Global-Scale Threshold AES (and SHA256)

Xiao Wang

Based on joint works with Jonathan Katz, Xiao Lan, Samuel Ranellucci, Mike Rosulek, Chenkai Weng, Kang Yang, Jiang Zhang

Authenticated Garbling Blueprint

Each layer based on a lot of prior effort in the community

Authenticated Bits

Authenticated Shares

Authenticated ANDs

Authenticated Garbled Circuits

Authenticated Bits

[BDOZ11 , NNOB12]

A bit : x

MAC

x

$\oplus$

y

=

$x \oplus y$

$x \oplus x = x\Delta_B$

$y \oplus y = y\Delta_B$

$x \oplus y \oplus x \oplus y = (x \oplus y)\Delta_B$

x

Key

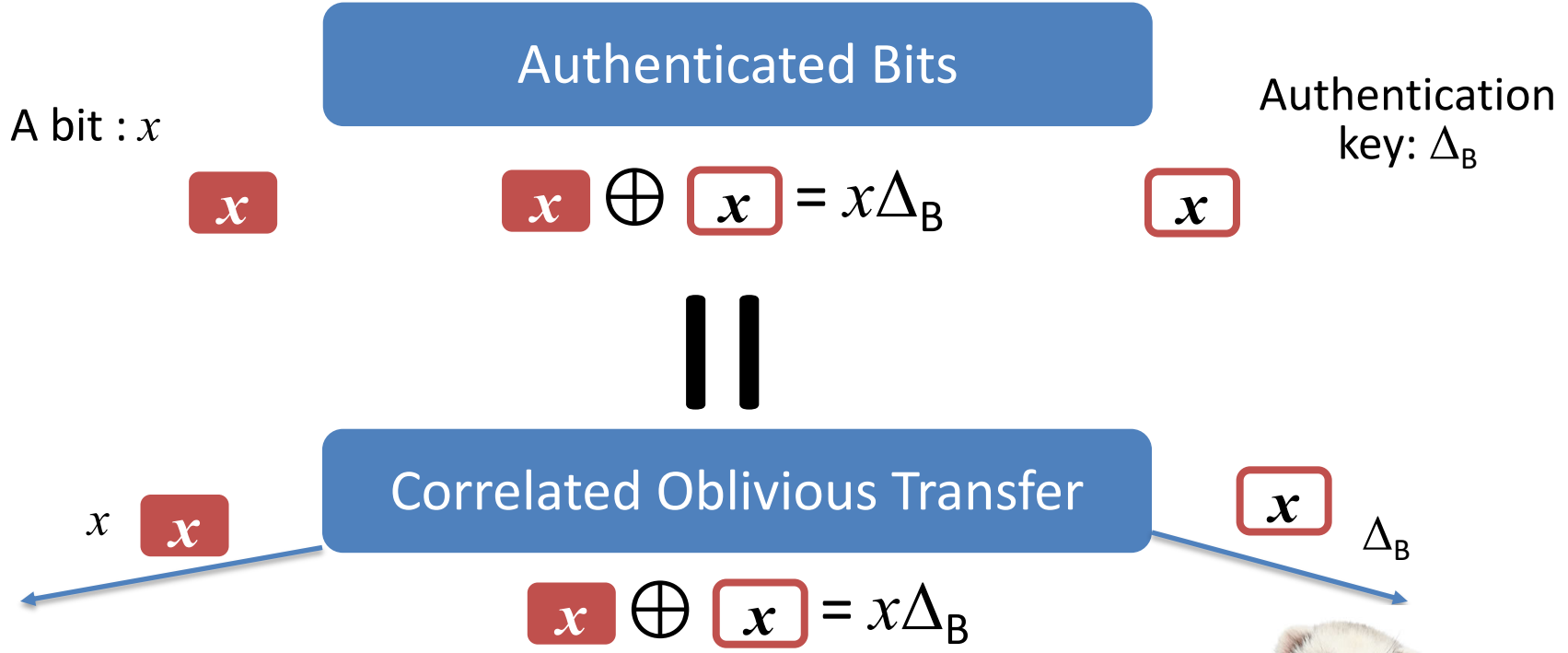
$\oplus$

y

=

$x \oplus y$

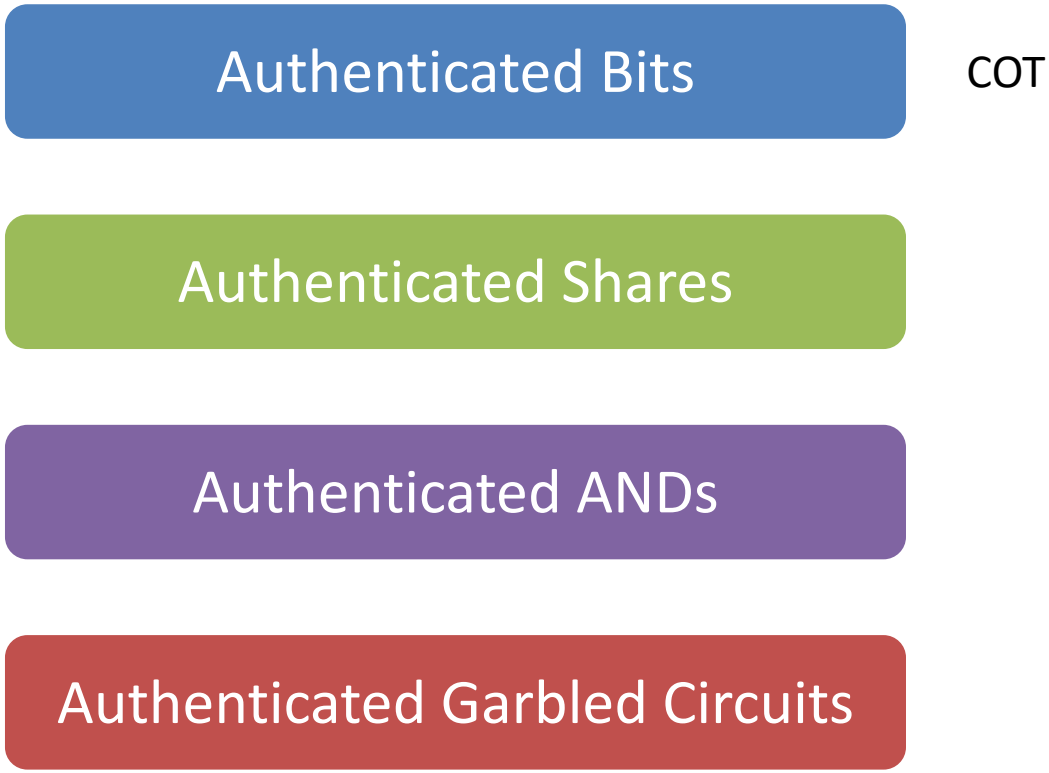
Authentication key: Δ_B



1. IKNP without the last hash function call [IKNP03,ALSZ13,KOS15]
2. Pseudorandom Correlation Generators [BCGIKS19, BCGIKRS19]

FERRET: ~60 million COT per second under 50Mbps

Authenticated Garbling Blueprint



Authenticated Shares

Δ_A

Δ_B

$x = x_1 \oplus x_2$

MAC x_1

x_1 Key

Key x_2

x_2 MAC

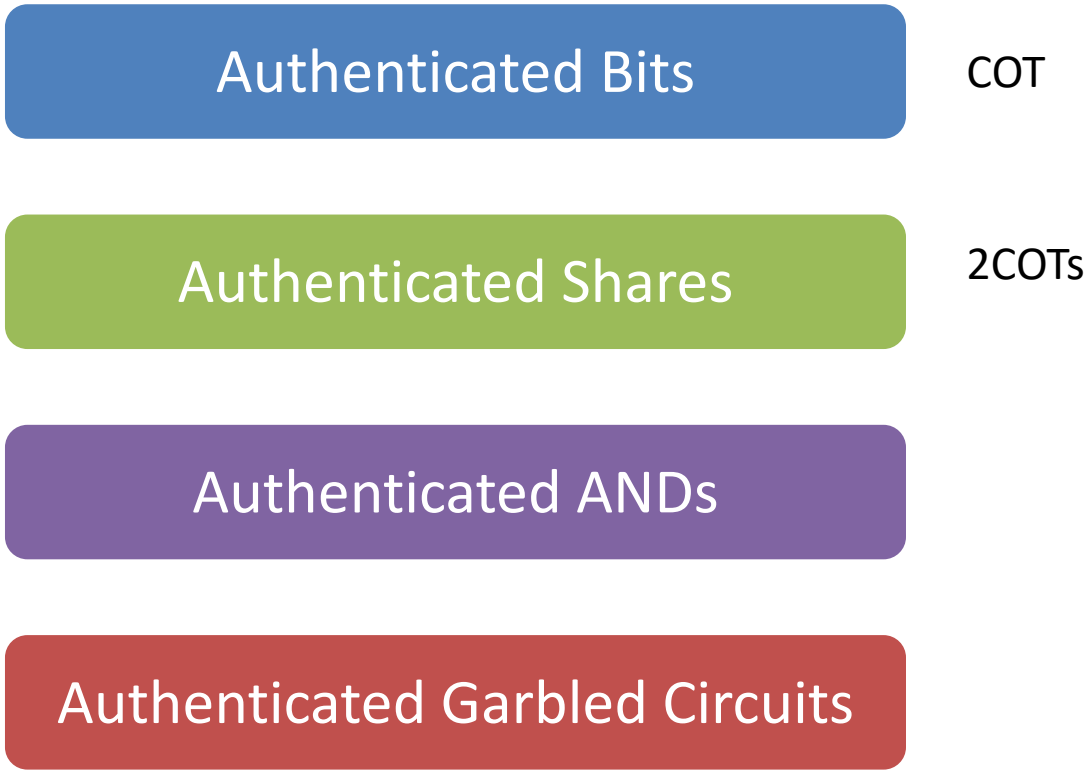
Only knows x_1

Only knows x_2

$$x_1 \oplus x_1 = x_1\Delta_B$$

$$x_2 \oplus x_2 = x_2\Delta_A$$

Authenticated Garbling Blueprint



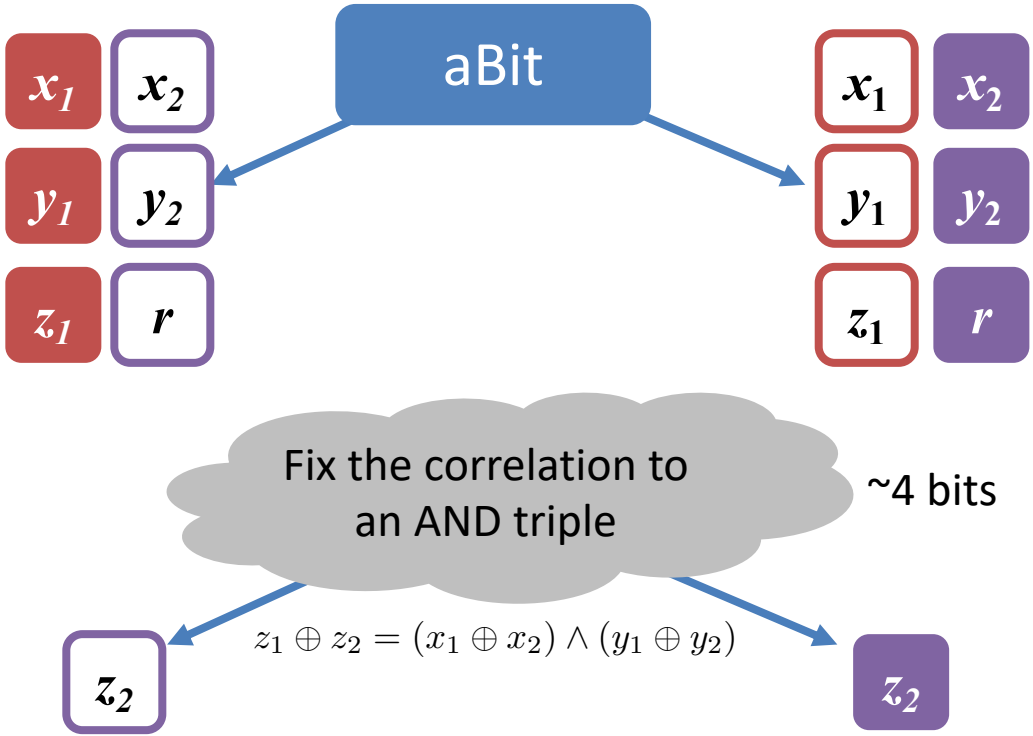
Authenticated ANDs

[NNOB12,FKOS15,WRK17,KRRW18]

- Goal: parties obtain authenticated shares [x], [y], [z] such that

$$x \wedge y = z$$

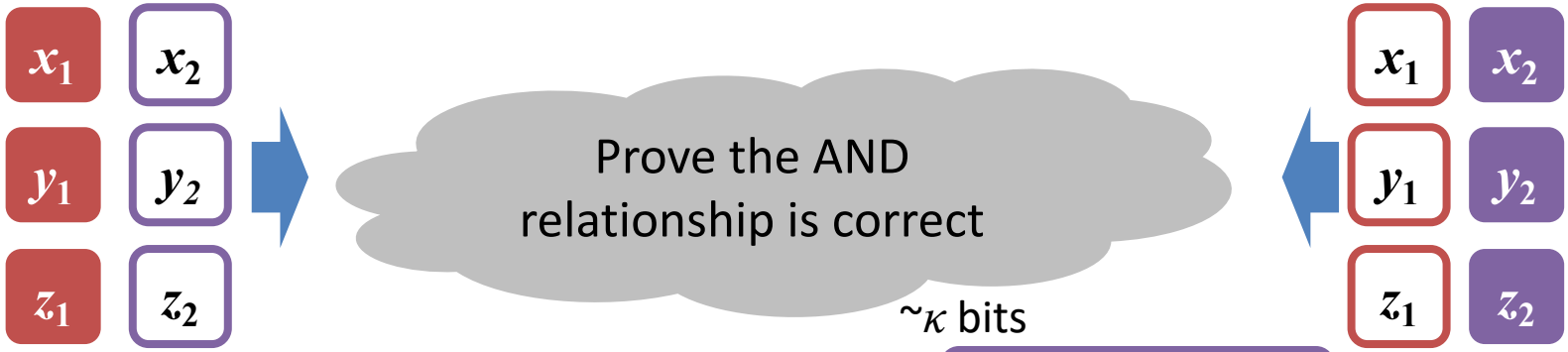
First step: Compute AND triples



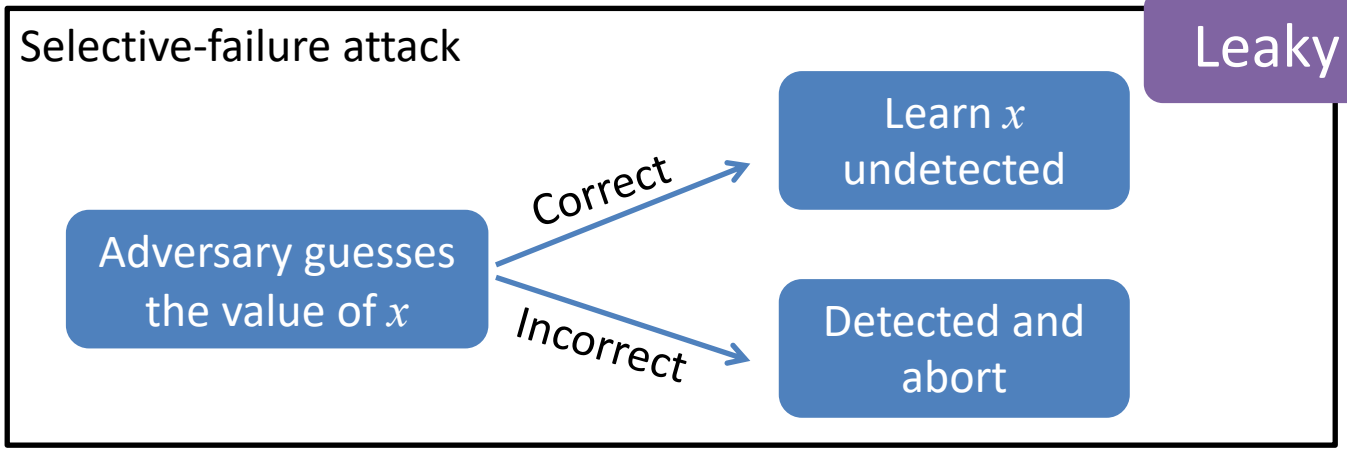
Privacy against malicious adversaries;
Correctness only for semi-honest adversaries

Second step: Check

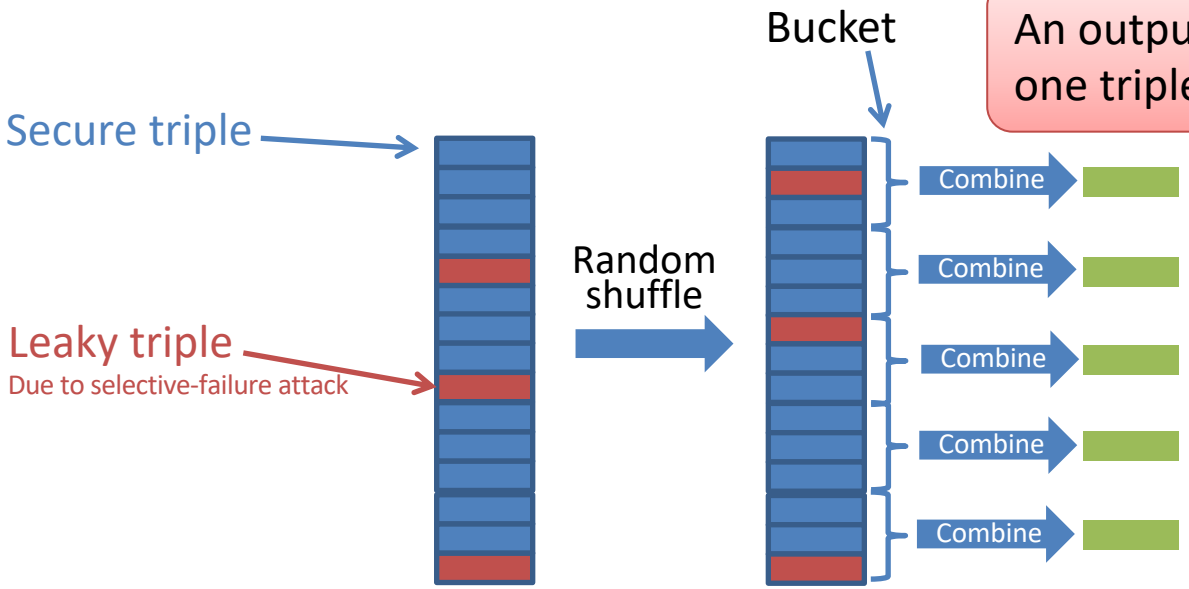
Correct and private against malicious adversaries except vulnerable to a specific selective-failure attack



Leaky triple

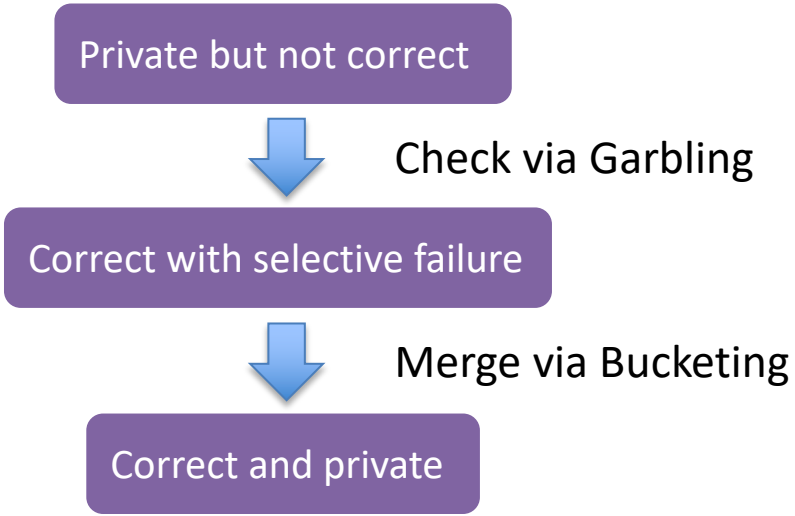


Third step: Bucketing

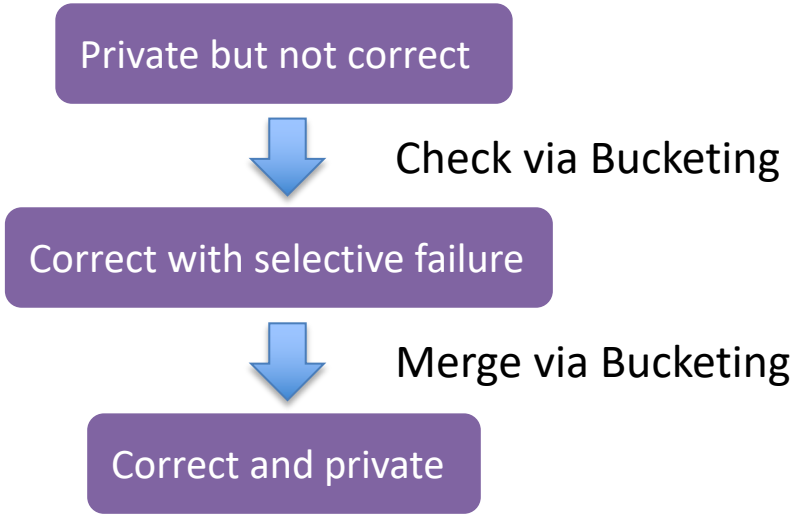


An output triple is secure as long as one triple in the bucket is secure

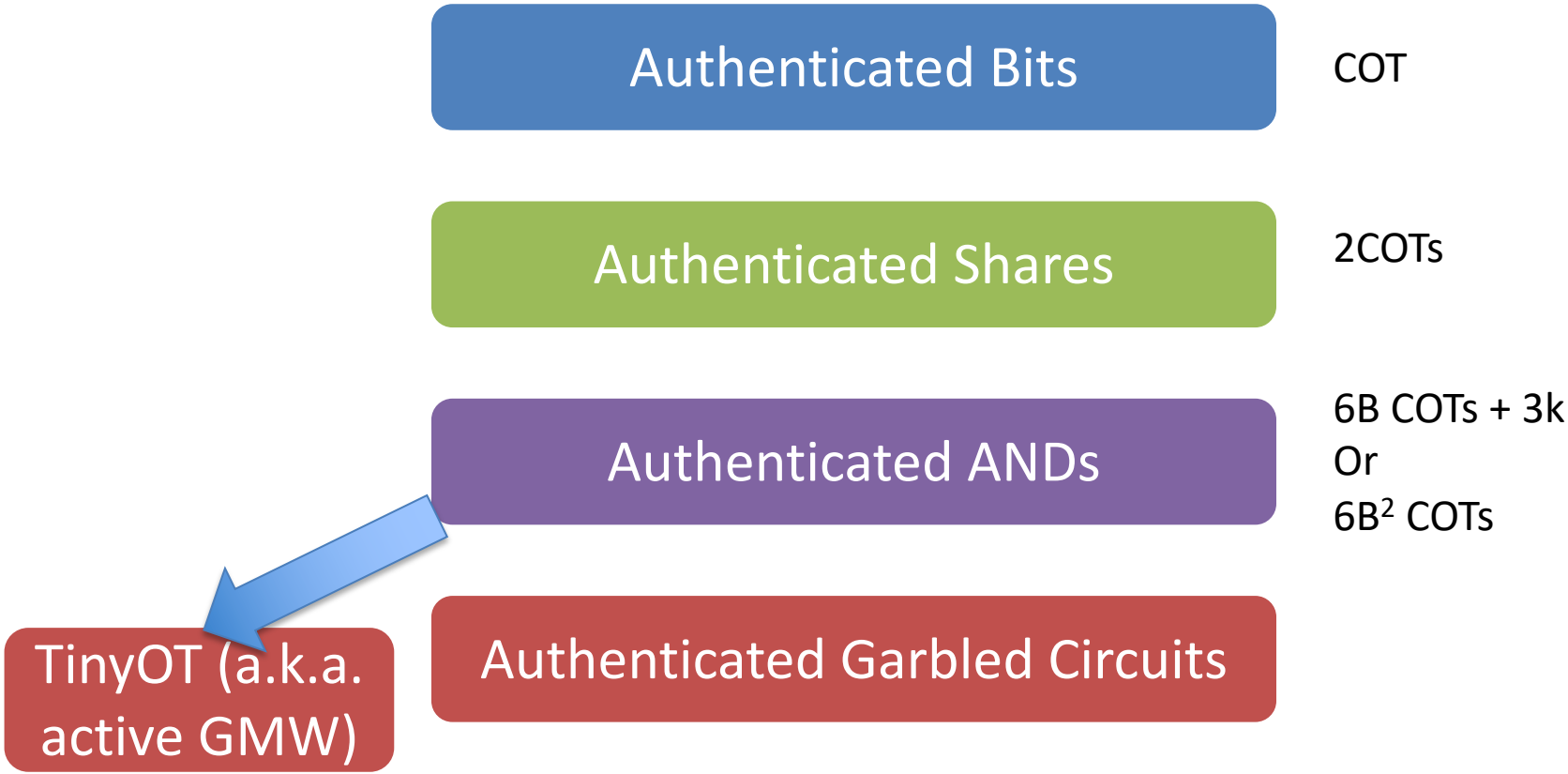
Correct and private against malicious adversaries



Alternative [FKOS15]



Authenticated Garbling Blueprint



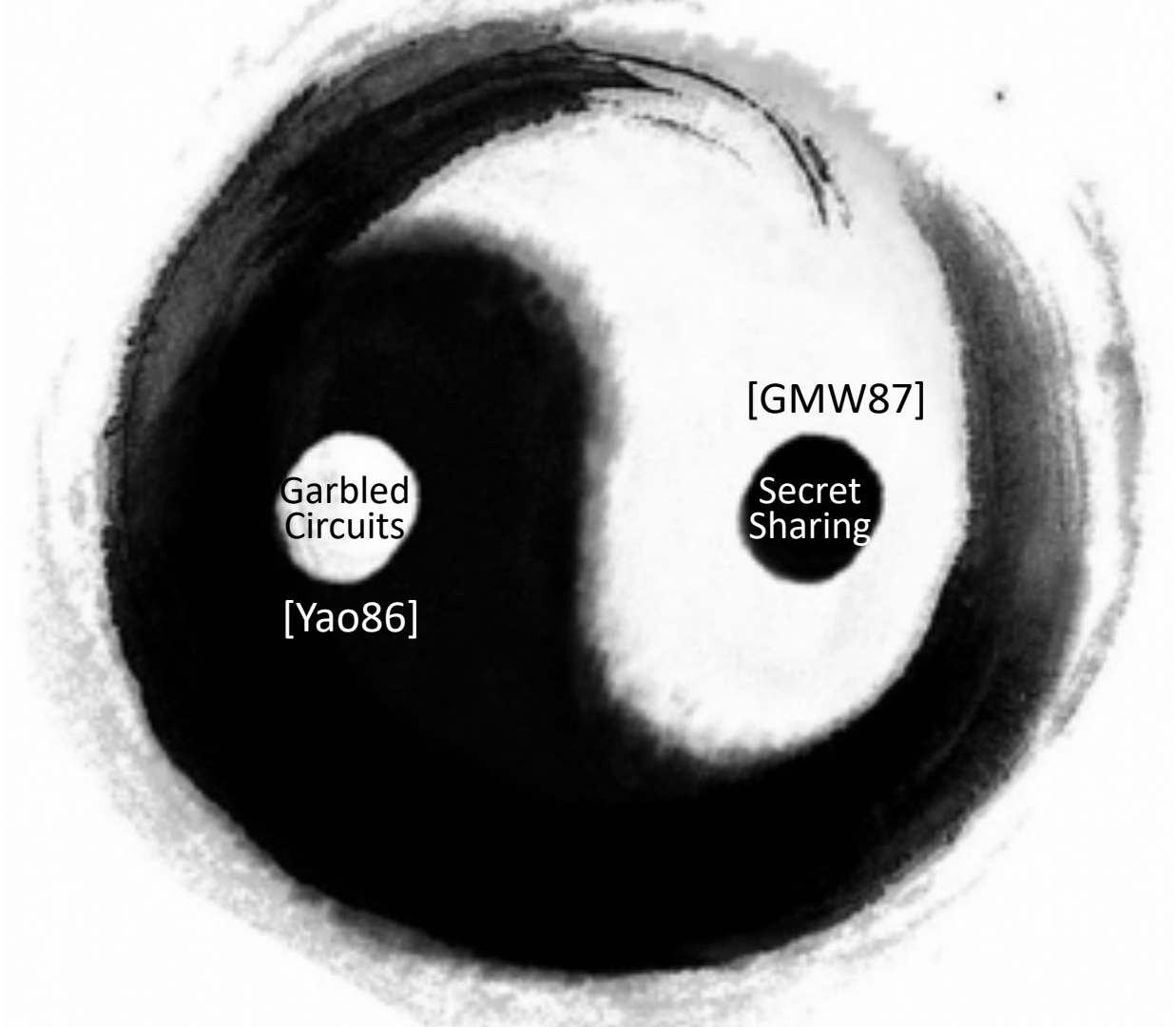
Authenticated Garbling Blueprint

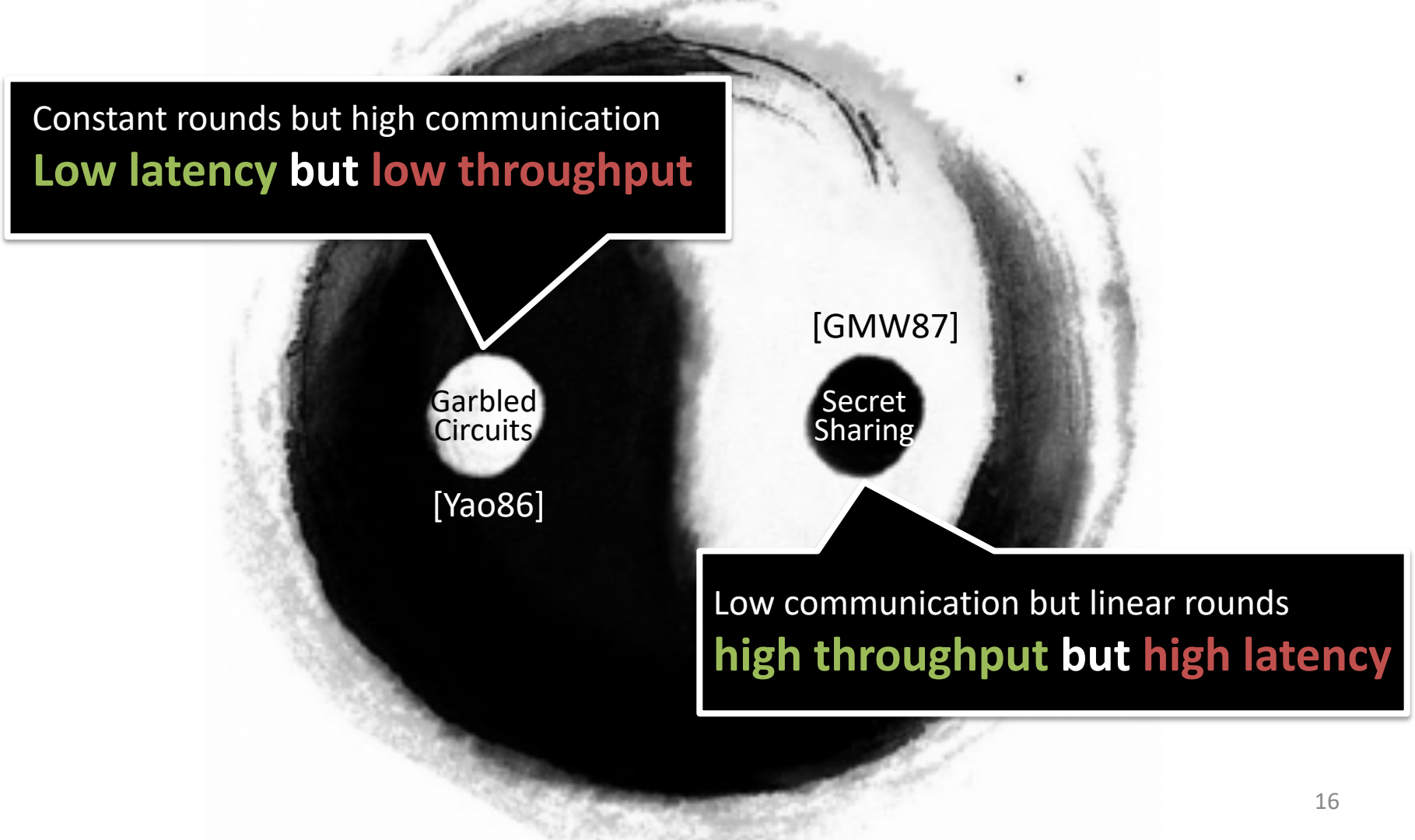
Authenticated Bits

Wolverine
Designated Verifier ZK
- 200 ns per AND
- 1 µs per 61-bit multiplication

Authenticated ANDs

TinyOT (a.k.a. active GMW)


[GMW87]
Secret Sharing
Garbled Circuits
[Yao86]


Constant rounds but high communication
Low latency but low throughput
Garbled Circuits
[Yao86]
[GMW87]
Secret Sharing
Low communication but linear rounds
high throughput but high latency

Constant rounds and low communication

Low latency and high throughput

Malicious

Our protocol

=

Semi-honest

Garbled Circuits

+

Malicious

Secret Sharing

Selective-failure Attack

garbled table
$H(\mathsf{L}_{\alpha,0}, \mathsf{L}_{\beta,0}) \oplus \mathsf{L}_{\gamma,0}$
$H(\mathsf{L}_{\alpha,0}, \mathsf{L}_{\beta,1}) \oplus \mathsf{L}_{\gamma,0}$
$H(\mathsf{L}_{\alpha,1}, \mathsf{L}_{\beta,0}) \oplus \mathsf{L}_{\gamma,0}$
$H(\mathsf{L}_{\alpha,1}, \mathsf{L}_{\beta,1}) \oplus \mathsf{L}_{\gamma,1}$

Corrupt

Garbler knows how rows are permuted

Selectively corrupt one or more rows

Learn information about which row is evaluated

Learn information about inputs

Preventing Selective-failure Attack [LPSY15,LSS16]

Secret share of GT

garbled

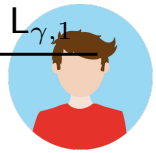
$H(\mathsf{L}_{\alpha,0}, \mathsf{L}_{\beta}$
$H(\mathsf{L}_{\alpha,0}, \mathsf{L}_{\beta}$
$H(\mathsf{L}_{\alpha,1}, \mathsf{L}_{\beta}$
$H(\mathsf{L}_{\alpha,1}, \mathsf{L}_{\beta}$

Secret share of GT

l table

$,0) \oplus \mathsf{L}_{\gamma,0}$
$,0) \oplus \mathsf{L}_{\gamma,0}$
$,1) \oplus \mathsf{L}_{\gamma,1}$

Corrupt



Evaluate as normal

Garbler does not know how garbled table is permuted

Selectively corrupt one or more rows

Learn information about which row is evaluated

Learn information about inputs

Compute shares of masked garbled labels [WRK17]

share of the AND of two secret masks!

Free-XOR with Delta = global key

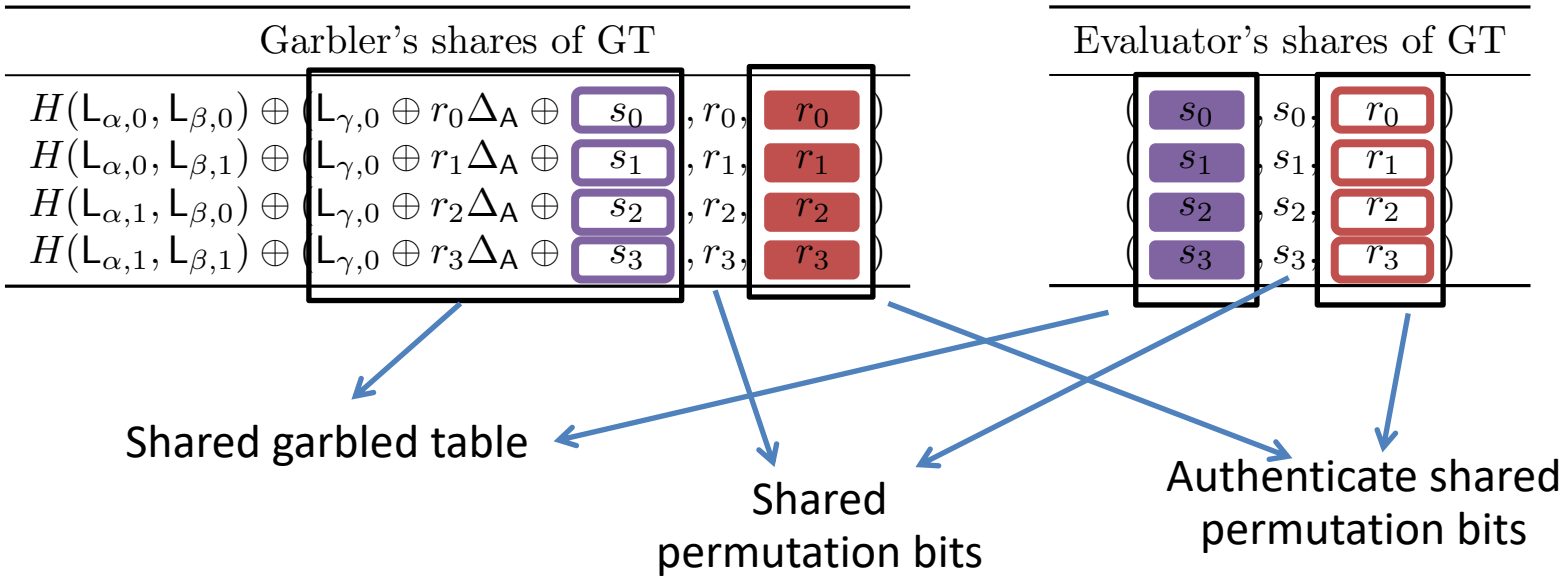
$$\mathsf{L}_{r,z_0 \oplus \lambda_\gamma} = \mathsf{L}_{r,0} \oplus (z_0 \oplus \lambda_\gamma)\Delta_A$$

Share of mask bit

Locally computable by the garbler

Locally computable by the evaluator

Putting Everything Together



All ■□□■ in the table can be computed from **one** TinyOT AND-triple

x	y	truth table
λ_α	λ_β	$z_0 = \lambda_\alpha \wedge \lambda_\beta$
λ_α	$\bar{\lambda}_\beta$	$z_1 = \lambda_\alpha \wedge \bar{\lambda}_\beta$
$\bar{\lambda}_\alpha$	λ_β	$z_2 = \bar{\lambda}_\alpha \wedge \lambda_\beta$
$\bar{\lambda}_\alpha$	$\bar{\lambda}_\beta$	$z_3 = \bar{\lambda}_\alpha \wedge \bar{\lambda}_\beta$

Constant rounds and less communication

Low latency and high throughput

Malicious

Our protocol

=

Semi-honest

Garbled Circuits

+

Malicious

Secret Sharing

Make TinyOT constant-round using cheap garbling techniques


DeMo's
PIZZERIA & DELI

Thanks!

https://github.com/emp-toolkit