Maliciously Secure SCALES Protocols

Anasuya Acharya [1], Carmit Hazay[1], Vladimir Kolesnikov[2], Manoj Prabhakaran[3]

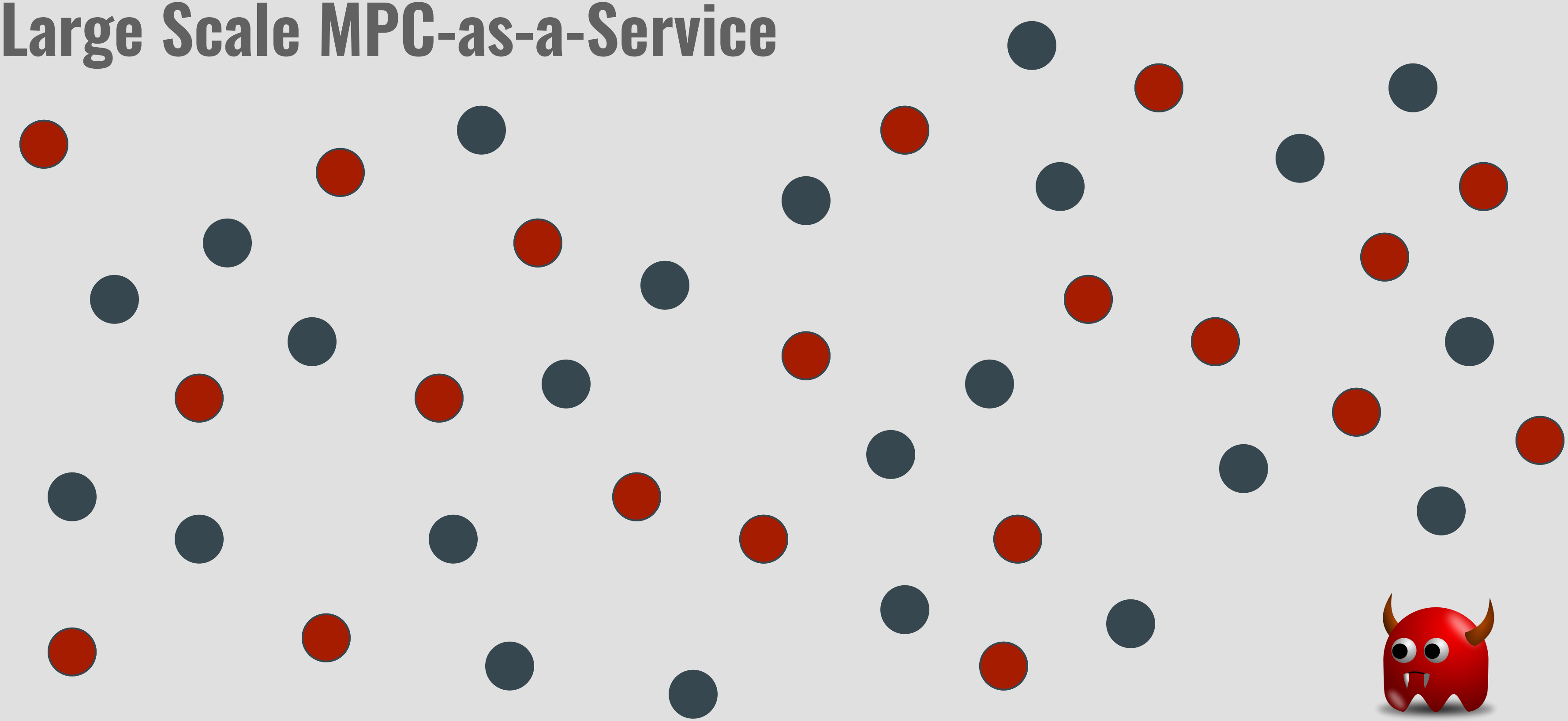
[1]Bar-Ilan University
[2]Georgia Institute of Technology
[3]IIT Bombay

Center for Research in Applied Cryptography and Cyber Security

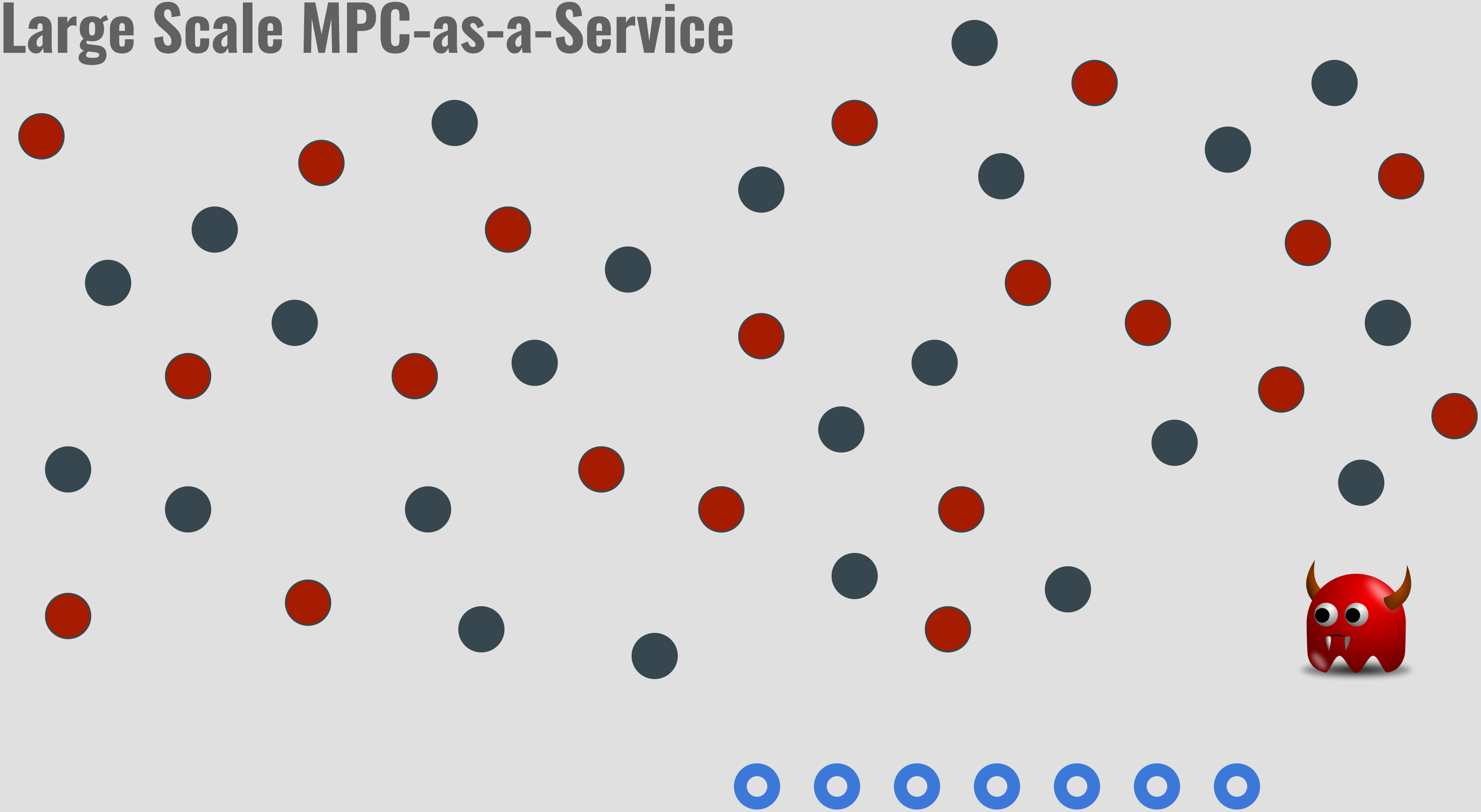
MPC

MPC-as-a-Service

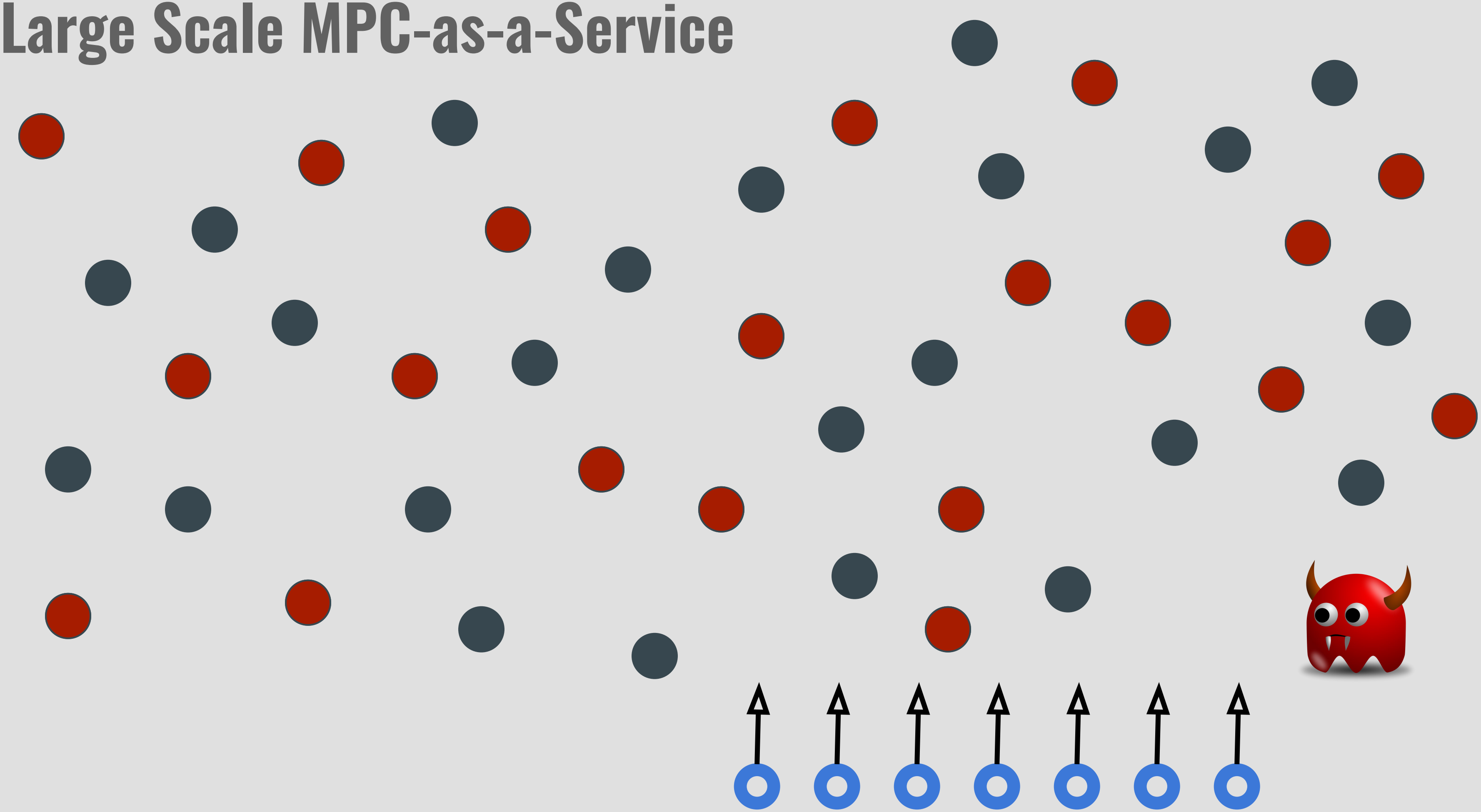
Large Scale MPC-as-a-Service



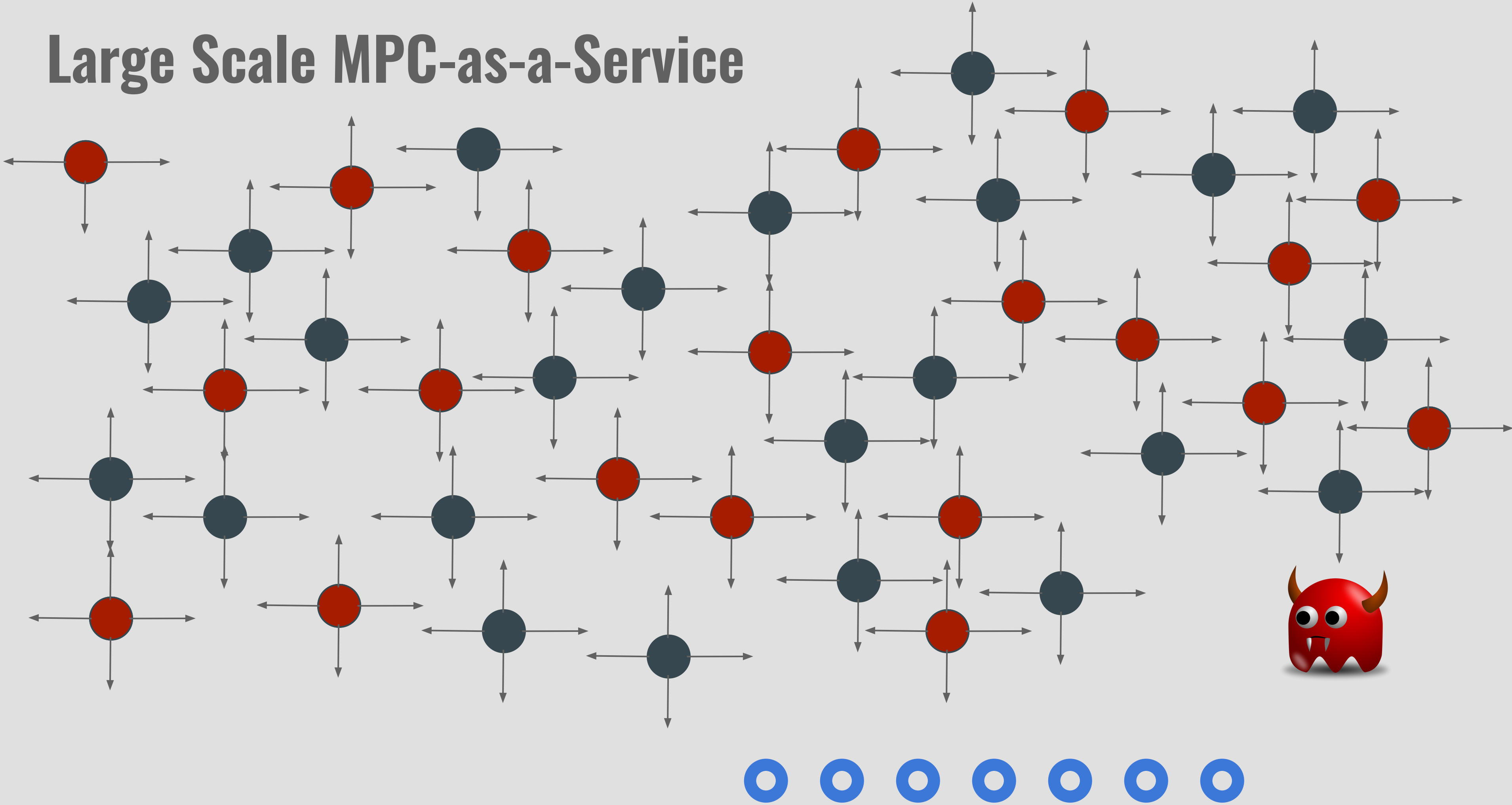
Large Scale MPC-as-a-Service



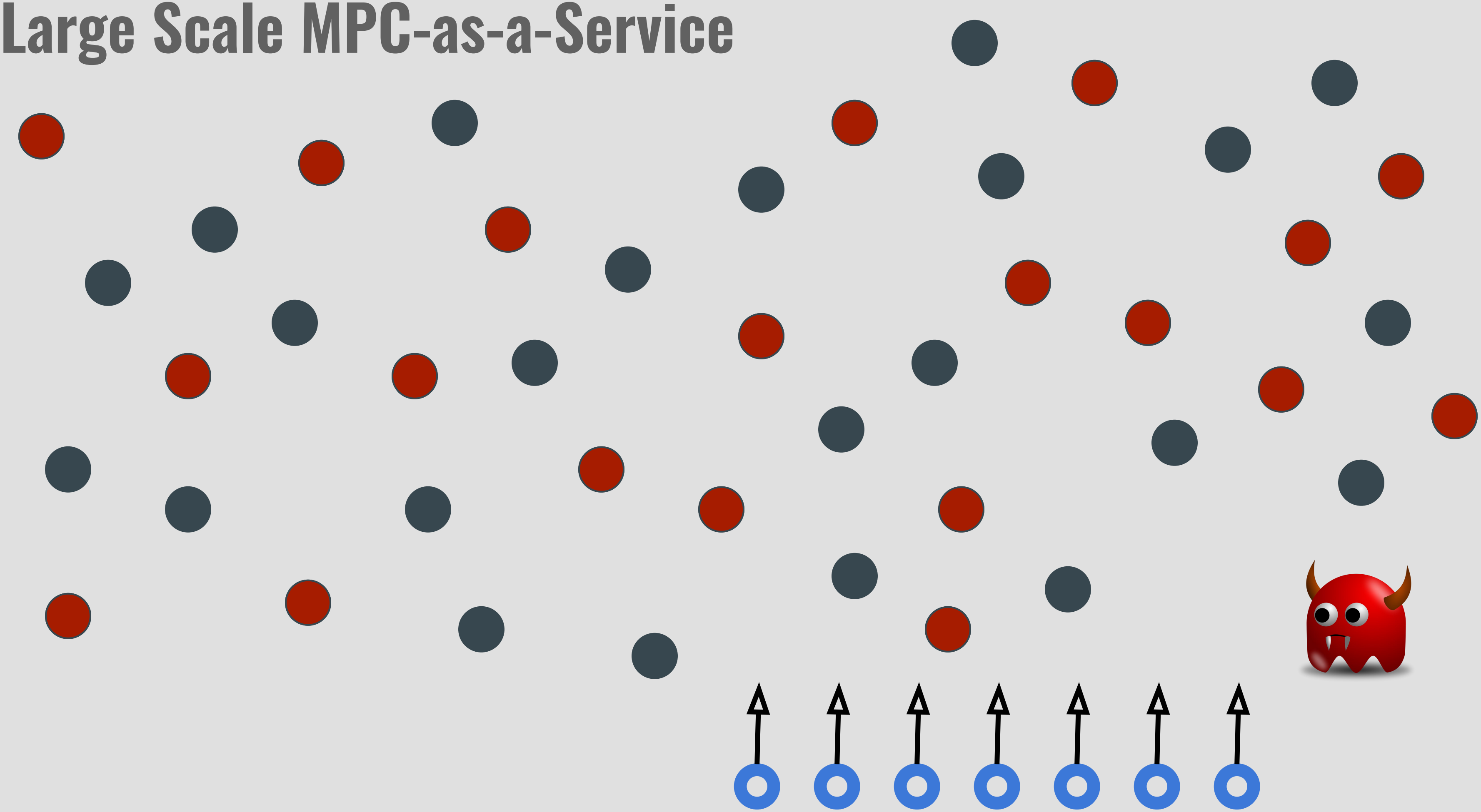
Large Scale MPC-as-a-Service



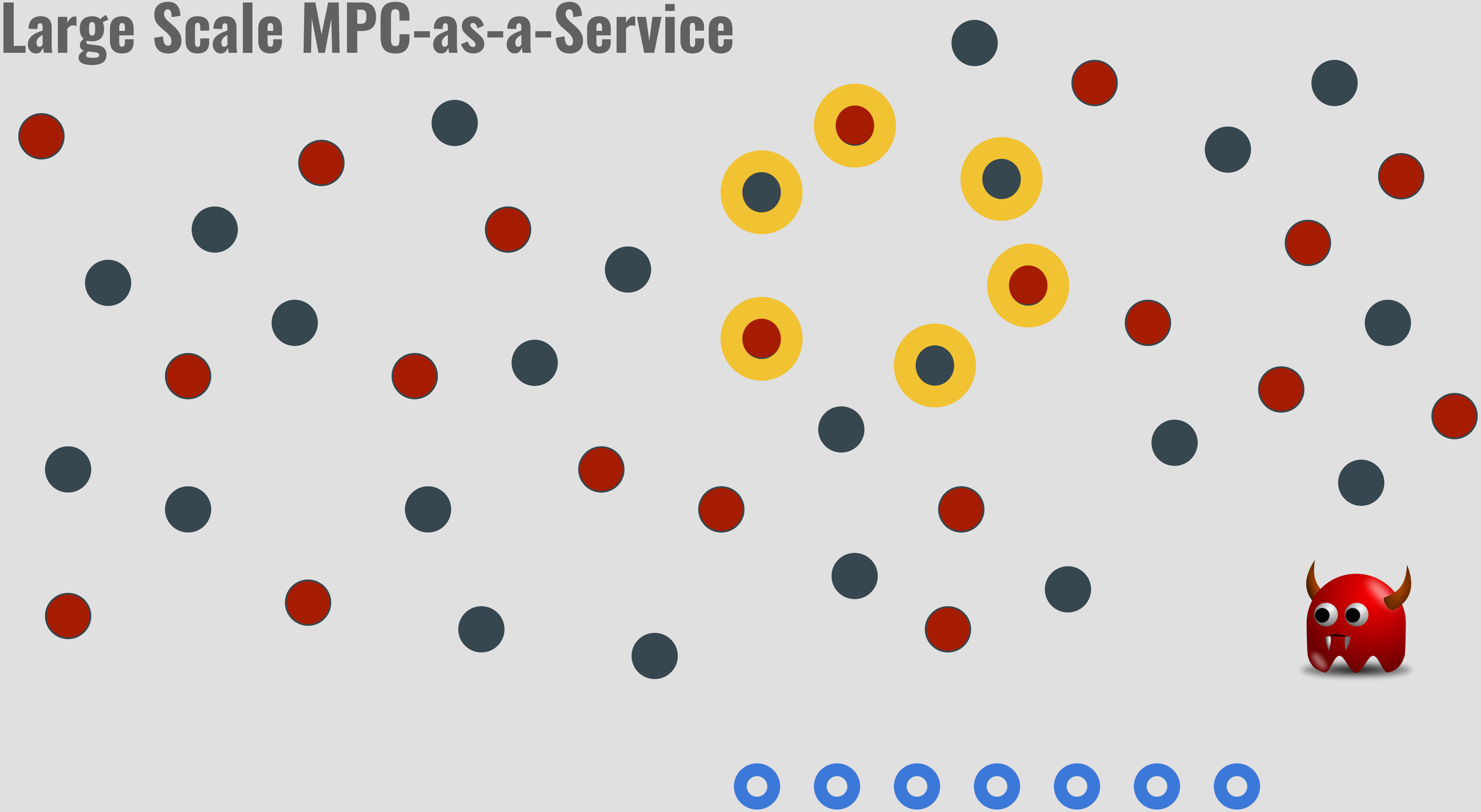
Large Scale MPC-as-a-Service



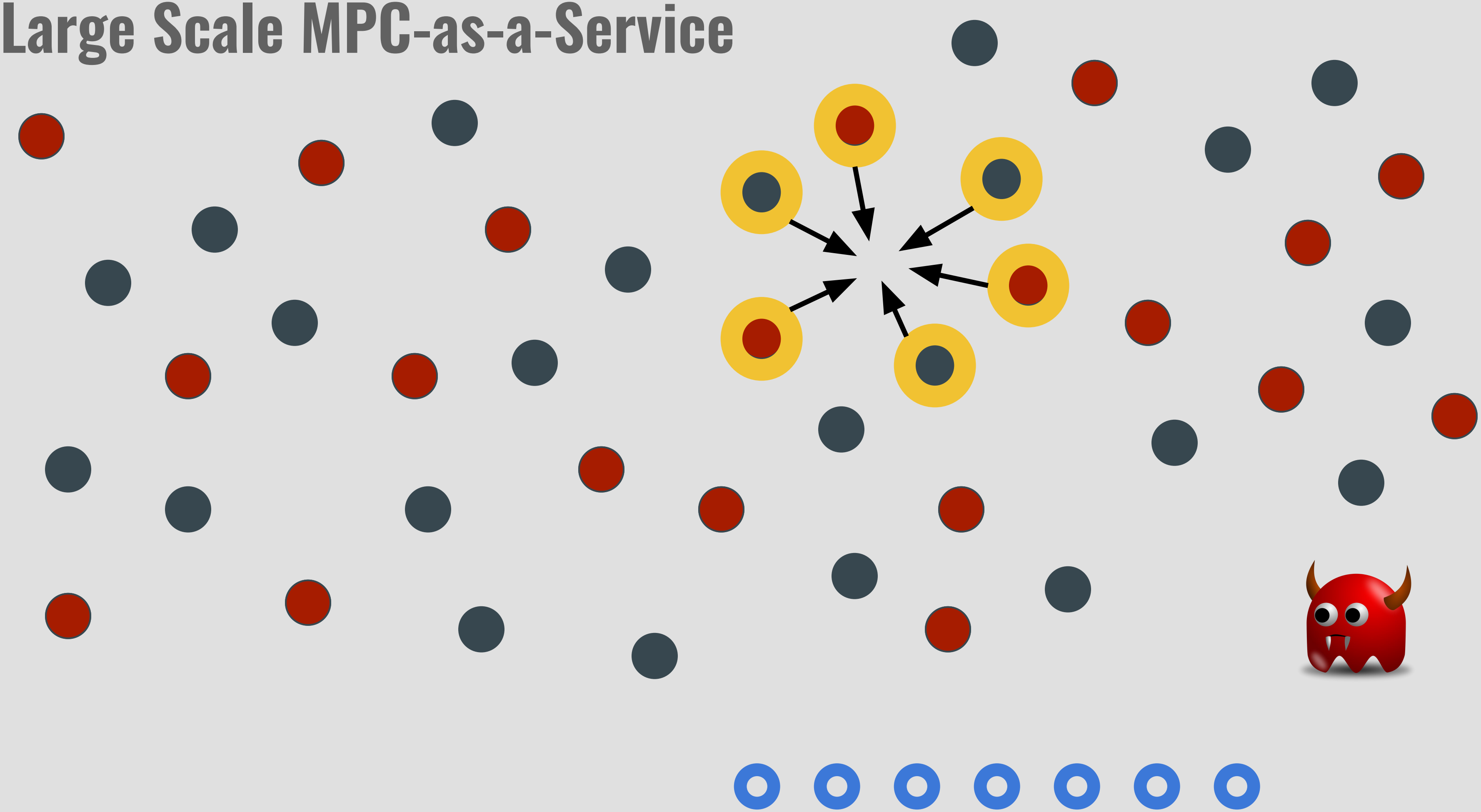
Large Scale MPC-as-a-Service



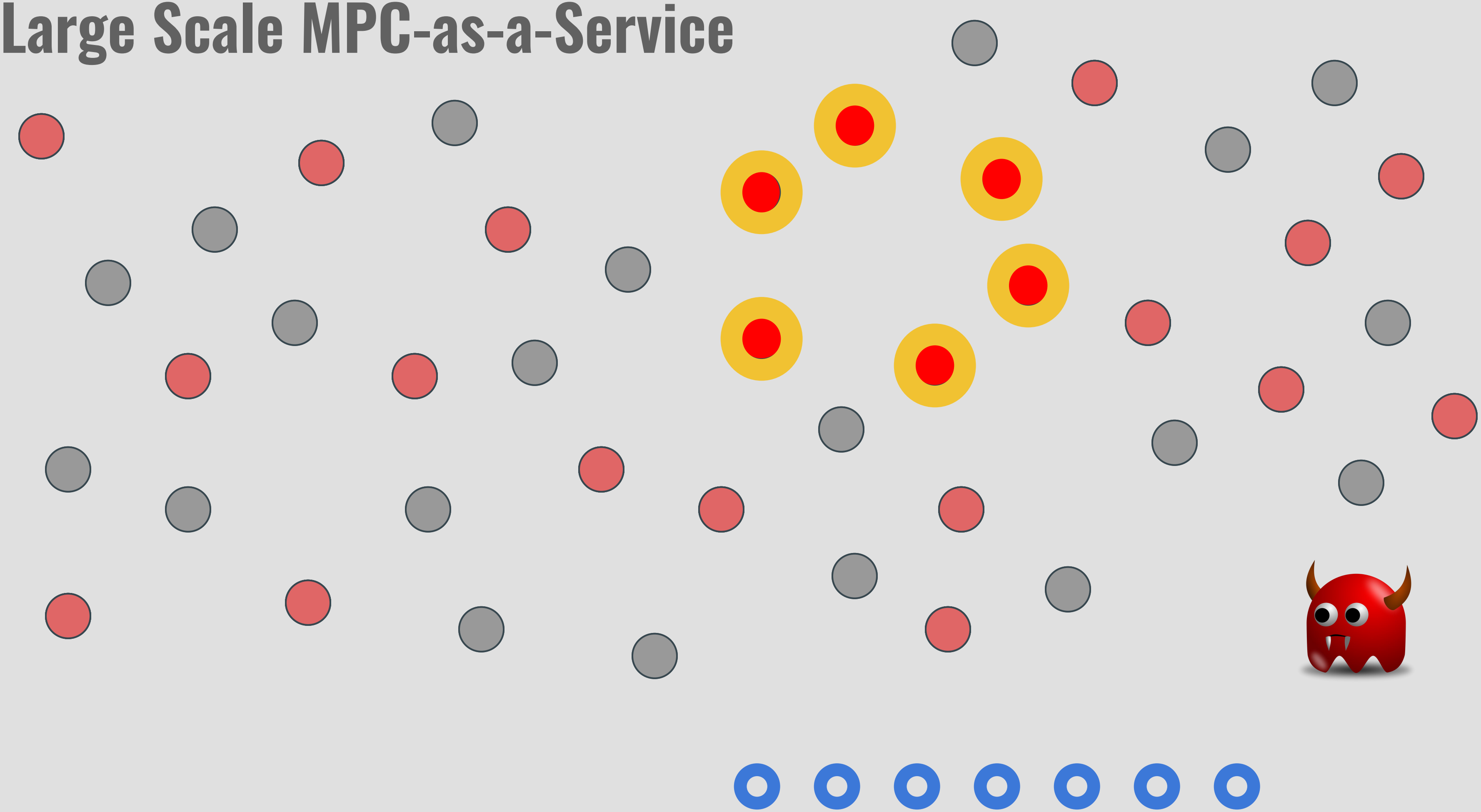
Large Scale MPC-as-a-Service



Large Scale MPC-as-a-Service



Large Scale MPC-as-a-Service



Overview

SCALES Model

Rerandomizable Garbling Schemes

Semi-Honest SCALES Protocol

Lifting to Malicious Security

The SCALES Model

[TCC ’22]

Small **C**lients **A**nd **L**arge **E**phemeral **S**ervers

Small **C**lients **A**nd **L**arge **E**phemeral **S**ervers

Small **C**lients **A**nd **L**arge **E**phemeral **S**ervers

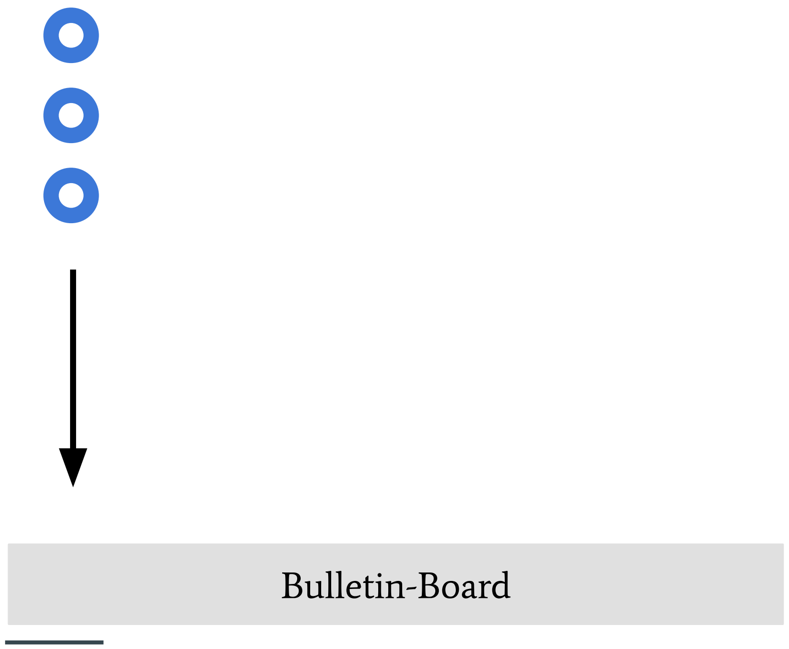
Small Clients And Large Ephemeral Servers

Bulletin-Board

Small Clients And Large Ephemeral Servers

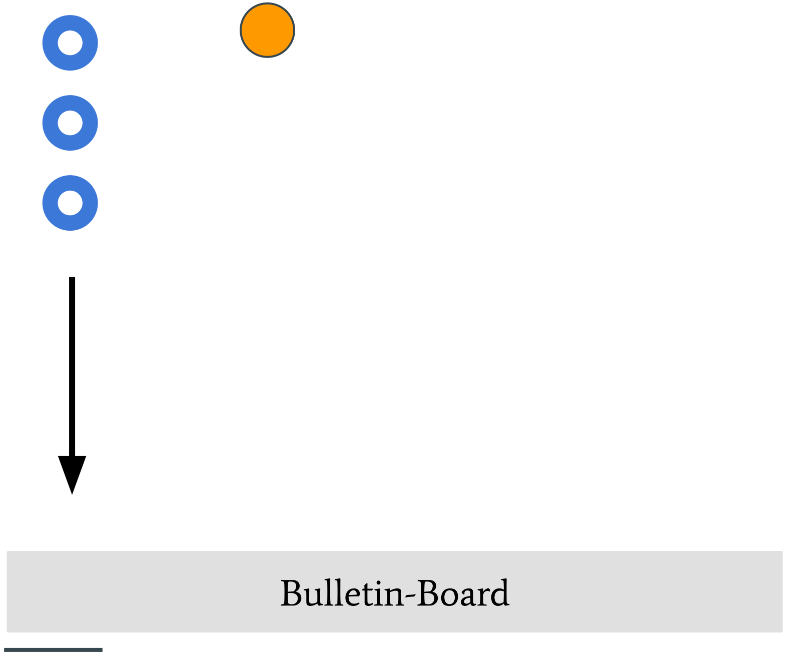
Bulletin-Board

Small Clients And Large Ephemeral Servers



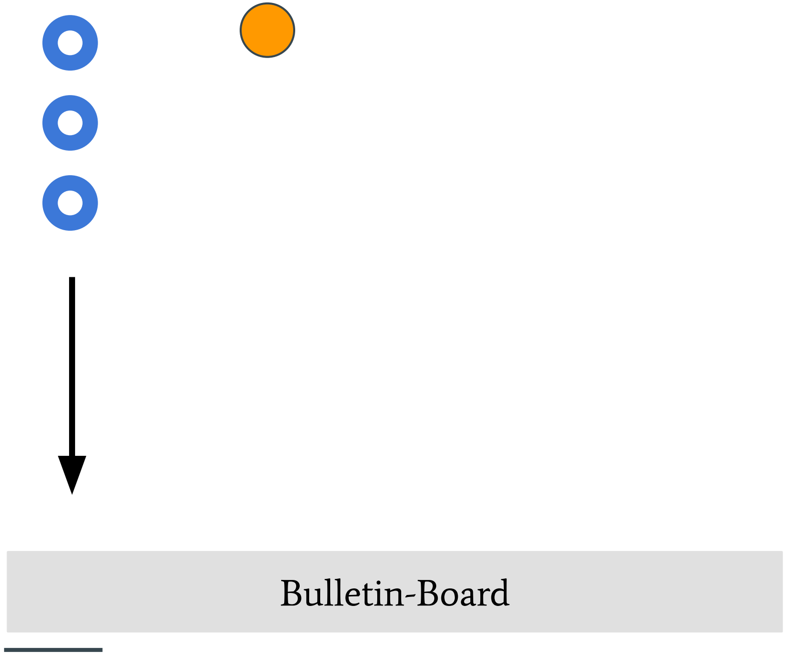
Bulletin-Board

Small Clients And Large Ephemeral Servers



Small Clients And Large Ephemeral Servers

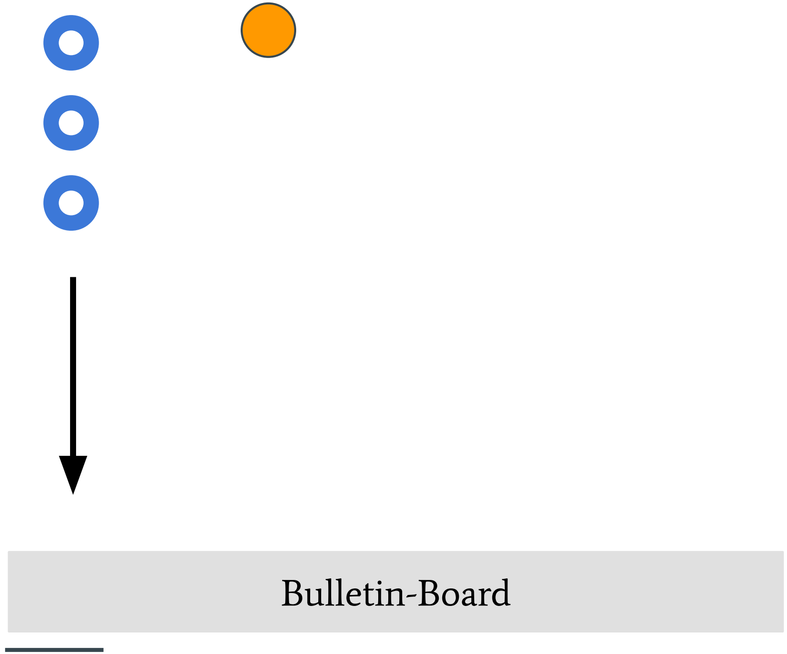
Secretly elected

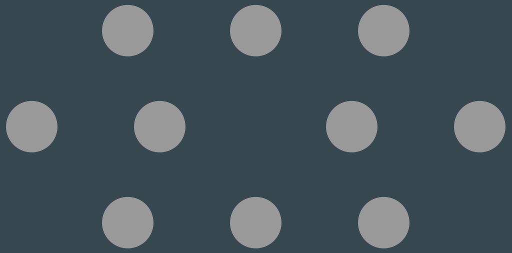


Bulletin-Board

Small Clients And Large Ephemeral Servers

Secretly elected
Silently compute



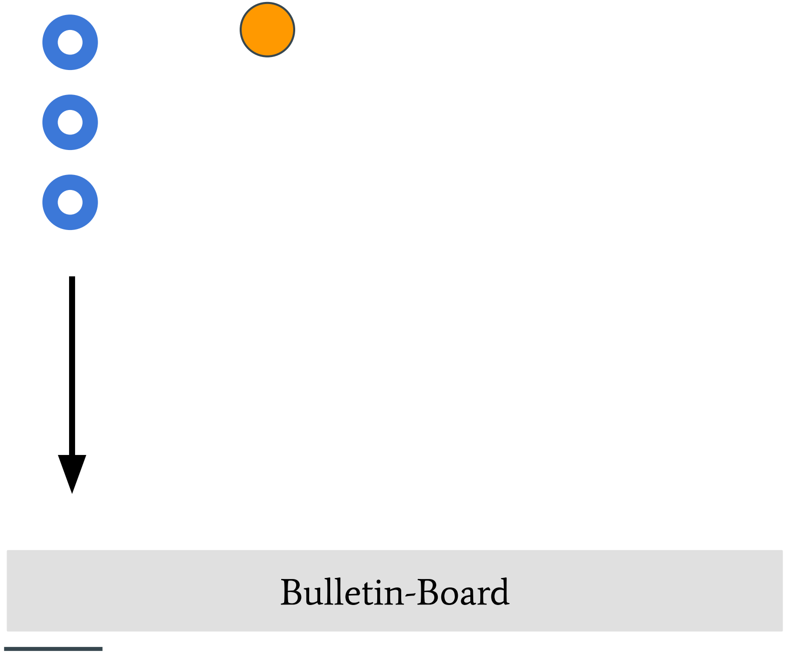


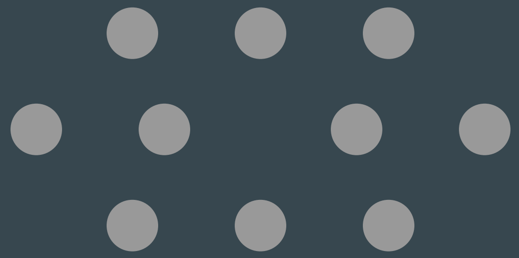
Small Clients And Large Ephemeral Servers

Secretly elected

Silently compute

Erase their state





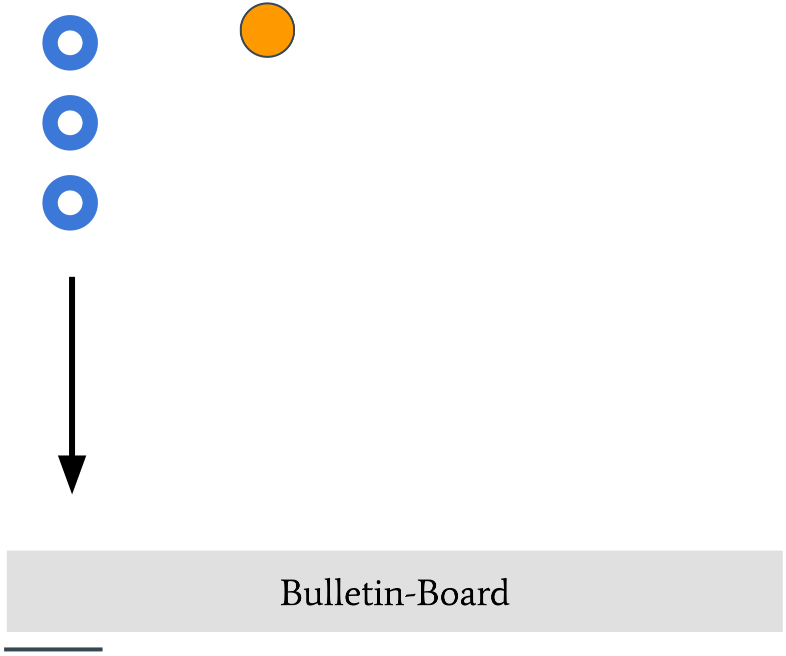
Small Clients And Large Ephemeral Servers

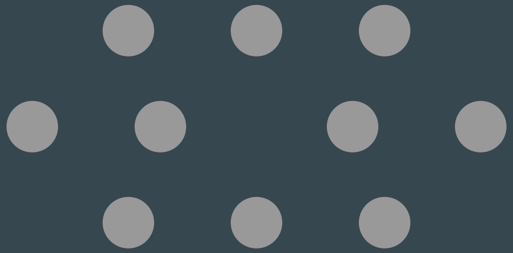
Secretly elected

Silently compute

Erase their state

'Speak Once'





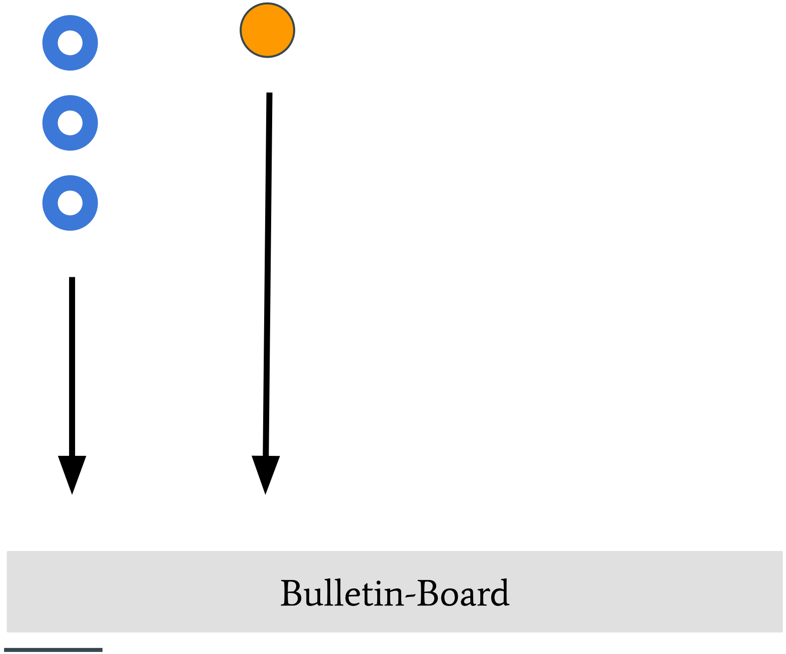
Small Clients And Large Ephemeral Servers

Secretly elected

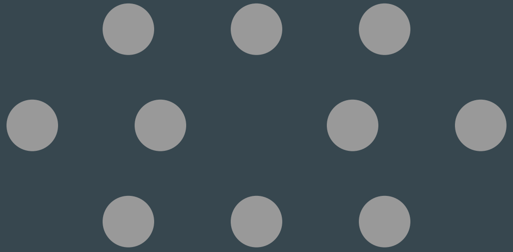
Silently compute

Erase their state

'Speak Once'



Bulletin-Board



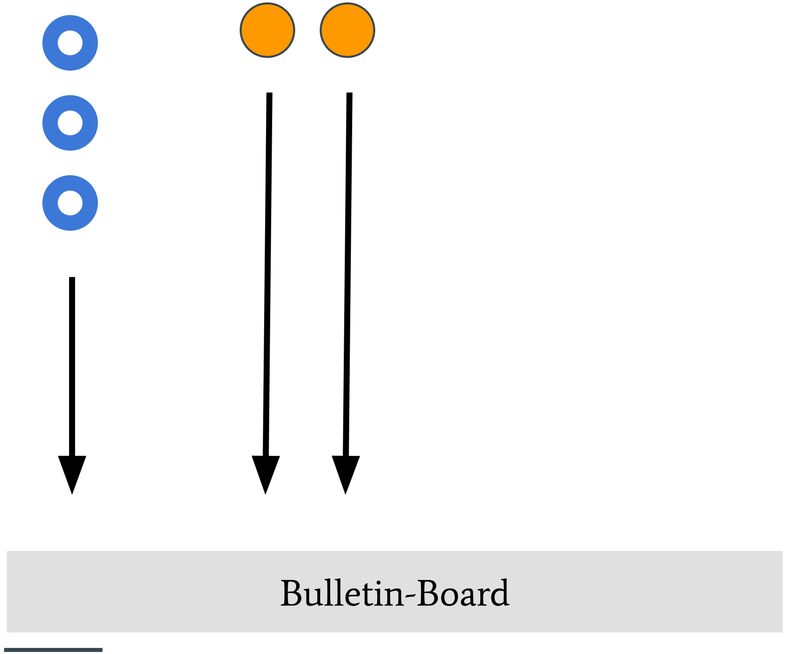
Small Clients And Large Ephemeral Servers

Secretly elected

Silently compute

Erase their state

'Speak Once'



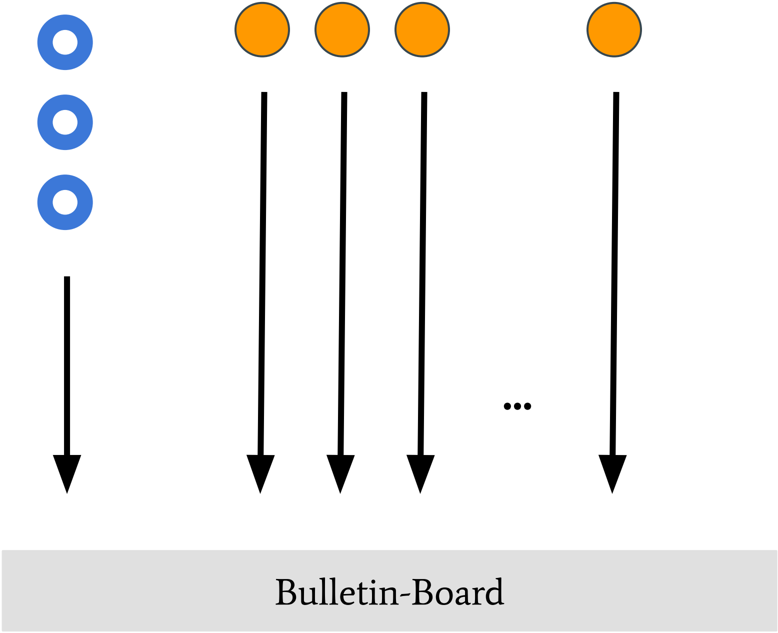
Small Clients And Large Ephemeral Servers

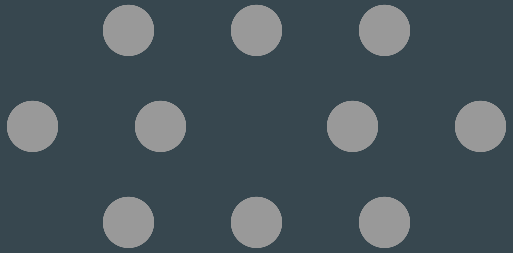
Secretly elected

Silently compute

Erase their state

`Speak Once’

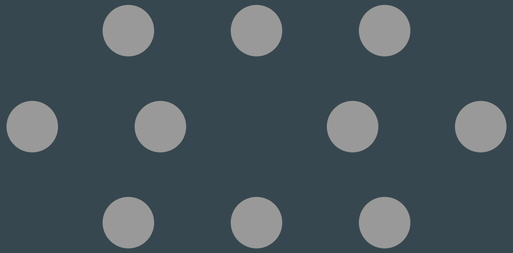




Small Clients And Large Ephemeral Servers

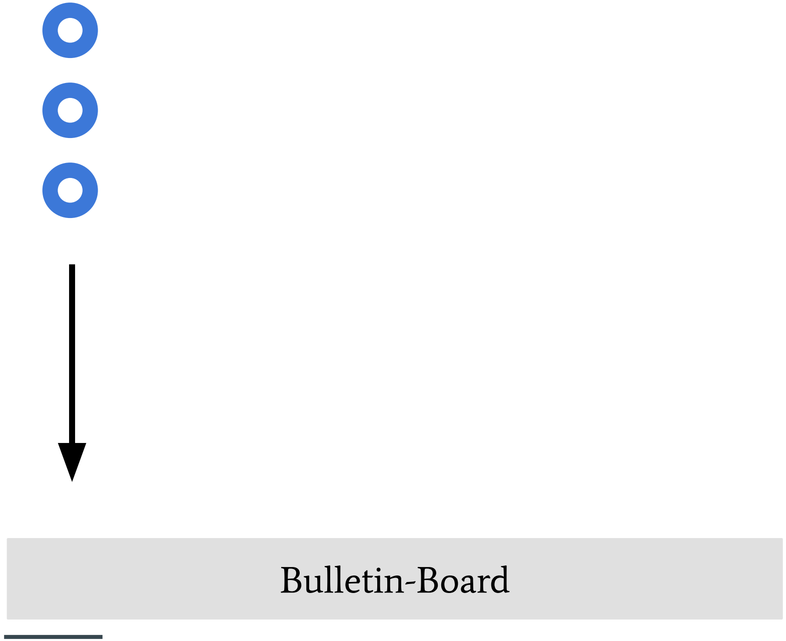
Secretly elected
Silently compute
Erase their state
`Speak Once'

Bulletin-Board



Small Clients And Large Ephemeral Servers

Secretly elected
Silently compute
Erase their state
`Speak Once'



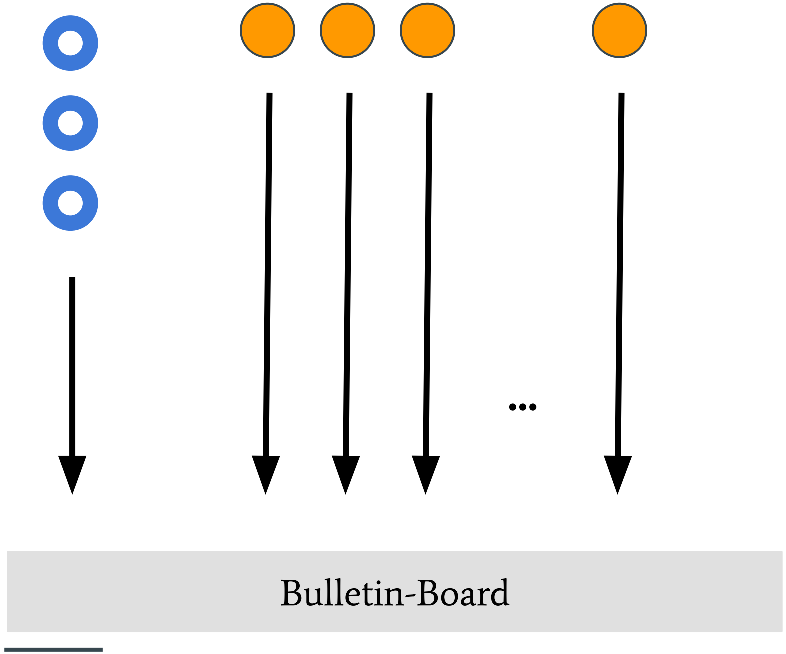
Small Clients And Large Ephemeral Servers

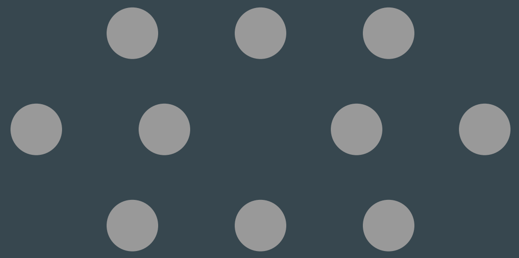
Secretly elected

Silently compute

Erase their state

‘Speak Once’





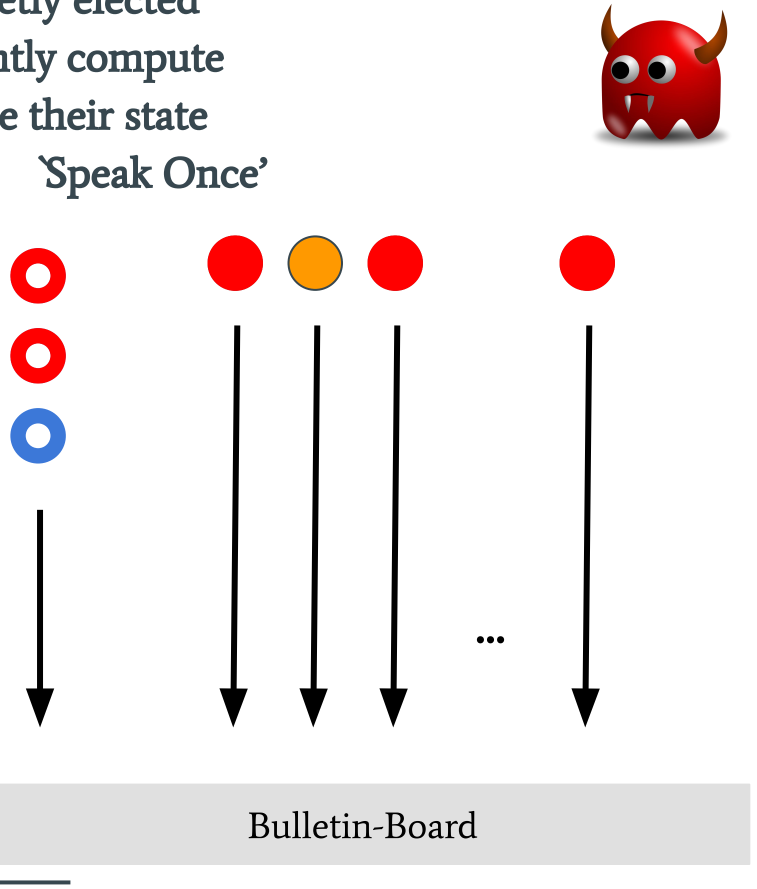
Small Clients And Large Ephemeral Servers

Secretly elected

Silently compute

Erase their state

'Speak Once'



Semi-Honest SCALES Protocol

Garbled Circuits

Garbled Circuits

Garbler

Evaluator

Garbled Circuits

Garbler

$f : \{ 0,1 \}^n \rightarrow \{ 0,1 \}^m$

$x = (x_1 , \dots , x_n)$

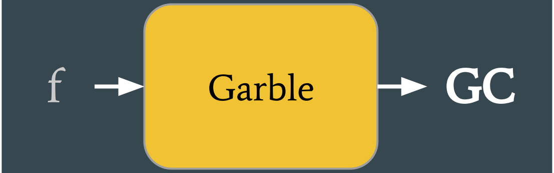
Evaluator

Garbled Circuits

Garbler

$f : \{ 0,1 \}^n \rightarrow \{ 0,1 \}^m$

$x = (x_1 , \dots , x_n)$



f

Garble

GC

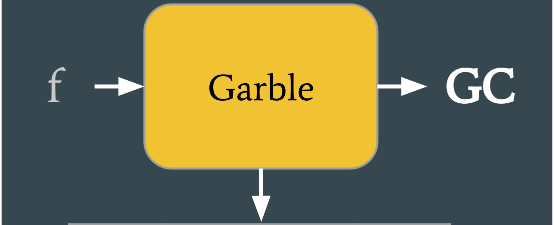
Evaluator

Garbled Circuits

Garbler

$f : \{ 0,1 \}^n \rightarrow \{ 0,1 \}^m$

$x = (x_1 , \dots , x_n)$



L^1_0	L^2_0	…	L^n_0
L^1_1	L^2_1	…	L^n_1

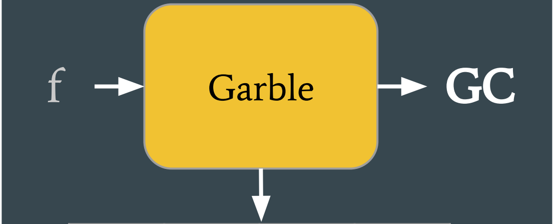
Evaluator

Garbled Circuits

Garbler

$$f : \{ 0,1 \}^n \rightarrow \{ 0,1 \}^m$$

$$x = (x_1 , \dots , x_n)$$



L^1_0	L^2_0	…	L^n_0
L^1_1	**L^2_1**	…	**L^n_1**

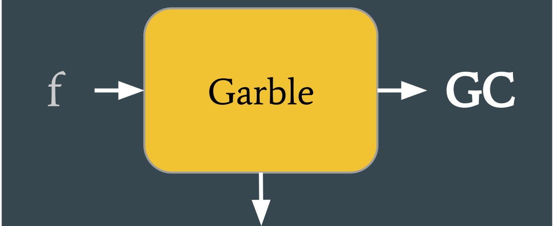
Evaluator

Garbled Circuits

Garbler

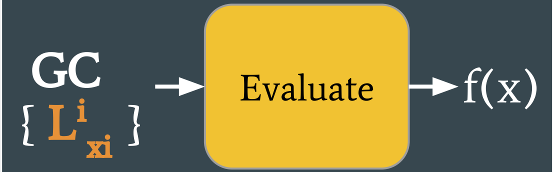
$$f : \{ 0,1 \}^n \rightarrow \{ 0,1 \}^m$$

$$x = (x_1 , \dots , x_n)$$



L^1_0	L^2_0	…	L^n_0
L^1_1	L^2_1	…	L^n_1

Evaluator

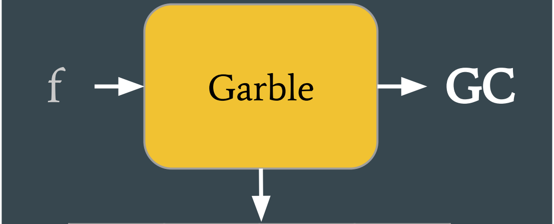


Rerandomizable Garbled Circuits

Garbler

$f : \{ 0,1 \}^n \rightarrow \{ 0,1 \}^m$

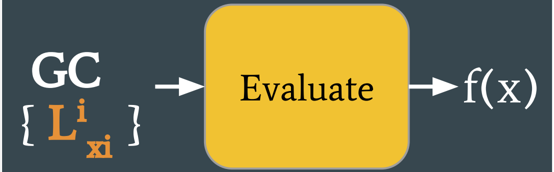
$x = (x_1 , \ldots , x_n)$



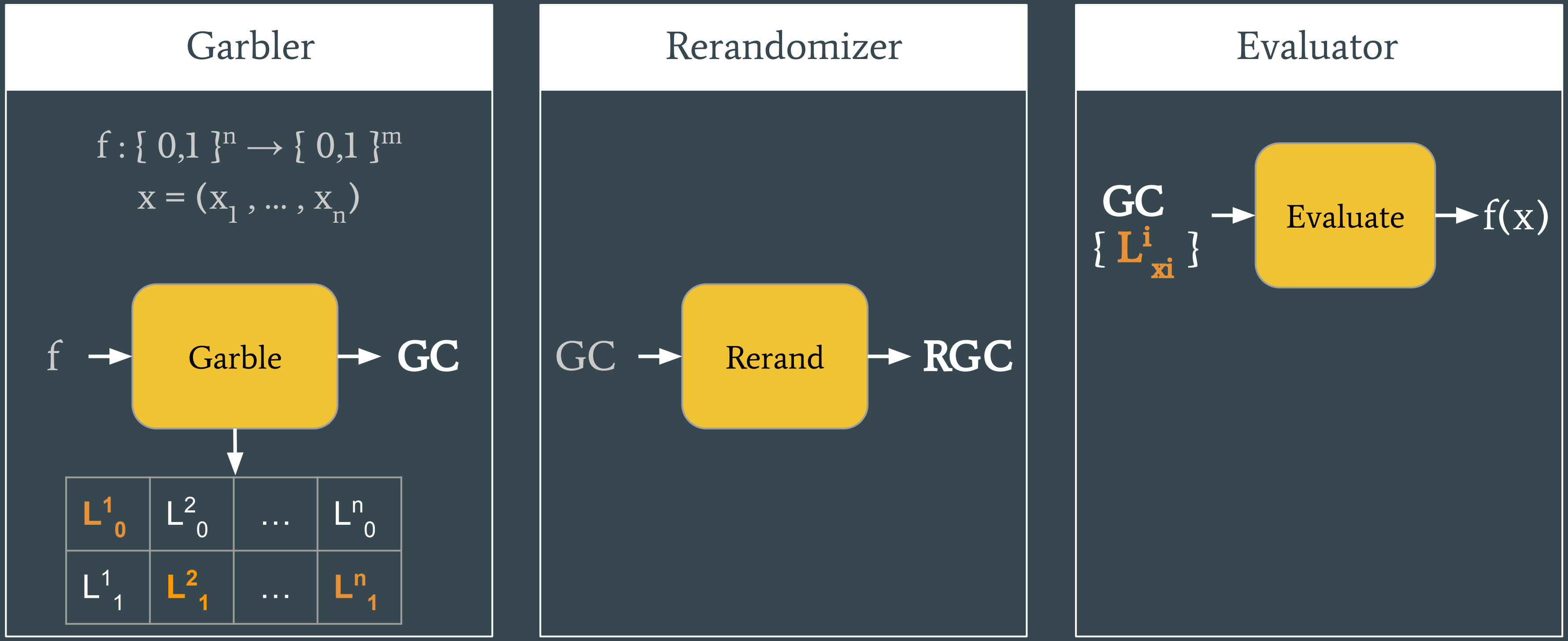
L^1_0	L^2_0	…	L^n_0
L^1_1	L^2_1	…	L^n_1

Rerandomizer

Evaluator



Rerandomizable Garbled Circuits



Garbler

$f : \{ 0,1 \}^n \to \{ 0,1 \}^m$

$x = (x_1 , \dots , x_n)$

f → Garble → GC

L^1_0	L^2_0	…	L^n_0
L^1_1	L^2_1	…	L^n_1

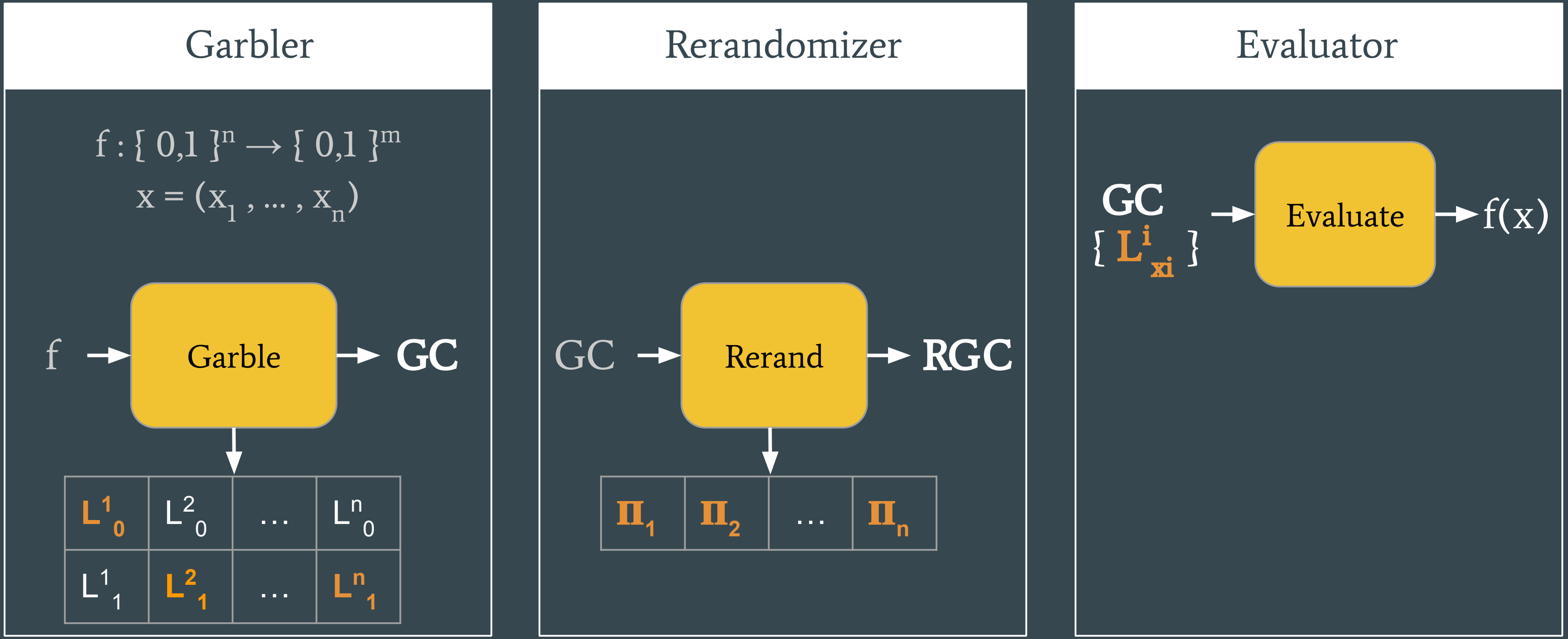
Rerandomizer

GC → Rerand → RGC

Evaluator

GC $\{ L^i_{xi} \}$ → Evaluate → f(x)

Rerandomizable Garbled Circuits



Garbler

$f : \{0,1\}^n \rightarrow \{0,1\}^m$

$x = (x_1 , \dots , x_n)$

$f \rightarrow$ Garble $\rightarrow$ GC

L^1_0	L^2_0	…	L^n_0
L^1_1	L^2_1	…	L^n_1

Rerandomizer

GC $\rightarrow$ Rerand $\rightarrow$ RGC

π_1	π_2	…	π_n

Evaluator

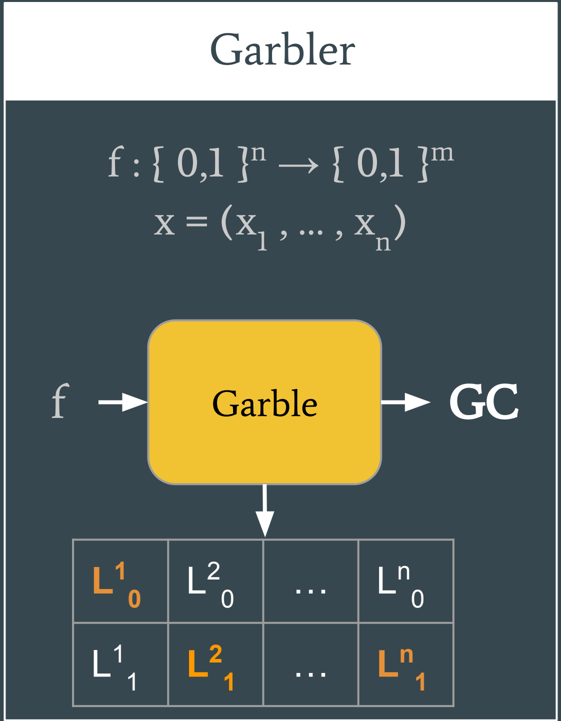
GC $\{ L^i_{xi} \}$ $\rightarrow$ Evaluate $\rightarrow f(x)$

Rerandomizable Garbled Circuits

Garbler

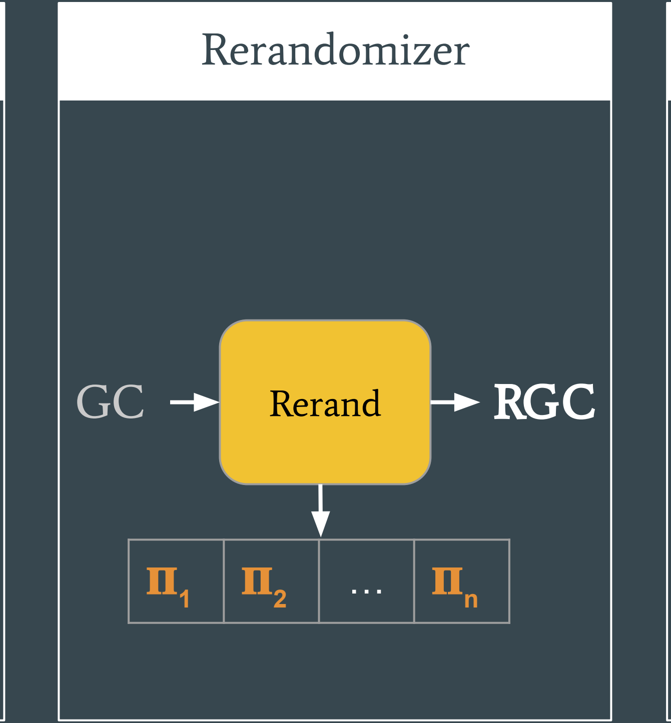
$f : \{0,1\}^n \rightarrow \{0,1\}^m$

$x = (x_1, \dots, x_n)$



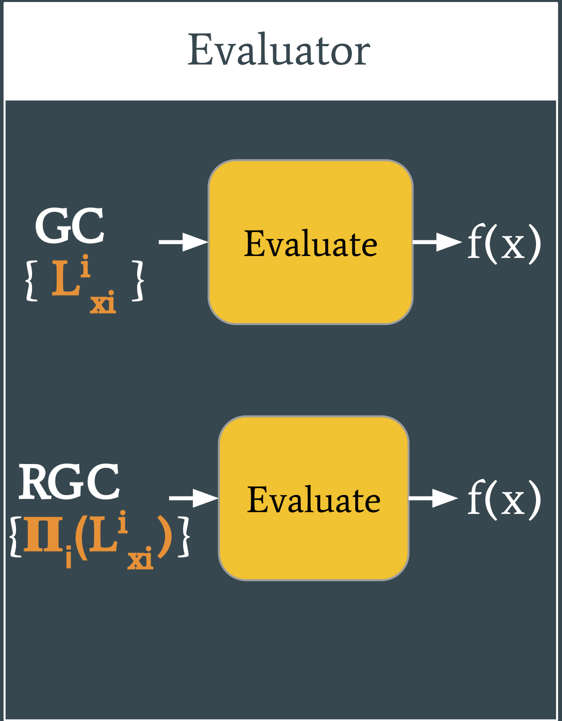
L^1_0	L^2_0	…	L^n_0
L^1_1	L^2_1	…	L^n_1

Rerandomizer



Π_1	Π_2	…	Π_n

Evaluator

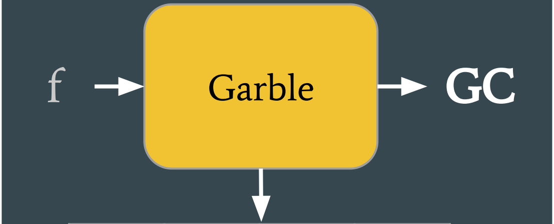


Rerandomizable Garbled Circuits

Garbler

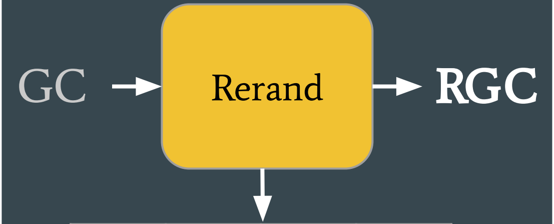
$f : \{ 0,1 \}^n \rightarrow \{ 0,1 \}^m$

$x = (x_1 , \dots , x_n)$



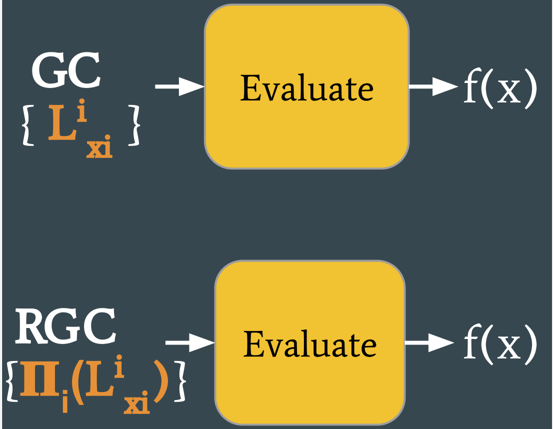
L^1_0	L^2_0	…	L^n_0
L^1_1	L^2_1	…	L^n_1

Rerandomizer



Π_1	Π_2	…	Π_n

Evaluator



Semi-Honest Protocol

Semi-Honest Protocol

Phase 1

Garble the circuit

Phase 2

Evaluate the garbling

Bulletin-Board

Semi-Honest Protocol

Phase 1

Garble the circuit

Phase 2

Evaluate the garbling

Bulletin-Board

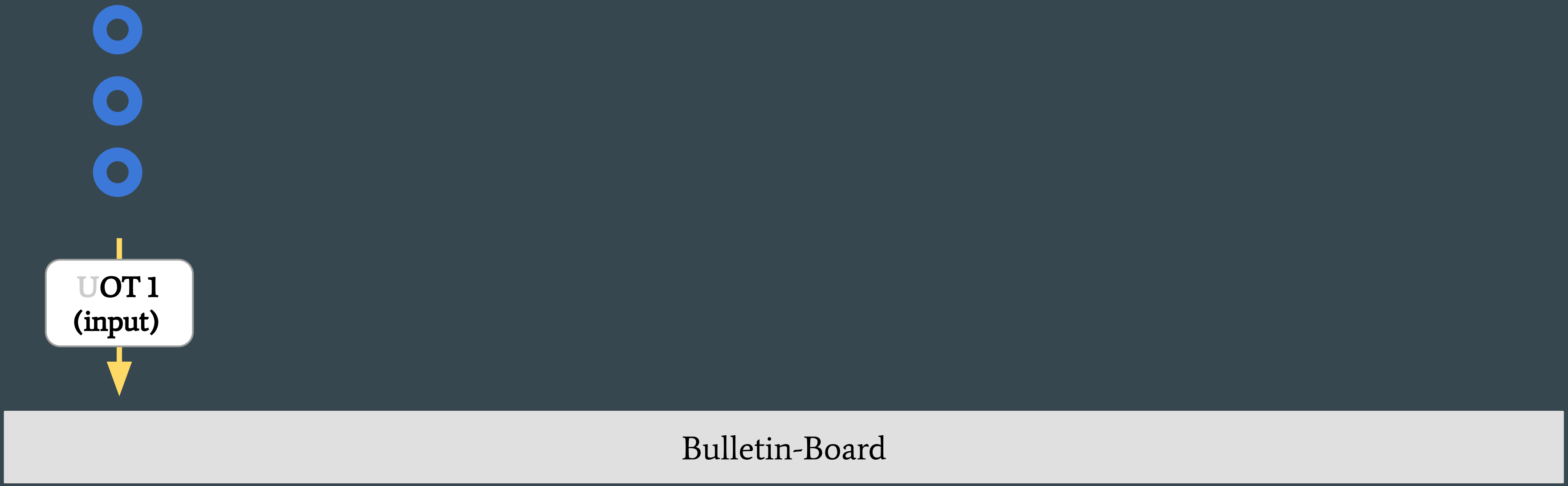
Semi-Honest Protocol

Phase 1

Garble the circuit

Phase 2

Evaluate the garbling



UOT 1
(input)

Bulletin-Board

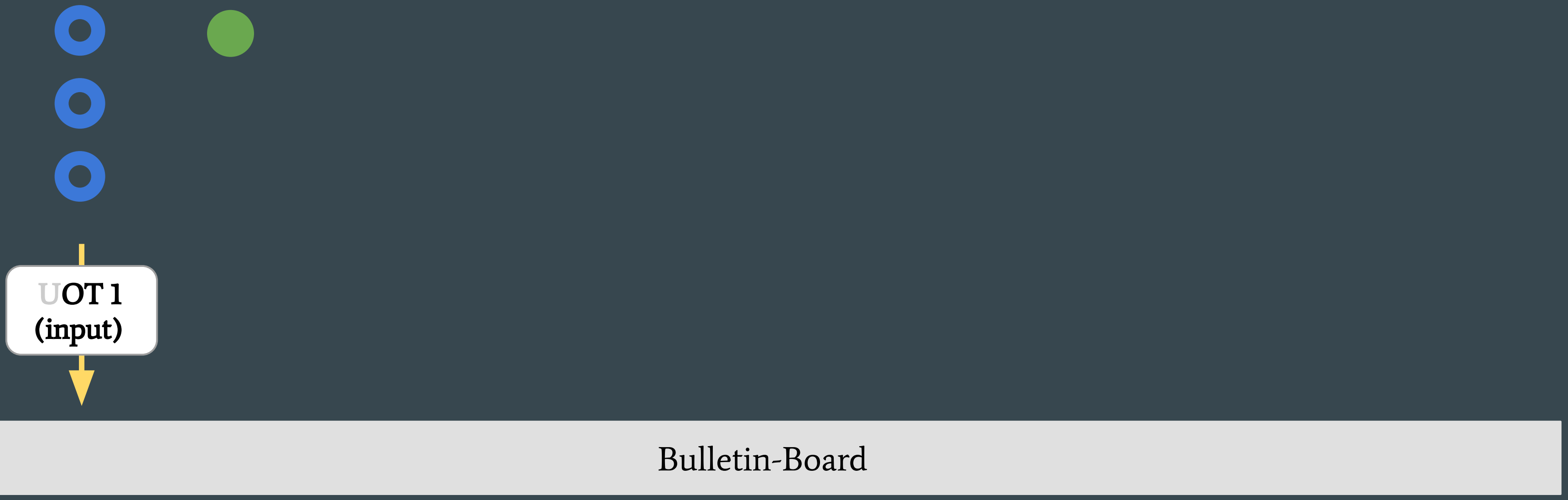
Semi-Honest Protocol

Phase 1

Garble the circuit

Phase 2

Evaluate the garbling



UOT 1
(input)

Bulletin-Board

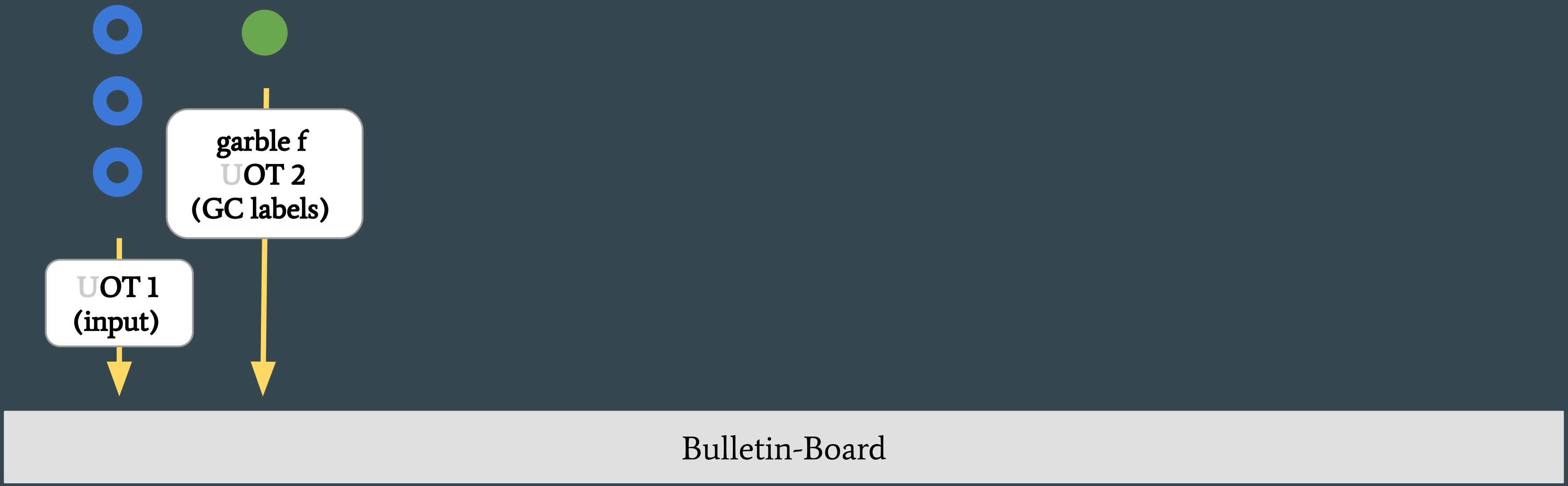
Semi-Honest Protocol

Phase 1

Garble the circuit

Phase 2

Evaluate the garbling



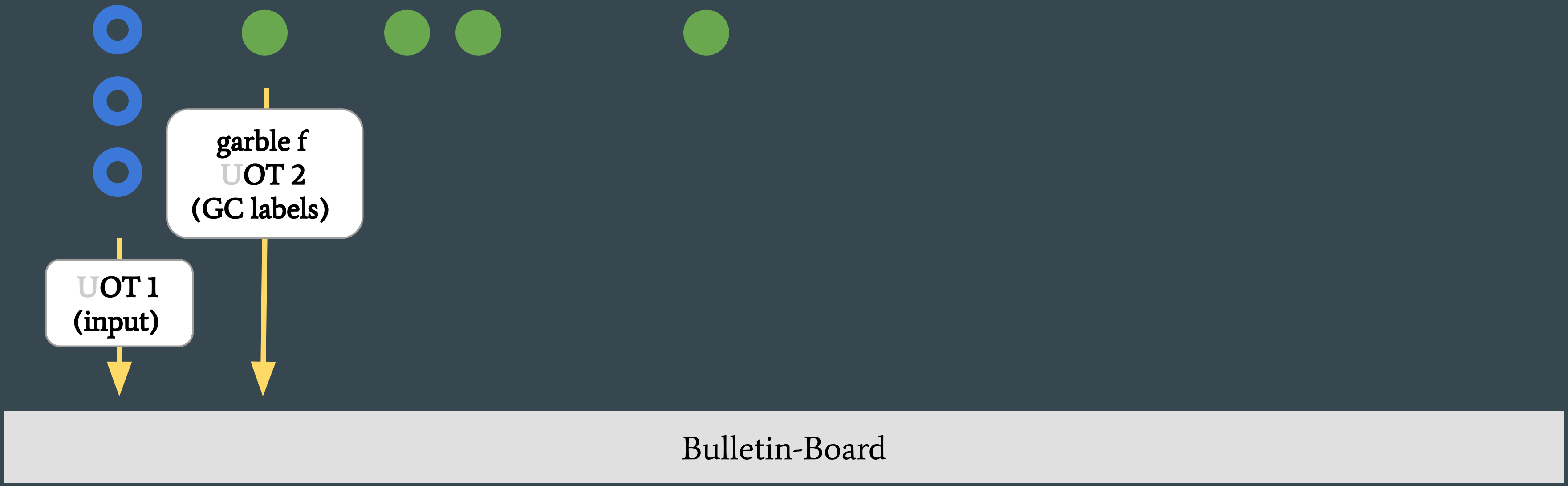
Semi-Honest Protocol

Phase 1

Garble the circuit

Phase 2

Evaluate the garbling



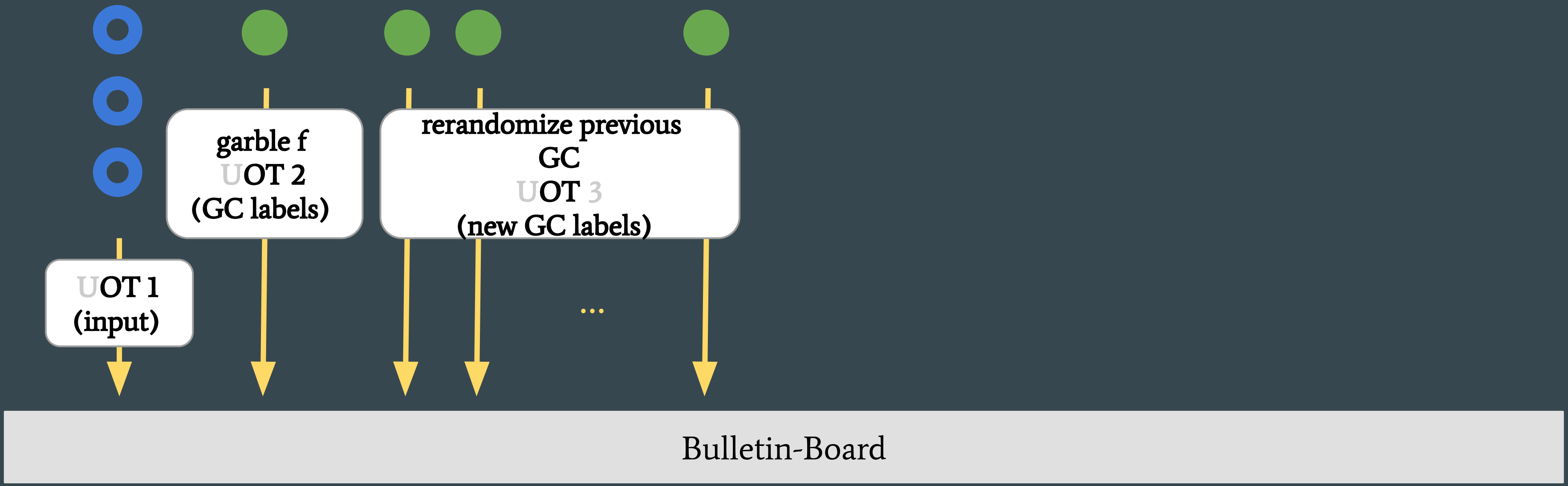
Semi-Honest Protocol

Phase 1

Garble the circuit

Phase 2

Evaluate the garbling



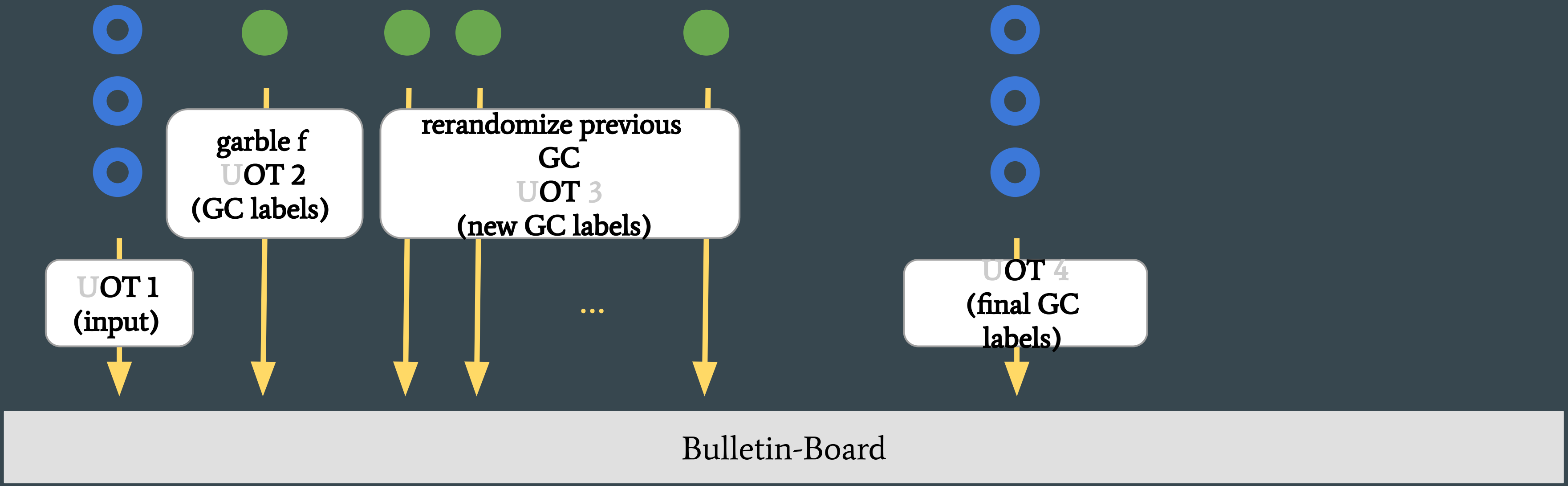
Semi-Honest Protocol

Phase 1

Garble the circuit

Phase 2

Evaluate the garbling



UOT 1
(input)

garble f
UOT 2
(GC labels)

rerandomize previous
GC
UOT 3
(new GC labels)

...

UOT 4
(final GC
labels)

Bulletin-Board

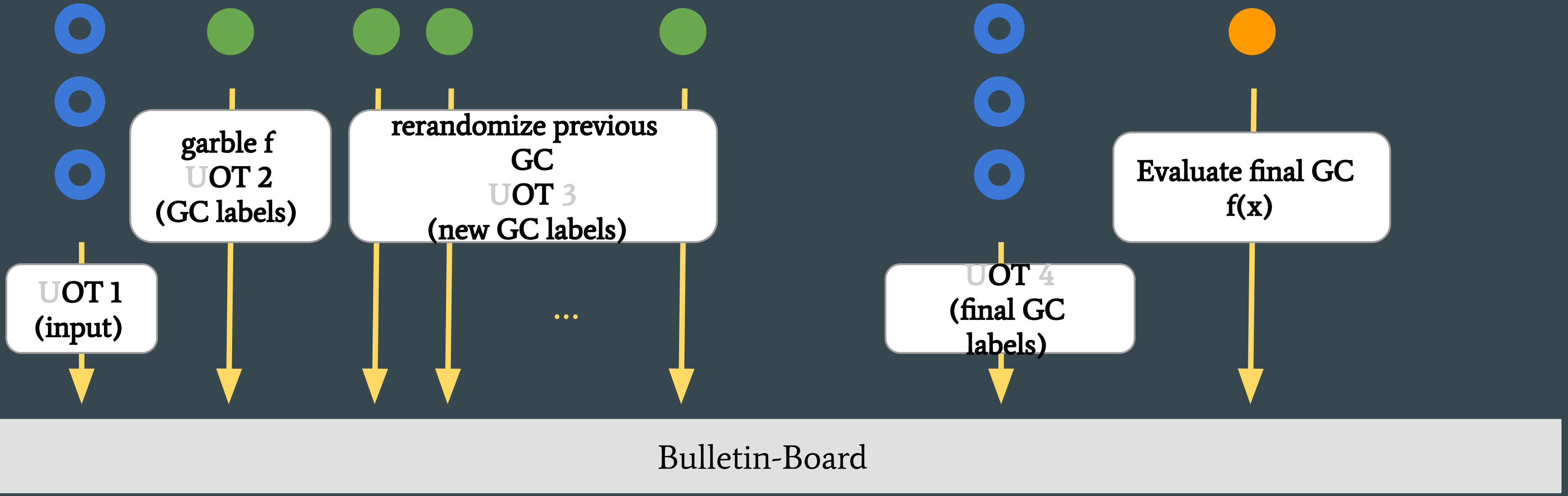
Semi-Honest Protocol

Phase 1

Garble the circuit

Phase 2

Evaluate the garbling



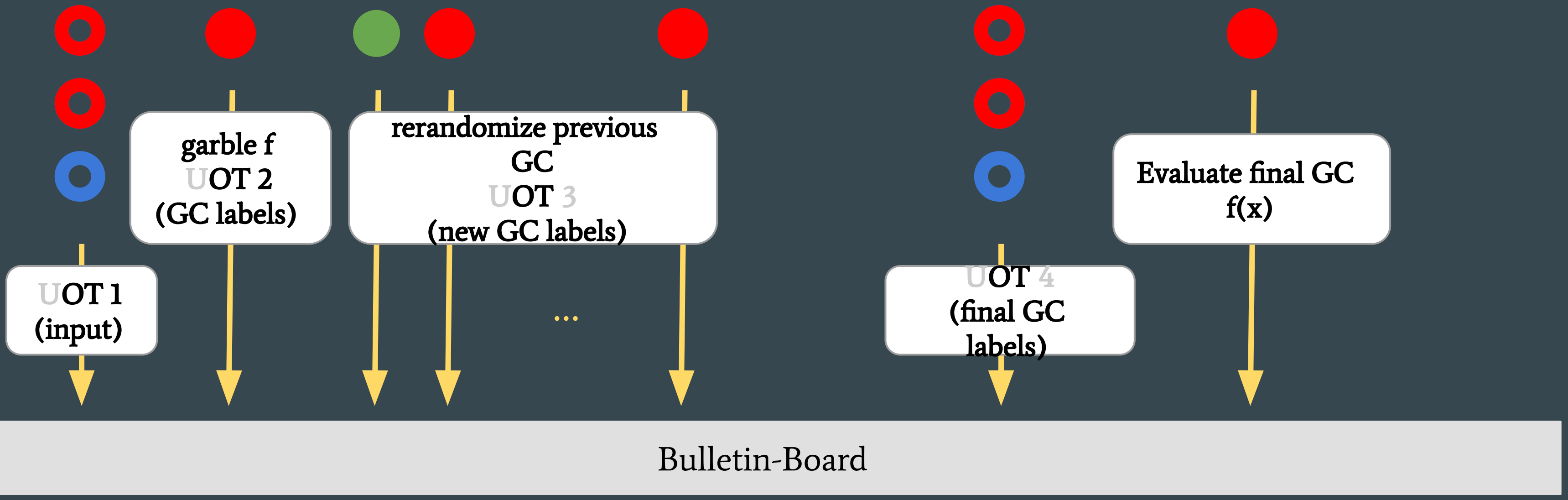
Semi-Honest Protocol

Phase 1

Garble the circuit

Phase 2

Evaluate the garbling



UOT 1
(input)

garble f
UOT 2
(GC labels)

rerandomize previous GC
UOT 3
(new GC labels)

…

UOT 4
(final GC labels)

Evaluate final GC
f(x)

Bulletin-Board

Lifting to Malicious Security

Lifting to Malicious Security

Security with Abort

Lifting to Malicious Security

Lifting to Malicious Security

Semi-honest secure UOT → Maliciously secure UOT

Lifting to Malicious Security

Semi-honest secure UOT → Maliciously secure UOT

2-round OT (with special structure) : CRS model

Lifting to Malicious Security

Semi-honest secure UOT → Maliciously secure UOT

2-round OT (with special structure) : CRS model

Forcing Semi-honest behaviour

Lifting to Malicious Security

Semi-honest secure UOT → Maliciously secure UOT

2-round OT (with special structure) : CRS model

Forcing Semi-honest behaviour

Approach 1: Generic SNARKs (CRS/RO model)

Lifting to Malicious Security

Semi-honest secure UOT → Maliciously secure UOT

2-round OT (with special structure) : CRS model

Forcing Semi-honest behaviour

Approach 1: Generic SNARKs (CRS/RO model)

Expensive Prover Computation

Custom Made ZK Proofs

•••

Black-Box in RGS

ZK Proof for Correct Garbling

ZK Proof for Correct Garbling

Prover

Verifier

ZK Proof for Correct Garbling

Prover	**Verifier**
Witness: randomness **r**	Public: **f, GC**

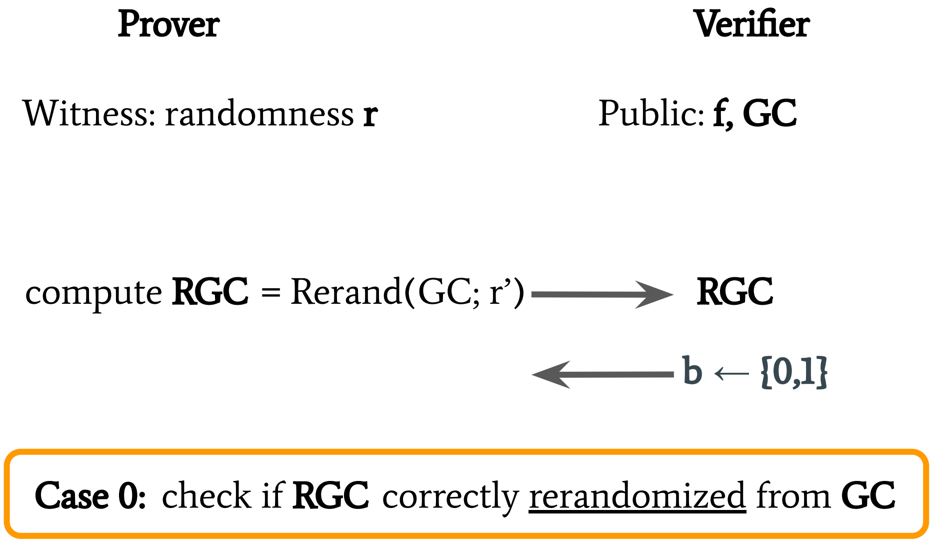
ZK Proof for Correct Garbling

Prover	**Verifier**
Witness: randomness **r**	Public: **f, GC**
compute **RGC** = Rerand(GC; r’) →	**RGC**

ZK Proof for Correct Garbling

Prover		**Verifier**
Witness: randomness **r**		Public: **f, GC**
compute **RGC** = Rerand(GC; r’)	→	**RGC**
	←	**b ← {0,1}**

ZK Proof for Correct Garbling



Prover

Verifier

Witness: randomness **r**

Public: **f, GC**

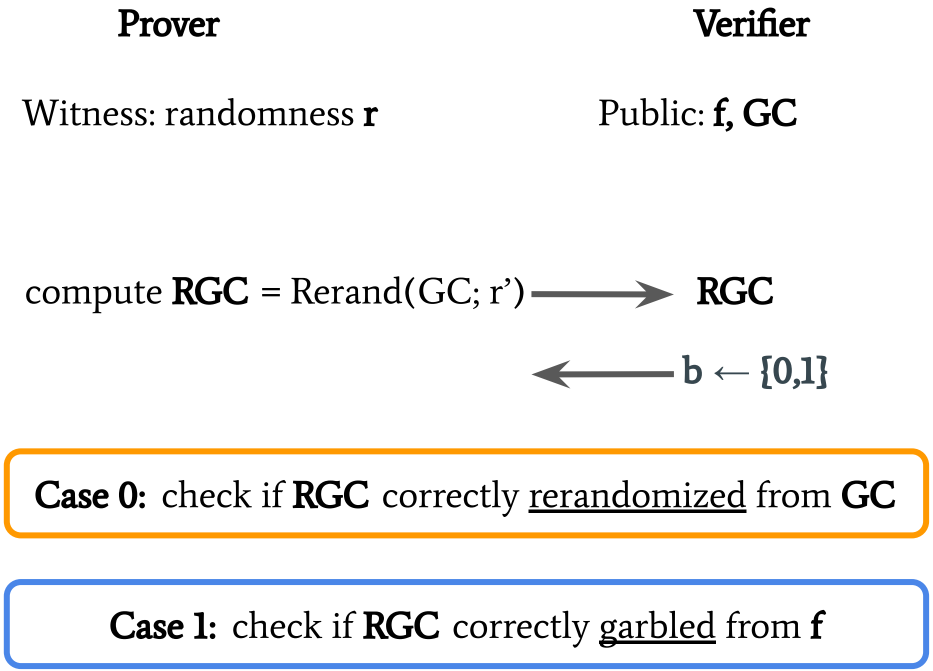
compute **RGC** = Rerand(GC; r’)

RGC

b ← {0,1}

Case 0: check if **RGC** correctly <u>rerandomized</u> from **GC**

ZK Proof for Correct Garbling



Prover

Verifier

Witness: randomness **r**

Public: **f, GC**

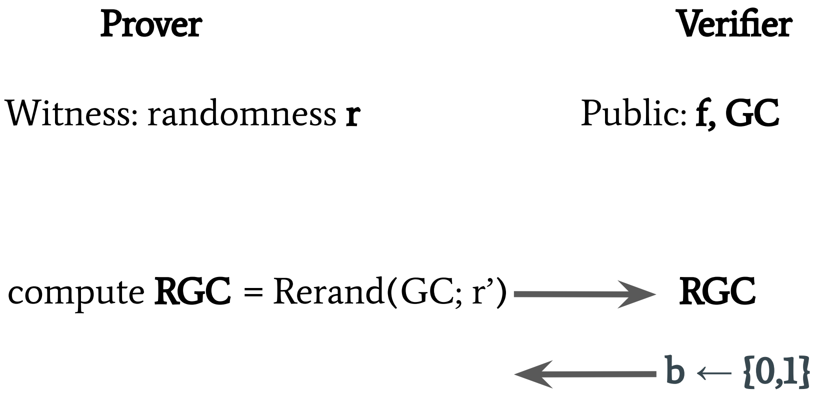
compute **RGC** = Rerand(GC; r’) → **RGC**

← b ← {0,1}

Case 0: check if **RGC** correctly <u>rerandomized</u> from **GC**

Case 1: check if **RGC** correctly <u>garbled</u> from **f**

ZK Proof for Correct Garbling



Prover | **Verifier**

Witness: randomness **r** | Public: **f, GC**

compute **RGC** = Rerand(GC; r') → **RGC**

← b ← {0,1}

Case 0: check if **RGC** correctly rerandomized from **GC**

Case 1: check if **RGC** correctly garbled from **f**

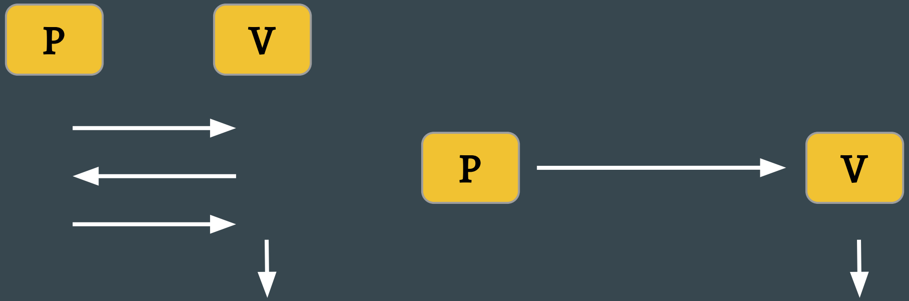
soundness ½
amplify by parallel repetition

Approach 2: Ephemeral Prover Zero-Knowledge

Approach 2: Ephemeral Prover Zero-Knowledge

Option 1

RO Model – Fiat-Shamir transform



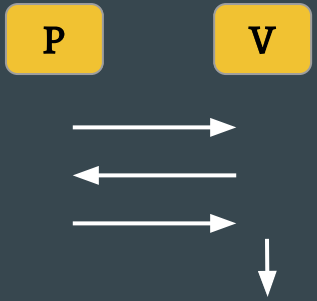
Approach 2: Ephemeral Prover Zero-Knowledge

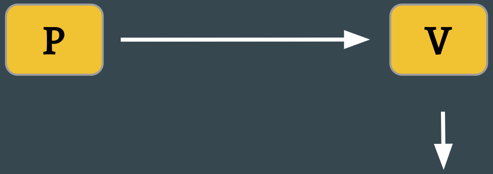
Option 1

RO Model – Fiat-Shamir transform

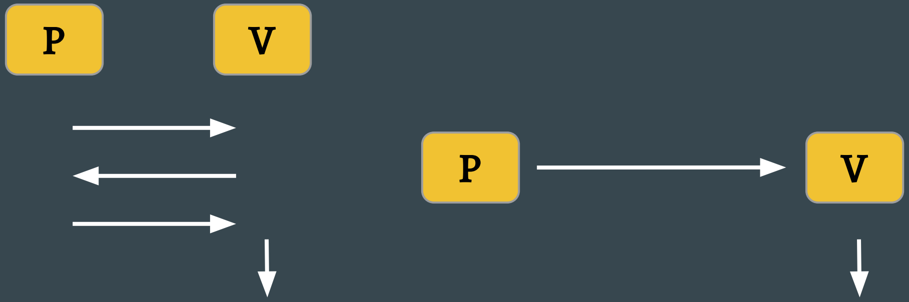
Option 2

CRS Model – Distributed Committed-Index OT





Approach 2: Ephemeral Prover Zero-Knowledge



Option 1

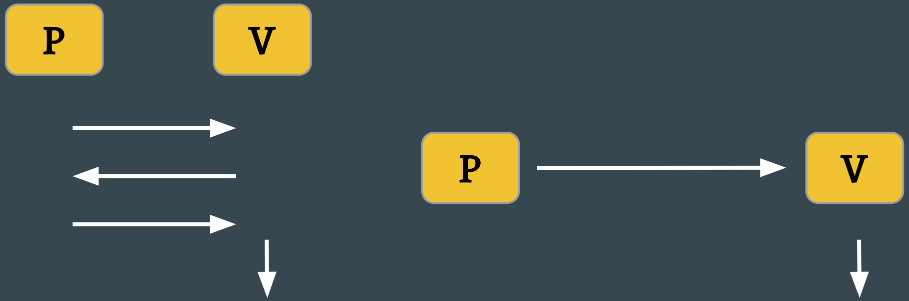
RO Model – Fiat-Shamir transform

Option 2

CRS Model – Distributed Committed-Index OT

multi-receiver **OT Protocol:**

Approach 2: Ephemeral Prover Zero-Knowledge



Option 1

RO Model – Fiat-Shamir transform

Option 2

CRS Model – Distributed Committed-Index OT

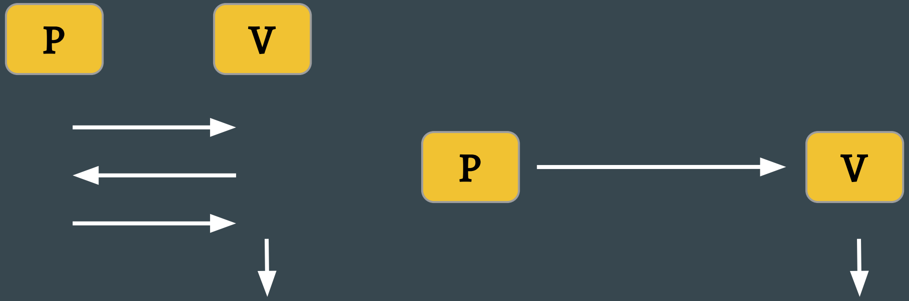
multi-receiver **OT Protocol:**

- choice bit is XOR of all receiver inputs

Approach 2: Ephemeral Prover Zero-Knowledge

Option 1

RO Model – Fiat-Shamir transform



Option 2

CRS Model – Distributed Committed-Index OT

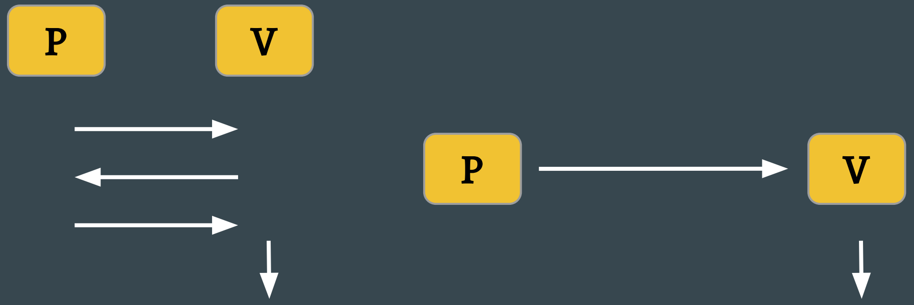
multi-receiver **OT Protocol:**

- choice bit is XOR of all receiver inputs
- choice bit committed before sender message

Approach 2: Ephemeral Prover Zero-Knowledge

Option 1

RO Model – Fiat-Shamir transform

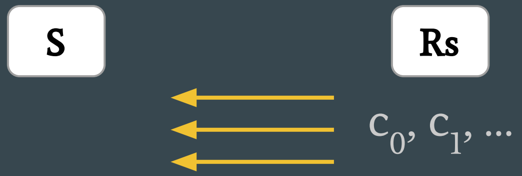


Option 2

CRS Model – Distributed Committed-Index OT

multi-receiver **OT Protocol:**

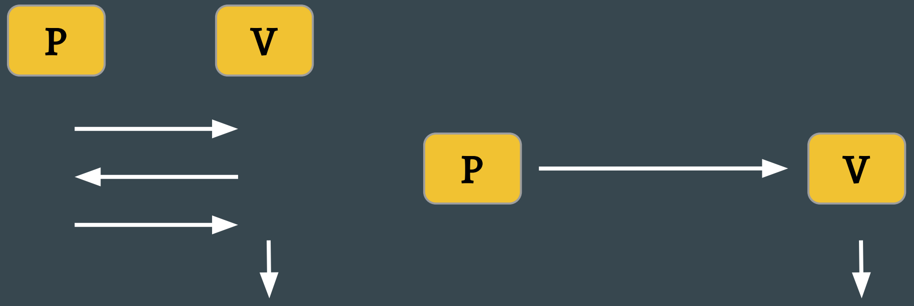
- choice bit is XOR of all receiver inputs
- choice bit committed before sender message



Approach 2: Ephemeral Prover Zero-Knowledge

Option 1

RO Model – Fiat-Shamir transform

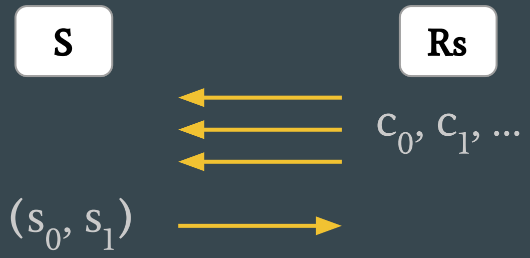


Option 2

CRS Model – Distributed Committed-Index OT

multi-receiver **OT Protocol:**

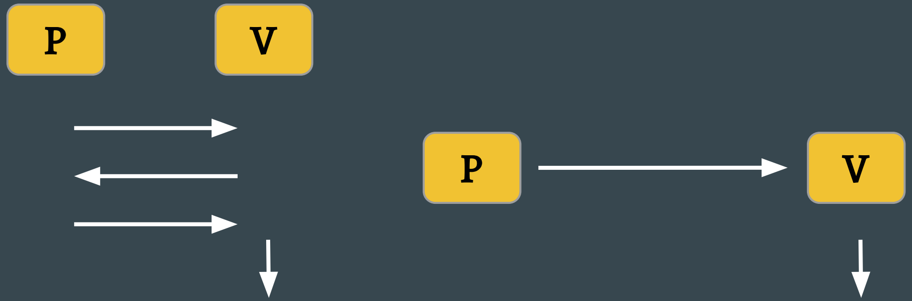
- choice bit is XOR of all receiver inputs
- choice bit committed before sender message
- server sends **first** proof message and OTs the **third** proof message



Approach 2: Ephemeral Prover Zero-Knowledge

Option 1

RO Model – Fiat-Shamir transform

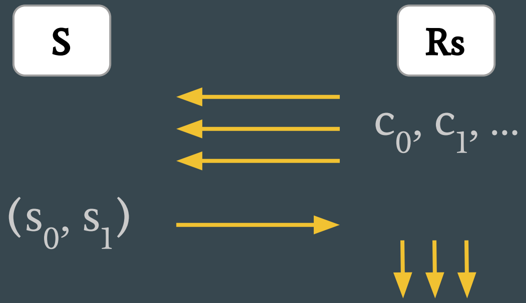


Option 2

CRS Model – Distributed Committed-Index OT

multi-receiver **OT Protocol:**

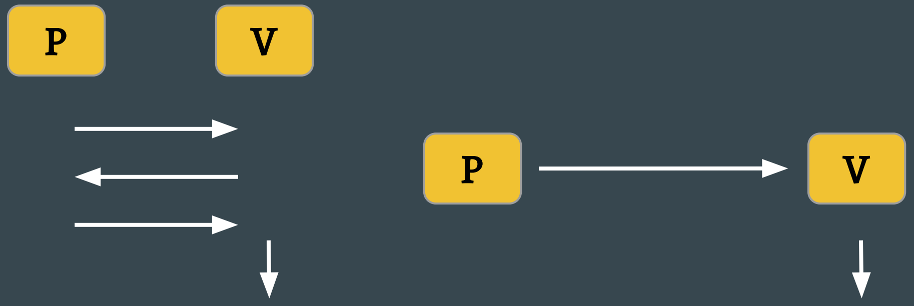
- choice bit is XOR of all receiver inputs
- choice bit committed before sender message
- server sends **first** proof message and OTs the **third** proof message



Approach 2: Ephemeral Prover Zero-Knowledge

Option 1

RO Model – Fiat-Shamir transform

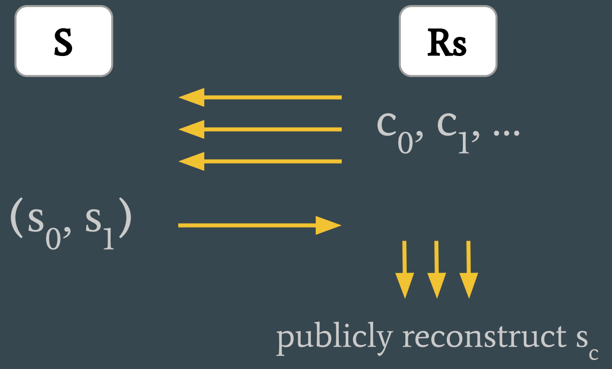


Option 2

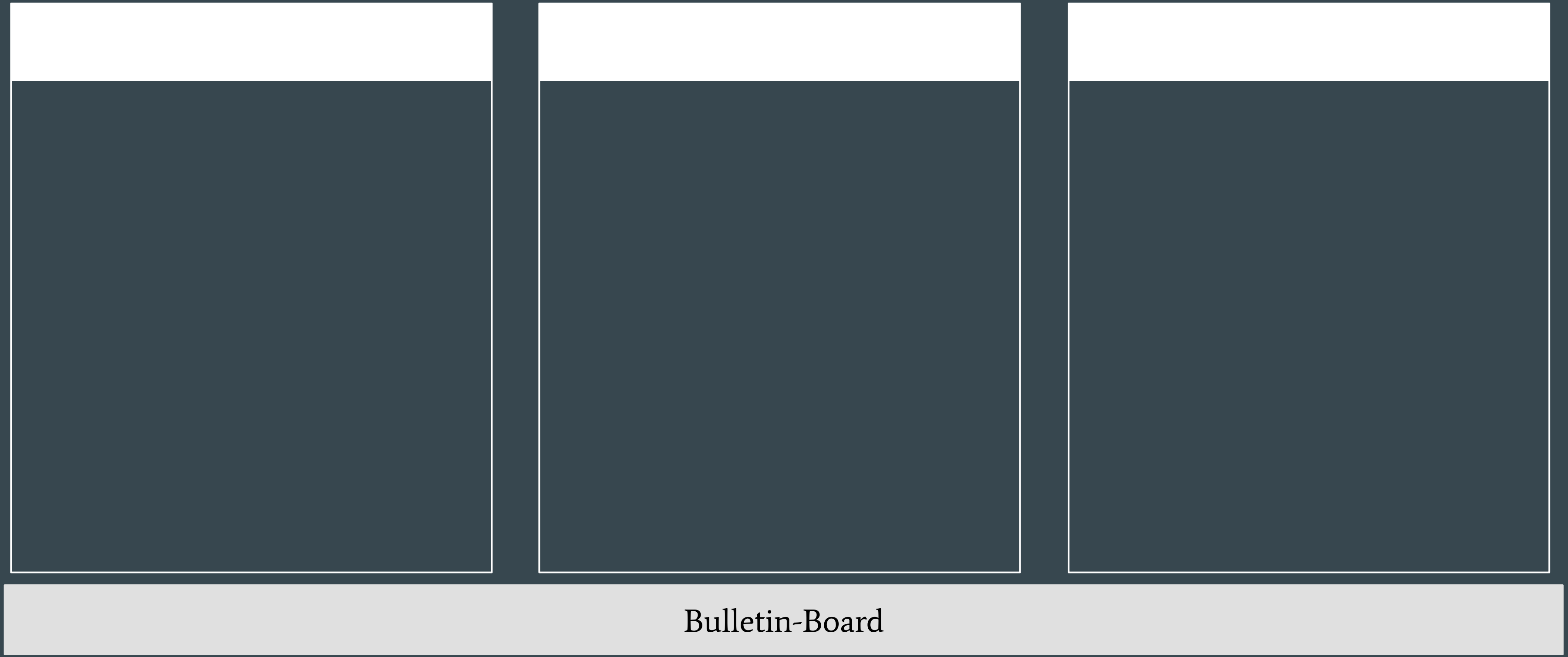
CRS Model – Distributed Committed-Index OT

multi-receiver **OT Protocol:**

- choice bit is XOR of all receiver inputs
- choice bit committed before sender message
- server sends **first** proof message
  and OTs the **third** proof message

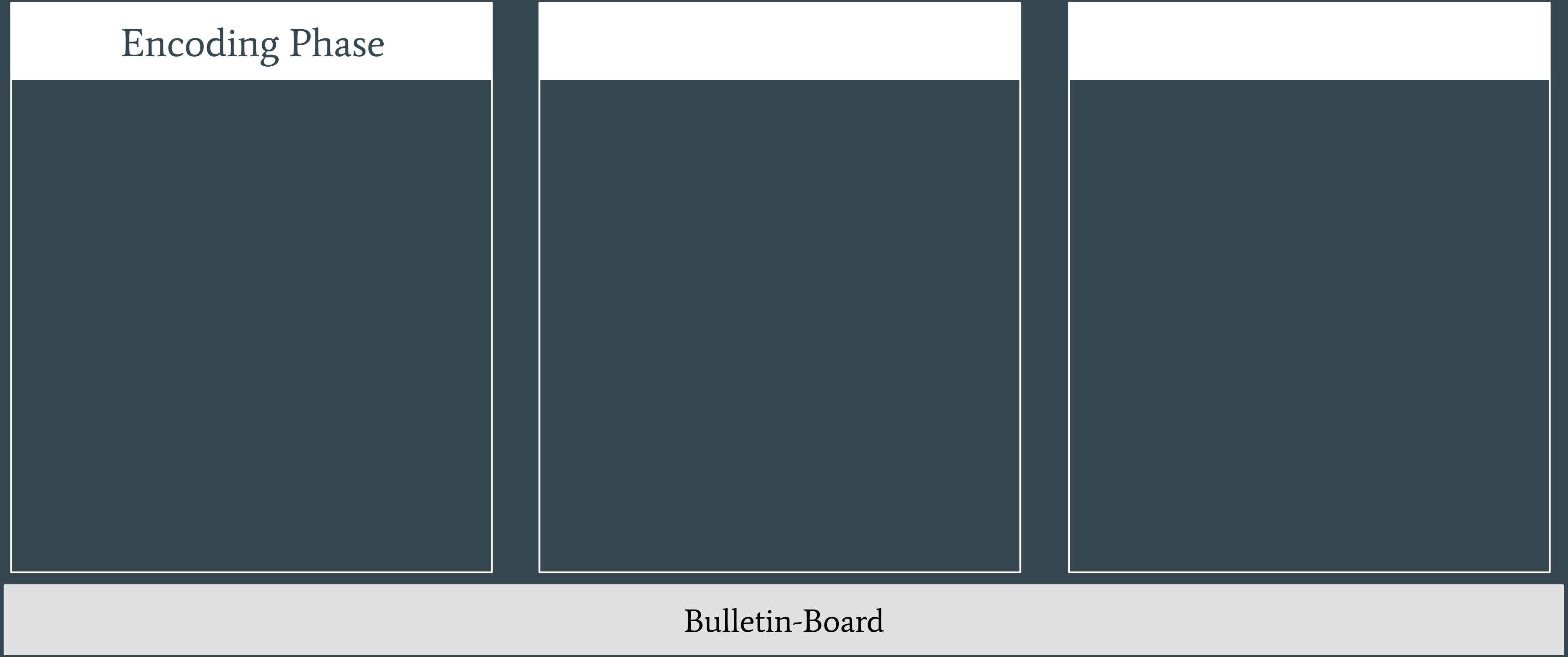


Maliciously Secure SCALES Protocol



Bulletin-Board

Maliciously Secure SCALES Protocol



Encoding Phase

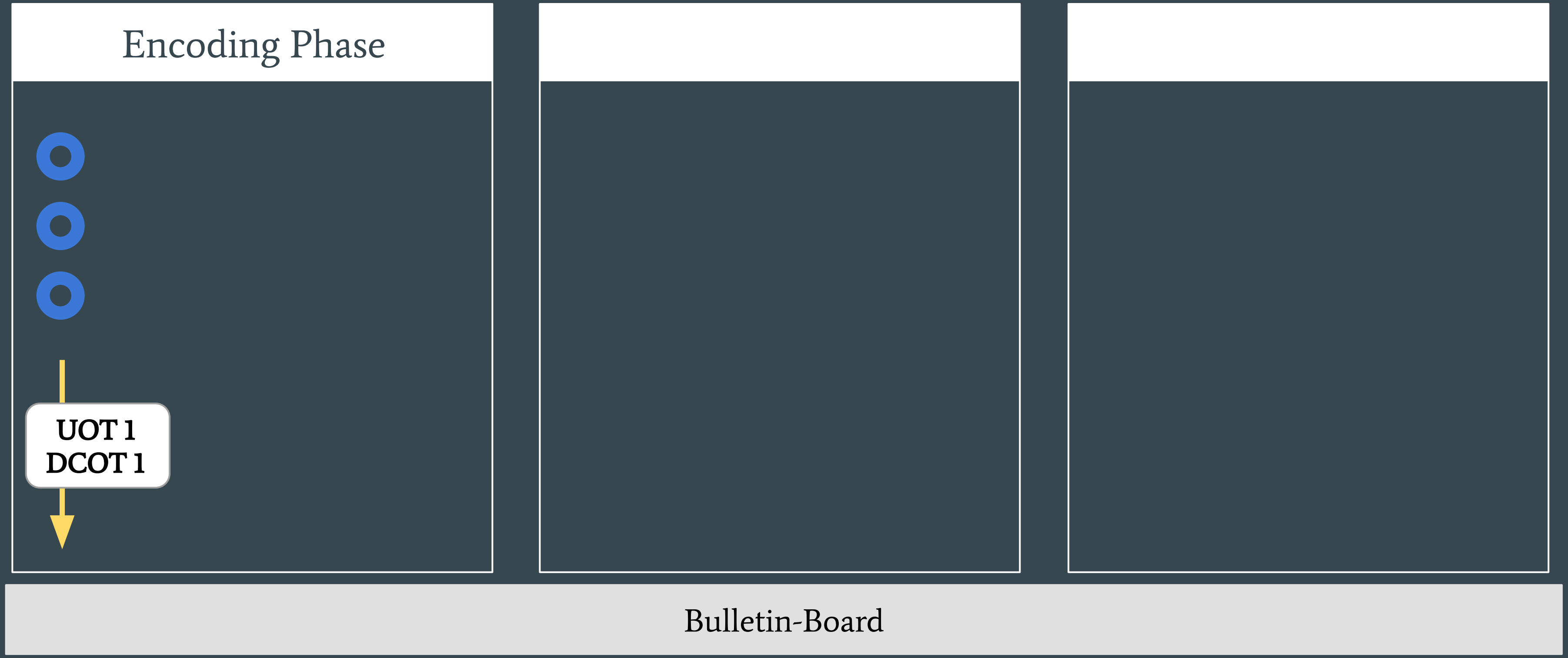
Bulletin-Board

Maliciously Secure SCALES Protocol

Encoding Phase

Bulletin-Board

Maliciously Secure SCALES Protocol

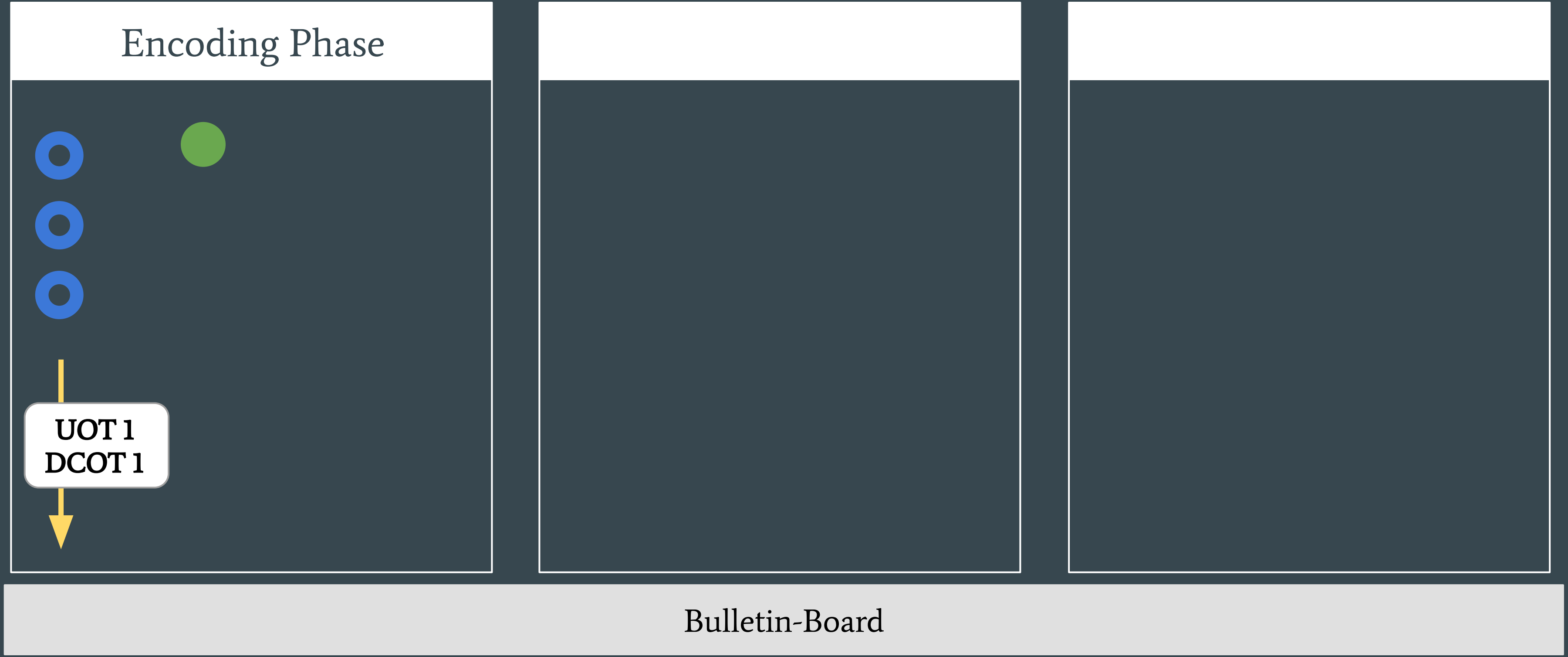


Encoding Phase

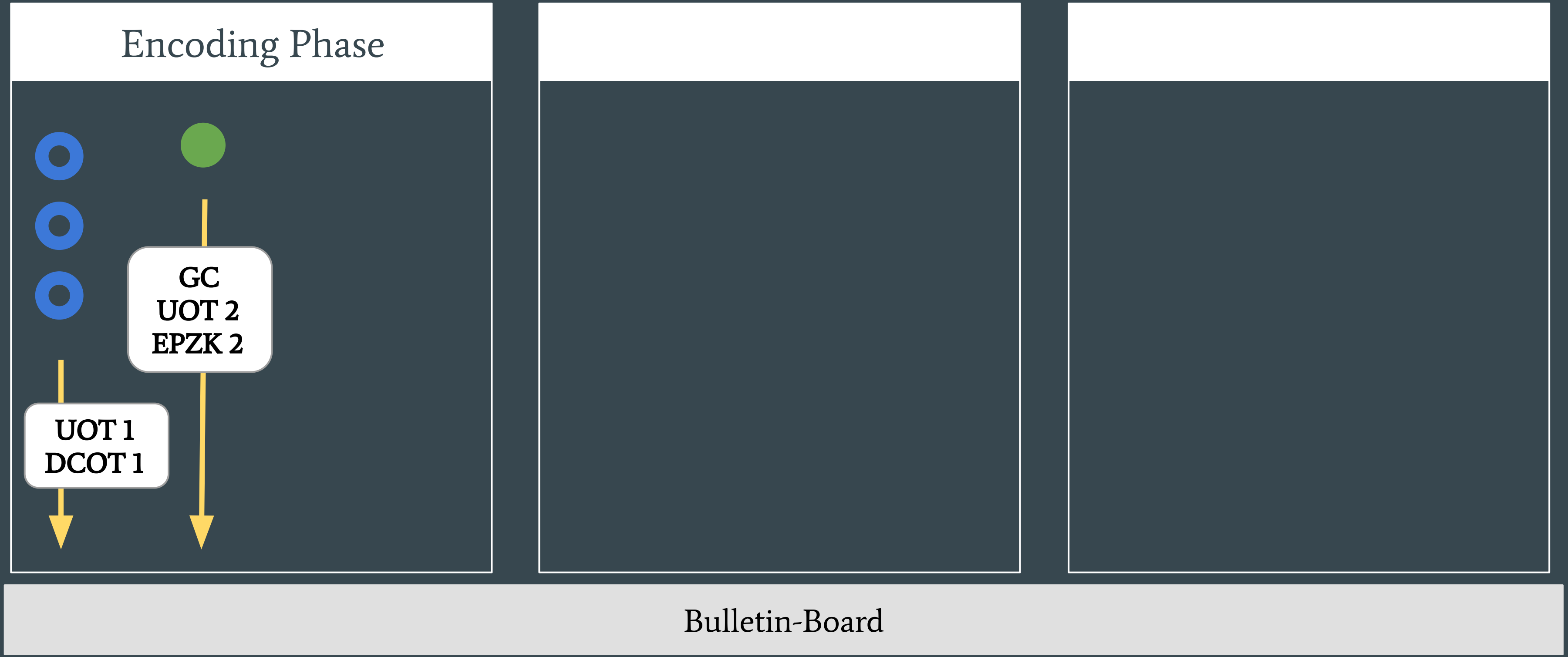
UOT 1
DCOT 1

Bulletin-Board

Maliciously Secure SCALES Protocol



Maliciously Secure SCALES Protocol



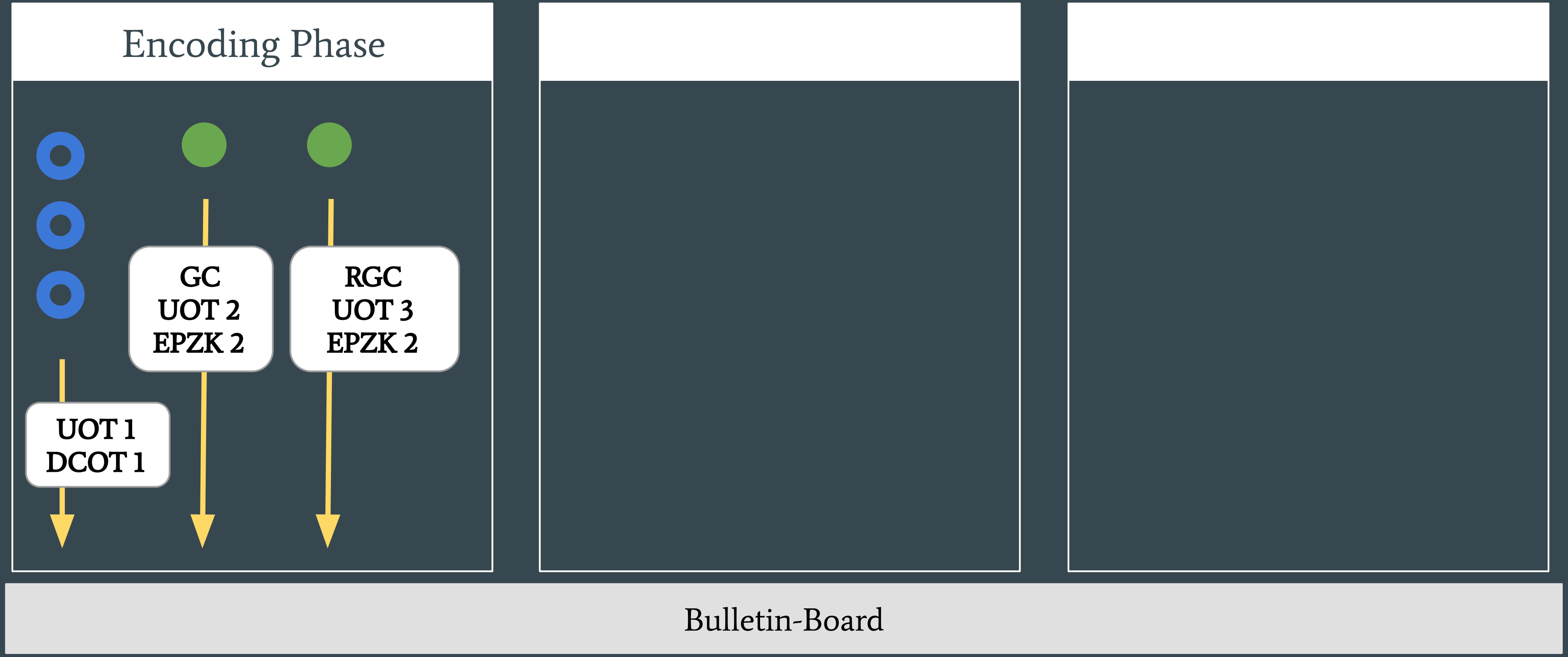
Encoding Phase

GC
UOT 2
EPZK 2

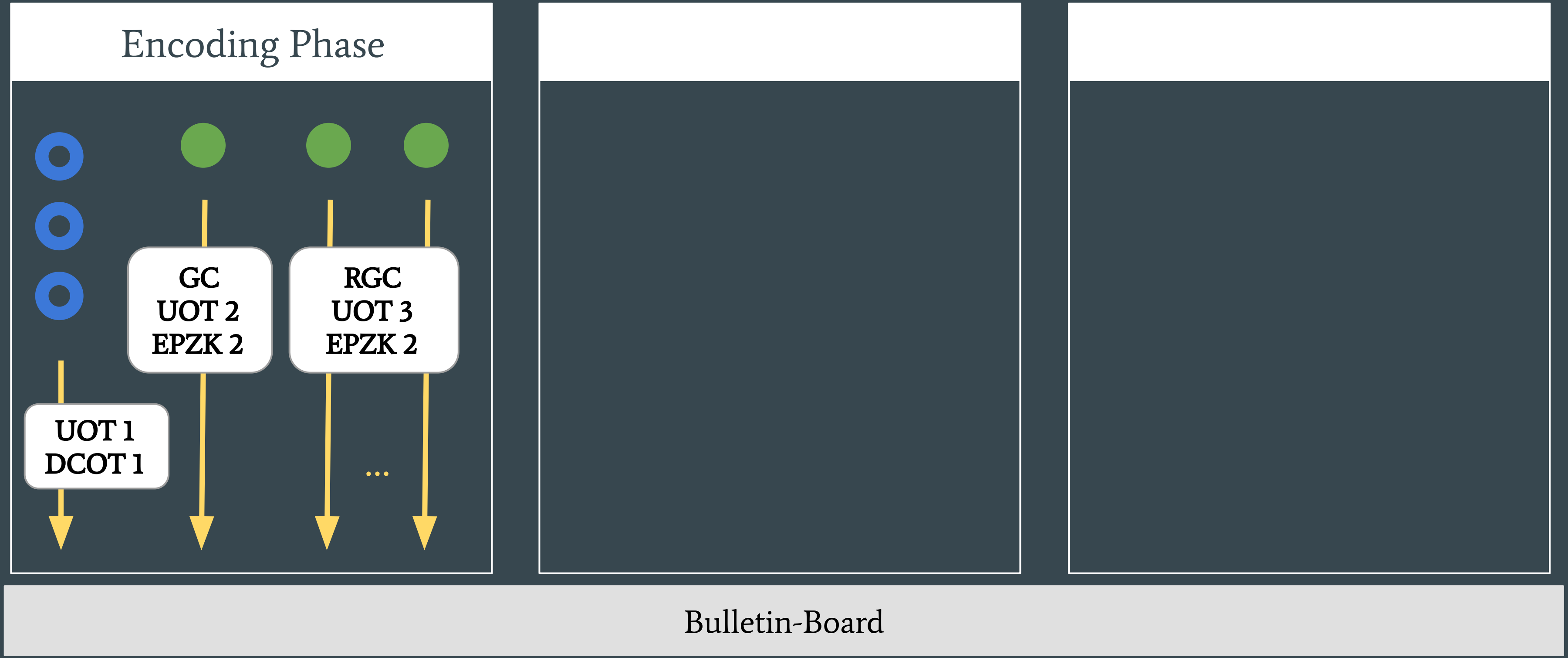
UOT 1
DCOT 1

Bulletin-Board

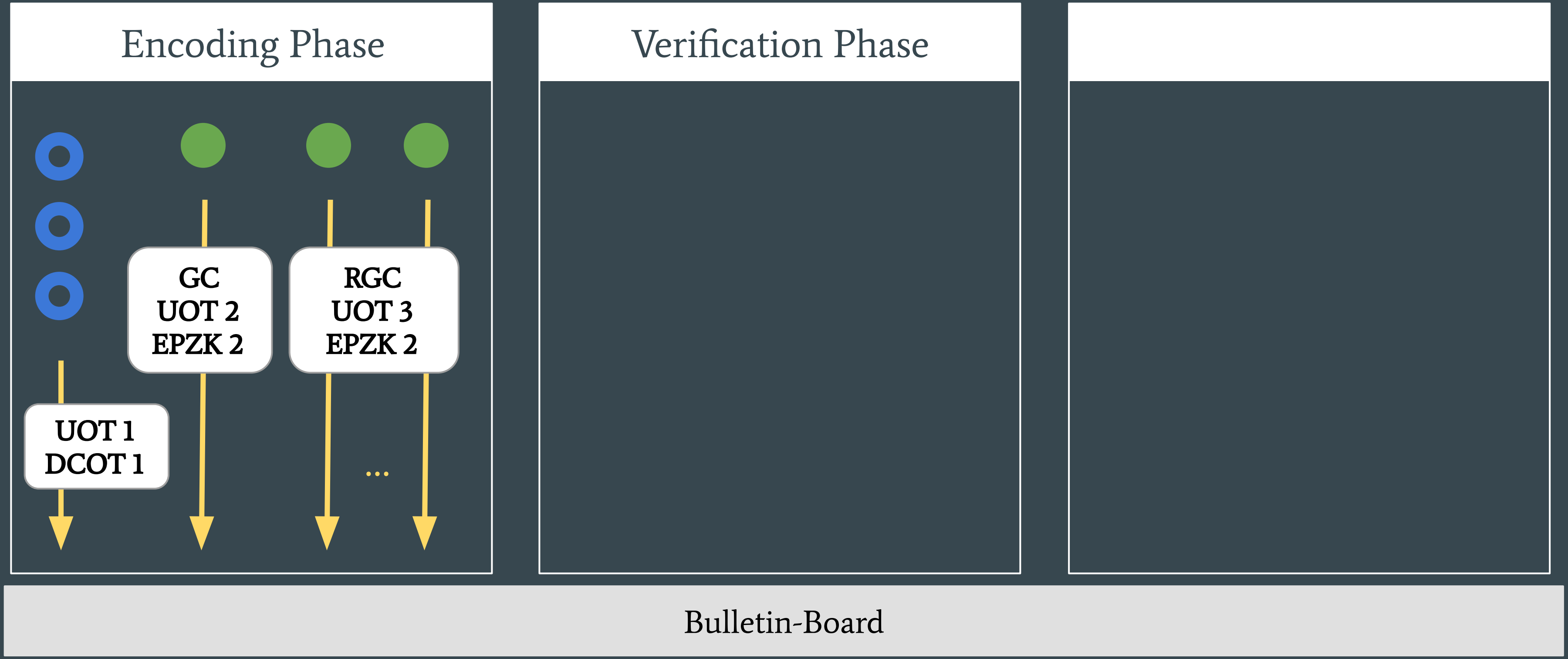
Maliciously Secure SCALES Protocol



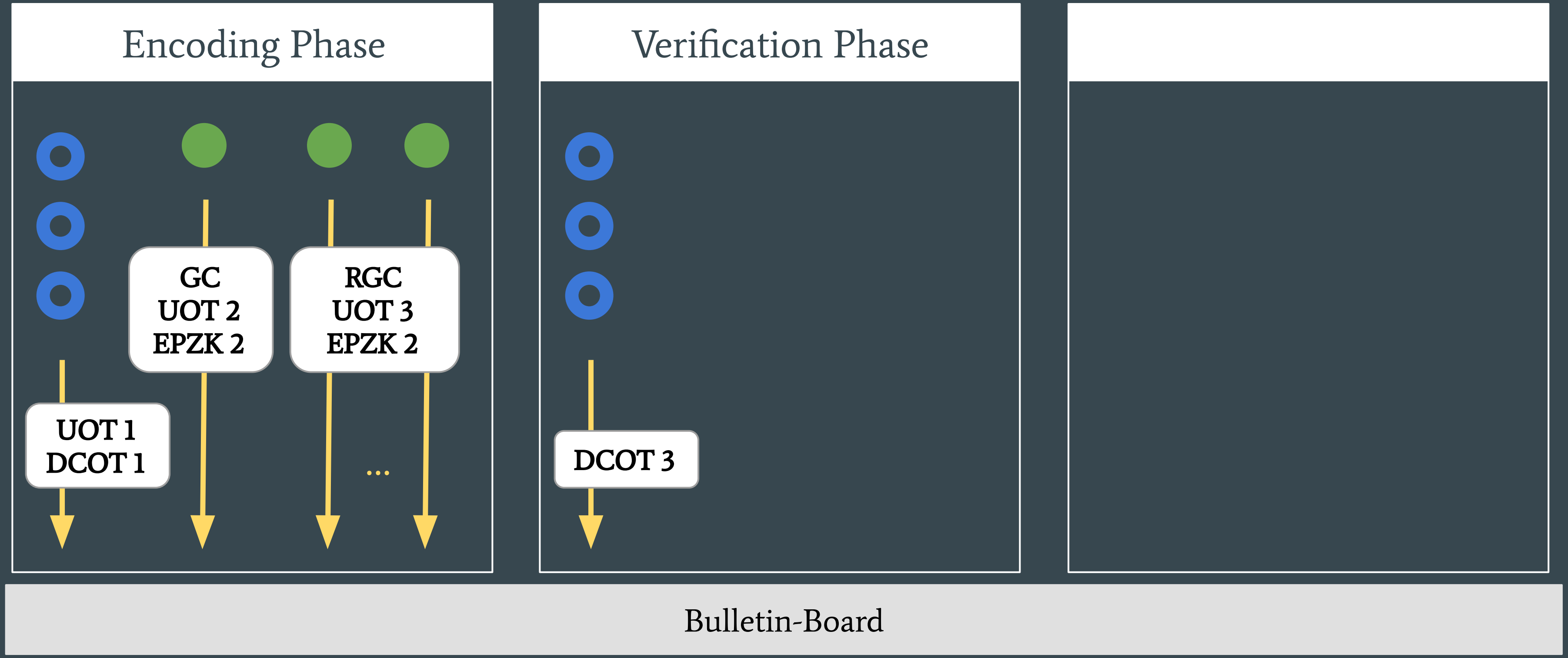
Maliciously Secure SCALES Protocol



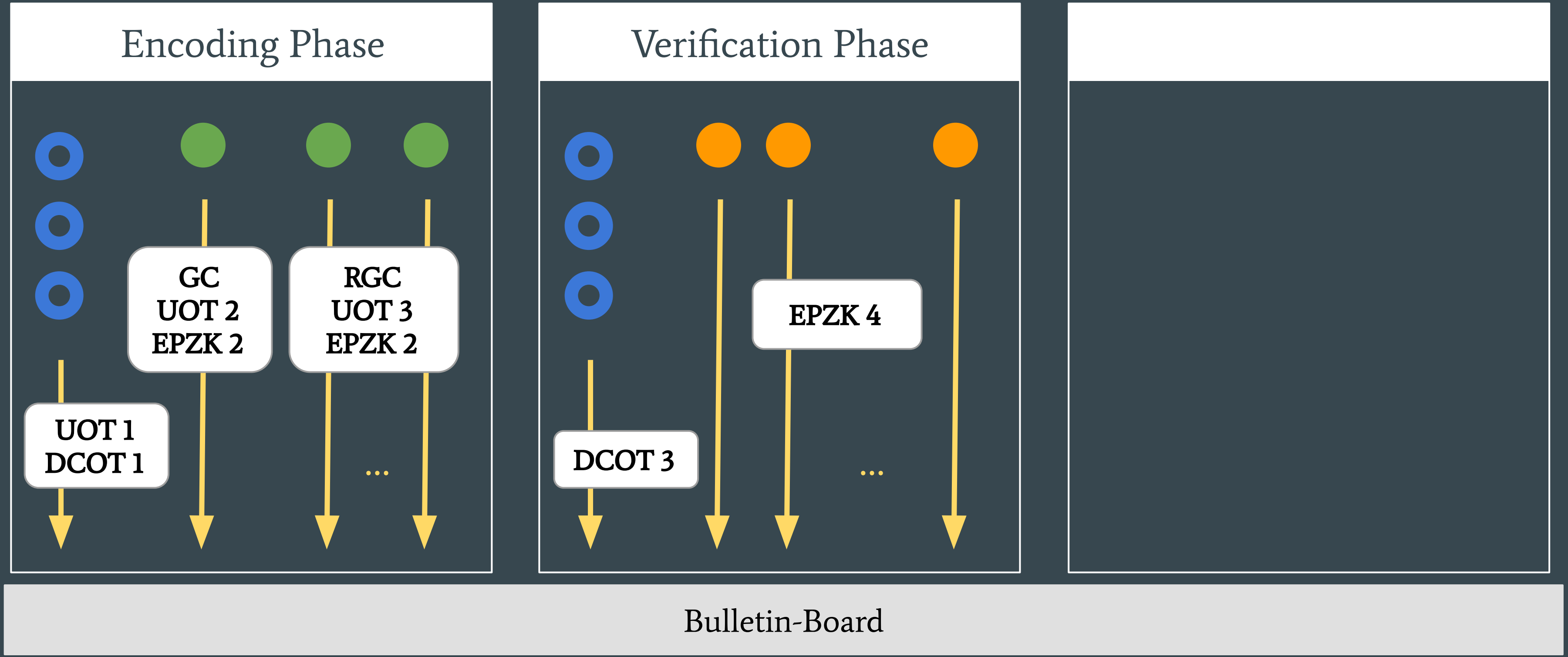
Maliciously Secure SCALES Protocol



Maliciously Secure SCALES Protocol



Maliciously Secure SCALES Protocol



Encoding Phase

GC
UOT 2
EPZK 2

RGC
UOT 3
EPZK 2

UOT 1
DCOT 1

...

Verification Phase

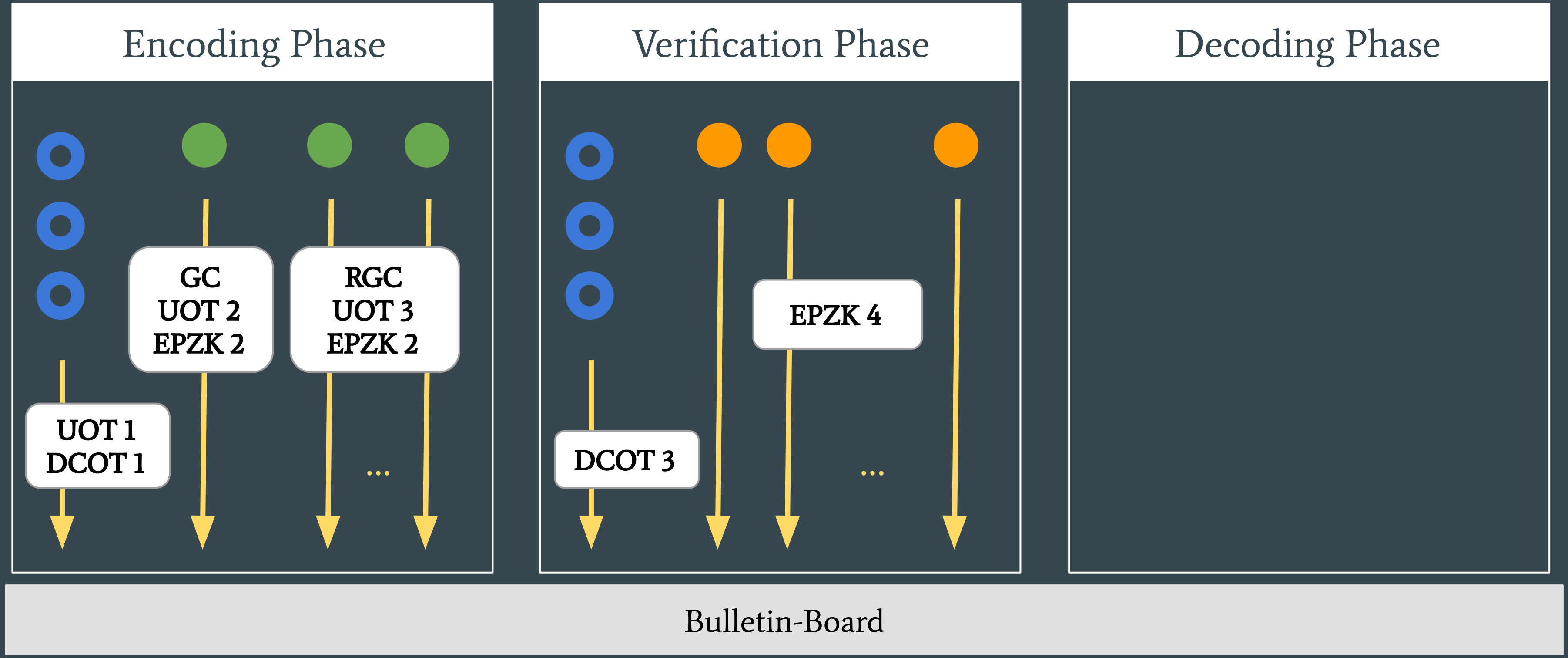
EPZK 4

DCOT 3

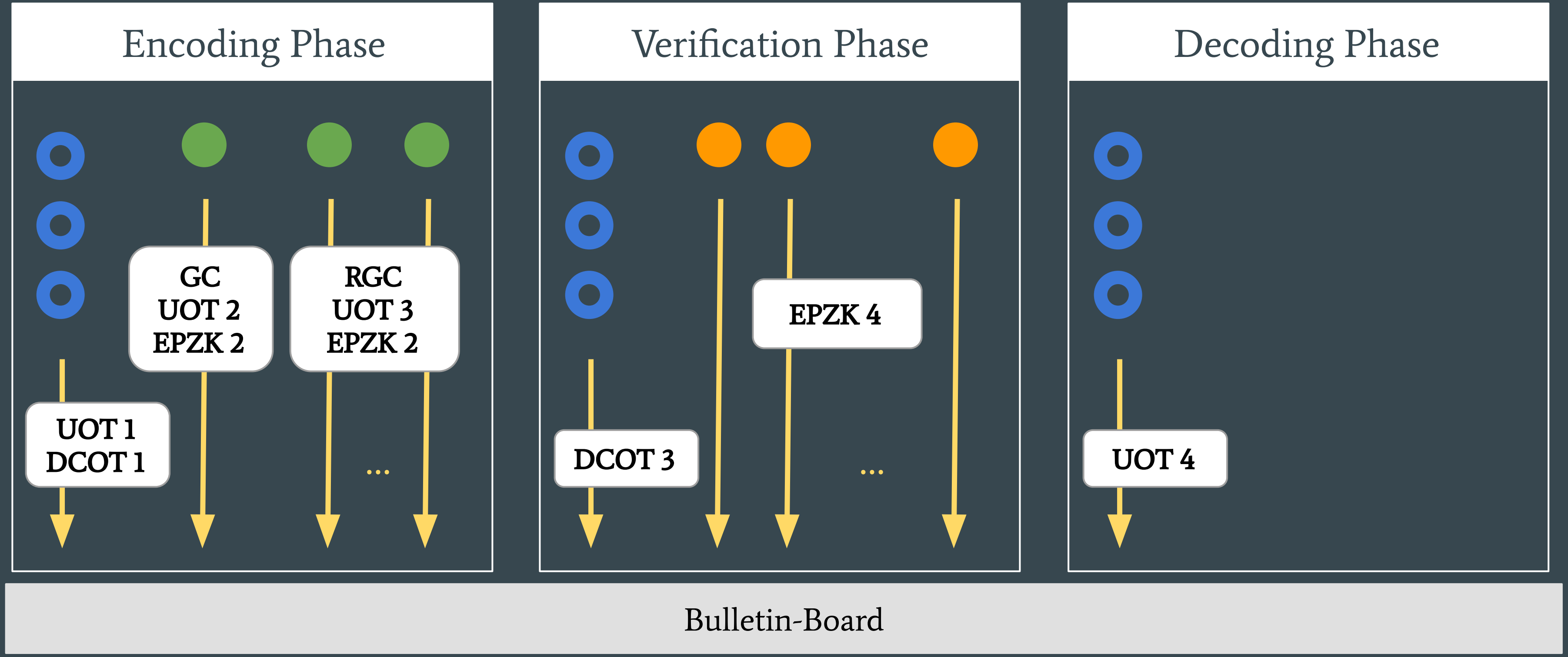
...

Bulletin-Board

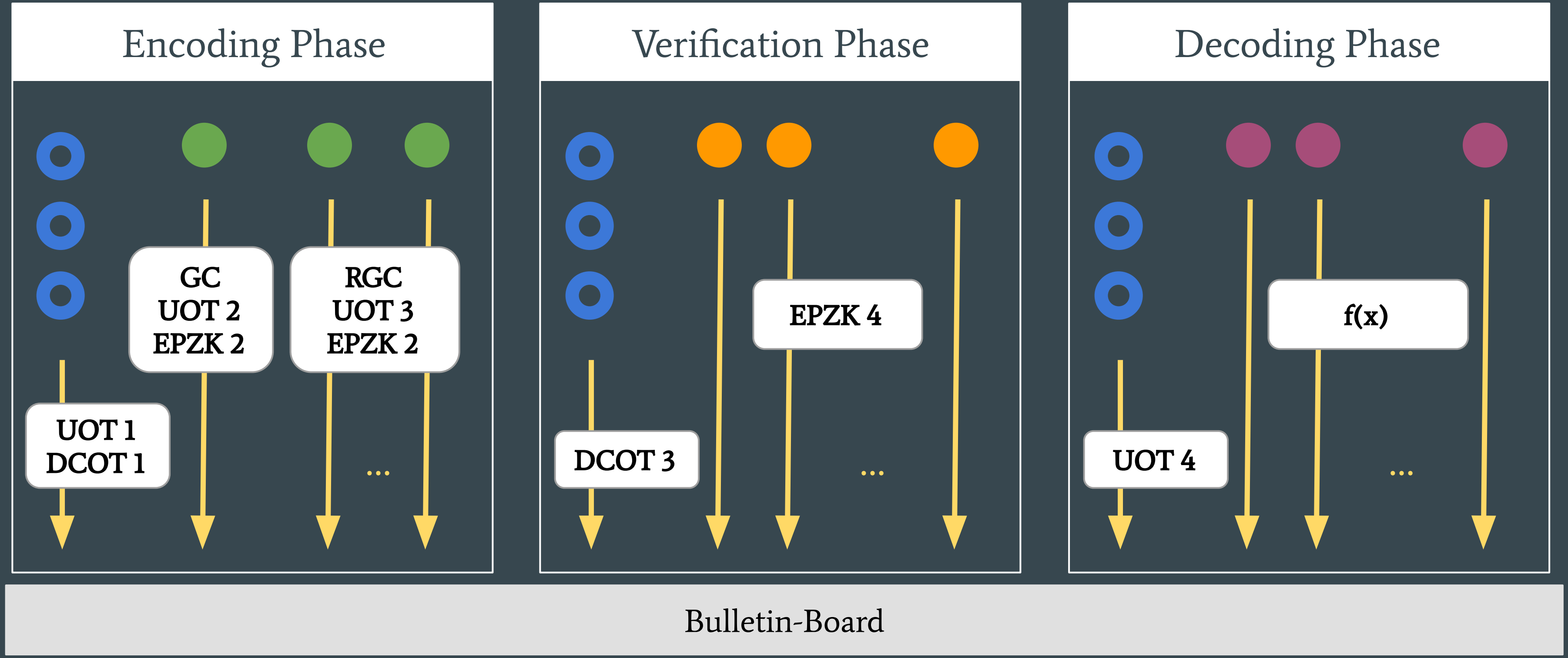
Maliciously Secure SCALES Protocol



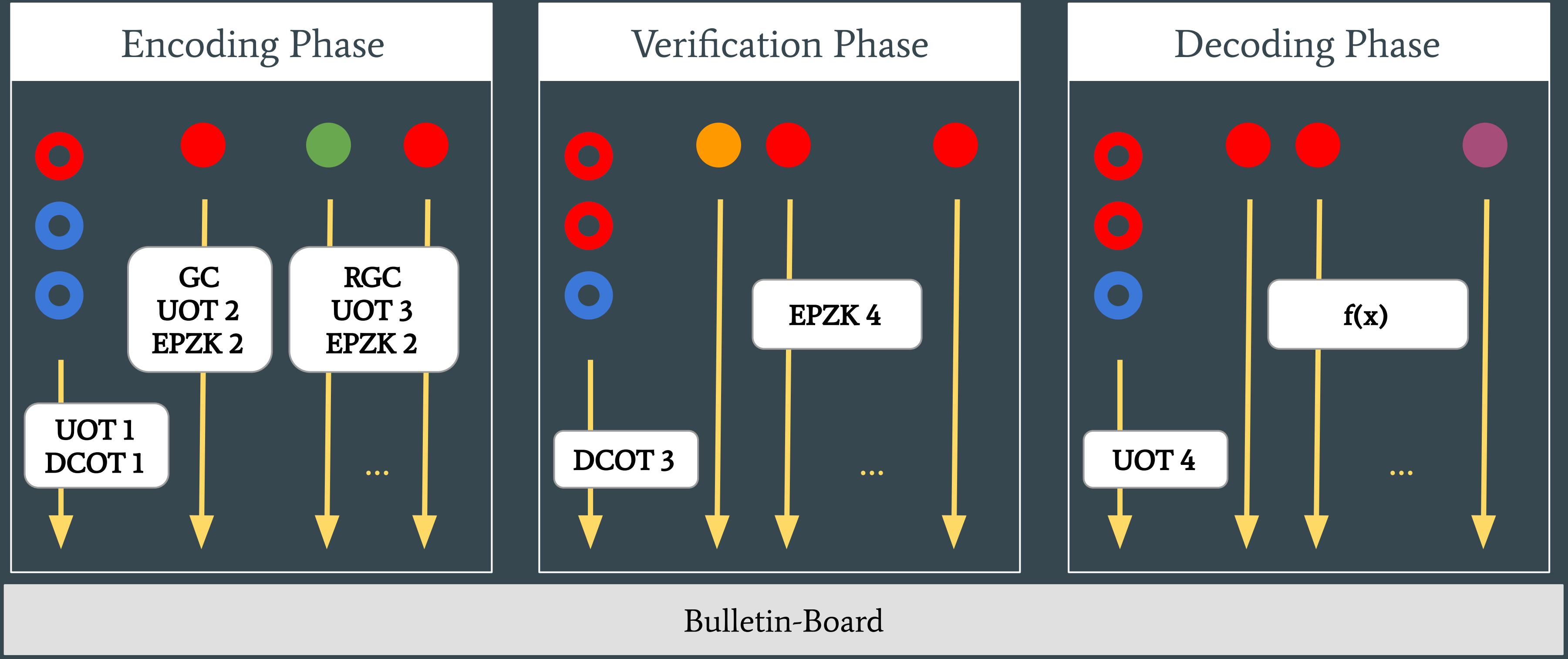
Maliciously Secure SCALES Protocol



Maliciously Secure SCALES Protocol



Maliciously Secure SCALES Protocol



Summary

Summary

Assuming DDH

Maliciously Secure SCALES Protocols –

Summary

Maliciously Secure SCALES Protocols –

- using Custom ZK proofs   2 phases – CRS + RO model

Assuming DDH

Summary

Assuming DDH

Maliciously Secure SCALES Protocols –

- using Custom ZK proofs
  2 phases – CRS + RO model
  3 phases – CRS model

Summary

Maliciously Secure SCALES Protocols –

- using Custom ZK proofs   2 phases – CRS + RO model
  3 phases – CRS model

Assuming DDH

Universally Composable

Summary

Maliciously Secure SCALES Protocols –

- using Custom ZK proofs   2 phases – CRS + RO model
  3 phases – CRS model

Assuming DDH

Universally Composable

Open Problems –

Summary

Assuming DDH

Maliciously Secure SCALES Protocols –

- using Custom ZK proofs   2 phases – CRS + RO model
  3 phases – CRS model

Universally Composable

Open Problems –

- SCALES protocols with Guaranteed Output Delivery

Summary

Assuming DDH

Maliciously Secure SCALES Protocols –

- using Custom ZK proofs   2 phases – CRS + RO model
  3 phases – CRS model

Universally Composable

Open Problems –

- SCALES protocols with Guaranteed Output Delivery
- SCALES protocols in the RO model only (or in the plain model)

Summary

Assuming DDH

Maliciously Secure SCALES Protocols –

- using Custom ZK proofs 2 phases – CRS + RO model
  3 phases – CRS model

Universally Composable

Open Problems –

- SCALES protocols with Guaranteed Output Delivery
- SCALES protocols in the RO model only (or in the plain model)
- SCALES from other hardness assumptions

Thank You!

https://eprint.iacr.org/2024/383