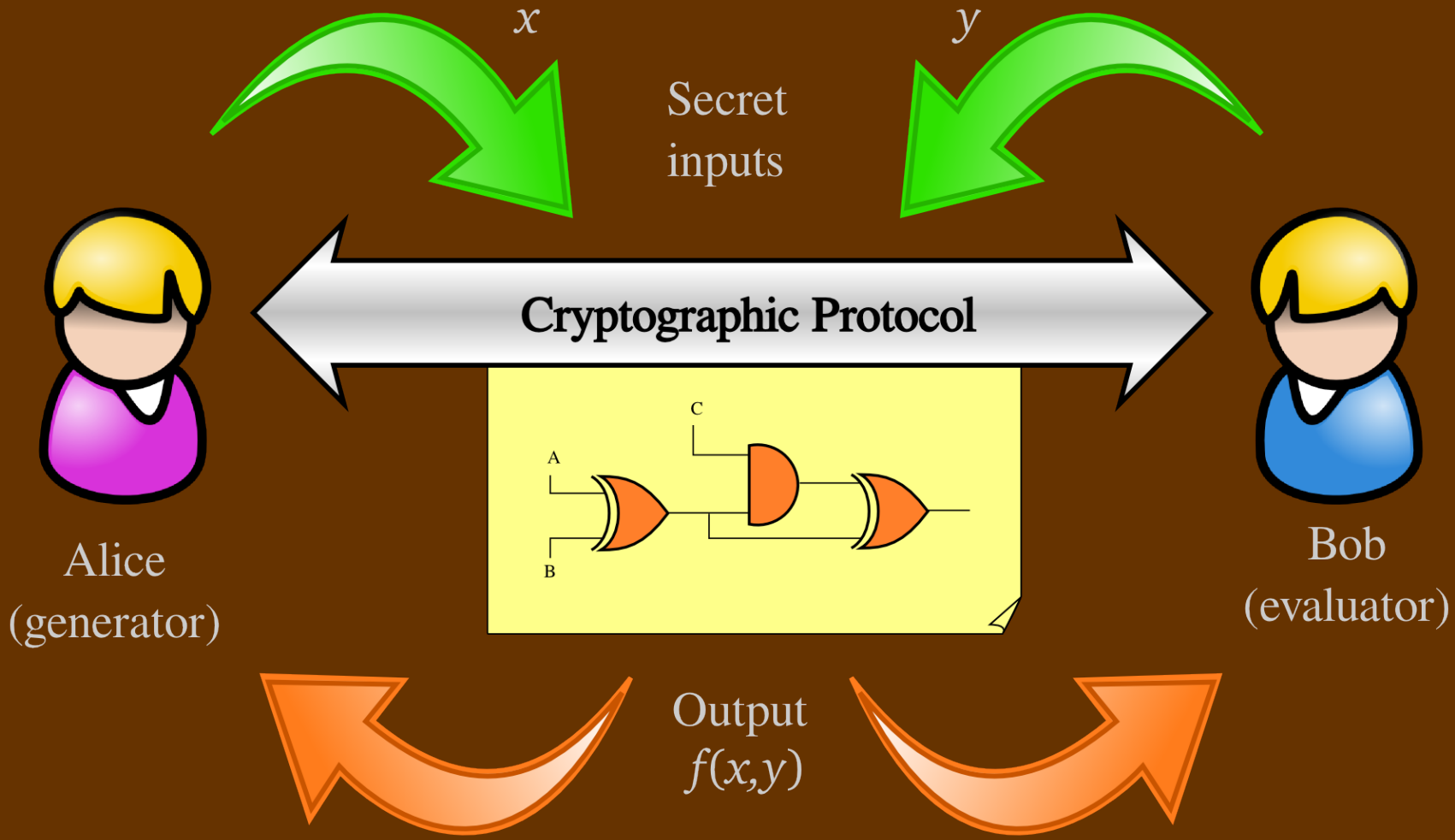
Two Halves Make a Whole

Reducing Data Transfer in Garbled Circuits using Half Gates

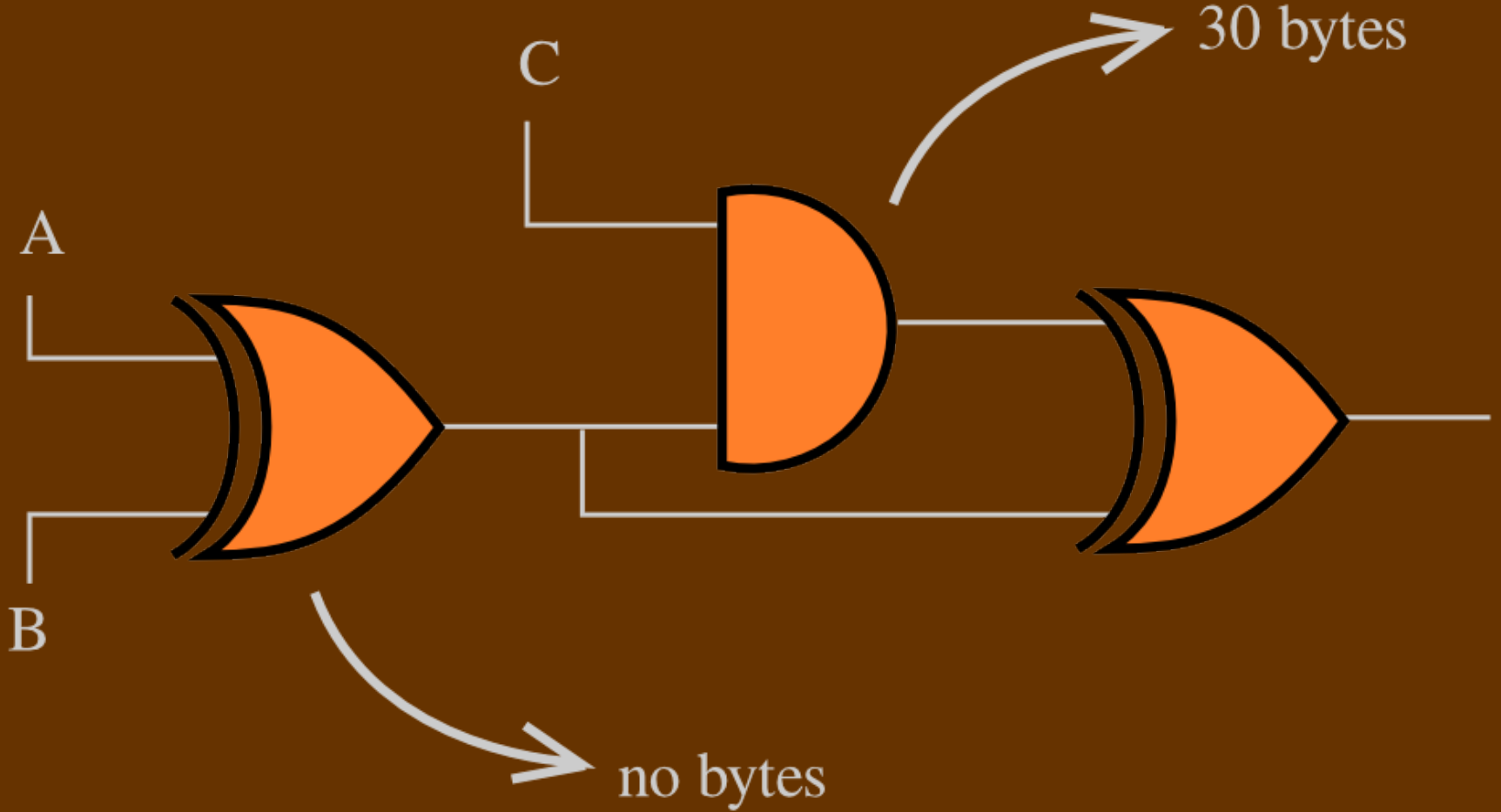
Samee Zahur <samee@virginia.edu>
Mike Rosulek <rosulekm@eecs.oregonstate.edu>
David Evans <evans@virginia.edu>

Two Party Computation (2PC)

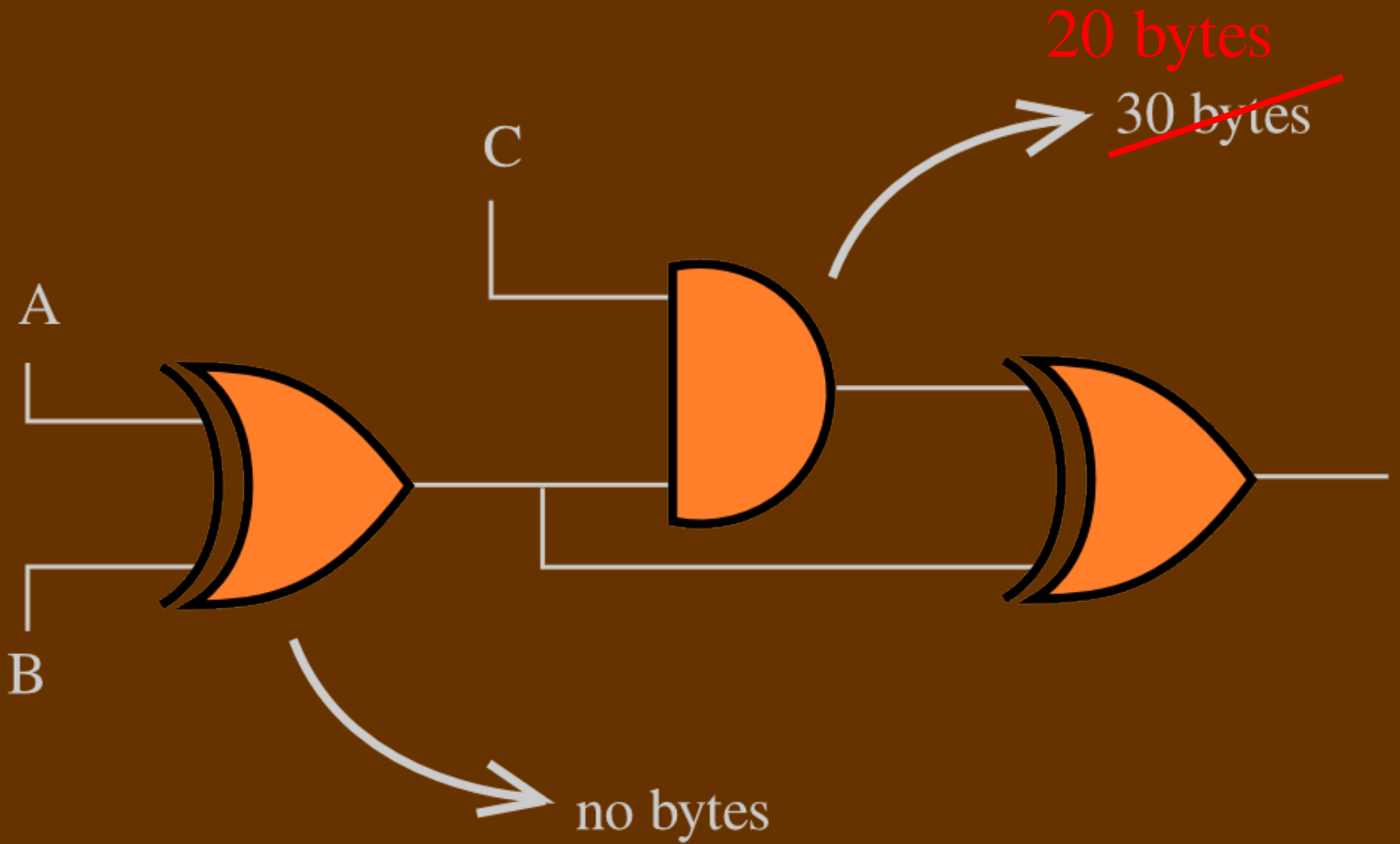


Yao : Constant-round protocol

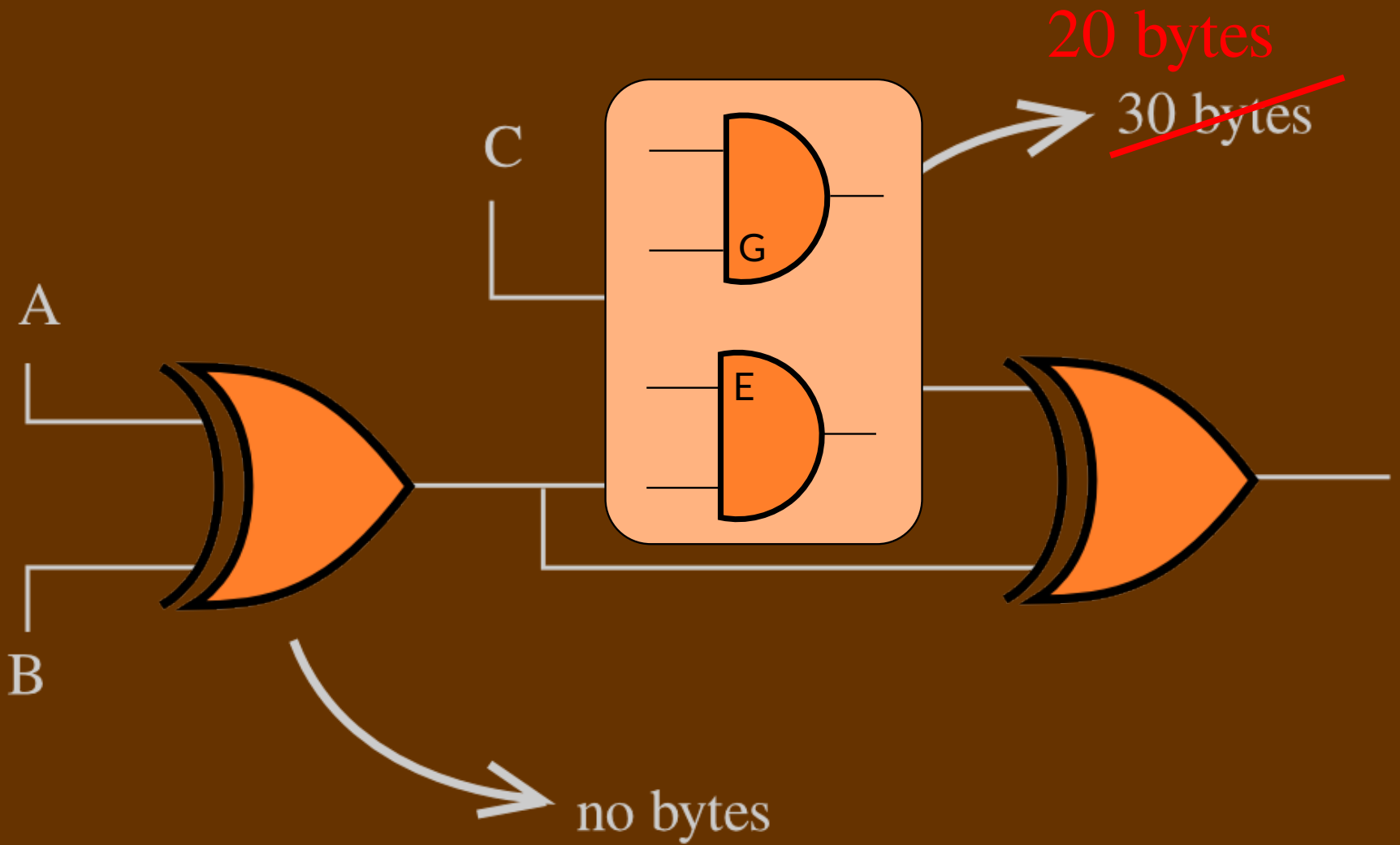
Result – Lower Bandwidth



Result – Lower Bandwidth

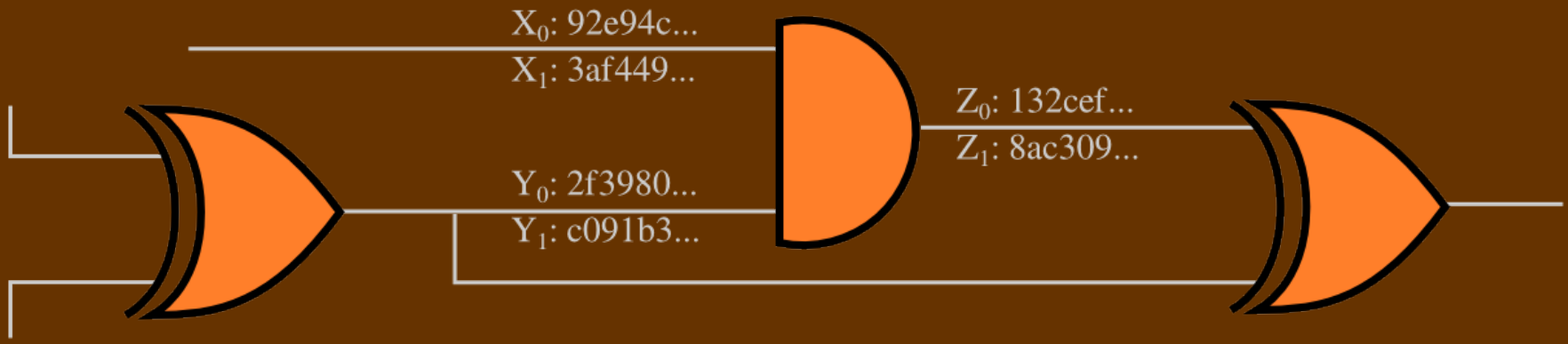


Result – Lower Bandwidth

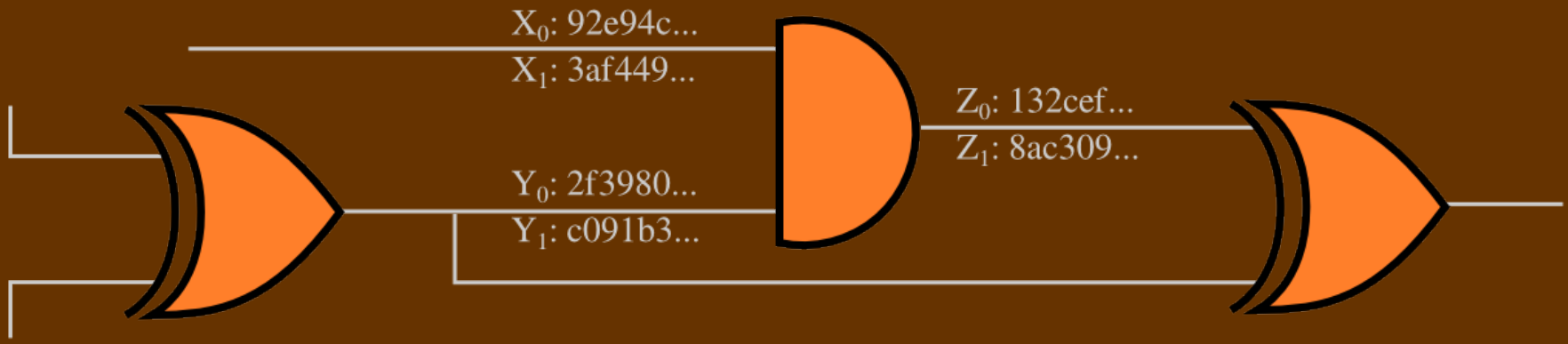


Whirlwind Primer of Yao's Protocol

Yao : Classical Version



Yao : Classical Version

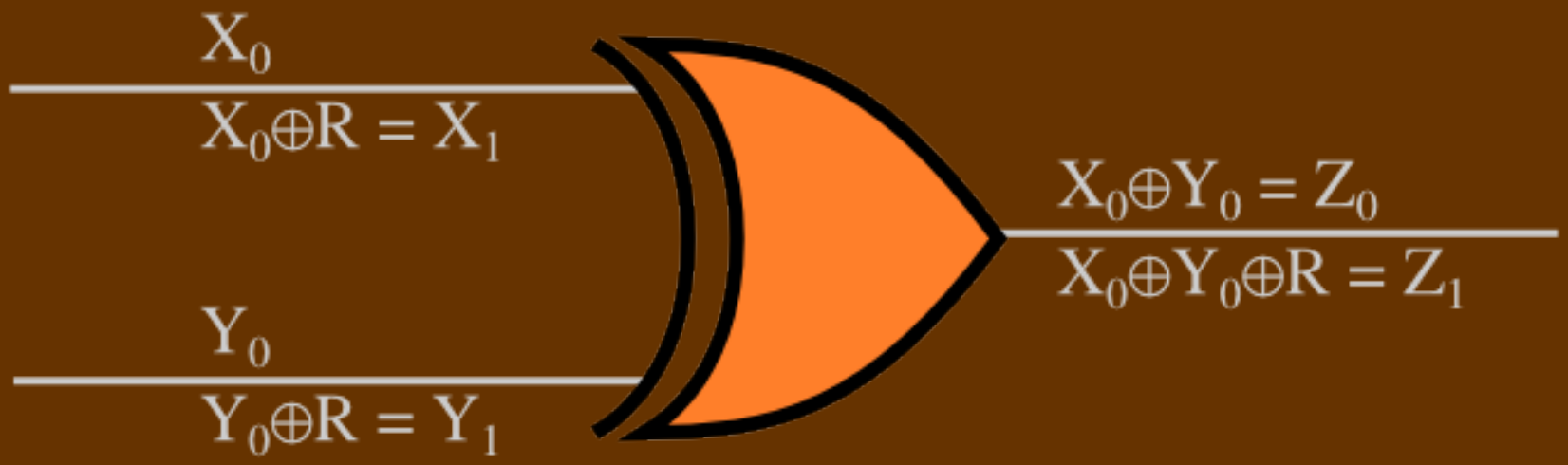


Input 1	Input 2	Output	Encrypted Output
X_0	Y_0	Z_0	$Enc(X_0 \| Y_0, Z_0)$
X_0	Y_1	Z_0	$Enc(X_0 \| Y_1, Z_0)$
X_1	Y_0	Z_0	$Enc(X_1 \| Y_0, Z_0)$
X_1	Y_1	Z_1	$Enc(X_1 \| Y_1, Z_1)$

Yao : Free XORs

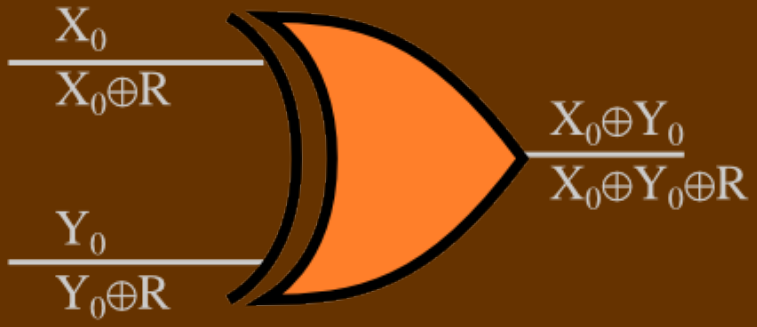
[Kolesnikov, Schneider; 2008]

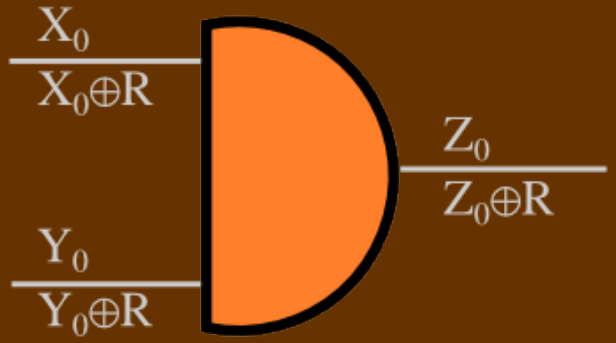
Generator's secret : R



Yao : Garbled Row Reduction (GRR)

[Naor, Pinkas, Sumner; 1999]



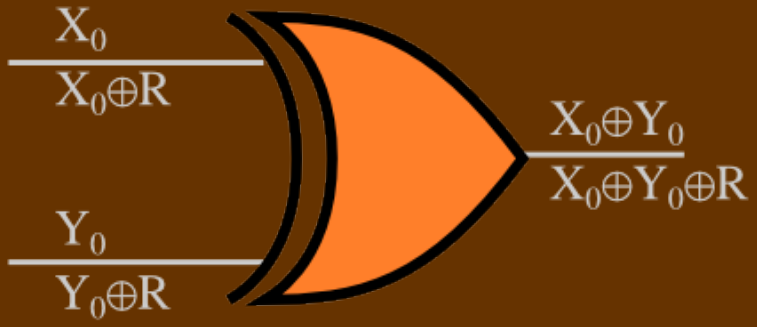


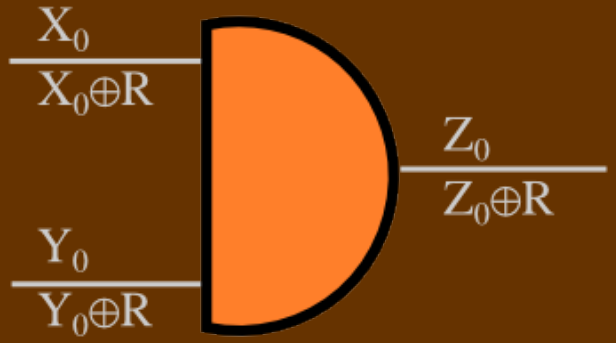
No data transfer

Inputs		Encrypted Output
X_0	Y_0	$Enc(X_0 \| Y_0, Z_0)$
X_0	Y_1	$Enc(X_0 \| Y_1, Z_0)$
X_1	Y_0	$Enc(X_1 \| Y_0, Z_0)$
X_1	Y_1	$Enc(X_1 \| Y_1, Z_1)$

Yao : Garbled Row Reduction (GRR)

[Naor, Pinkas, Sumner; 1999]





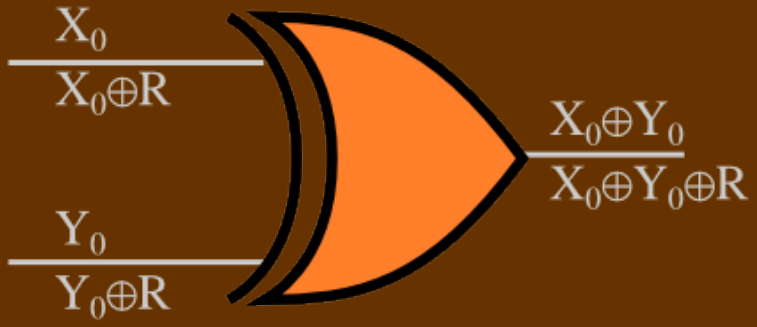
$Z_0 = Dec(X_0 || Y_0, \text{“00...0”})$

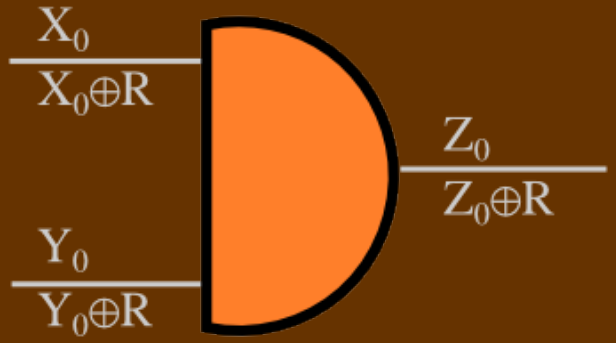
No data transfer

Inputs		Encrypted Output
X_0	Y_0	$Enc(X_0 \| Y_0, Z_0)$
X_0	Y_1	$Enc(X_0 \| Y_1, Z_0)$
X_1	Y_0	$Enc(X_1 \| Y_0, Z_0)$
X_1	Y_1	$Enc(X_1 \| Y_1, Z_1)$

Yao : Garbled Row Reduction (GRR)

[Naor, Pinkas, Sumner; 1999]





$Z_0 = \text{Dec}(X_0 \| Y_0, \text{“00...0”})$

No data transfer

Inputs		Encrypted Output
X_0	Y_0	$\text{Enc}(X_0 \| Y_0, Z_0)$ → “00..0”
X_0	Y_1	$\text{Enc}(X_0 \| Y_1, Z_0)$
X_1	Y_0	$\text{Enc}(X_1 \| Y_0, Z_0)$
X_1	Y_1	$\text{Enc}(X_1 \| Y_1, Z_1)$

Comparison

		Size per gate	
		XOR	AND
Classical	[Yao '86, BMR '90]	4	4
Row-reduction, GRR3	[NPS '99]	3	3
Free XOR + GRR3	[KS '08]	0	3

Comparison

		Size per gate	
		XOR	AND
Classical	[Yao '86, BMR '90]	4	4
Row-reduction, GRR3	[NPS '99]	3	3
Free XOR + GRR3	[KS '08]	0	3
Row-reduction, GRR2	[PSSW '09]	2	2
fleXOR	[KMR '14]	0/1/2	2

Comparison

		Size per gate	
		XOR	AND
Classical	[Yao '86, BMR '90]	4	4
Row-reduction, GRR3	[NPS '99]	3	3
Free XOR + GRR3	[KS '08]	0	3
Row-reduction, GRR2	[PSSW '09]	2	2
fleXOR	[KMR '14]	0/1/2	2
Half gates	**[this work]**	**0**	**2**

Our Approach

Observation

"Half gates" are cheap!

Inputs		Encrypted Output
X_0	Y_0	$Enc(X_0 \| Y_0, Z_0)$
X_0	Y_1	$Enc(X_0 \| Y_1, Z_0)$
X_1	Y_0	$Enc(X_1 \| Y_0, Z_0)$
X_1	Y_1	$Enc(X_1 \| Y_1, Z_1)$

"00..0"

$Z = X \cdot Y$

Nobody knows any input

Input	Encrypted Output
X_0	$Enc(X_0, Z_0)$
X_1	$Enc(X_1, Z_r)$

"00..0"

$Z = X \cdot r$

Generator knows an input

Evaluator-Side Half Gates

Input	Encrypted Output
Y_0	$Enc(Y_0, Z_0)$
Y_1	$Enc(Y_1, X_0 \oplus Z_0)$

$$Z = X \cdot y$$

Two Halves Make a Whole

$$
\begin{aligned}
a\,b &= a\,(r \oplus r \oplus b) \\
&= a\,r \oplus a\,(r \oplus b)
\end{aligned}
$$

Two Halves Make a Whole

$$a\,b = a\,(r \oplus r \oplus b)$$

$$= a\,r \oplus a\,(r \oplus b)$$

Generator knows r

Evaluator knows $(r \oplus b)$

Performance

	Resource usage reduction		
Benchmark	**Bandwidth**	**Time**	**Energy**
Edit distance	33.3%	25.7%	21.0%
AES	33.3%	7%	5.3%
Set intersection	32.2%	19.7%	15.5%

Security Assumption

Traditionally, random oracle model

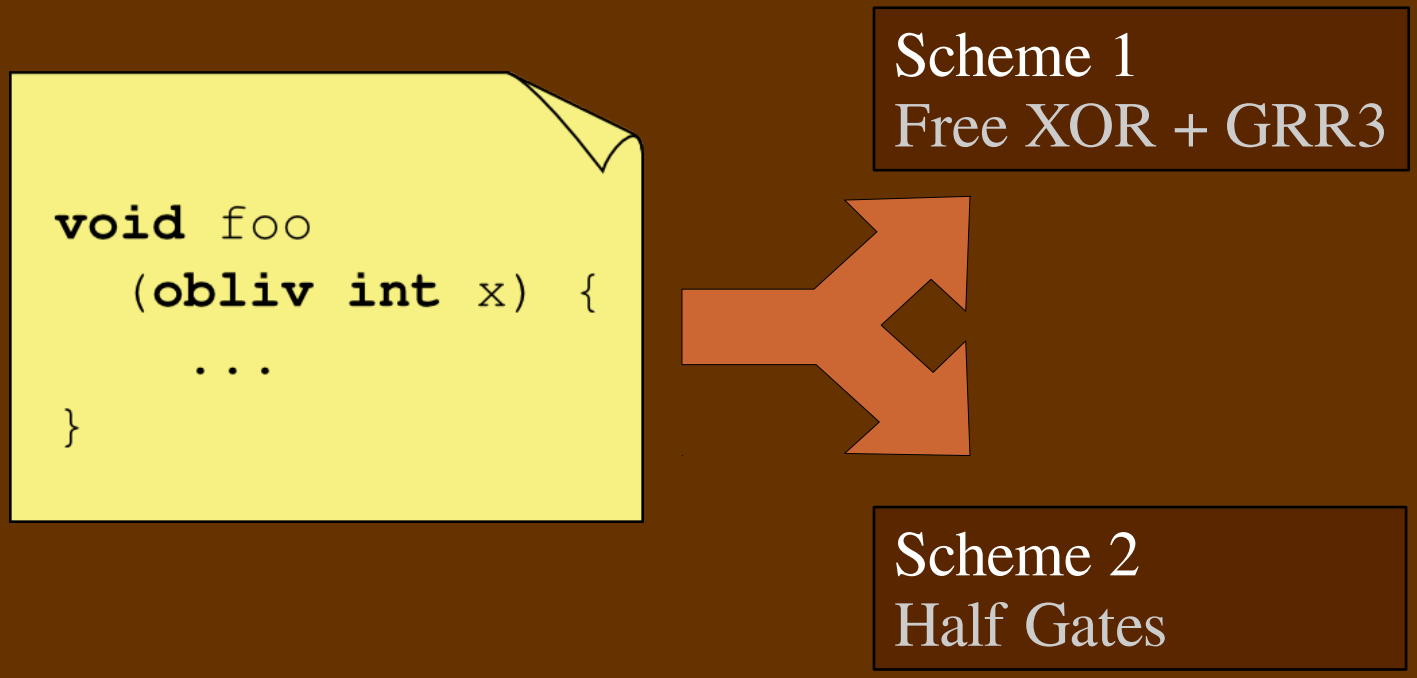
$$\mathrm{Enc}_i(K, X) = \mathrm{Dec}_i(K, X) = \mathrm{H}(K, i) \oplus X$$

Alternate assumptions:

- Circular Correlation Robustness [CKKZ '12]
- Ideal random permutation, for AES-NI [BHK '13]

Obliv-C

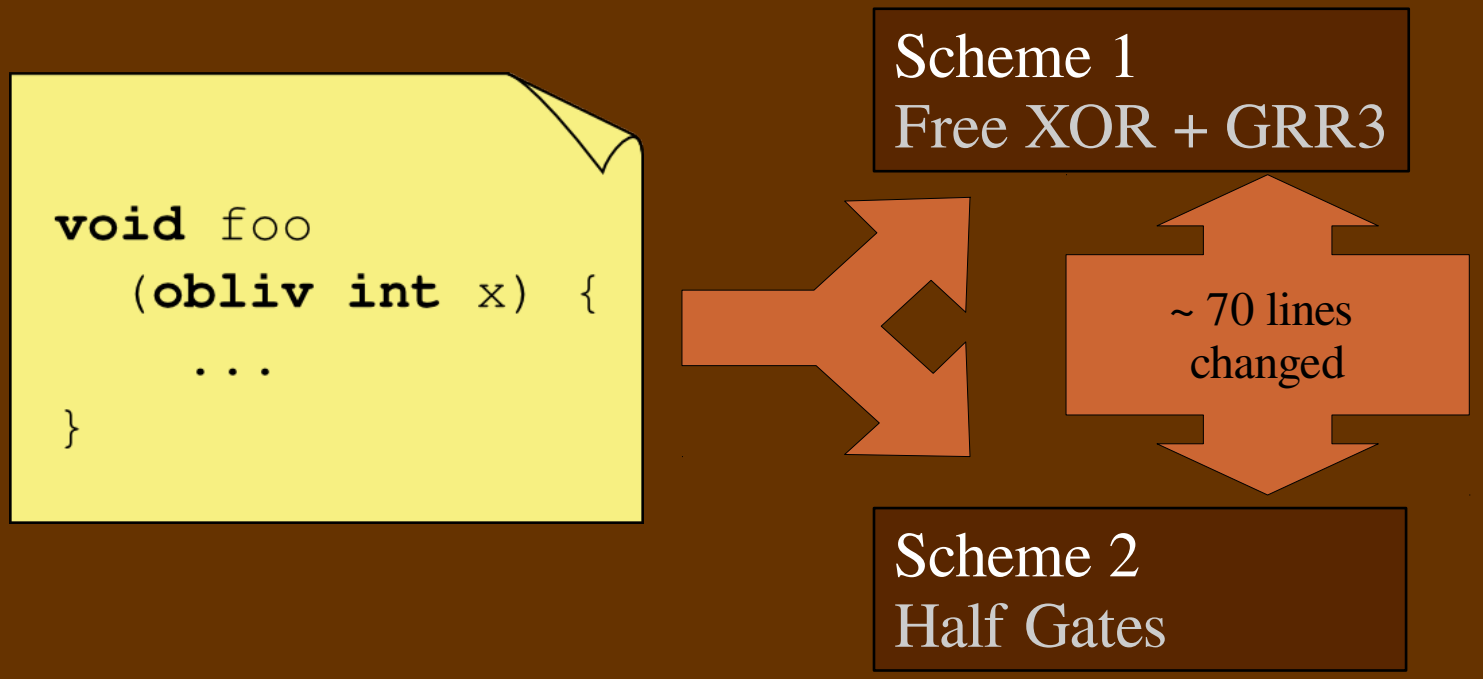
http://oblivc.org/



~2.2 M (non-XOR) gates per second

Obliv-C

http://oblivc.org/

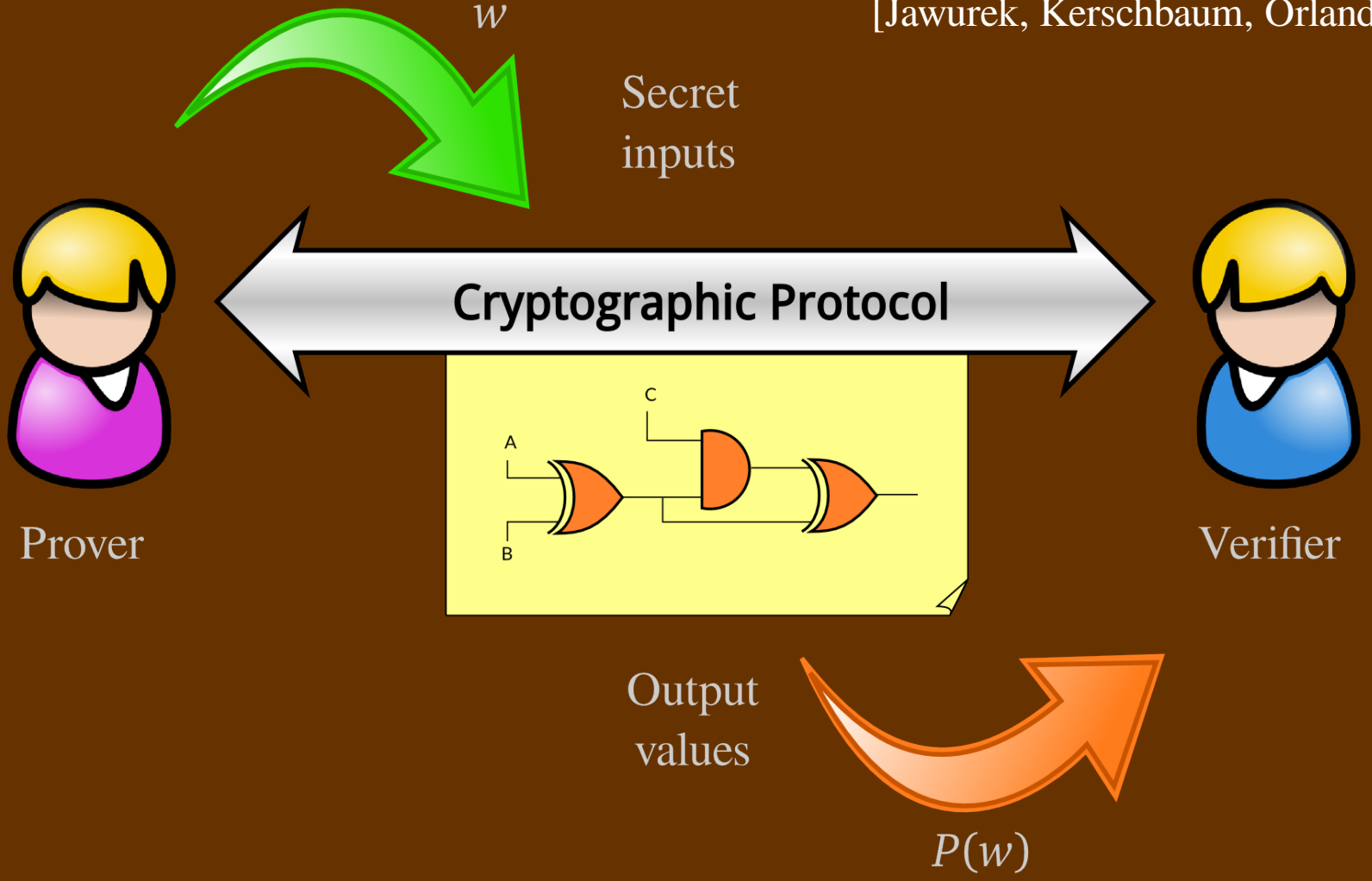


~2.2 M (non-XOR) gates per second

Applications in Zero-Knowledge

ZK Proofs with (Privacy-Free) Yao

[Jawurek, Kerschbaum, Orlandi; 2013]



Looking Ahead

Could we do better?
Maybe only one ciphertext row per gate?
Maybe try sacrificing free-XOR (again)?

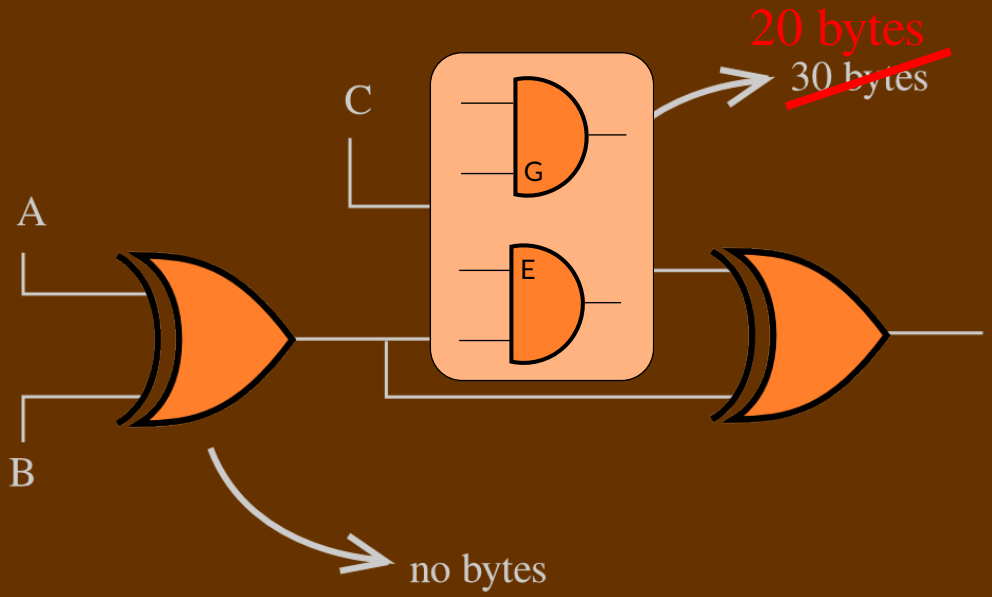
Conditions Necessary for Improvement

Theorem: *one cannot securely garble AND gates with fewer than 2k bits for any garbling scheme where the gate output labels are a GF(2^k)-linear combination of the input labels and their (non-nested) hashes.*

For further improvements, one must:

- Use nested hashes,
- Use other non-linear operations, and/or
- Consider larger blocks of circuits

Conclusion



Full gates are *never* necessary.

Further improvement needs significantly different techniques

The End

http://oblivc.org/

Evaluator-Side Half Gate

Input	Encrypted Output
Y_0	$\text{Enc}(Y_0, Z_0)$
Y_1	$\text{Enc}(Y_1, X_0 \oplus Z_0)$

$$Z = X \cdot y$$

Evaluator-Side Half Gate

If $y = 0$ then $z = 0$

Decrypt $\rightarrow Z_0$

Input	Encrypted Output
Y_0	$\text{Enc}(Y_0, Z_0)$
Y_1	$\text{Enc}(Y_1, X_0 \oplus Z_0)$

$Z = X \cdot y$

Evaluator-Side Half Gate

If $y = 0$ then $z = 0$

Decrypt $\rightarrow Z_0$

If $y = 1$ then $z = x$

Decrypt $\rightarrow X_0 \oplus Z_0$

Input	Encrypted Output
Y_0	$Enc(Y_0, Z_0)$
Y_1	$Enc(Y_1, X_0 \oplus Z_0)$

$Z = X \cdot y$

Evaluator-Side Half Gate

If $y = 0$ then $z = 0$

Decrypt $\rightarrow Z_0$

If $y = 1$ then $z = x$

Decrypt $\rightarrow X_0 \oplus Z_0$

XOR with $X_0 \rightarrow Z_0$

$X_0 \oplus R \rightarrow Z_0 \oplus R$

Input	Encrypted Output
Y_0	$\text{Enc}(Y_0, Z_0)$
Y_1	$\text{Enc}(Y_1, X_0 \oplus Z_0)$

$Z = X \cdot y$

Evaluator-Side Half Gate

If $y = 0$ then $z = 0$

Decrypt $\rightarrow Z_0$

If $y = 1$ then $z = x$

Decrypt $\rightarrow X_0 \oplus Z_0$

XOR with $X_0 \rightarrow Z_0$

$X_0 \oplus R \rightarrow Z_0 \oplus R$

Input	Encrypted Output
Y_0	$Enc(Y_0, Z_0)$
Y_1	$Enc(Y_1, X_0 \oplus Z_0)$

"00..0"

$Z = X \cdot y$

Comparison – Privacy Free

	Gate size		Hash calls per gate			
			generator		evaluator	
	XOR	AND	XOR	AND	XOR	AND
row-reduction (GRR1)	1	1	0	3	0	1
free XOR + GRR2	0	2	0	3	0	1
fleXOR	0/1/2	1	0	3	0	1
Half gates [this work]	**0**	**1**	**0**	**2**	**0**	**1**