F_{OT}

Malicious Sender OT (ms-OT)---OT secure for malicious sender & SH receiver

$\mathcal{S}$ $(m_0 m_1)$

$\mathcal{R}$(b)

2n ms-OT executions with random inputs

OT_1 ·· OT_k ·· OT_{2n}

BB Solution:
Cut & Choose

$$\xrightarrow{T \subset [2n],\ |T| = n}$$

$$\xleftarrow{\text{Open Randomness in } OT_k \text{ for } k \in T}$$

Cut & Choose ➔ R behave honestly in most OTs [IKLP06,Wee10]

Theorem 2: CCA + mS-OT → BB implementation of F_{OT}

Malicious Sender OT (ms-OT)---OT secure for malicious sender & SH receiver

$\mathcal{S}$ $(m_0 m_1)$

$\mathcal{R}(b)$

2n ms-OT executions with random inputs

OT_1 ·· OT_k ·· OT_{2n}

BB Solution: Cut & Choose

$T \subset [2n], |T| = n$ →

Open Randomness in OT_k for $k \in T$ ←

OT Combiner

Cut & Choose → R behave honestly in most OTs [IKLP06,Wee10]

Theorem 2: CCA + mS-OT ➔ BB implementation of F_{OT}

Malicious Sender OT (ms-OT)---OT secure for malicious sender & SH receiver

$\mathcal{S}$ $(m_0 m_1)$

$\mathcal{R}(b)$

2n ms-OT executions with random inputs

OT_1 ·· OT_k ·· OT_{2n}

BB Solution: Cut & Choose

$T \subset [2n]$, $|T| = n$

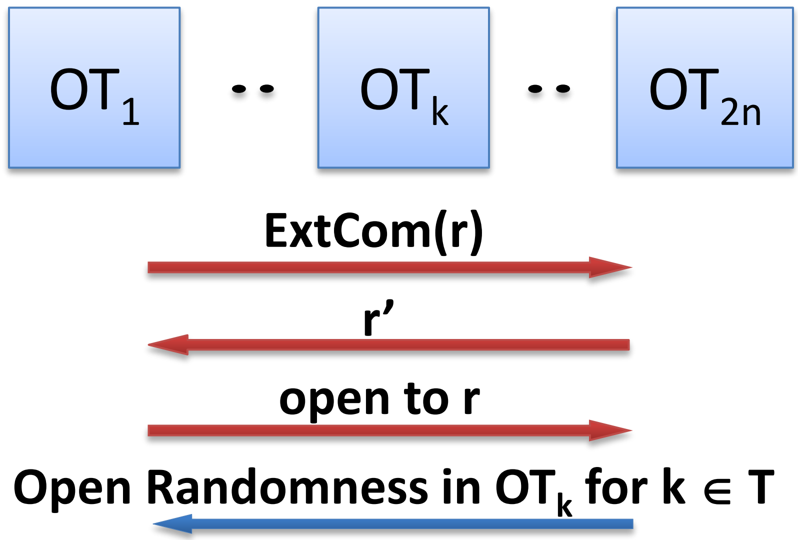
Open Randomness in OT_k for $k \in T$

To prove security against a malicious sender, Simulator needs to bias the set T to be cut

Theorem 2: CCA + mS-OT ➔ BB implementation of F_{OT}

$\mathcal{S}\ (m_0 m_1)$ $\mathcal{R}(b)$

2n ms-OT executions
with random inputs



BB Solution:
Cut & Choose

To prove security against a malicious sender, Simulator needs to bias the set T to be cut

Theorem 2: CCA + mS-OT → BB implementation of F_{OT}

$\mathcal{S}$ $(m_0 m_1)$

$\mathcal{R}$(b)

2n ms-OT executions with random inputs

OT_1 ·· OT_k ·· OT_{2n}

ExtCom(r) →

← r'

open to r →

← Open Randomness in OT_k for $k \in T$

BB Solution:
Cut & Choose

T = r XOR r'

To prove security against a malicious sender, Simulator needs to bias the set T to be cut

Theorem 2: CCA + mS-OT ➔ BB implementation of F_{OT}

$\mathcal{S}\,(m_0 m_1)$ $\mathcal{R}(b)$

2n ms-OT executions with random inputs

OT_1 ·· OT_k ·· OT_{2n}

ExtCom(r) →

r' ←

open to r →

Open Randomness in OT_k for $k \in T$ ←

BB Solution: Cut & Choose

T = r XOR r'

Using Coin Tossing,

Simulator can bias the set T to be cut

Informally, SH-OT + Coin-Tossing
→ Ideal OT in stand-alone setting [IKLP06,Wee10]

In the concurrent setting,

Main issue: **simulation-sound** coin tossing

In the concurrent setting,

Main issue: **simulation-sound** coin tossing

No adv can bias the coin tossing results,

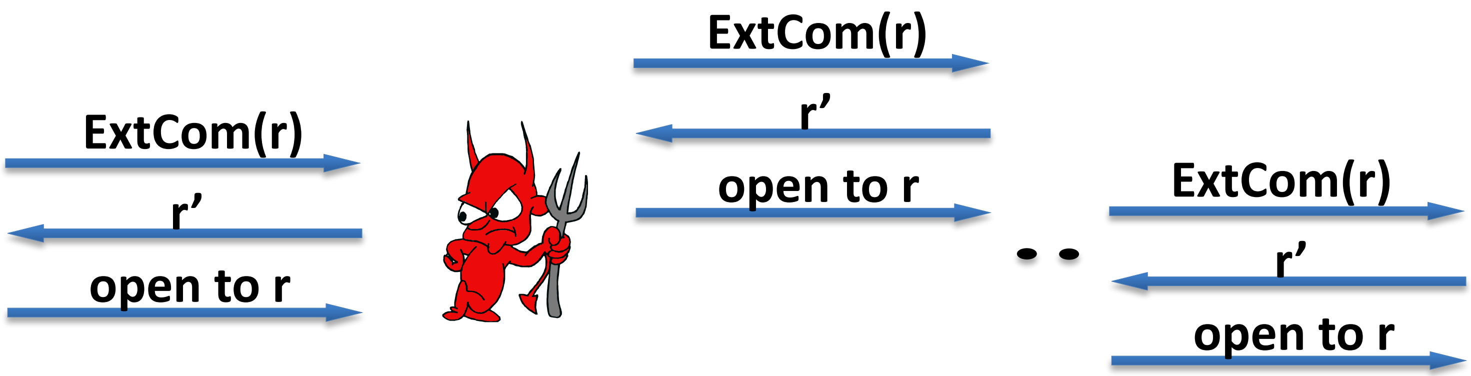
even when the simulator is doing so

In the concurrent setting,

Main issue: **simulation-sound** coin tossing

No adv can bias the coin tossing results,

even when the simulator is doing so

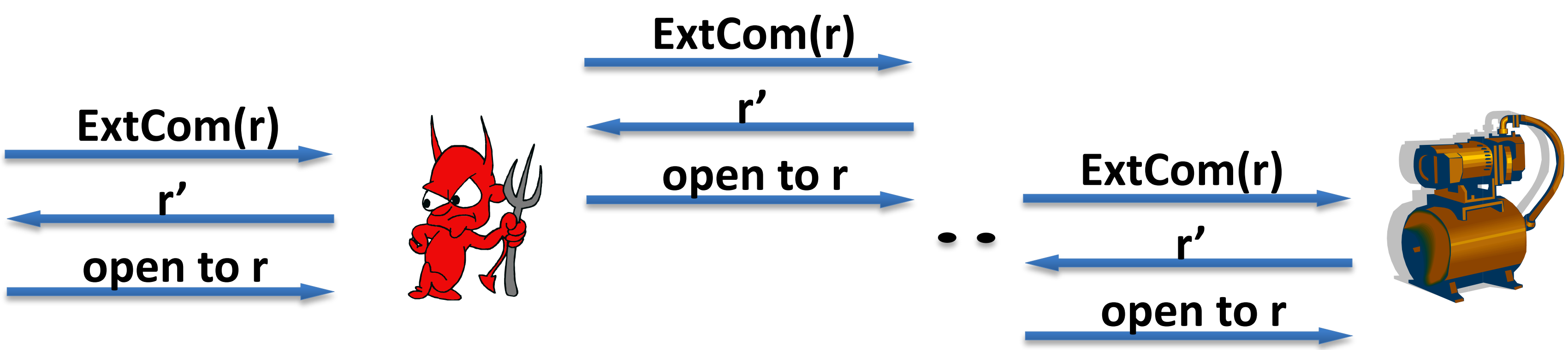


In the concurrent setting,

Main issue: **simulation-sound** coin tossing

No adv can bias the coin tossing results,

even when the simulator is doing so

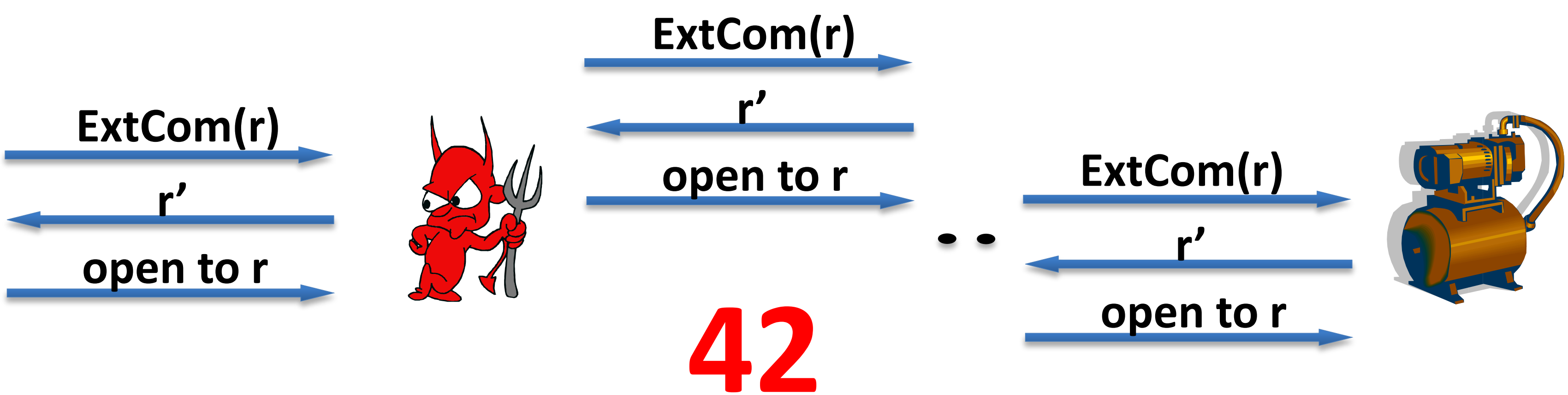


In the concurrent setting,

Main issue: **simulation-sound** coin tossing

No adv can bias the coin tossing results,

even when the simulator is doing so

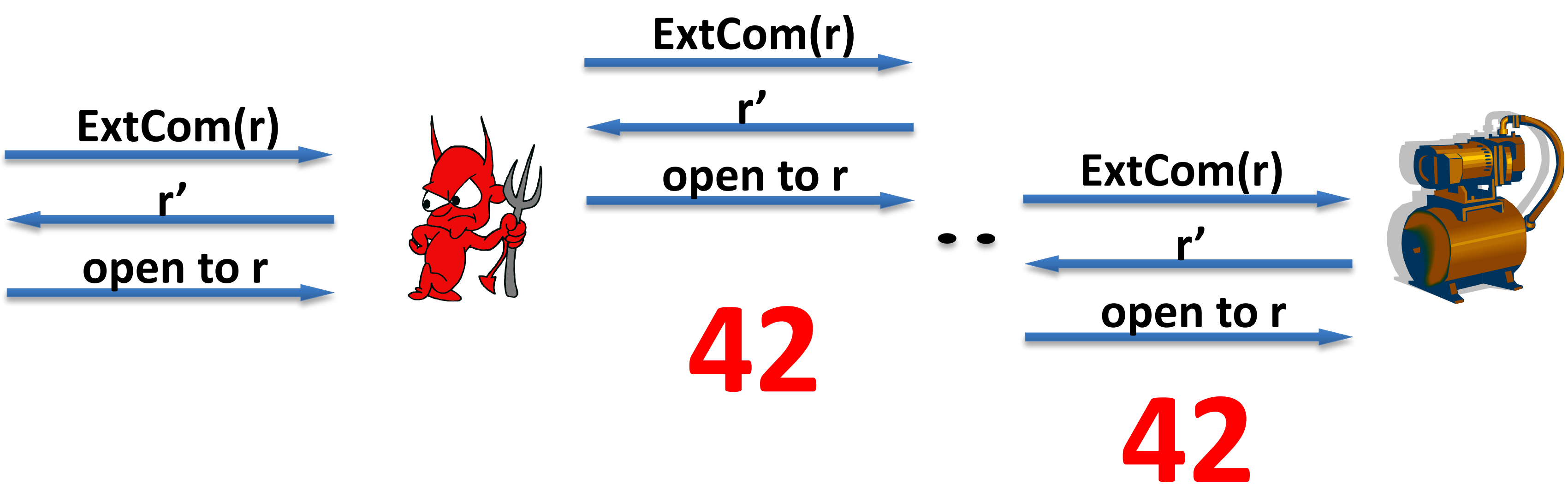


In the concurrent setting,

Main issue: **simulation-sound** coin tossing

No adv can bias the coin tossing results,

even when the simulator is doing so

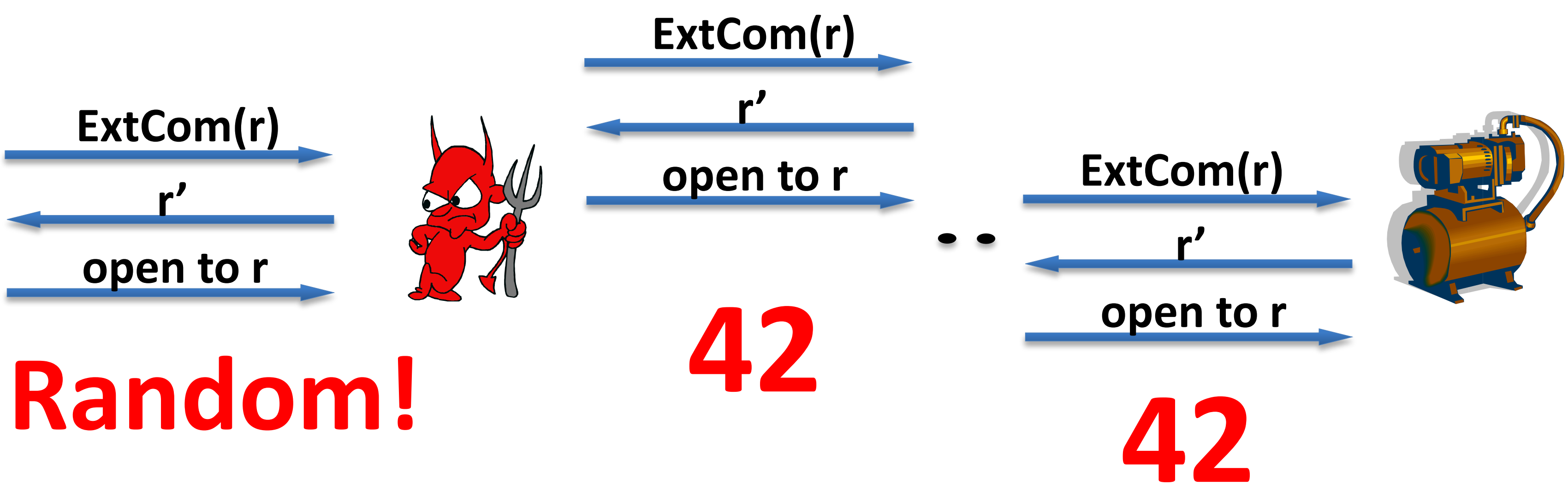


In the concurrent setting,

Main issue: **simulation-sound** coin tossing

No adv can bias the coin tossing results,

even when the simulator is doing so

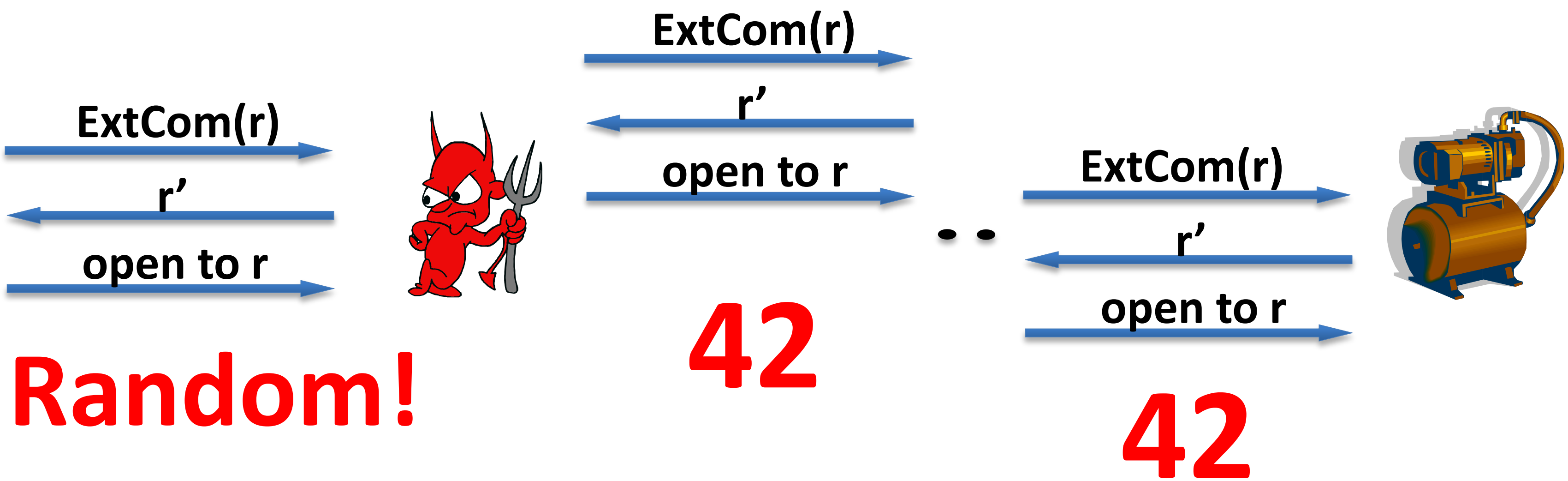


In the concurrent setting,

Main issue: **simulation-sound** coin tossing

No adv can bias the coin tossing results,

even when the simulator is doing so



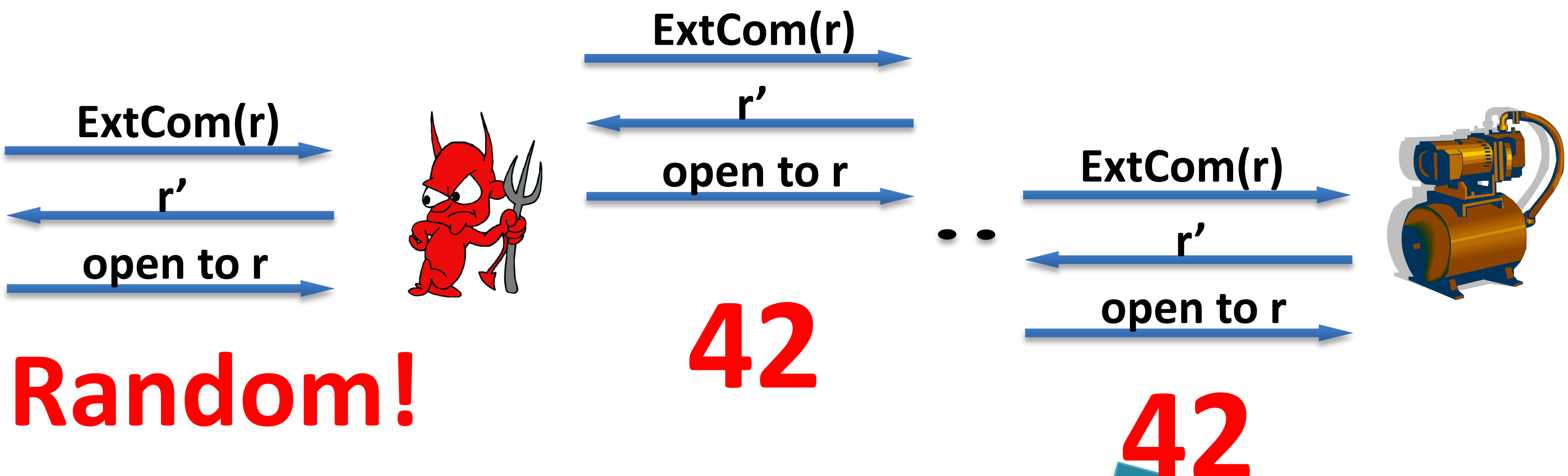
Informally, SH-OT + **simulation sound coin tossing**

➔ Ideal OT in **concurrent setting**

In the concurrent setting,

Main issue: **simulation-sound** coin tossing

No adv can bias the coin tossing results,
even when the simulator is doing so



from CCA Com

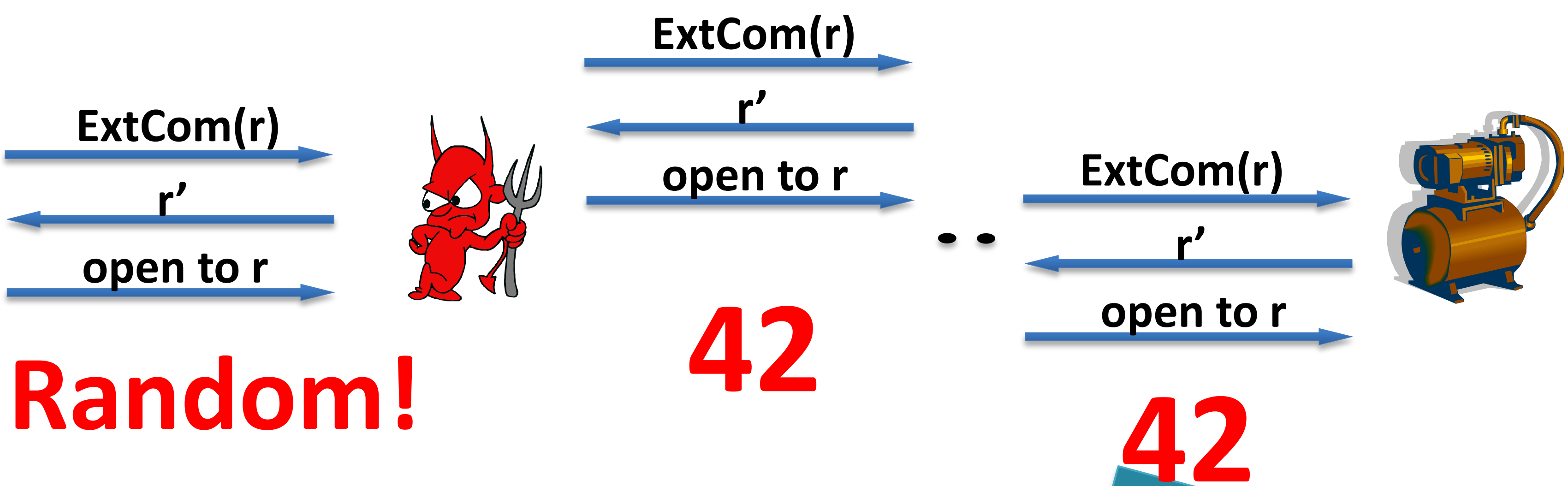
Informally, SH-OT + **simulation sound coin tossing**
➔ Ideal OT in **concurrent setting**

In the concurrent setting,

Main issue: **simulation-sound** coin tossing

No adv can bias the coin tossing results,

even when the simulator is doing so



Informally, SH-OT + **simulation sound coin tossing**

➔ Ideal OT in **concurrent setting**

Concurrent Coin Tossing from CCA

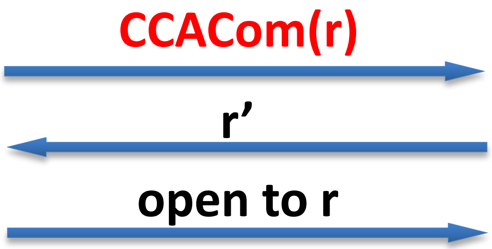
Concurrent Coin Tossing from CCA

ExtCom(r) →

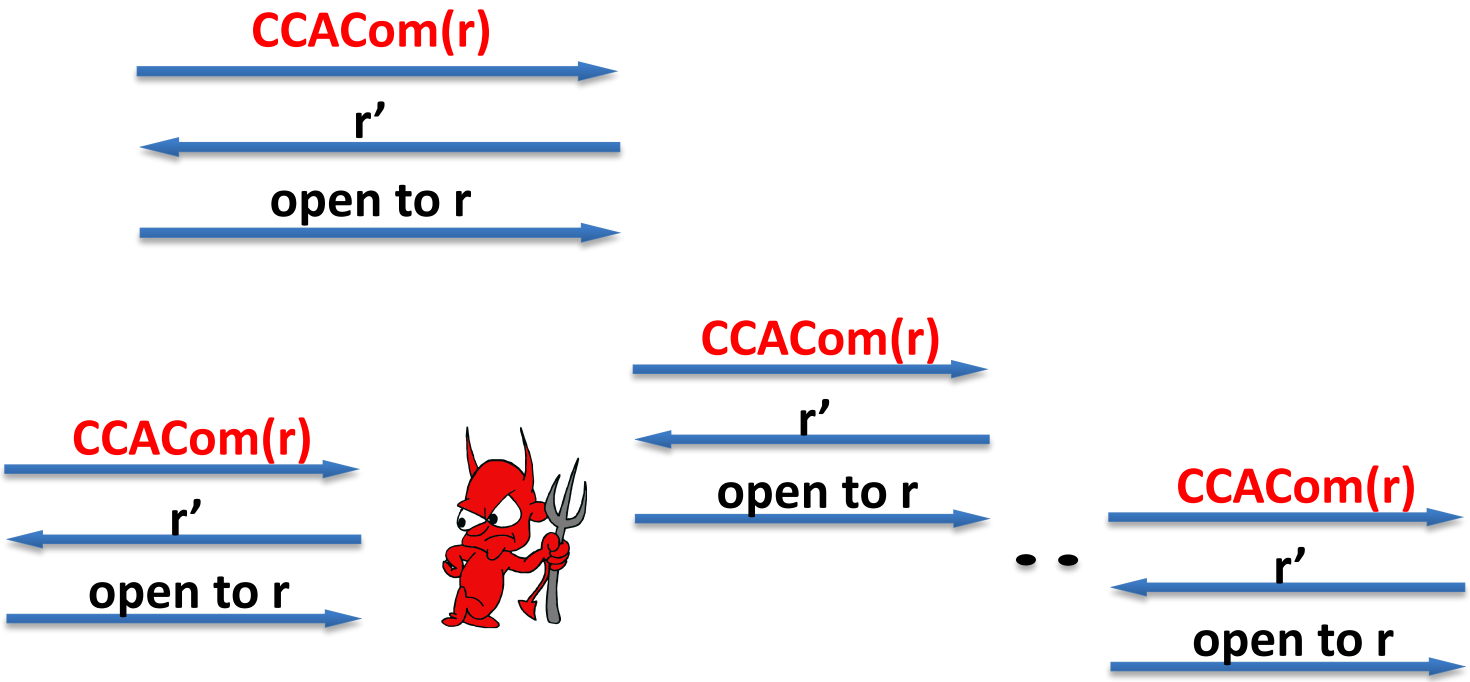
← **r’**

open to r →

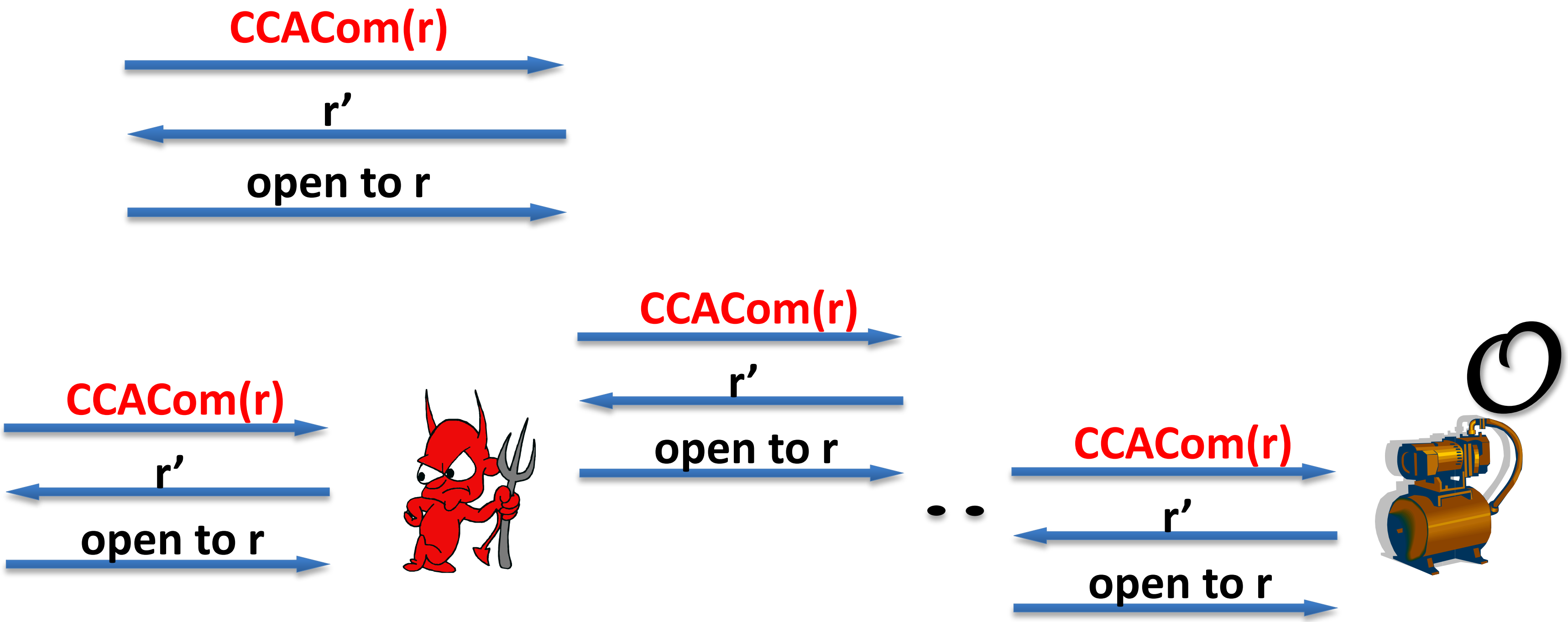
Concurrent Coin Tossing from CCA



Concurrent Coin Tossing from CCA

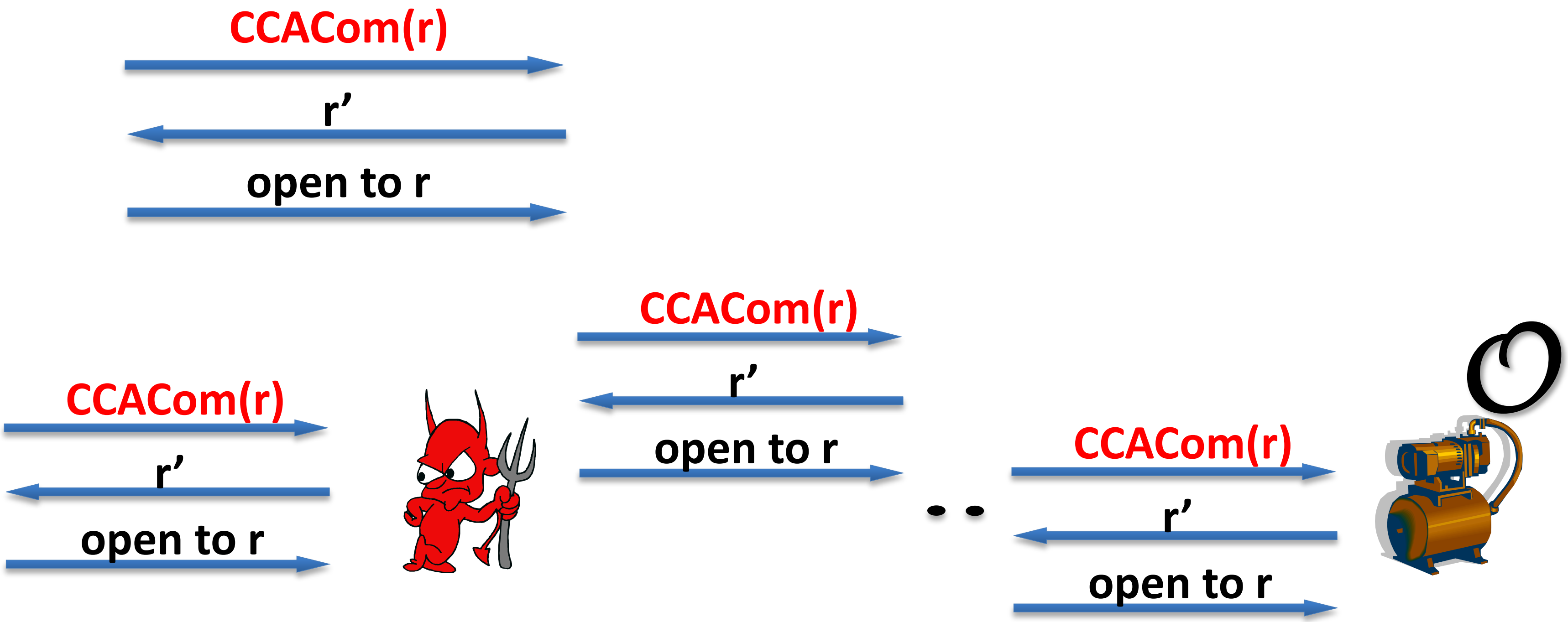


Concurrent Coin Tossing from CCA



Simulator can bias coins, by using oracle to break CCACom from adv

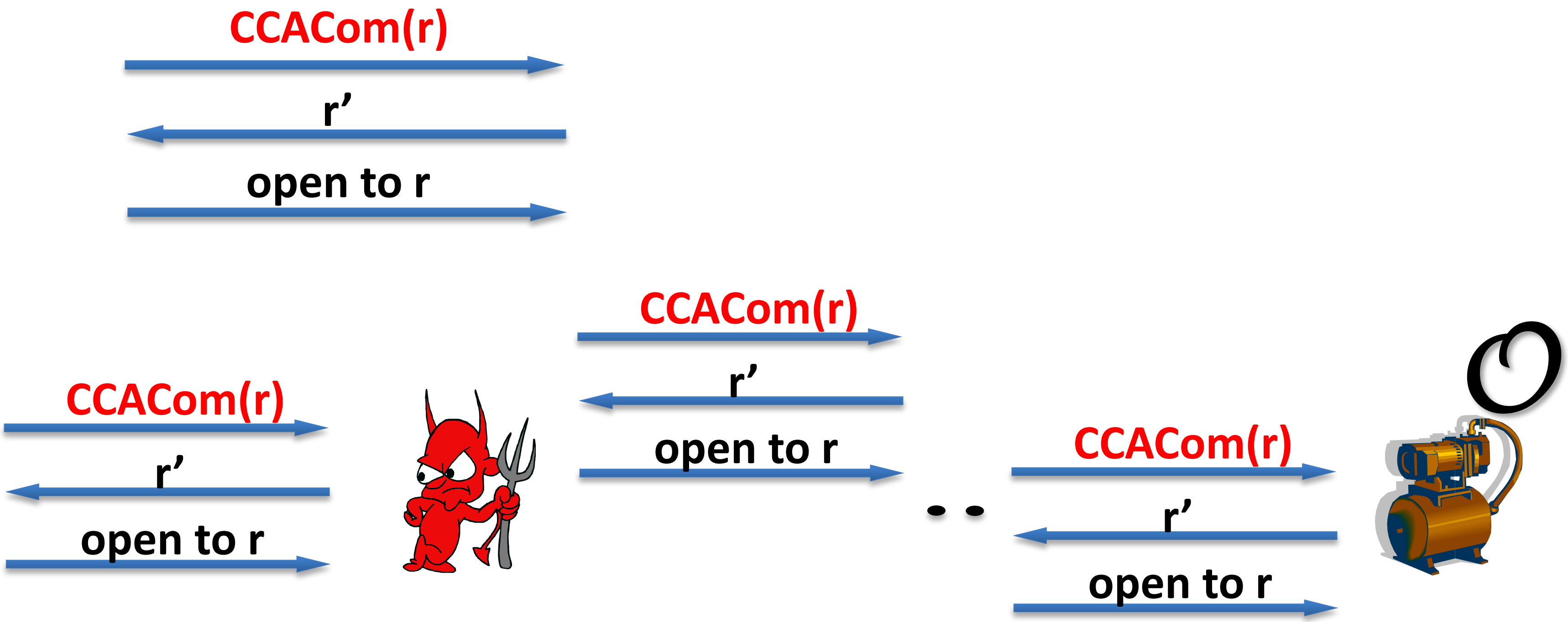
Concurrent Coin Tossing from CCA



Simulator can bias coins, by using oracle to break CCACom from adv

The adv cannot bias coins, as the CCACom from honest player is still hiding

Concurrent Coin Tossing from CCA



Simulator can bias coins, by using oracle to break CCACom from adv

The adv cannot bias coins, as the CCACom from honest player is still hiding

Theorem 2: CCA + SH-OT ➔ BB implementation of F_{OT}

Our Result (informal) :

BB construction of concurrently secure MPC protocols

- In the plain model
- Assuming Semi-Honest Oblivious Transfer protocols
- Security in the UC with super-poly helper model [CLP10]
    - Implies SPS security
    - Closed under universal composition

Our Result (informal) :

BB construction of concurrently secure MPC protocols

- In the plain model
- Assuming Semi-Honest Oblivious Transfer protocols
- Security in the UC with super-poly helper model [CLP10]
  - Implies SPS security
  - Closed under universal composition

BB CCA Commitments

Our Result (informal) :

BB construction of concurrently secure MPC protocols

- In the plain model
- Assuming Semi-Honest Oblivious Transfer protocols
- Security in the UC with super-poly helper model [CLP10]
    - Implies SPS security
    - Closed under universal composition

BB CCA Commitments

KEY Notion

Our Result (informal) :

BB construction of concurrently secure MPC protocols

- In the plain model
- Assuming Semi-Honest Oblivious Transfer protocols
- Security in the UC with super-poly helper model [CLP10]
  - Implies SPS security
  - Closed under universal composition

KEY Notion

BB CCA Commitments

O(n)-round, better round-complexity?

Thank you!