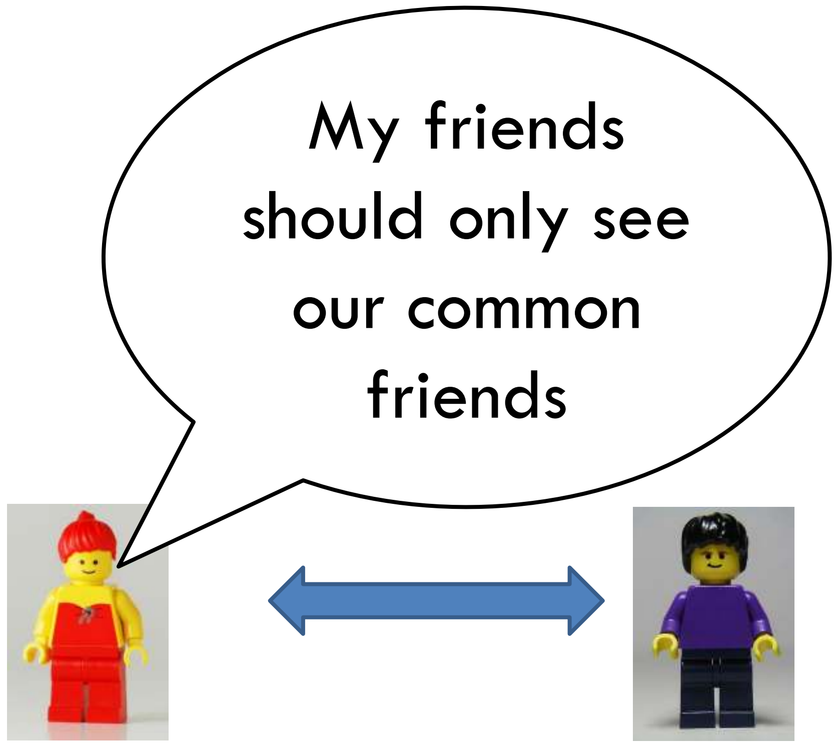
AARHUS UNIVERSITY

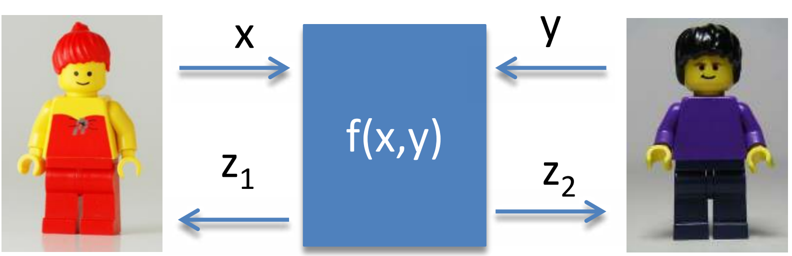
Hiding the Input Size in Secure Two-Party Computation

Yehuda Lindell, Kobbi Nissim, Claudio Orlandi

Privacy on *(or a more privacy sensitive social network)*



Secure Computation



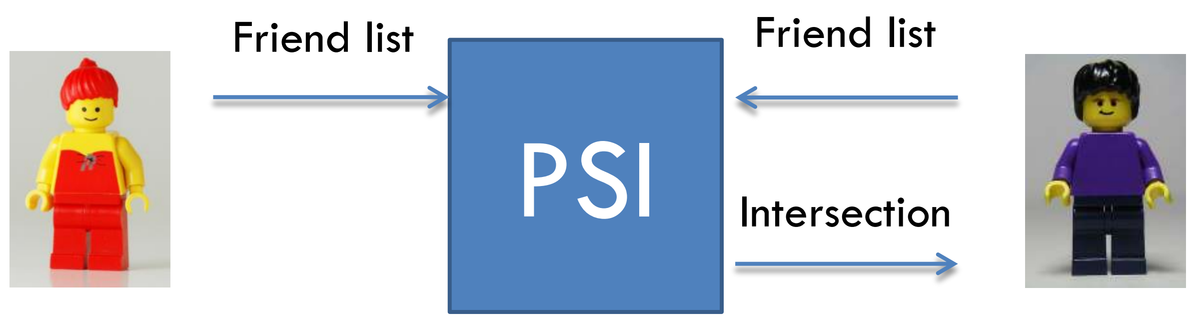
Trusted Party

Cryptographic Protocol

- Privacy
- Correctness
- Input Independence
- “The protocol is as secure as the ideal world”

Or is it?

Privacy on (*or a more privacy sensitive social network*)



Privacy on

(or a more privacy sensitive social network)

Friend list

Friend list

Yao's protocol

Intersection **+ size of friend list!**

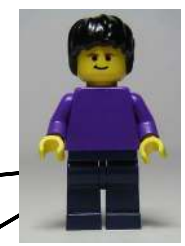
Privacy on

(or a more privacy sensitive social network)

You learned more than you were supposed to!

Fr

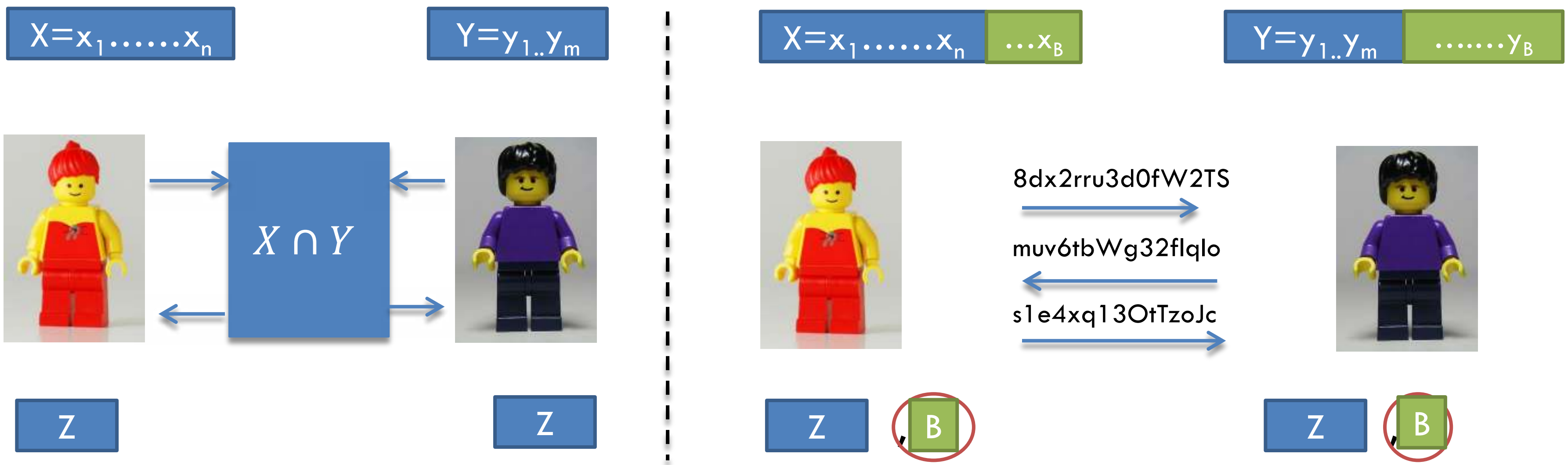
Friend list



Don't worry, it's only metadata!

size of friend list!

Padding?



- Just add a lot of “fake entries” to your DB
- Requires an upper bound ☹
- **Inherent** inefficiency ☹

Impossibility of Size-Hiding: Proof by Authority

[G04] *"…making no restriction on the relationship among the lengths of the two inputs disallows the existence of secure protocols for computing any nondegenerate functionality…"*

[IP07] *"…hiding the size of* both *inputs is impossible for interesting functions…"*

[HL10] *"…We remark that some restriction on the input lengths is unavoidable because, as in the case of encryption, to some extent such information is always leaked…"*

Impossibility of Size-Hiding: Proof by Authority

[G04] *"…making no restriction on the relationship among the lengths of the two inputs disallows the existence of secure protocols for computing* ***any nondegenerate functionality****…"*

[IP07] *"…hiding the size of* both *inputs is impossible for* ***interesting functions****…"*

[HL10] *"…We remark that some restriction on the input lengths is unavoidable because, as in the case of encryption, to some extent such information is* ***always leaked****…"*

Impossibility

- Is it impossible for
  - Any nondegenerate functionality?
    - What is nondegenerate?
    - What does no restriction mean?
  - All interesting functions?
    - What is interesting?
    - What about hiding one party's input?
- Is it really like encryption? Is length information always leaked?

This Work

- Part of a general research effort to revisit the foundations of secure computation
- Do we have any proof that it's impossible?
  - If yes, where and for what functions?
- Is it impossible always or sometimes?
  - If sometimes, can we characterize when?
- How do we define size hiding?

- Compare to recent work on fairness…

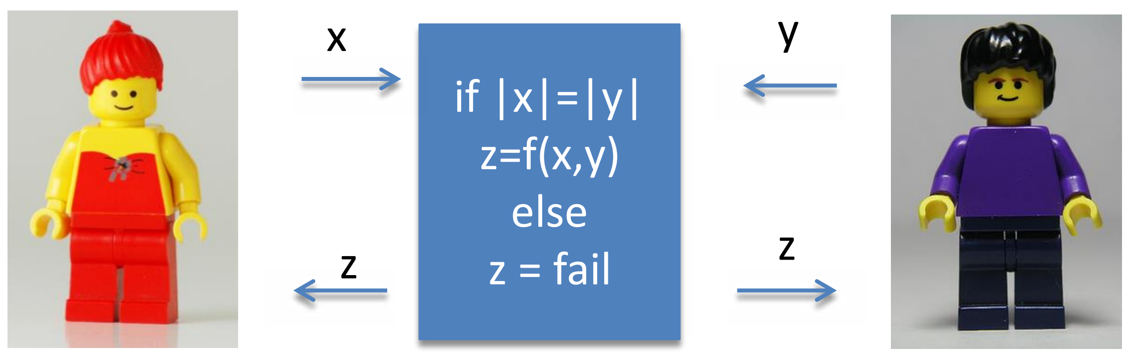
Input Size Can be Hidden Sometimes

- MicaliRabinKilian'03 (and many subsequent work…):

  Zero Knowledge Sets (check membership without revealing the size of the set)
- IshaiPaskin'07:
  - Branching programs (reveal length of the branching program but nothing else about input size)
    - Implies set intersection, server input size is hidden
- AtenieseDeCristofaroTsudik'11:
  - Specific protocol for set intersection, client input size is hidden; efficient, in random oracle model
- Note: all these are for **specific problems**/restricted class, and all hide **only one party's input**

A Test Case: Standard Definition

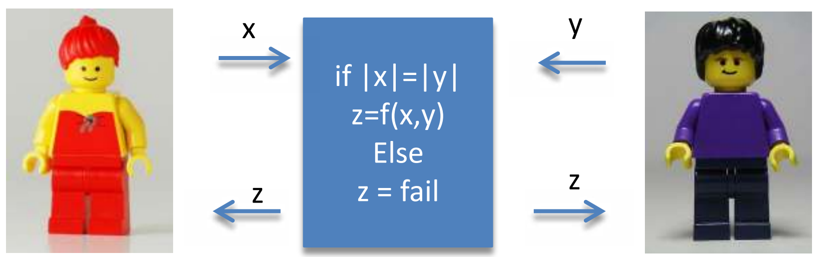
- Standard definition, e.g. [Gol04]



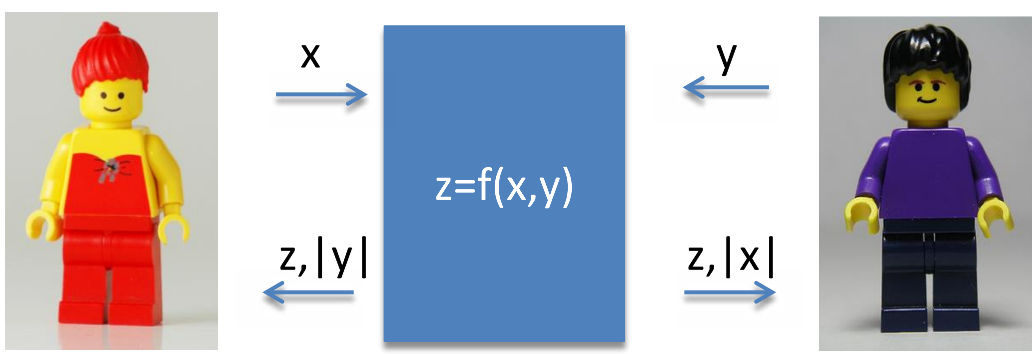
- Need to know other party's size in advance
  - Introduces problem of input **size** dependence
  - One party can choose its input after knowing the size of the other party's input (outside the scope of the protocol)

Defining Non-Input-Size Hiding

- Formulation [G04]:



- Our formulation:

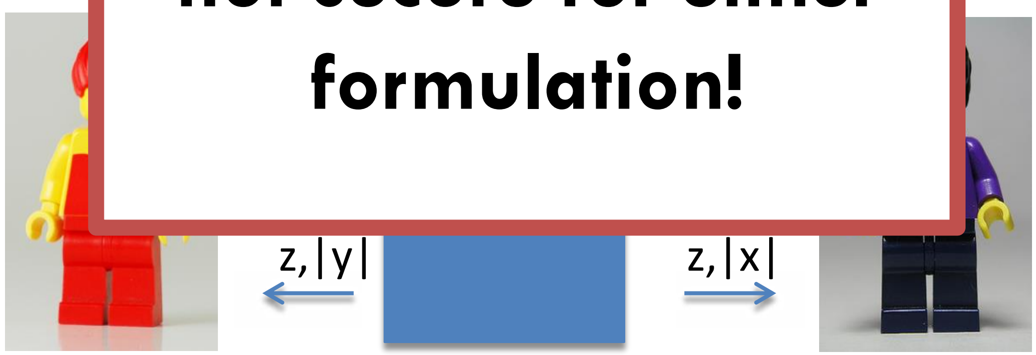


- Security guarantees incomparable

Defining Non-Input-Size Hiding

- Formulation [G04]:

- Our forr



Standard protocols are not secure for either formulation!

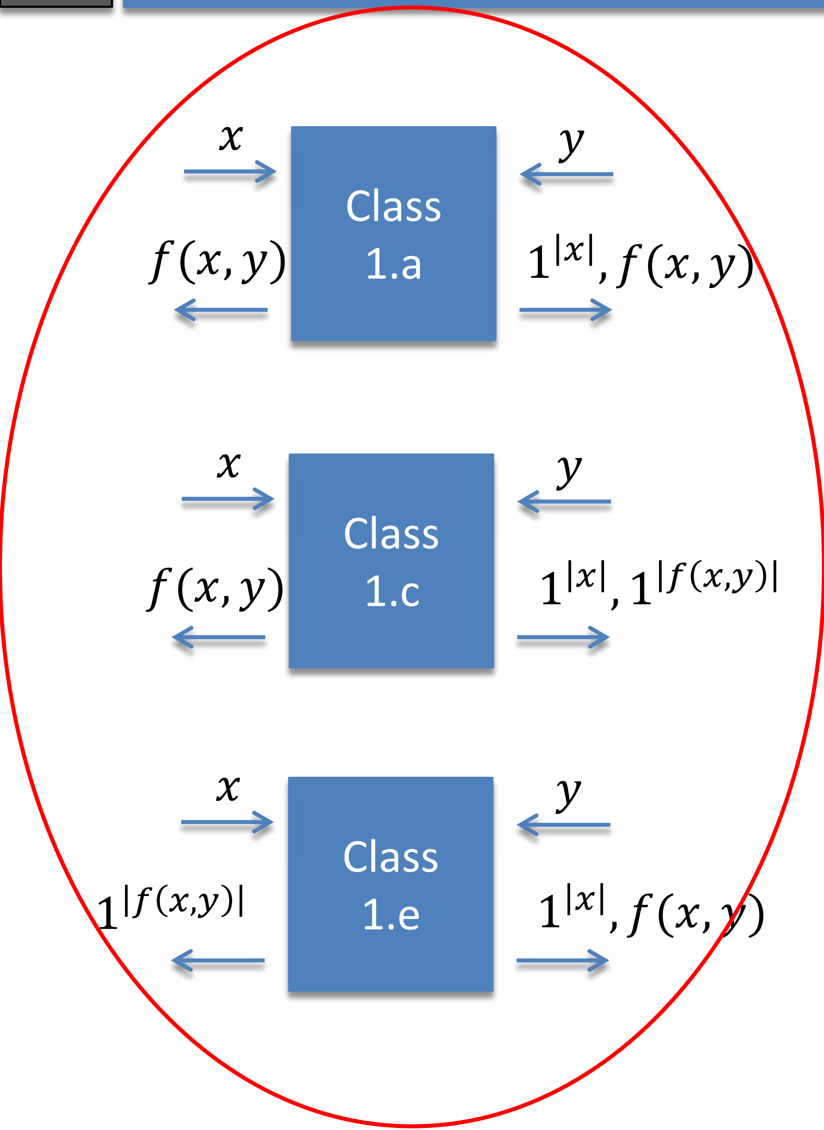
- Security guarantees incomparable

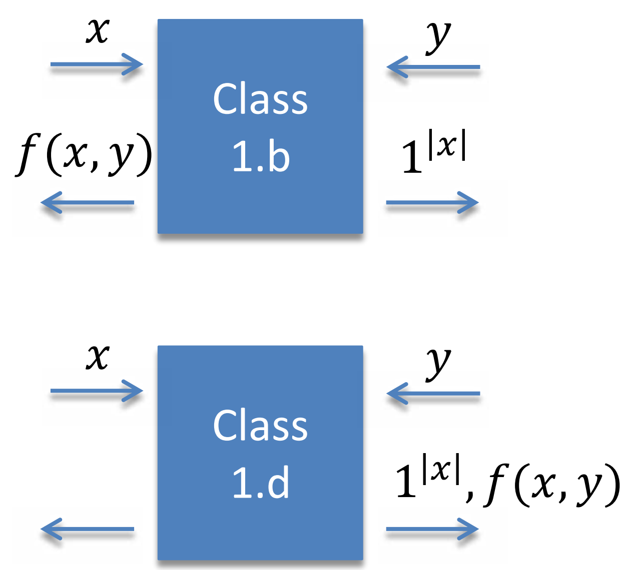
Ideal Model - Classes

- Classes
  - 0: both input-sizes are leaked
  - 1: Bob learns $|x|$, Alice does not learn $|y|$
  - 2: both input-sizes are not revealed
- Subclasses
  - Who gets output?
  - Is the output size leaked?
- Our classification is complete for symmetric functions $f(x, y) = f(y, x)$

Class 0

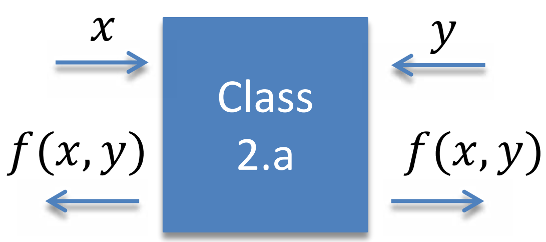
Class 1

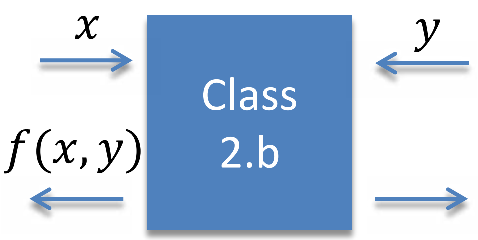


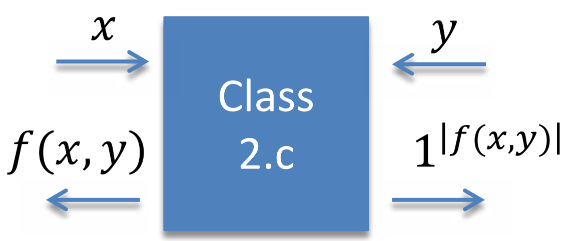


Essentially equivalent classes (outputs have same length)

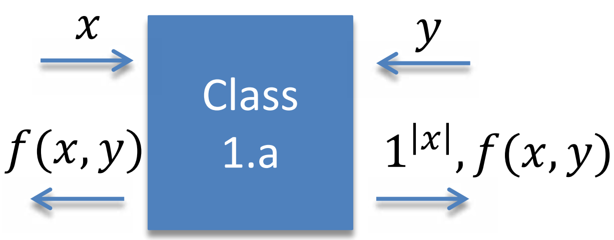
Class 2

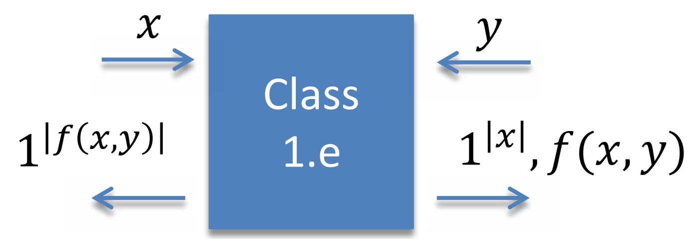


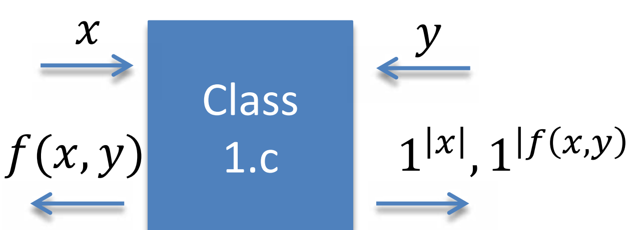




Positive Results







Tools

- Fully Homomorphic Encryption

$$(G, E, D, Eval)$$

  - Correctness:

$$D_{sk}(Eval_{pk}(f, E_{pk}(x))) = f(x)$$

  - Circuit privacy:

$$Eval_{pk}(f, E_{pk}(x)) \approx E_{pk}(f(x))$$

Class 1.a

x → Class 1.a ← y

$f(x,y)$ ← Class 1.a → $1^{|x|}, f(x,y)$

$(pk, sk) \leftarrow Gen(1^k)$

$c_x \leftarrow Enc_{pk}(x)$

$\xrightarrow{pk, c_x}$

$c_z = Eval_{pk}(f(\cdot, y), c)$

$\xleftarrow{c_z}$

$z = Dec_{sk}\,(c_z)$

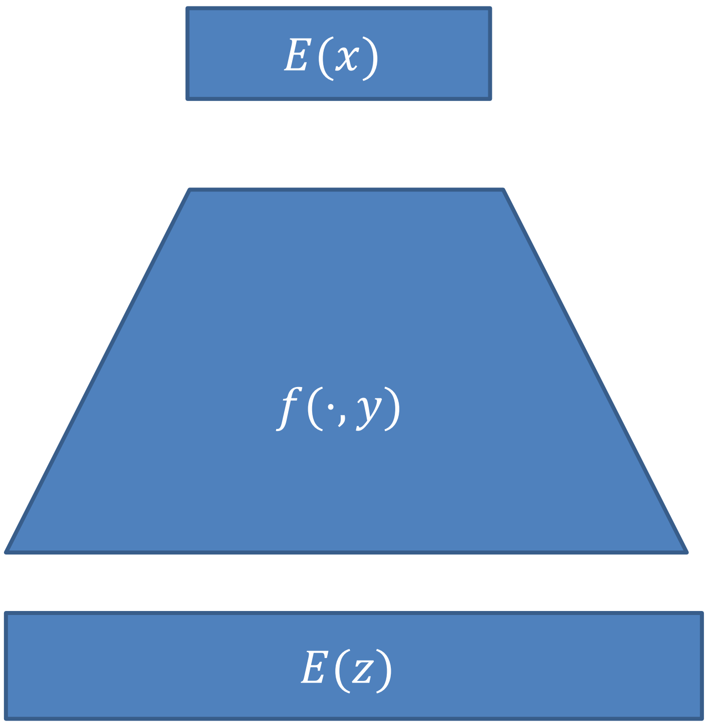
$\xrightarrow{z}$

Class 1.a

- The devil is in the details
  - In order to compute c_z, a circuit computing $f(\cdot, y)$ must be known, but this involves knowing the output length

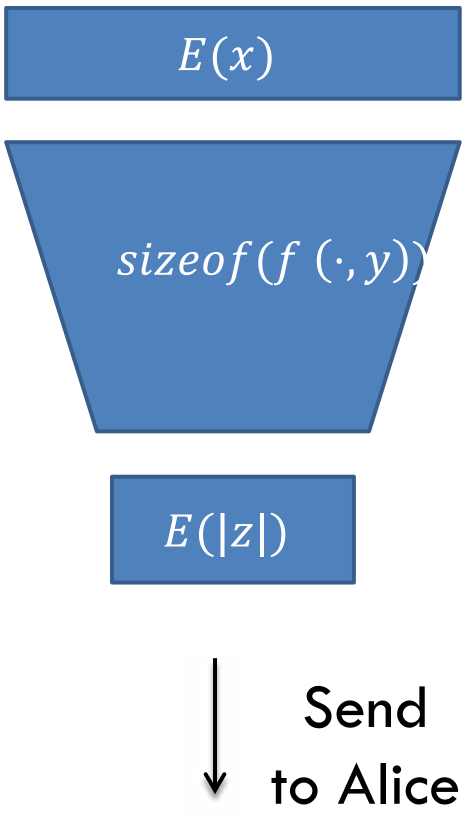
- Solution: P_2 computes an upper bound (it can do this since it knows $|x|$ and y

Computing an Upper Bound

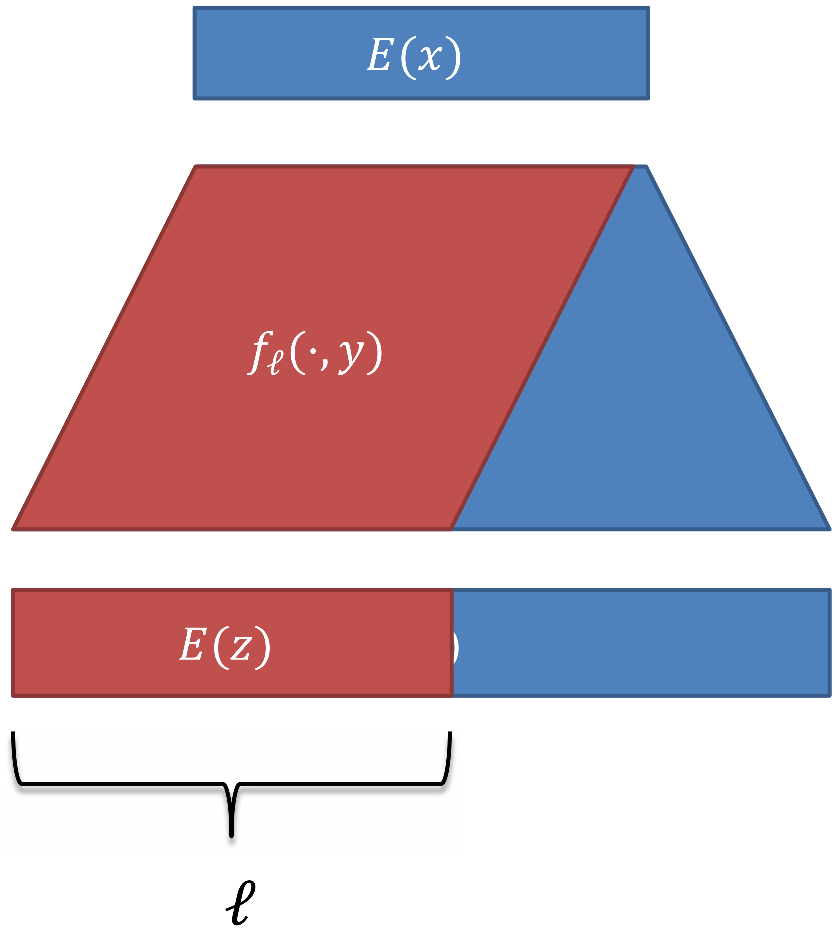


- Example: set union
  - $z = x \cup y$
- Clear that $|z| \leq |x| + |y|$
- But how long exactly?
  Any upper bound reveals information about $|y|$

The Solution



Alice opens $\ell = |z|$



Class 1.a

$x \rightarrow$ Class 1.a $\leftarrow y$

$f(x,y) \leftarrow$ Class 1.a $\rightarrow 1^{|x|}, f(x,y)$

$(pk, sk) \leftarrow Gen(1^k)$

$c_x \leftarrow Enc_{pk}(x)$

$\xrightarrow{pk, c_x}$

$c_\ell = Eval_{pk}(\boldsymbol{sizeof}(f(\cdot, y)), c)$

$\xleftarrow{c_\ell}$

$\ell = Dec_{sk}(c_\ell)$

$\xrightarrow{\ell}$

$c_z = Eval_{pk}(\boldsymbol{f_\ell}(\cdot, y), c)$

$\xleftarrow{c_z}$

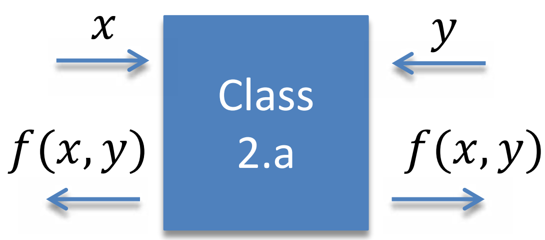
$z = Dec_{sk}(c_z)$

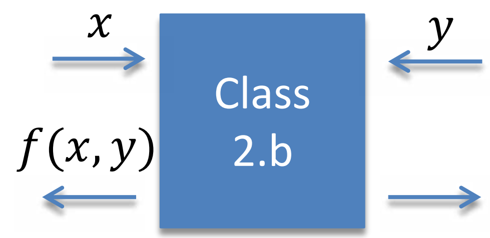
$\xrightarrow{z}$

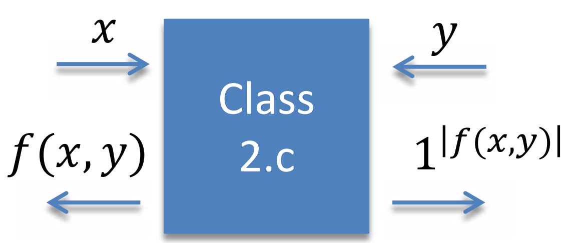
The circuit for output of length exactly ℓ

- Thm: FHE $\Rightarrow \forall f$ can be securely computed in Classes 1.a/c/e

Positive Results







Two-Size Hiding Protocols

- **Theorem:** If FHE exists, then the following functions can be securely computed in class 2 (semi-honest)
  - Greater than (Millionaire's problem)
  - And other functions:
    - Equality
    - Mean
    - Variance
    - Median

Two-Size Hiding Protocols

- **Theorem:** If FHE exists, then the following functions can be securely computed in class 2 (sem
  - Gr
  - An
    - E
    - M
    - V
    - M

First example of protocols for interesting functions where the size of the input of both parties is protected

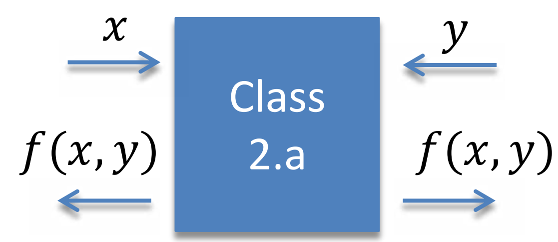
Size Independent Protocols

- π is size independent for f if
  - Correct (except for $negl(k)$)
  - Computation efficient (runtime $poly(input+k)$)
  - Communication efficient (bounded by $poly(k)$)

- Construction idea: "compile" these insecure protocols using FHE.
- (Concrete protocol for "greater than" in the paper)

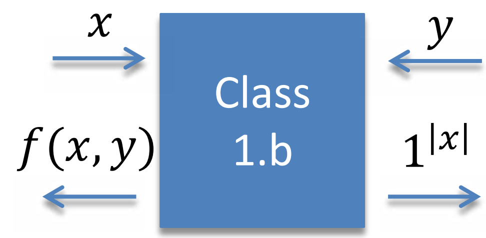
Negative Results

Lower Bounds

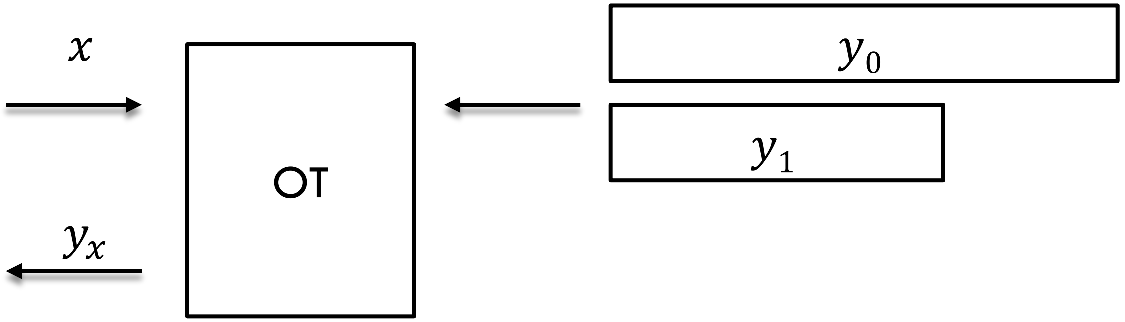


- **Theorem:** There exist functions that cannot be computed while hiding both parties' input size
  - Not everything can be computed in Class 2
- Examples: Inner product, Set Intersection, Hamming distance, etc.
  - Any protocol with "high" communication complexity

Class 1.b



- **Theorem:** There exist functions that cannot be securely computed in class 1.b
- **Proof:** size-hiding OT
  - x = selection bit
  - $y = (y_0, y_1)$ two strings of different length
  - $f(x,y) = y_x$



Conclusions and Open Problems

Conclusions and Open Problems

- Open Problems
  - (More) efficient protocols for specific tasks?
  - Malicious security?
  - Dealing with side-channel attacks (timing)?

- Hiding the input size is (sometimes) possible.
  - Don't give up!
- Landscape of size-hiding 2PC is very rich
  - Many positive and negative results.

Summary of Feasibility

	All f (bounded output)	**All** f (even unbounded output)	GT ($x > y$)	vecxor	Intersection	OT	omprf
2.a	×	×	✓	✓	×	✓	✓
2.b	×	×	✓	×	×	×	✓
2.c	×	×	✓	✓	×	✓	✓
1.a	✓	✓	✓	✓	✓	✓	✓
1.b	✓	×	✓	✓	✓	×	✓
1.c	✓	✓	✓	✓	✓	✓	✓
1.d	✓	×	✓	✓	✓	✓	×
1.e	✓	✓	✓	✓	✓	✓	✓