2-party Secure Computation

Malicious Adversaries

Bar-Ilan Winter School, Feb 2015

abhi shelat

Brief Survey

x

$$f(x, y)$$

y

"...and nothing else"

image credit: http://wikis.zum.de/rmg/Benutzer:Deininger_Matthias/Facharbeit/Alice_Bob_und_Mallory

The age of optimism

80s	90s	00s	10s	20s
PKE SFE	PKE	PKE		
Invented	Practical	Ubiquit		
		SFE	SFE	SFE
		Feasible	Practical	Ubiquit

MNPS04
MNPS08
KS06, K08

Fairplay
Honest but curious

4k gates,
600 gates/sec

MNPS04 MNPS08	Fairplay Honest but curious	4k gates, 600 gates/sec
KS06, K08		
LP04, LP07, LPS08	Cut-and-choose Malicious adv	1k gates, 4 gates/sec

2009

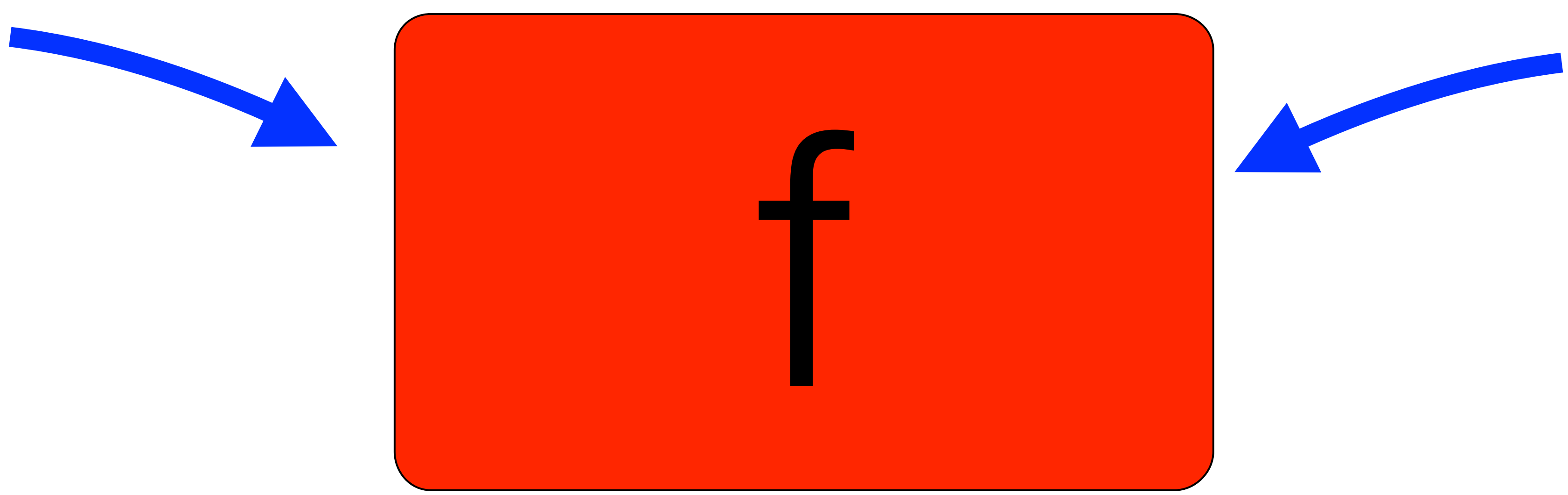
10x

PSSW09

AES circuit
Malicious adv

40k gates,
35 gates/sec
(2^{-40} security)

Alice
x



Bob
y

$AES_x(y)$

MNPS04 MNPS08 KS06, K08	Fairplay Honest but curious	4k gates, 600 gates/sec
LP04, LP07, LPS08 PSSW09	Cut-and-choose Malicious adv	1k gates, 4 gates/sec
LP10	Hybrid, C&C+ZK Malicious adv	
JS07 JKS08	Yao + ZK Malicious adv	
NO09	Lego+ Malicious adv	

MNPS04 MNPS08 KS06, K08	Fairplay Honest but curious	4k gates, 600 gates/sec
LP04, LP07, LPS08 PSSW09	Cut-and-choose Malicious adv	1k gates, 4 gates/sec
LP10	Hybrid, C&C+ZK Malicious adv	
JS07 JKS08	Yao + ZK Malicious adv	
NO09	Lego+ Malicious adv	
IPS08,09, LOP11	Better BB Cut-and-choose Malicious adv	
HL08, HL08b	Tamper proof model	

			$(2^{-40}$ security)
	PSSW09	AES circuit Malicious adv	40k gates, 35 gates/sec
	SS11	Hybrid CC+ZK Malicious adv	40k gates, 130 gates/sec
Amortized	NNOB11	GMW + OT Ext Malicious adv	40k gates, 20k gates/sec $(2^{-58}$ security) 3s/bloc
	DPSZ11	GMW + Beaver Malicious adv	100k gates, 10 gates/sec 10ms/gate ~ 100 g/s

5000x

2011

HEKM11

Pipeline + Circuit Lib
Honest but curious

40k gates
12k gates/sec

Bottleneck became the Compiler

2011

HEKM11	Pipeline + Circuit Lib Honest but curious	40k gates 12k gates/sec

Bottleneck became the Compiler

JKS08	200x200 edit distance	660s

2011

HEKM11 Pipeline + Circuit Lib 40k gates

Honest but curious 12k gates/sec

Bottleneck became the Compiler

JKS08 200x200 edit distance 660s

HEKM11 1.2B nonxor gates

96k g/s

2k x 10k edit distance

2012

1.2B nonxor gates   96k g/s

2k x 10k edit distance

KSS12

6B gates   4K x 4K edit distance   86k gates/sec

260m gate   RSA-256   125k gates/sec

330m gate   2k x 2s Edit   123k gates/sec

MNPS04 MNPS08 KS06, K08	Fairplay Honest but curious	4k gates, 600 gates/sec
LP04, LP07, LPS08	Cut-and-choose Malicious adv	1k gates, 4 gates/sec
PSSW09	AES circuit Malicious adv	40k gates, 35 gates/sec
LP11	Hybrid, C&C+ZK Malicious adv	
SS11	Hybrid C&C+ZK Malicious adv	40k gates, 130 gates/sec
KSS12	Hybrid CC+ZK, Parallel Malicious adv	6B gates, 130k gates/sec
SS13	CC, Parallel Malicious adv	B gates, 1M gates/sec

More Garbled Circuits work

K08	Output Auth	
KS08	Free XOR-trick	
CKKZ12	Using circular 2-corr RHF	
HEKM11	Pipeline + Circuit Lib Honest but curious	40k- 1.2B gates 12k-96k gates/sec
HS13	Less Memory, Parallel Honest but curious	
FN12 HMSG13	GPU system Honest but curious	35M gates/se

Amortized

JS07 JKS08 — Yao + ZK
Malicious adv

NO09 — Lego+
Malicious adv

IPS08,09, LOP11 — Better BB Cut-and-choose
Malicious adv

NNOB11 — GMW + OT Ext
Malicious adv

DPSZ11 — GMW + Beaver + SHE — 100k ops
Malicious adv
10ms/"op" ~ 100 ops/s

DKLPSS12 — 500 ops/s

SZ13 — GMW + OT Ext — Fast
??

Advanced Techniques

- Cut & choose
  - Lindell13
  - Huang-Evans-Katz13
- Amortization: C&C + LEGO
  - Huang-Katz-Kolesnikov-Kumaresan-Malozemoff14
  - Lindell-Riva14
- Garbling
  - Zahur-Evans-Rosulek14
  - Malkin-Pastro-shelat15
- Algorithmic
  - Venkatasubramanian-shelat15

ORAM Secure Computation

Gordon-Katz-Kolesnikov-Krell-Malkin12

Keller-Scholl14

- 4 accesses/second to oblivious array of size one million
- Dijkstra's algorithm:
  - 2^11 vertices and 2^12 edges in 10 hours
  - 2^18 vertices and 2^19 edges in 14 months
    (estimated from running a fully functional program)

Wang-Huang-Chan-shelat-Shi14

SCORAM: 4m gates/ORAM op

Wang-Chan-Shi14

CORAM: 500k gates/ORAM op

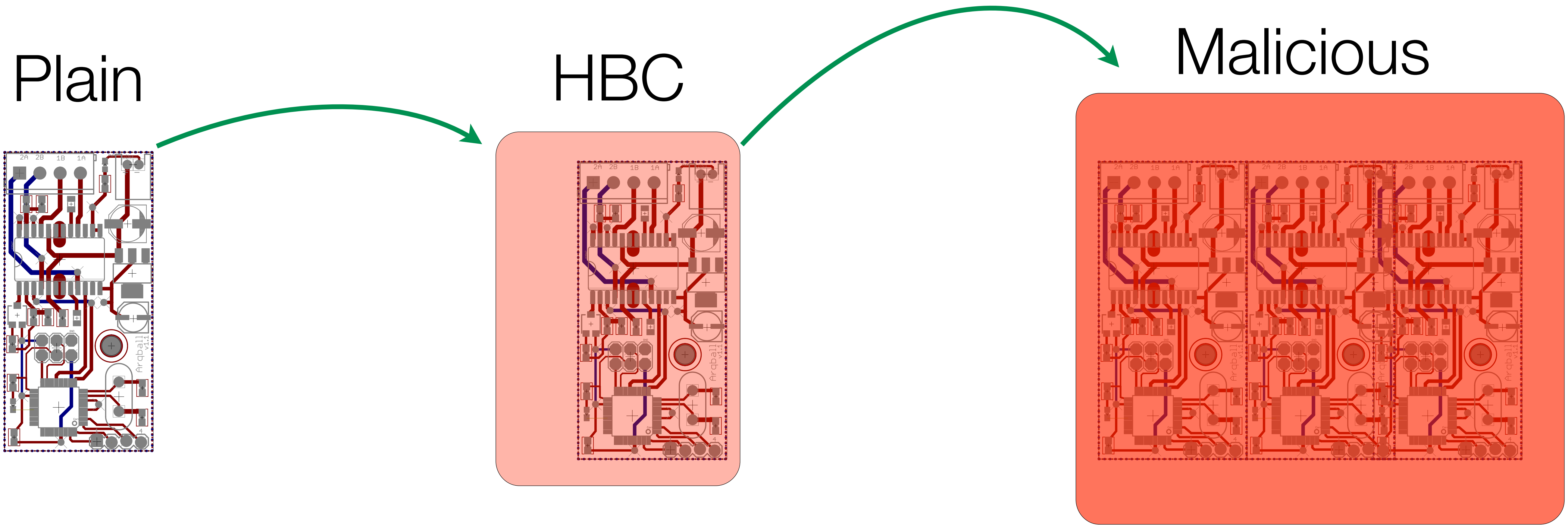
BenSasson-Chiesa-Tromer-Virza14

TinyRAM

Question: which secure computation techniques are preferable?

overhead

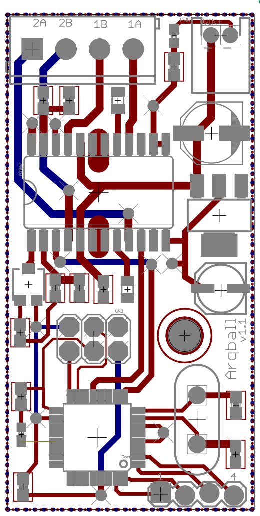
2-party Secure computation



overhead

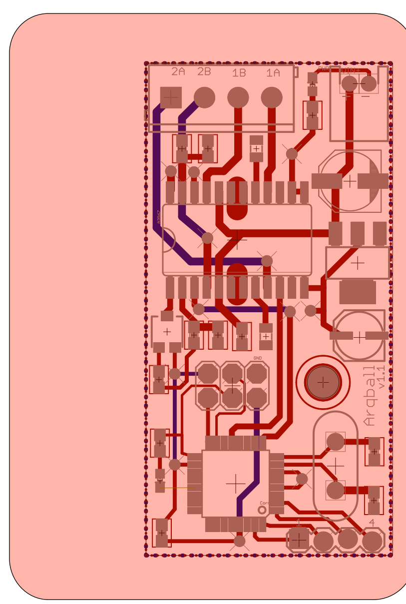
2-party Secure computation

Plain



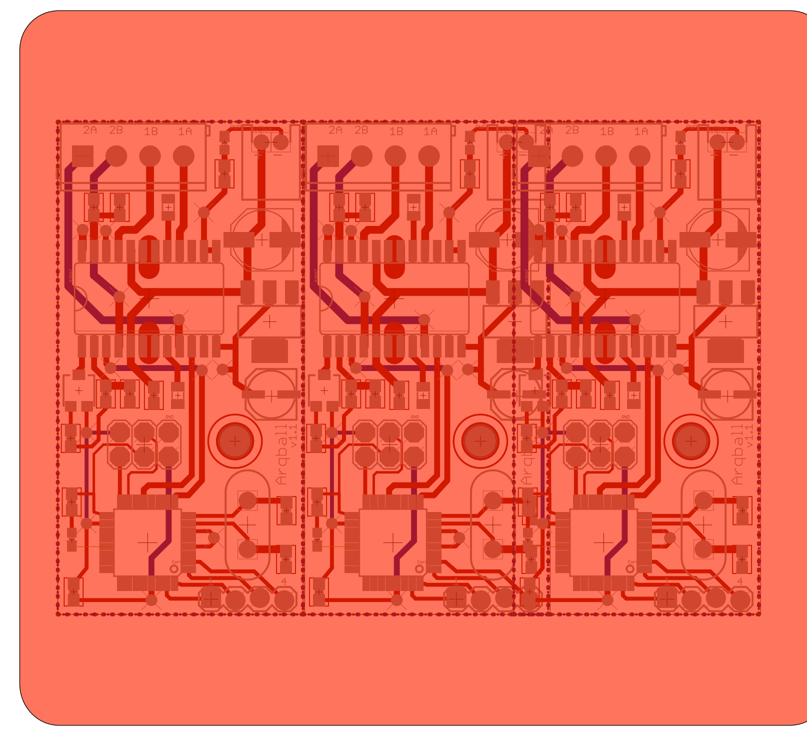
Comm
Comp
Assumption

HBC



Comm
Comp
Assumption

Malicious

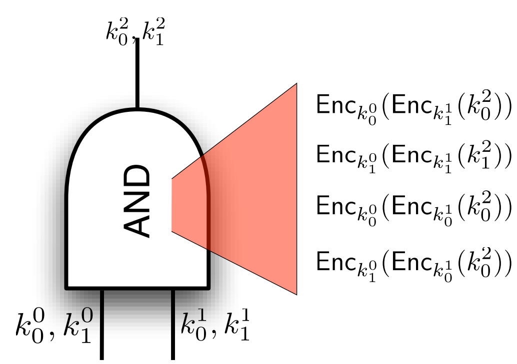


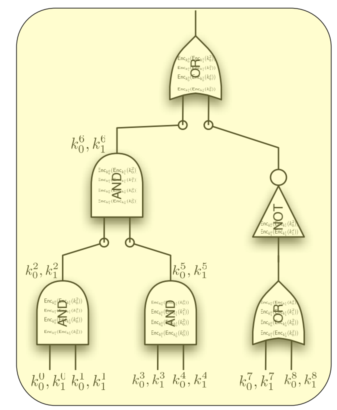
Parallelizability is KEY

Basic Protocols

Garbled circuits

Y82





0 1 c

Oblivious Transfer

Garbled gates + Composition + Key Mgmt

Honest-but-curious

f(x,y)

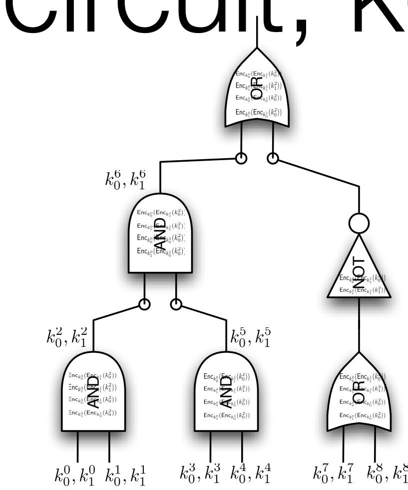
x

OT 1st msg

y

OT 2nd msg

Garbled circuit, keys for x



2 round!

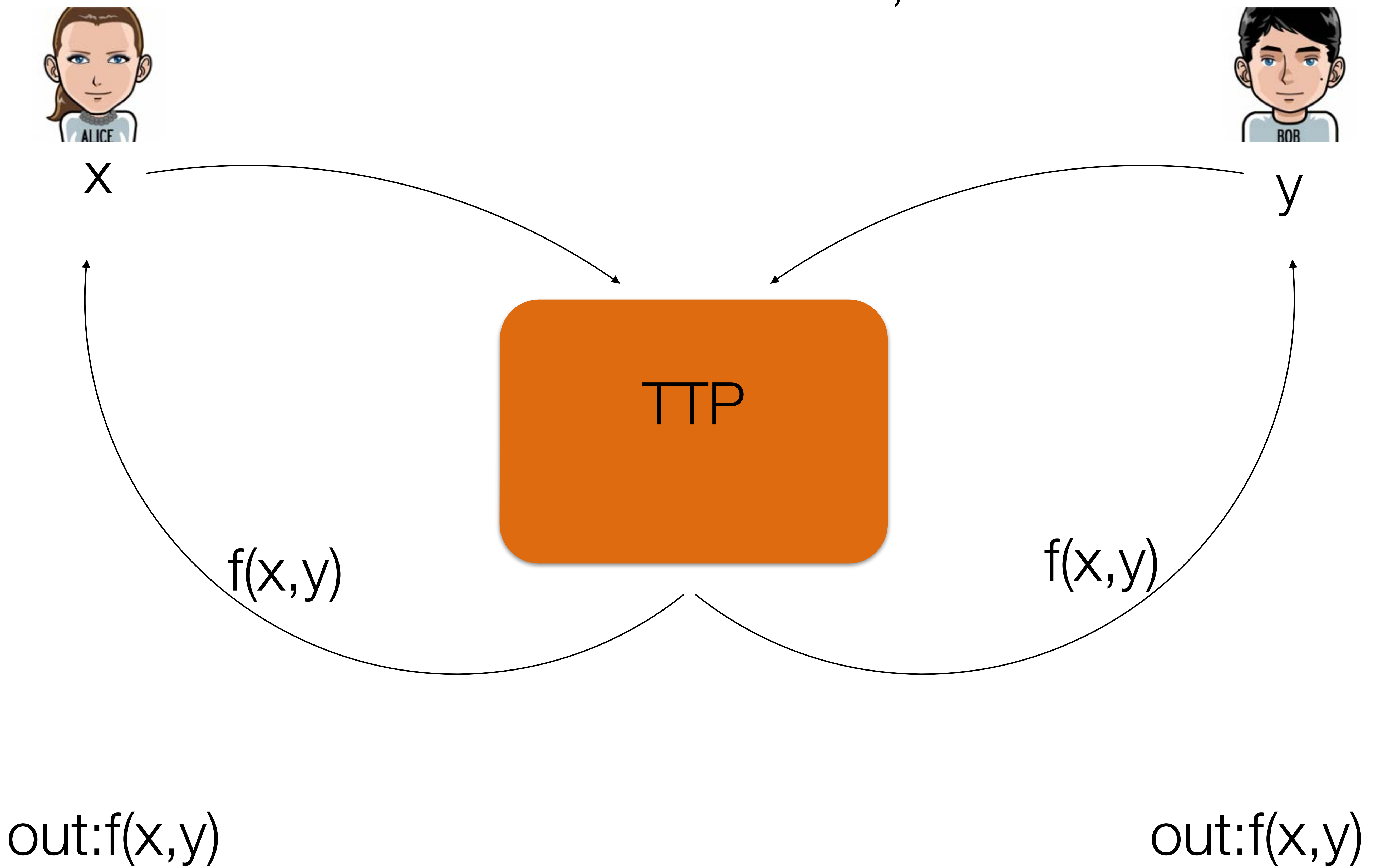
1. Incrementally construct maliciously-secure protocol

Definition

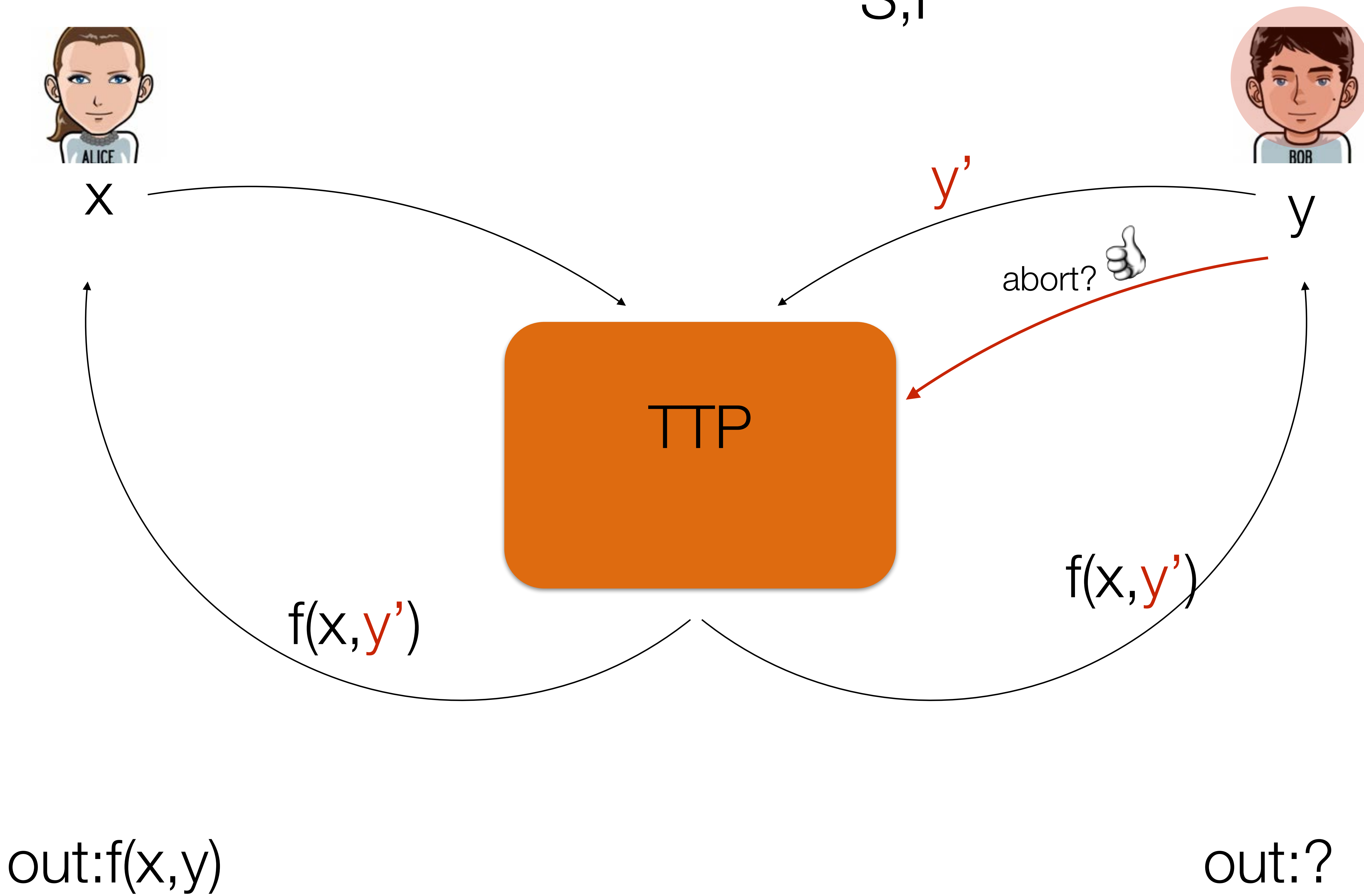
$$\forall A \; \exists S \; \forall (x_1, x_2), z$$

$$\textbf{IDEAL}_{f,S(z),I}(x_1, x_2, k) \approx_c \textbf{REAL}_{f,A(z),I}(x_1, x_2, k)$$

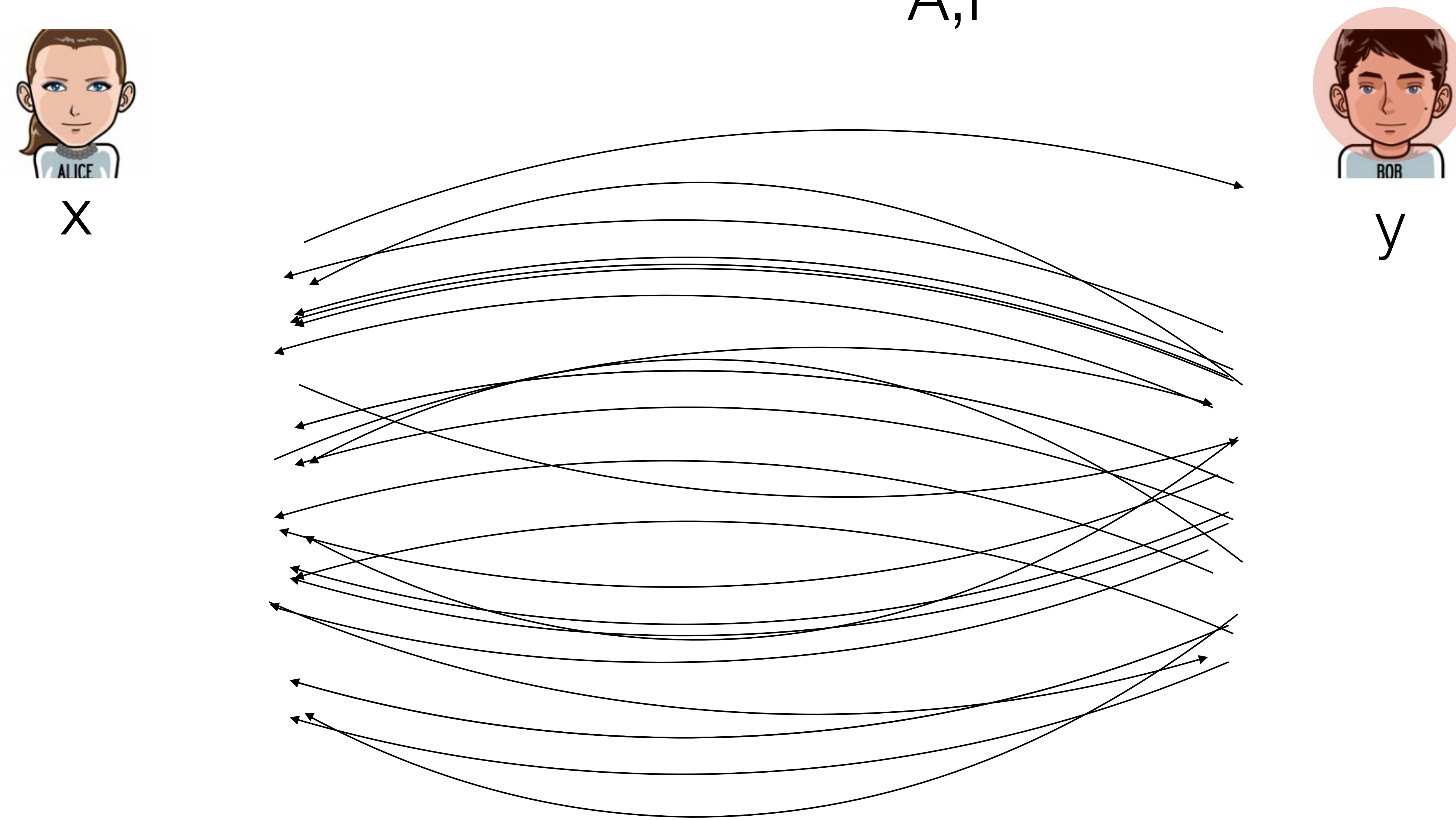
IDEAL$_{S,I}$



IDEAL$_{S,I}$



REAL $_{A,I}$



out:f(x,y)

out:?

Definition

$$\forall A \; \exists S \; \forall (x_1, x_2), z$$

$$\mathbf{IDEAL}_{f,S(z),I}(x_1, x_2, k) \approx_c \mathbf{REAL}_{f,A(z),I}(x_1, x_2, k)$$

1. Incrementally construct maliciously-secure protocol

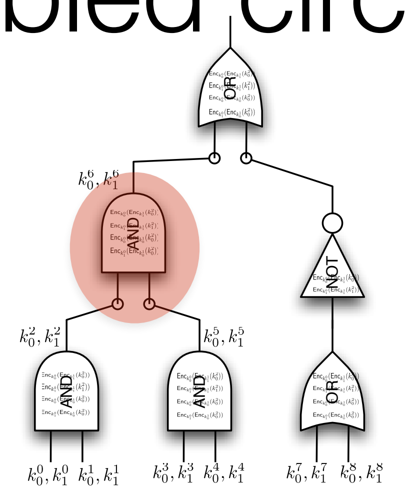
2. Optimize

What can go wrong?

OT 1st msg

OT 2nd msg

Garbled circuit



sending a bad circuit

Prove circuit is good

GMW,Jarecki-Shmatikov07

$$\bigwedge_{g\in G} \mathsf{CorrectGarble}_g \wedge \bigwedge_{w\in W} \mathsf{GoodKeys}_w \wedge \bigwedge_{w\in W_S} \mathsf{CorrectInput}_w \wedge \bigwedge_{w\in W_R} \mathsf{ZKS}_w \wedge \bigwedge_{w\in W_O} \mathsf{CorrectOutput}_w$$

where

$$\begin{aligned}
\mathsf{GoodKeys}_w &= \mathsf{ZKNotEq}(C_0^w, C_1^w) \\
\mathsf{CorrectInput}_w &= (\mathsf{ZKDL}(\mathfrak{g}, C_0^w/\alpha^{x_{b_w}^w}) \wedge \mathsf{ZKDL}(\mathfrak{g}, C_b)) \vee \\
&\quad (\mathsf{ZKDL}(\mathfrak{g}, C_1^w/\alpha^{x_{b_w}^w}) \wedge \mathsf{ZKDL}(\mathfrak{g}, C_b/\alpha)), \text{ where } C_b \text{ is the} \\
&\quad \text{sCS commitment inside } \mathsf{Com}_{cid_{Si}} \text{ if } w \text{ is the } i^{th} \text{ input wire of} S \\
\mathsf{CorrectOutput}_w &= \mathsf{ZKPlainEq2}(E_0^w, C_0^w, 0) \;\wedge\; \mathsf{ZKPlainEq2}(E_1^w, C_1^w, 1)
\end{aligned}$$

$$\begin{aligned}
\mathsf{CorrectGarble}_g &= \mathsf{CorrectShuffle}(0,0) \;\vee\; \mathsf{CorrectShuffle}(0,1) \vee \\
&\quad \mathsf{CorrectShuffle}(1,0) \;\vee\; \mathsf{CorrectShuffle}(1,1)
\end{aligned}$$

$$\begin{aligned}
\mathsf{CorrectShuffle}(\alpha,\beta) &= \mathsf{CorrectCipher}(0,0,\alpha,\beta) \;\wedge\; \mathsf{CorrectCipher}(0,1,\alpha,\beta) \;\wedge \\
&\quad \mathsf{CorrectCipher}(1,0,\alpha,\beta) \;\wedge\; \mathsf{CorrectCipher}(1,1,\alpha,\beta)
\end{aligned}$$

$$\begin{aligned}
\mathsf{CorrectCipher}(\sigma_A,\sigma_B,\alpha,\beta) &= \mathsf{ZKPlainEq}(F_{\alpha\beta}^{(1)}, C_{\alpha\oplus\sigma_A}^A, D_{\alpha\beta}) \;\wedge \\
&\quad \mathsf{ZKPlainEq}(F_{\alpha\beta}^{(2)}, C_{\beta\oplus\sigma_B}^B, (C_{g(\alpha\oplus\sigma_A,\beta\oplus\sigma_B)}^C/D_{\alpha\beta}))
\end{aligned}$$

$$\begin{aligned}\mathsf{CorrectGarble}_g = {} & \mathsf{CorrectShuffle}(0,0) \ \vee \ \mathsf{CorrectShuffle}(0,1) \vee \\ & \mathsf{CorrectShuffle}(1,0) \ \vee \ \mathsf{CorrectShuffle}(1,1)\end{aligned}$$

$$\begin{aligned}\mathsf{CorrectShuffle}(\alpha,\beta) = {} & \mathsf{CorrectCipher}(0,0,\alpha,\beta) \ \wedge \ \mathsf{CorrectCipher}(0,1,\alpha,\beta) \ \wedge \\ & \mathsf{CorrectCipher}(1,0,\alpha,\beta) \ \wedge \ \mathsf{CorrectCipher}(1,1,\alpha,\beta)\end{aligned}$$

$$\begin{aligned}\mathsf{CorrectCipher}(\sigma_A,\sigma_B,\alpha,\beta) = {} & \mathsf{ZKPlainEq}(F^{(1)}_{\alpha\beta}, C^A_{\alpha\oplus\sigma_A}, D_{\alpha\beta}) \ \wedge \\ & \mathsf{ZKPlainEq}(F^{(2)}_{\alpha\beta}, C^B_{\beta\oplus\sigma_B}, (C^C_{g(\alpha\oplus\sigma_A,\beta\oplus\sigma_B)}/D_{\alpha\beta}))\end{aligned}$$

32-clause Sigma-protocol PER gate

Given $\mathsf{com}(K_x^0)$, $\mathsf{com}(K_x^1)$, $\mathsf{com}(K_y^0)$, $\mathsf{com}(K_y^1)$, $\mathsf{com}(K_w^0)$, and $\mathsf{com}(K_w^1)$, P_2 needs P_1 to prove that the AND gate $(\delta, T_4, T_5, \sigma, T_\sigma)$ is correctly computed. More specifically,

(a) P_1 sends $\mathsf{com}(\delta; r)$ to P_2, and P_1 proves that $\mathsf{com}(\delta; r) = g^\delta h^r$.

(b) For every $(b_0, b_1) \in \{0,1\}^2$, let $i = 2 * b_0 + b_1$, P_1 sends $\mathsf{com}(T_i)$ to P_2 and proves that

$$\Big(\mathsf{com}(K_x^{b_0})\mathsf{com}(K_y^{b_1})\mathsf{com}(\delta) = \mathsf{com}(K_x^{b_0} + K_y^{b_1} + \delta)\Big) \wedge$$
$$\Big(\mathsf{com}(T_i) = \mathsf{com}(K_x^{b_0} + K_y^{b_1} + \delta)^{K_x^{b_0} + K_y^{b_1} + \delta}\Big).$$

Moreover, P_1 proves that $T_i \in \mathbb{Z}_N^*$ for $i = 0, 1, 2, 3$.

(c) Let $\texttt{Mask}(b_0, b_1)$ denote the case that $(K_x^0)_N = b_0$ and $(K_y^0)_N = b_1$. P_1 proves to P_2 that

$$\texttt{mask}(0,0) \vee \texttt{mask}(0,1) \vee \texttt{mask}(1,0) \vee \texttt{mask}(1,1).$$

In particular, for case $\texttt{mask}(b_0, b_1)$, let

$$\begin{cases} a_0 = 2 \cdot b_0 + b_1 \\ a_1 = 2 \cdot b_0 + (1 - b_1) \end{cases} \text{and} \begin{cases} a_2 = 2 \cdot (1 - b_0) + b_1 \\ a_3 = 2 \cdot (1 - b_0) + (1 - b_1). \end{cases}$$

It is defined that

$$\texttt{mask}(b_0, b_1) = (P(a_0) = T_0) \wedge (P(a_1) = T_1) \wedge (P(a_2) = T_2) \wedge (Q(a_3) = T_3),$$

where $P(x)$ is the Lagrange polynomial coincides at points $(-1, K_w^0)$, $(4, T_4)$, $(5, T_5)$, and (σ, T_σ); and $Q(x)$ is the Lagrange polynomial coincides at points $(-1, K_w^1)$, $(4, T_4)$, $(5, T_5)$, and (σ, T_σ).

Can optimize to ~19-clauses PER gate

Open problem to optimize so as to outperform C&C

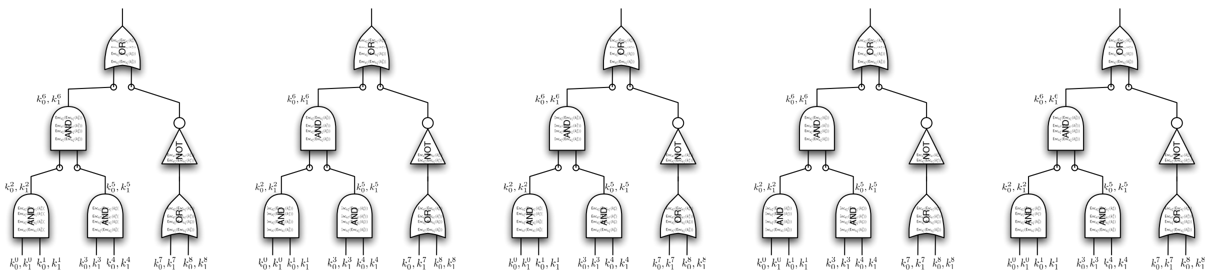
Cut & Choose

First Idea: Cut & Choose

OT 1st msg

OT 2nd msg

Send k fresh garbled circuits

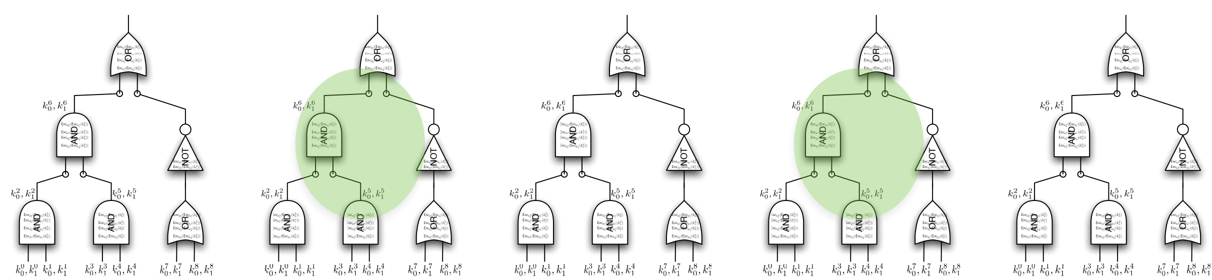


First Idea: Cut & Choose

OT 1st msg

OT 2nd msg

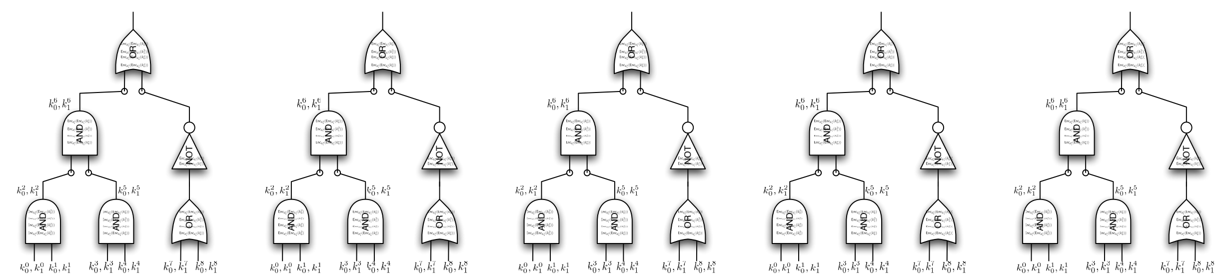
Send k fresh garbled circuits



“open” challenge set of t circuits

random coins for challenge

Cut and Choose



challenge

response

Garbler sends k circuits to Evaluator.

Evaluator selects t to test.

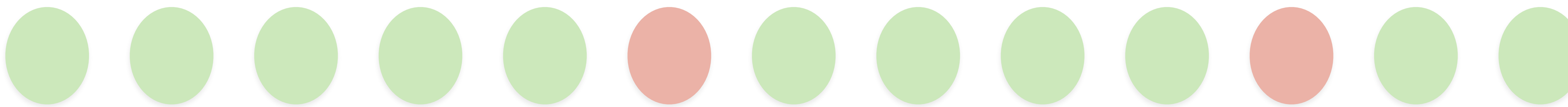
G asks E for random coins used to garble.

Evaluator verifies that all t circuits are valid.

What does this cut&choose test accomplish?

Balls & Bins

k circuits in total



Evaluator picks c circuits to corrupt.

Garbler picks t circuits to test.

\# of ways to pick only good: $\binom{k-c}{t}$

\# of ways to pick t: $\binom{k}{t}$

Given that evaluator checks t,
Pr that garbler succeeds in passing test:

$$\frac{\binom{k-c}{t}}{\binom{k}{t}}$$

setting t=k/2

Given that evaluator checks t,
Pr that garbler succeeds in passing test:

$$\frac{\binom{k-c}{t}}{\binom{k}{t}} = \frac{(k/2)(k/2-1)\cdots(k/2-c)}{k(k-1)(k-2)\cdots(k-c)}$$

setting t=k/2

Given that evaluator checks t,
Pr that garbler succeeds in passing test:

$$\frac{\binom{k-c}{t}}{\binom{k}{t}} = \frac{(k/2)(k/2-1)\cdots(k/2-c)}{k(k-1)(k-2)\cdots(k-c)} < 2^{-c}$$

setting t=k/2

NEGL probability that test passes if O(k) circuits are bad

$$2^{-c} > \frac{\binom{k-c}{t}}{\binom{k}{t}} = \frac{(k/2)(k/2-1)\cdots(k/2-c)}{k(k-1)(k-2)\cdots(k-c)} \geq \left(\frac{1}{2} - \frac{c}{k}\right)^{c}$$

setting t=k/2

$$\frac{k/2-c}{k-c} \geq \frac{k/2-c}{k} \geq \left(\frac{1}{2} - \frac{c}{k}\right)$$

Noticeable probability that O(1) circuits are corrupted

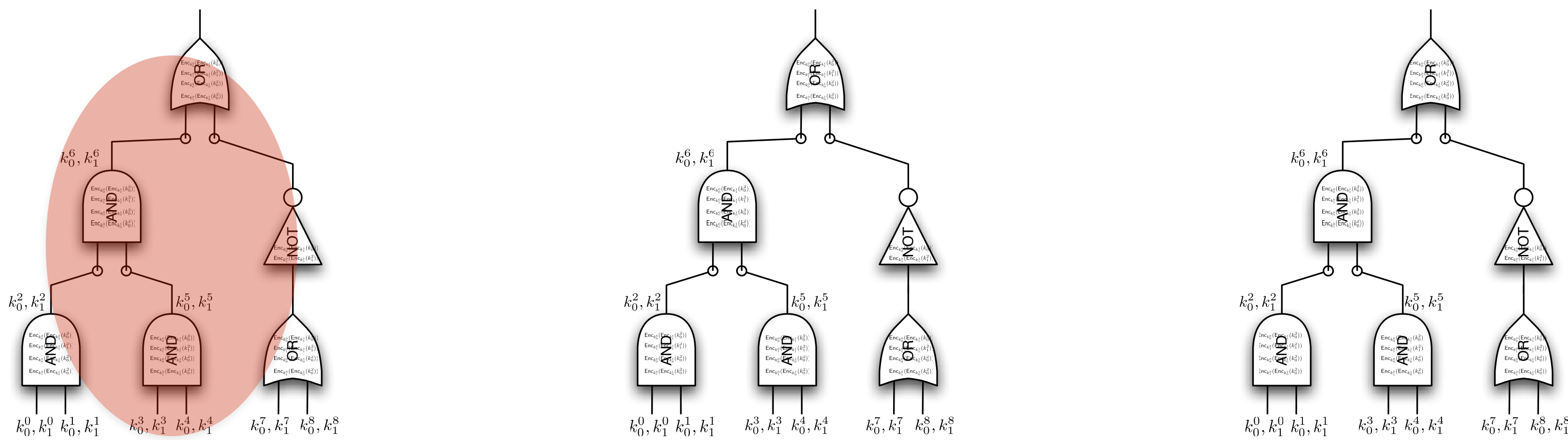
What do we do with the remaining circuits?

First idea:

Abort if outputs are not all the same.

First idea:

Abort if outputs are not all the same.

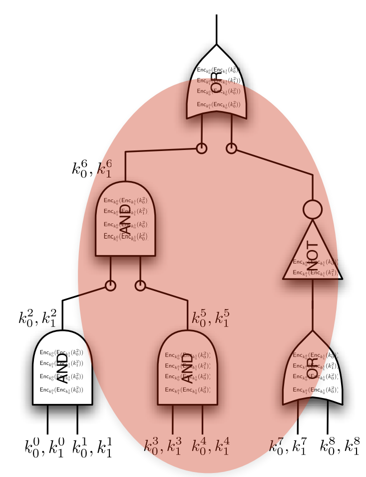


If y_1=0, output f(x,y)
else output f(x,y)+1

Test

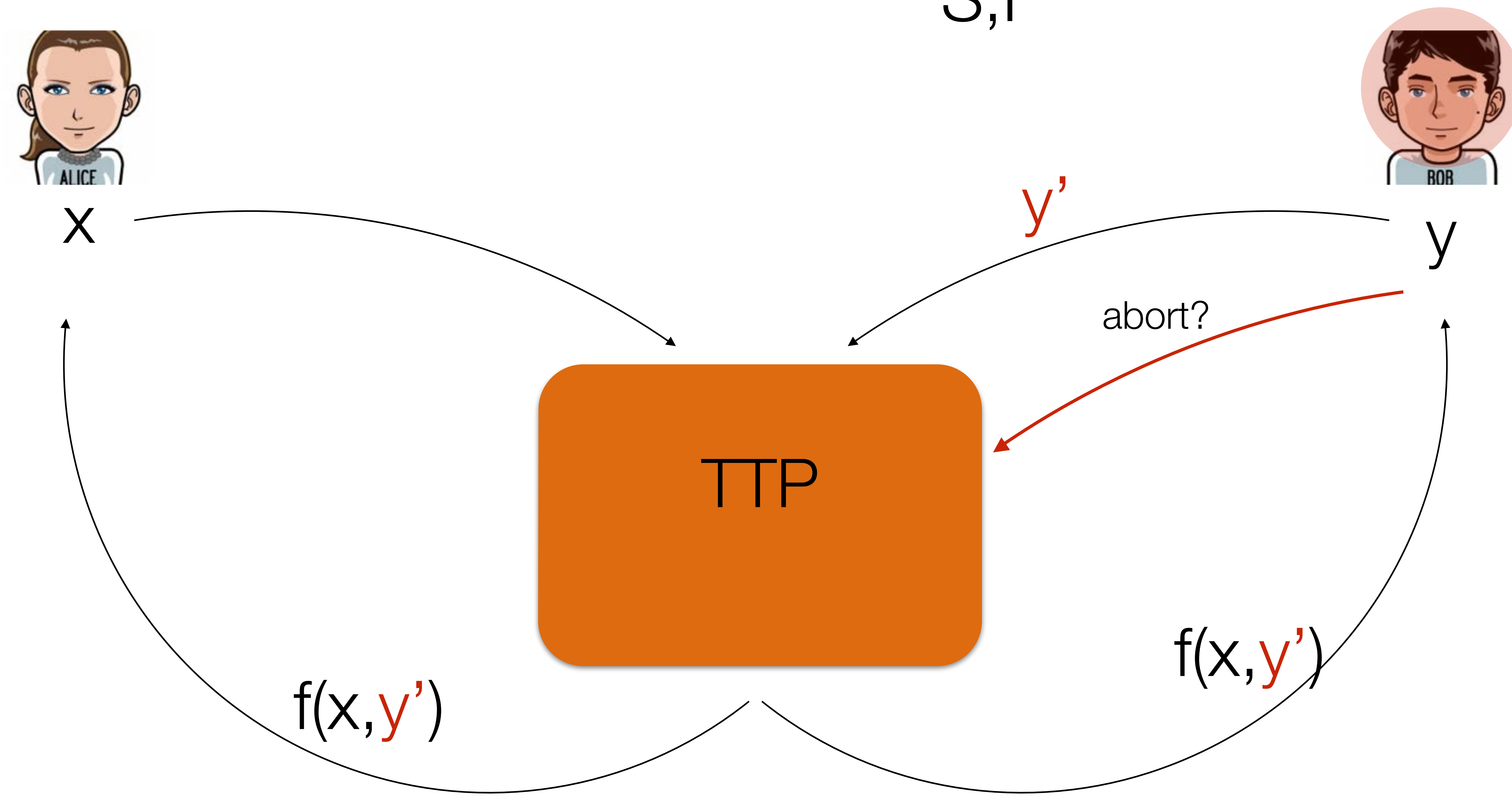
$$\forall A\ \exists S\ \forall (x_1, x_2), z$$

$$\mathbf{IDEAL}_{f,S(z),I}(x_1, x_2, k) \approx_c \mathbf{REAL}_{f,A(z),I}(x_1, x_2, k)$$



If y_1=0, output f(x,y)
else output f(x,y)+1

$\text{IDEAL}_{S,I}$



out:f(x,y)

out:?

Comment

In practice, all circuits must have same # of gates & same wiring.

Cheating restricted to changing gates.

Hard to analyze.

Second idea:

Eval all remaining circuits, take majority output.

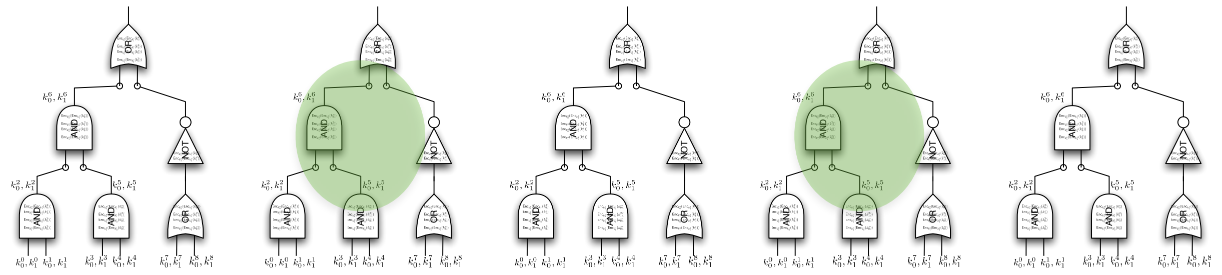
Third idea:

Eval all remaining circuits, exploit cheating later.

state-of-the-art [L13]

OT

Send k fresh garbled circuits



challenge set of t circuits

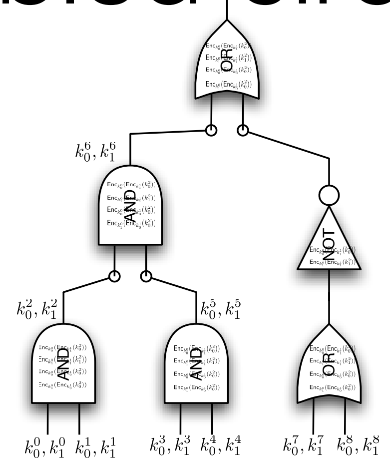
random coins for challenge

majority of Eval

Problem: Garblers' inputs

OT

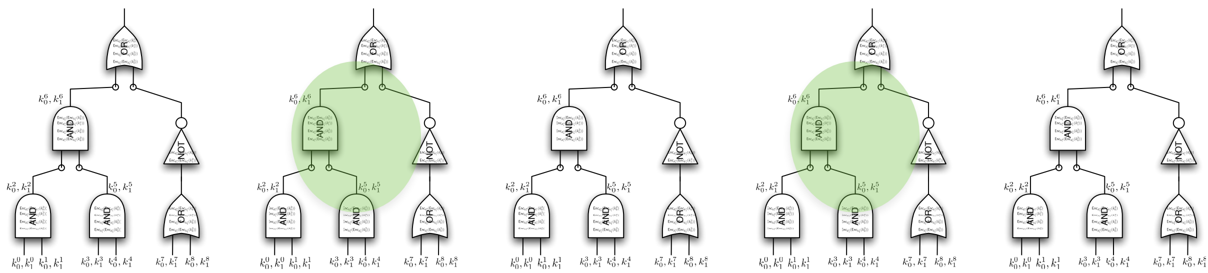
Garbled circuit



In basic protocol, garblers' input wires sent in this step.

OT

Send k fresh garbled circuits



Can't send garblers' inputs in this step anymore!

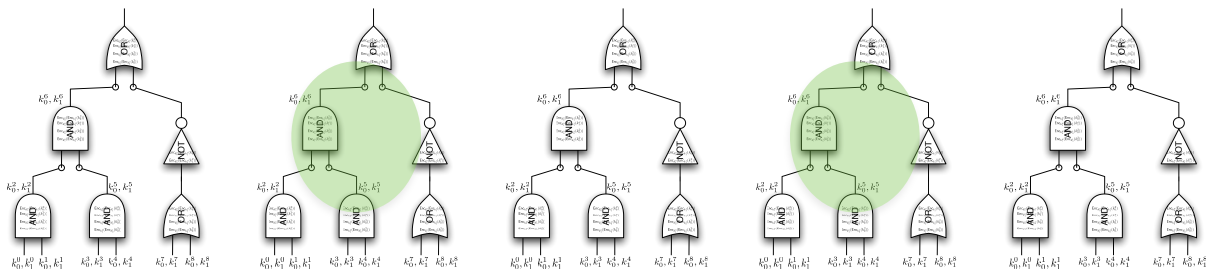
challenge set of t circuits

challenge response

majority of Eval

OT

Send k fresh garbled circuits



Can’t send garblers’ inputs in this step anymore!

challenge set of t circuits

challenge response G keys

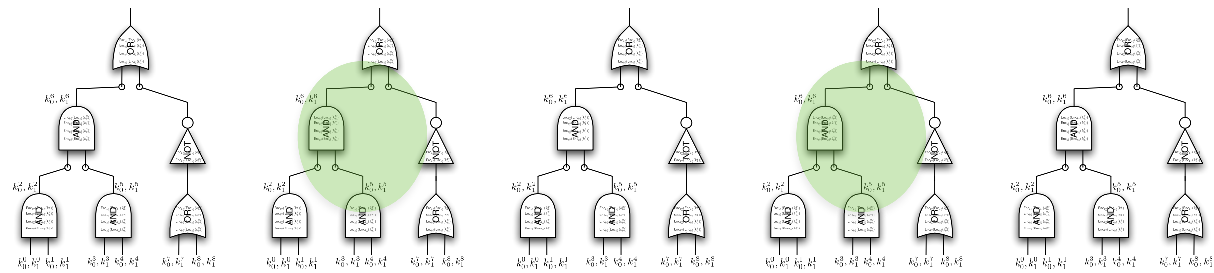
Send here instead.

majority of Eval

Problem: Input Consistency

OT

Send k fresh garbled circuits



l=k-t circuits
Needs all input keys.

challenge set of t circuits

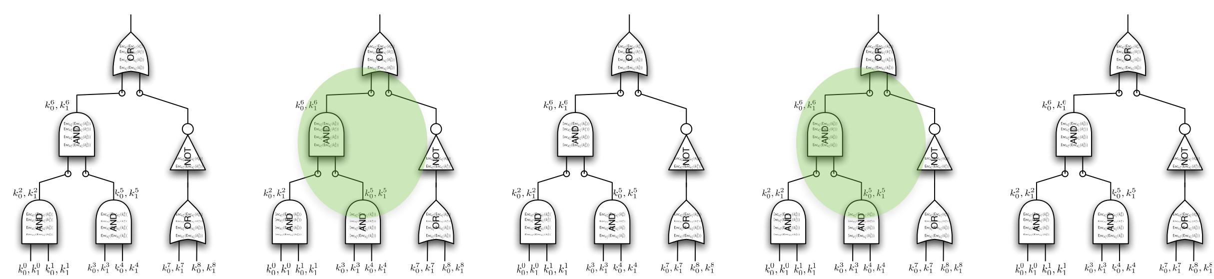
challenge response $K^1_{in}, K^2_{in}, \ldots, K^l_{in}$

majority of Eval

Problem: Input Consistency

OT

Send k fresh garbled circuits



l=k-t circuits
Needs all input keys.

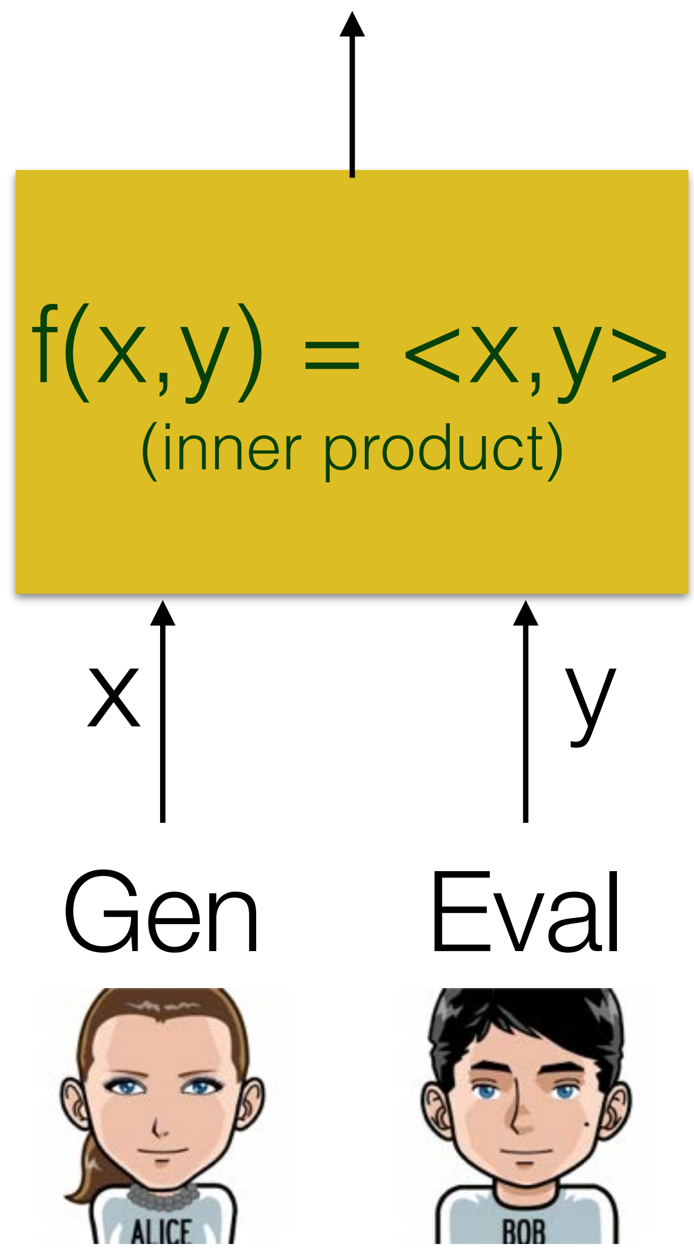
challenge set of t circuits

challenge response $K^1_{in}, K^2_{in},\dots,K^l_{in}$

majority of Eval

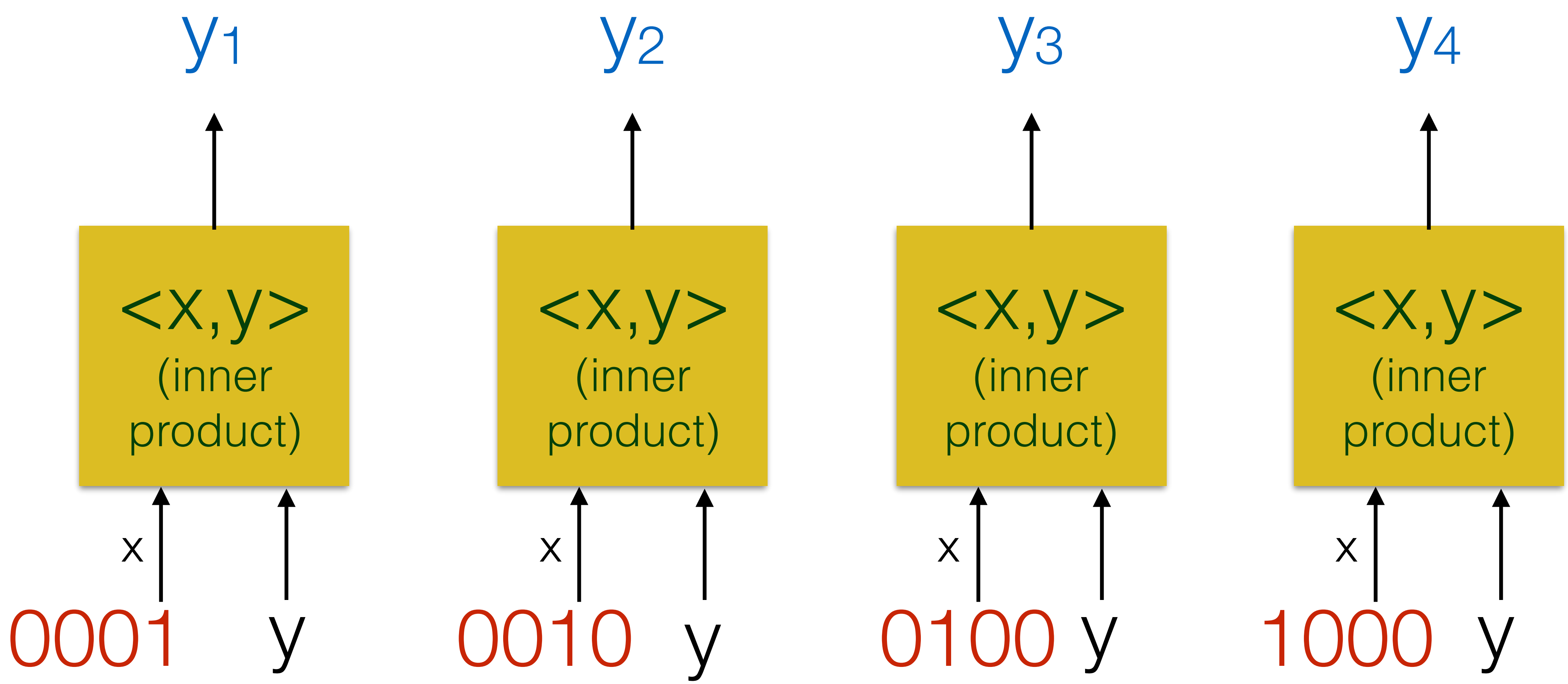
What if keys do not correspond to same input?

Input Consistency Attack



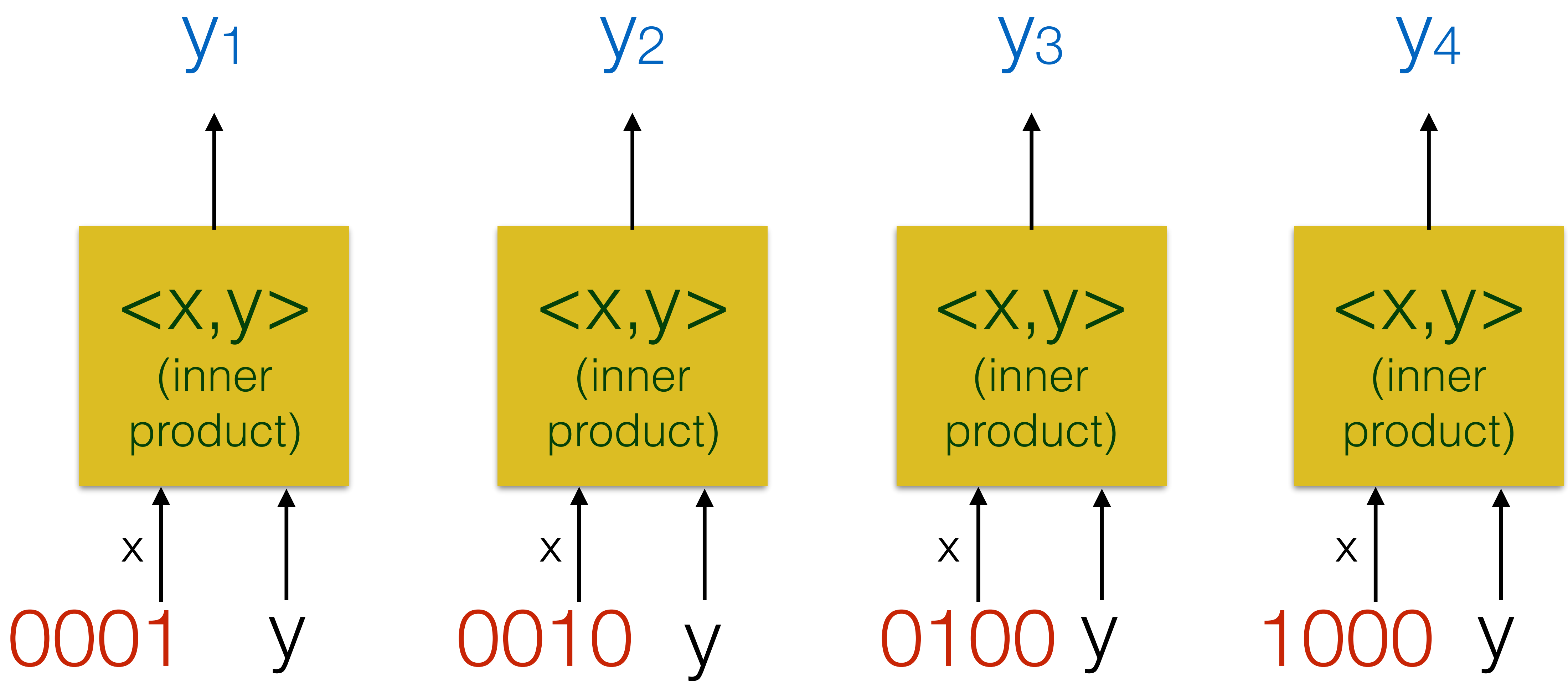
[Mohassel-Franklin06, Kiraz-Schoenmakers06, Lindell-Pinkas07]

Input Consistency Attack



[Mohassel-Franklin06, Kiraz-Schoenmakers06, Lindell-Pinkas07]

Input Consistency Attack



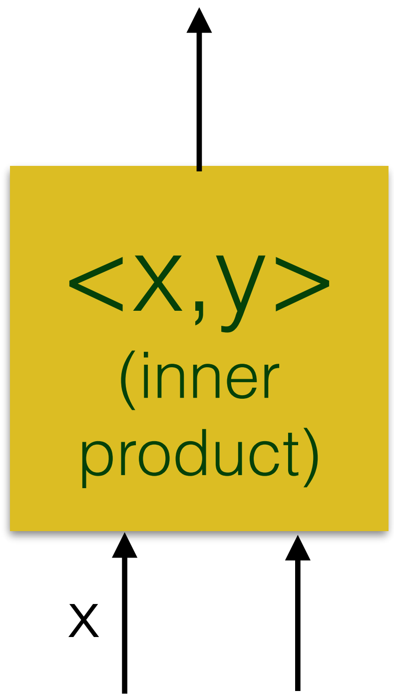
Majority(y_1,y_2,y_3,y_4)

[Mohassel-Franklin06, Kiraz-Schoenmakers06, Lindell-Pinkas07]

Test

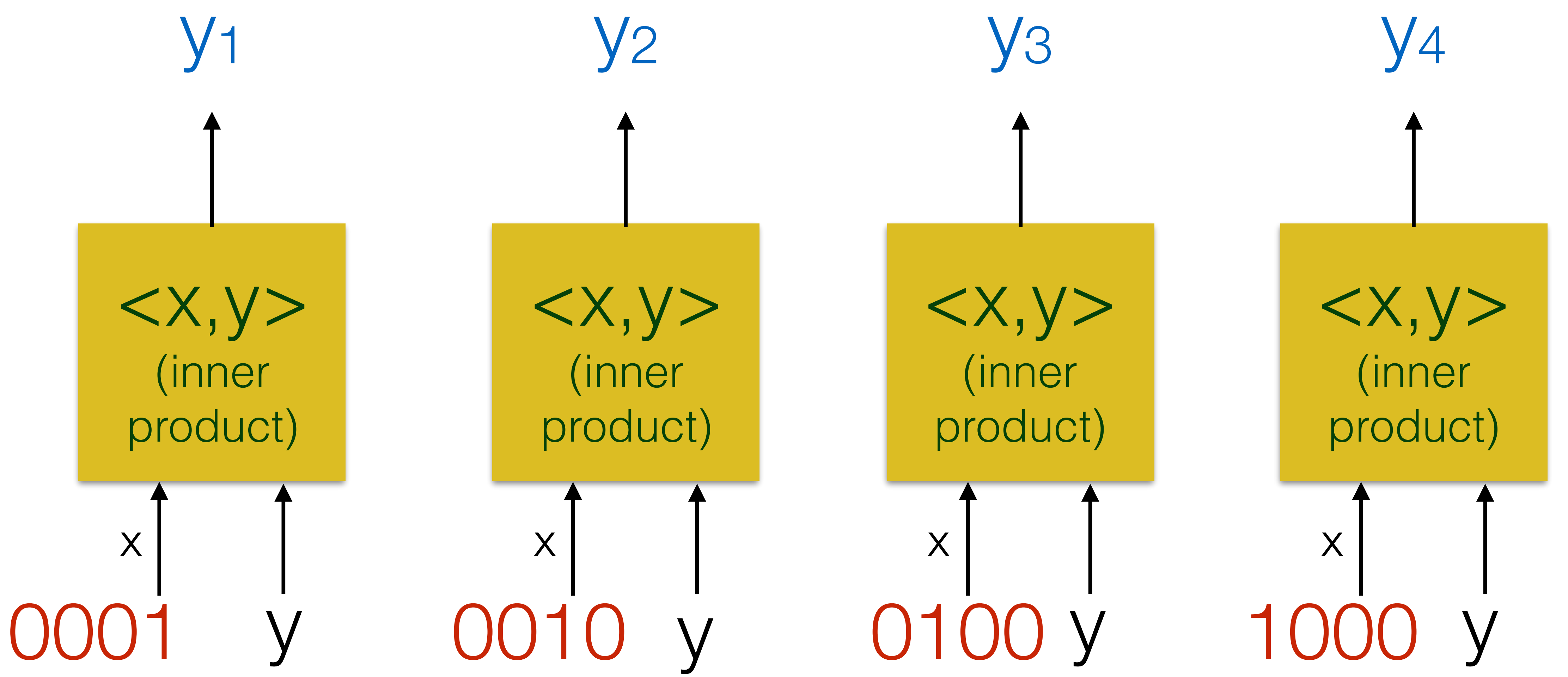
$$\forall A \; \exists S \; \forall (x_1, x_2), z$$

$$\mathbf{IDEAL}_{f,S(z),I}(x_1, x_2, k) \approx_c \mathbf{REAL}_{f,A(z),I}(x_1, x_2, k)$$



Majority(y_1,y_2,y_3,y_4)

Input Consistency Attack



Majority(y_1,y_2,y_3,y_4)

Bad!

[Mohassel-Franklin06, Kiraz-Schoenmakers06, Lindell-Pinkas07]

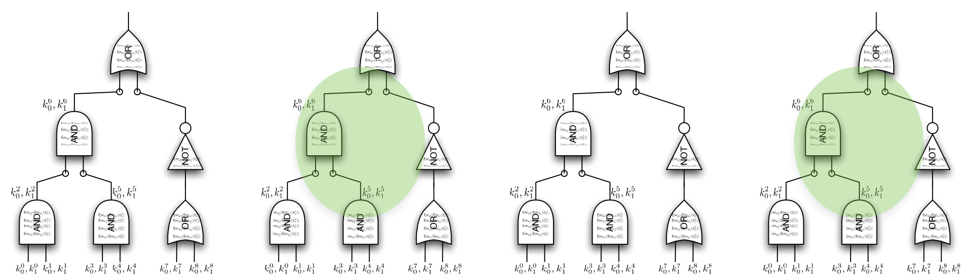
How to handle inconsistent inputs?

$K^1{}_{in}, K^2{}_{in}, \ldots, K^l{}_{in}$

Prove consistency

OT

Send k fresh garbled circuits



π_1

challenge set of t circuits

challenge resp $K^1_{in}, \ldots, K^l_{in}$ π_2

majority of Eval

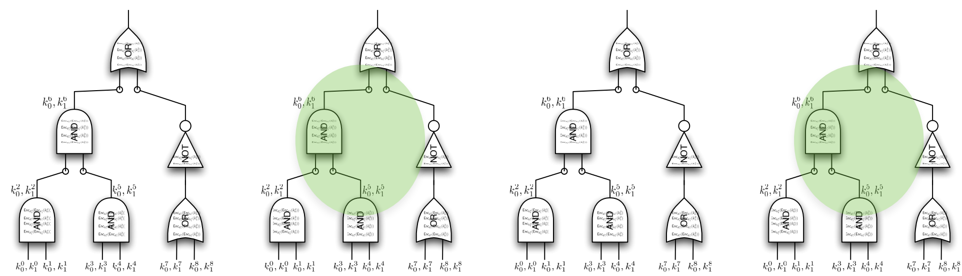
	Input Consistency		2-Outputs		OT +
	sym	pke	sym	pke	
LP07 “blackbox”	$\Theta(k^2 n)$		$\Theta(k^2 n)$		OWF
Kiraz08	$\Theta(k^2 n)$		$\Theta(kn)$	$\Theta(k)$	DLOG
LP11	$\Theta(kn)$	$\Theta(kn)$			DLOG
SS11	$\Theta(kn)$	$\Theta(kn)$	$\Theta(kn)$	$\Theta(kn)$	DLOG
KSS12	“	“	“	$\Theta(k)$	DLOG
SS13	$\Theta(kn)$		$\Theta(kn)$		OWF

Problem: Malicious OT

Use Malic-secure OT here. Is that enough?

OT

Send k fresh garbled circuits

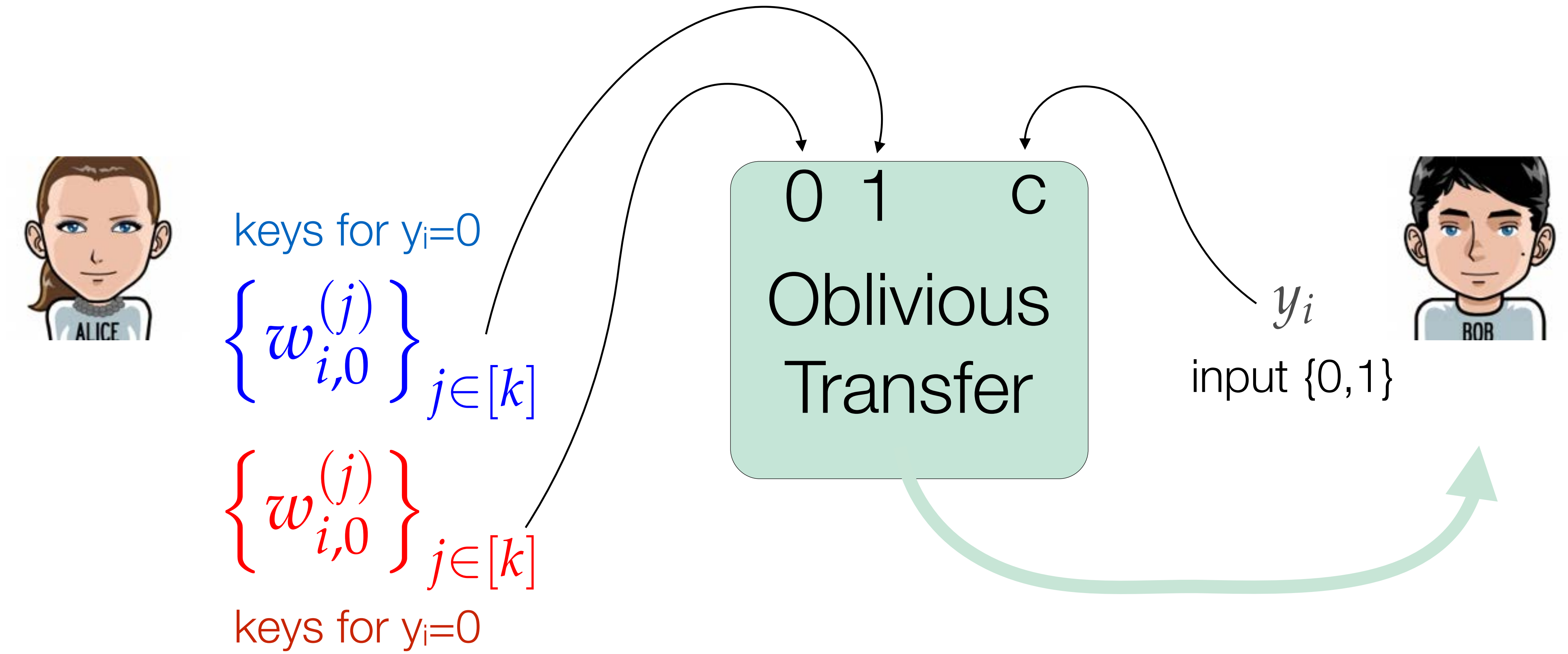


π_1

challenge set of t circuits

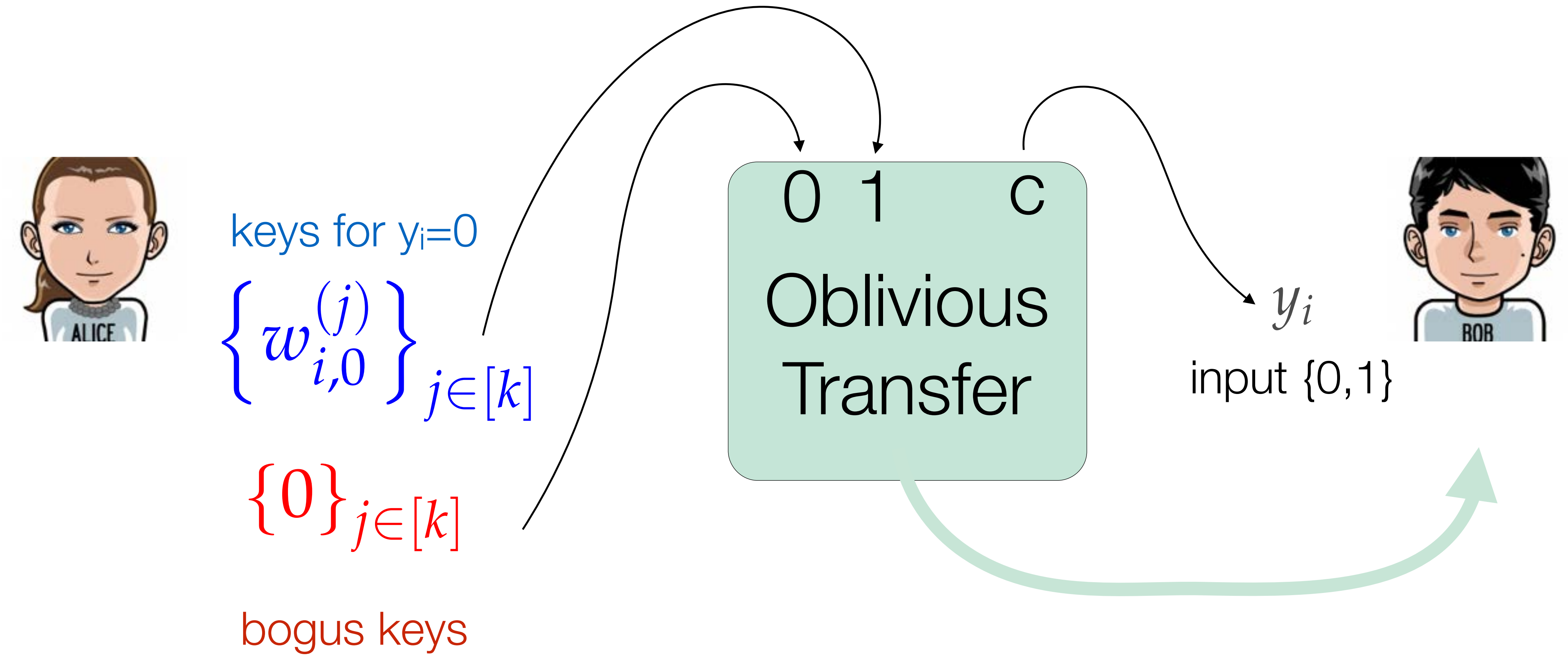
challenge resp $K^1_{in},\ldots,K^l_{in}$ π_2

majority of Eval



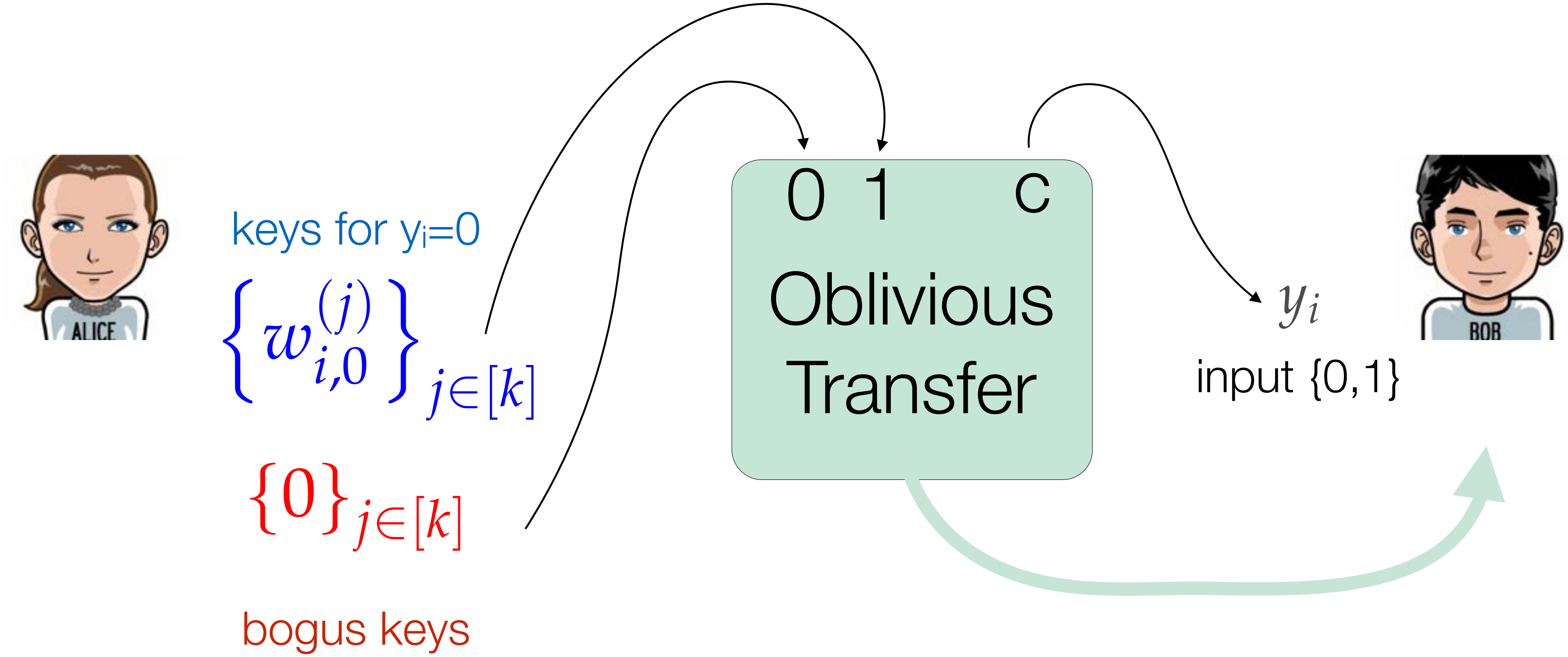
Input OT

“Key management”



What are the possible outcomes?

Selective Failure attack



What are the possible outcomes?

Input y_i = 0 OK

Input y_i = 1 FAIL: Cannot Eval

Selective Failure attack

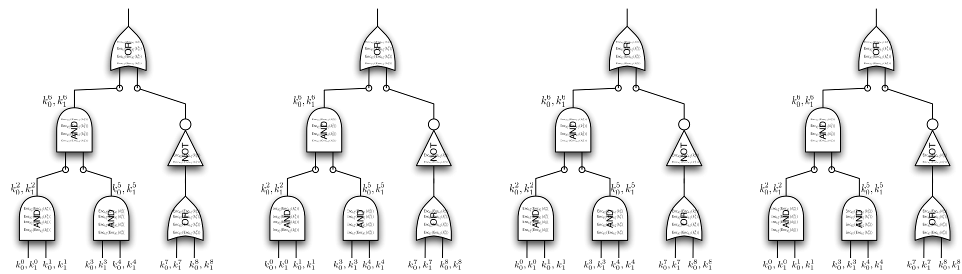
Selective Failure Solutions

Encode inputs

Prove consistency

OT π'_1

Send k fresh garbled circuits

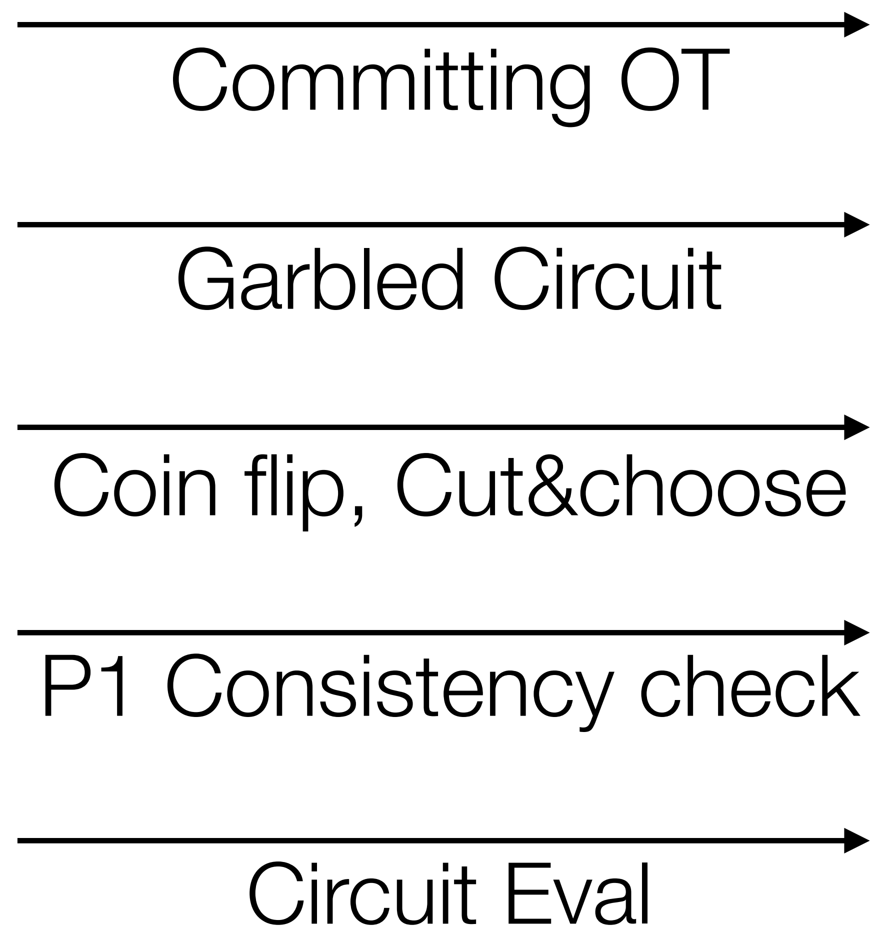


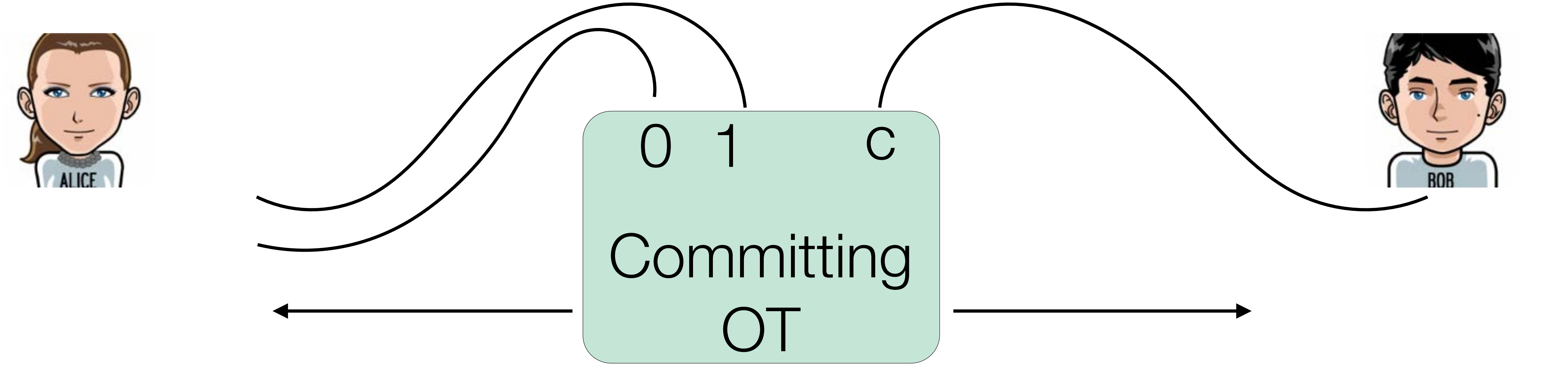
π_1 π'_2

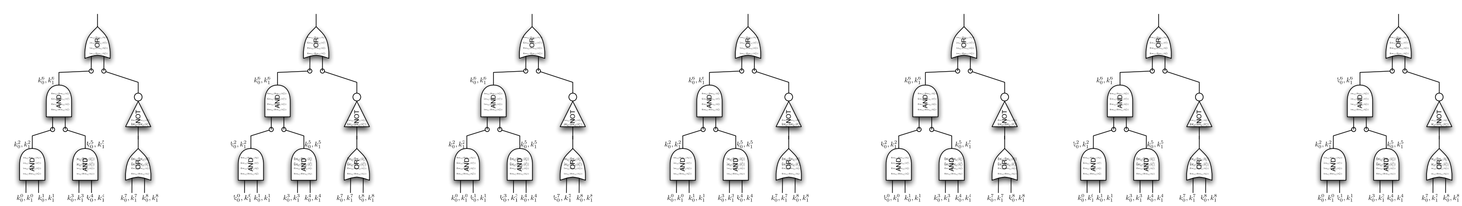
challenge set of t circuits

challenge resp $K^1{}_{in},\ldots,K^l{}_{in}$ π_2

majority of Eval


Committing OT
Garbled Circuit
Coin flip, Cut&choose
P1 Consistency check
Circuit Eval





Com(Alice's inputs)

Coin Flipping

Open Circuits, send

Eval

Key problems for Malicious Security

Circuit Consistency

Input Consistency

Selective Failure

Output Authentication
(2-output case)

Circuit Consistency

Given that evaluator checks t,
Pr that garbler succeeds in passing test:

$$\frac{\binom{k-c}{t}}{\binom{k}{t}}$$

k=10. Suppose evaluator checks 1.

Garbler can choose how many to corrupt.

0	1	2	3	4	5	6	7	8	9	10

k=10. Suppose evaluator checks 1.

Garbler can choose how many to corrupt.

0	1	2	3	4	5	6	7	8	9	10
0	0	0	0	0	1/2	2/5	3/10	1/5	1/10	0

k=10. Suppose evaluator checks 2.

Garbler can choose how many to corrupt.

0	1	2	3	4	5	6	7	8	9	10
0	0	0	0	1/3						

k=10. Suppose evaluator checks 2.

Garbler can choose how many to corrupt.

0	1	2	3	4	5	6	7	8	9	10
0	0	0	0	1/3	2/9	2/15	1/15	1/45	0	0

k=10

eval checks

of circuits garbler corrupts

	1	2	3	4	5	6	7	8	9
1	0	0	0	0	1/2	2/5	3/10	1/5	1/10
2	0	0	0	1/3	2/9	2/15	1/15	1/45	0
3	0	0	0	1/6	1/12	1/30	1/120	0	0
4	0	0	1/6	1/14	1/42	1/210	0	0	0
5	0	0	1/12	1/42	1/252	0	0	0	0
6	0	2/15	1/30	1/210	0	0	0	0	0
7	0	1/15	1/120	0	0	0	0	0	0
8	1/5	1/45	0	0	0	0	0	0	0
9	1/10	0	0	0	0	0	0	0	0

Pr garbler succeeds in corrupting
a majority of evaluated circuits

If eval checks t circuits, garbler should corrupt

$$\left\lfloor \frac{(k-t)+1}{2} \right\rfloor$$

majority of evaluated should be corrupt, no more

Evaluator should thus check t* circuits to minimize

$$\min_t \left[\frac{\binom{k - \left\lfloor \frac{(k-t)+1}{2} \right\rfloor}{t}}{\binom{k}{t}} \right]$$

[SS11]

s copies of the circuit can yield

$$2^{-0.32s}$$

if t* ~ 3/5s

Optimal for single choice of t.

But Eval can randomize choice of t.

Value of game

If Garbler wins, payoffs are (1,-1)
If Garbler looses, payoffs are (-1,1)

Both parties can run probabilistic strategies.

Game is zero-sum.

min payoff that Evaluator can force =
max payoff that Garbler can achieve

We want to solve

$$\min_{e_1,\ldots e_k} \max_{x_1,\ldots,x_k} \prod e_t x_c \left(\frac{\binom{k-c}{t}}{\binom{k}{t}} \right)$$

e_i : Pr that evaluator checks i

x_j : Pr that garbler corrupts j

Linear Program

Variables x_i : Pr that garbler corrupts i circuits

$$(x_1, x_2, \dots, x_n)$$

Table for Eval checking 1 circuit:

0	1	2	3	4	5	6	7	8	9	10
0	0	0	0	0	1/2	2/5	3/10	1/5	1/10	0

$$v_1 = (x_1, x_2, \dots, x_n) \cdot (0, 0, 0, 0, 0, \frac{1}{2}, \frac{2}{5}, \frac{3}{10}, \frac{1}{5}, \frac{1}{10}, 0)$$

Expected payoff if Eval check 1 circuit.

$$
\begin{bmatrix}
0 & 0 & 0 & 0 & 1/2 & 2/5 & 3/10 & 1/5 & 1/10 \\
0 & 0 & 0 & 1/3 & 2/9 & 2/15 & 1/15 & 1/45 & 0 \\
0 & 0 & 0 & 1/6 & 1/12 & 1/30 & 1/120 & 0 & 0 \\
0 & 0 & 1/6 & 1/14 & 1/42 & 1/210 & 0 & 0 & 0 \\
0 & 0 & 1/12 & 1/42 & 1/252 & 0 & 0 & 0 & 0 \\
0 & 2/15 & 1/30 & 1/210 & 0 & 0 & 0 & 0 & 0 \\
0 & 1/15 & 1/120 & 0 & 0 & 0 & 0 & 0 & 0 \\
1/5 & 1/45 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1/10 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}
\begin{bmatrix}
x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \\ x_9
\end{bmatrix}
$$

Evaluator chooses min row

To express as LP, add variable v.

maximize v

subject to

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 1/2 & 2/5 & 3/10 & 1/5 & 1/10 & -1 \\ 0 & 0 & 0 & 1/3 & 2/9 & 2/15 & 1/15 & 1/45 & 0 & -1 \\ 0 & 0 & 0 & 1/6 & 1/12 & 1/30 & 1/120 & 0 & 0 & -1 \\ 0 & 0 & 1/6 & 1/14 & 1/42 & 1/210 & 0 & 0 & 0 & -1 \\ 0 & 0 & 1/12 & 1/42 & 1/252 & 0 & 0 & 0 & 0 & -1 \\ 0 & 2/15 & 1/30 & 1/210 & 0 & 0 & 0 & 0 & 0 & -1 \\ 0 & 1/15 & 1/120 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \\ 1/5 & 1/45 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \\ 1/10 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \\ x_9 \\ v \end{bmatrix} \leq \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$$0 \leq x_i \leq 1$$

$$\sum x_i = 1$$

```
H-representation
begin
20 11 rational
0 0 0 0 0 1/2 2/5 3/10 1/5 1/10 -1
0 0 0 0 1/3 2/9 2/15 1/15 1/45 0 -1
0 0 0 0 1/6 1/12 1/30 1/120 0 0 -1
0 0 0 1/6 1/14 1/42 1/210 0 0 0 -1
0 0 0 1/12 1/42 1/252 0 0 0 0 -1
0 0 2/15 1/30 1/210 0 0 0 0 0 -1
0 0 1/15 1/120 0 0 0 0 0 0 -1
0 1/5 1/45 0 0 0 0 0 0 0 -1
0 1/10 0 0 0 0 0 0 0 0 -1
1 -1 -1 -1 -1 -1 -1 -1 -1 -1 0
-1 1 1 1 1 1 1 1 1 1 0
0 1 0 0 0 0 0 0 0 0 0
0 0 1 0 0 0 0 0 0 0 0
0 0 0 1 0 0 0 0 0 0 0
0 0 0 0 1 0 0 0 0 0 0
0 0 0 0 0 1 0 0 0 0 0
0 0 0 0 0 0 1 0 0 0 0
0 0 0 0 0 0 0 1 0 0 0
0 0 0 0 0 0 0 0 1 0 0
0 0 0 0 0 0 0 0 0 1 0
end
maximize
0 0 0 0 0 0 0 0 0 0 1
```

```
* cdd+: Double Description Method in C++:Version 0.77(August 19, 2003)
* Copyright (C) 1999, Komei Fukuda, fukuda@ifor.math.ethz.ch
* Compiled for Rational Exact Arithmetic with GMP
*cdd LP Result
*cdd input file : 10.ine  (20 x 11)
*LP solver: Dual Simplex
*LP status: a dual pair (x, y) of optimal solutions found.
*maximization is chosen.
*Objective function is
 0 + 0 X[1] + 0 X[2] + 0 X[3] + 0 X[4] +
 0 X[5] + 0 X[6] + 0 X[7] + 0 X[8] + 0 X[9] +
 1 X[10]
*LP status: a dual pair (x, y) of optimal solutions found.
begin
  primal_solution
  1 :  60/247
  2 :  575/1729
  3 :  440/1729
  4 :  30/247
  5 :  12/247
  6 :  0
  7 :  0
  8 :  0
  9 :  0
  10 :  6/247
  dual_solution
  19 :  23/1235
  20 :  53/2470
  17 :  23/2470
  18 :  147/9880
  1 :  7/247
  3 :  27/247
  5 :  63/247
  7 :  90/247
  9 :  60/247
  10 :  6/247
  optimal_value : 6/247
end
*number of pivot operations = 5
*Computation starts     at Sun Feb 15 06:50:05 2015
*           terminates at Sun Feb 15 06:50:05 2015
*Total processor time = 0 seconds
*                     = 0h 0m 0s
```

$6/247 \sim .02429$

$2^{-5.3}$

		60/247	575/1729	440/1729	30/247	12/247				
		1	2	3	4	5	6	7	8	9
7/247	1	0	0	0	0	1/2	2/5	3/10	1/5	1/10
	2	0	0	0	1/3	2/9	2/15	1/15	1/45	0
27/247	3	0	0	0	1/6	1/12	1/30	1/120	0	0
	4	0	0	1/6	1/14	1/42	1/210	0	0	0
63/247	5	0	0	1/12	1/42	1/252	0	0	0	0
	6	0	2/15	1/30	1/210	0	0	0	0	0
90/247	7	0	1/15	1/120	0	0	0	0	0	0
	8	1/5	1/45	0	0	0	0	0	0	0
60/247	9	1/10	0	0	0	0	0	0	0	0

Solution for k=10

$k=41$

```
primal_solution
  1 :  10645508192981161500/20055554759628164776001
```

k=117 `.00000000000034624553` $2^{-41.3}$

versus

k=125 in SS11