Prio: Private, Robust, and Scalable Computation of Aggregate Statistics

Henry Corrigan-Gibbs and Dan Boneh

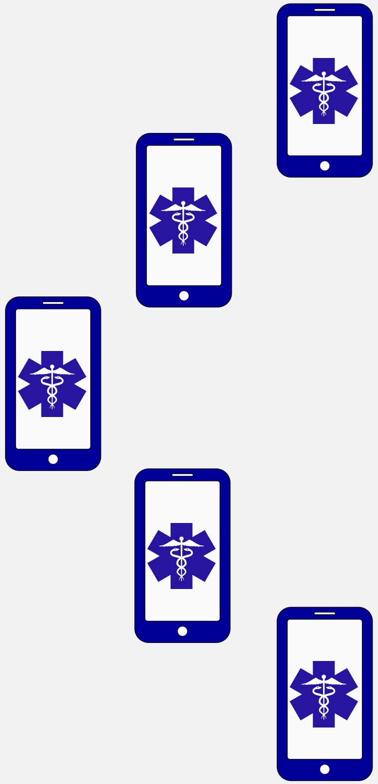
Stanford University

Presented by Elijah Grubb

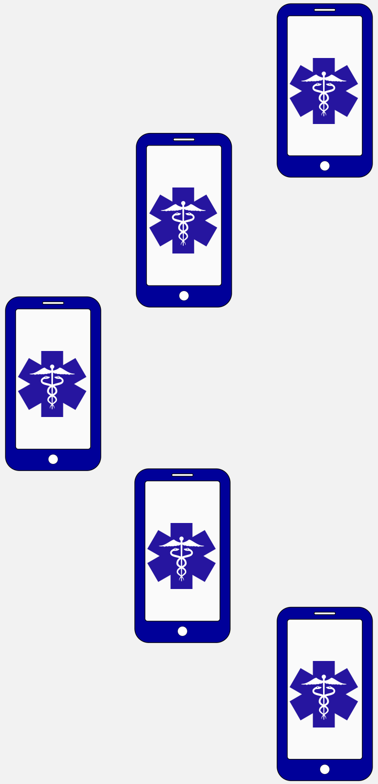
Problem: Privacy Preserving Statistics of Many Users

Problem: Privacy Preserving Statistics of Many Users

Problem: Privacy Preserving Statistics of Many Users



Problem: Privacy Preserving Statistics of Many Users



Problem: Privacy Preserving Statistics of Many Users

Problem: Privacy Preserving Statistics of Many Users

$x_i^{disease} \in \{0,1\}$

$\sum_j x_j$

System Goals

- Anonymity: Adversarial data collectors cannot tell which data value belongs to which client.
- Privacy: An adversary, who controls any number of clients and all but one server, learns nothing about an honest clients input outside of the aggregate function over the data $f(x_1, \dots, x_n)$.
- Robustness: A malicious client can only affect the result by misreporting their private data values within the function's input bounds.
- Efficiency

Related Work

Related Work

- Relax Correctness

Related Work

- Relax Correctness
  - Randomized Response (RAPPOR)

Related Work

- Relax Correctness
  - Randomized Response (RAPPOR)
- Relax Privacy

Related Work

- Relax Correctness
  - Randomized Response (RAPPOR)
- Relax Privacy
  - Semi-honest secure systems (SplitX)

Related Work

- Relax Correctness
  - Randomized Response (RAPPOR)
- Relax Privacy
  - Semi-honest secure systems (SplitX)
- Relax Robustness

Related Work

- Relax Correctness
  - Randomized Response (RAPPOR)
- Relax Privacy
  - Semi-honest secure systems (SplitX)
- Relax Robustness
  - Additively Homomorphic Encryption (PrivEx, VPriv)

Related Work

- Relax Correctness
  - Randomized Response (RAPPOR)
- Relax Privacy
  - Semi-honest secure systems (SplitX)
- Relax Robustness
  - Additively Homomorphic Encryption (PrivEx, VPriv)
- Relax Efficiency

Related Work

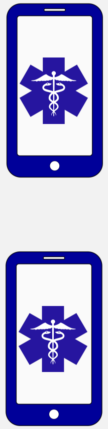
- Relax Correctness
  - Randomized Response (RAPPOR)
- Relax Privacy
  - Semi-honest secure systems (SplitX)
- Relax Robustness
  - Additively Homomorphic Encryption (PrivEx, VPriv)
- Relax Efficiency
  - General MPC
  - SNARKs (Pinocchio)

Example of a Simple Scheme

$p = 17$

Example of a Simple Scheme

$p = 17$



Example of a Simple Scheme

$p = 17$

1 

Example of a Simple Scheme

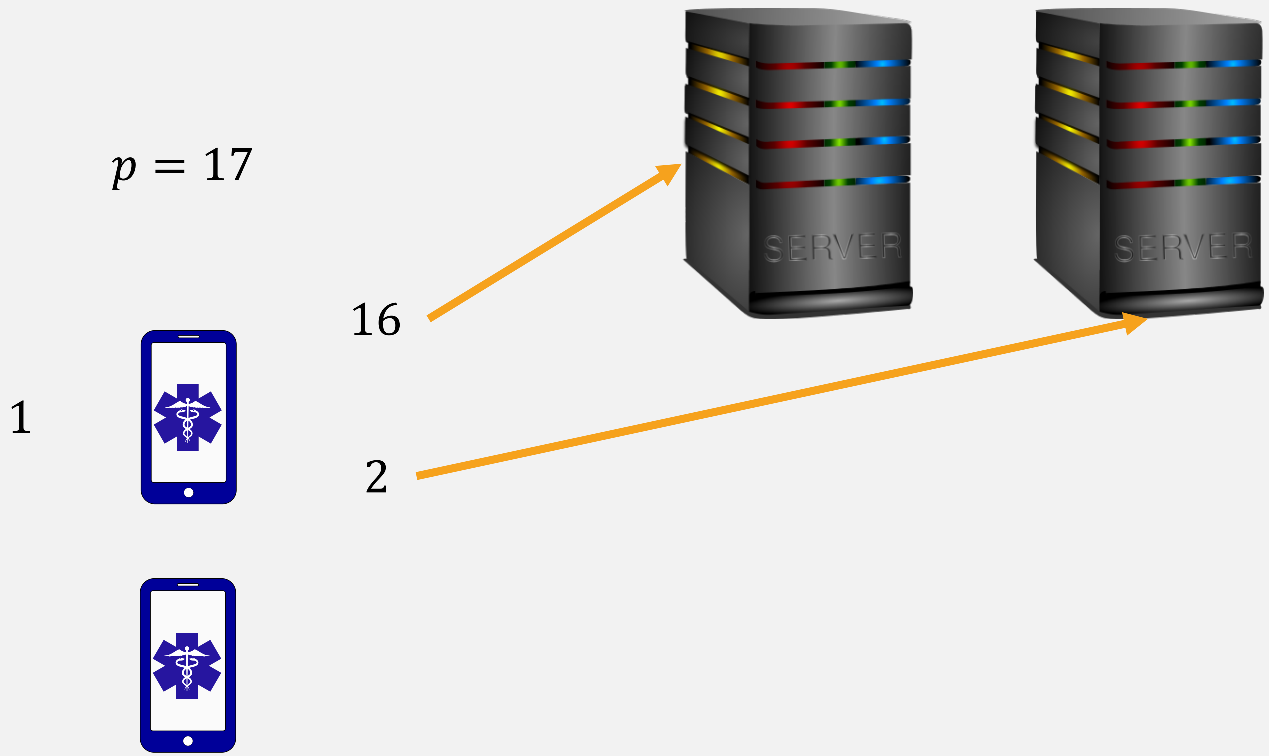
$p = 17$

16

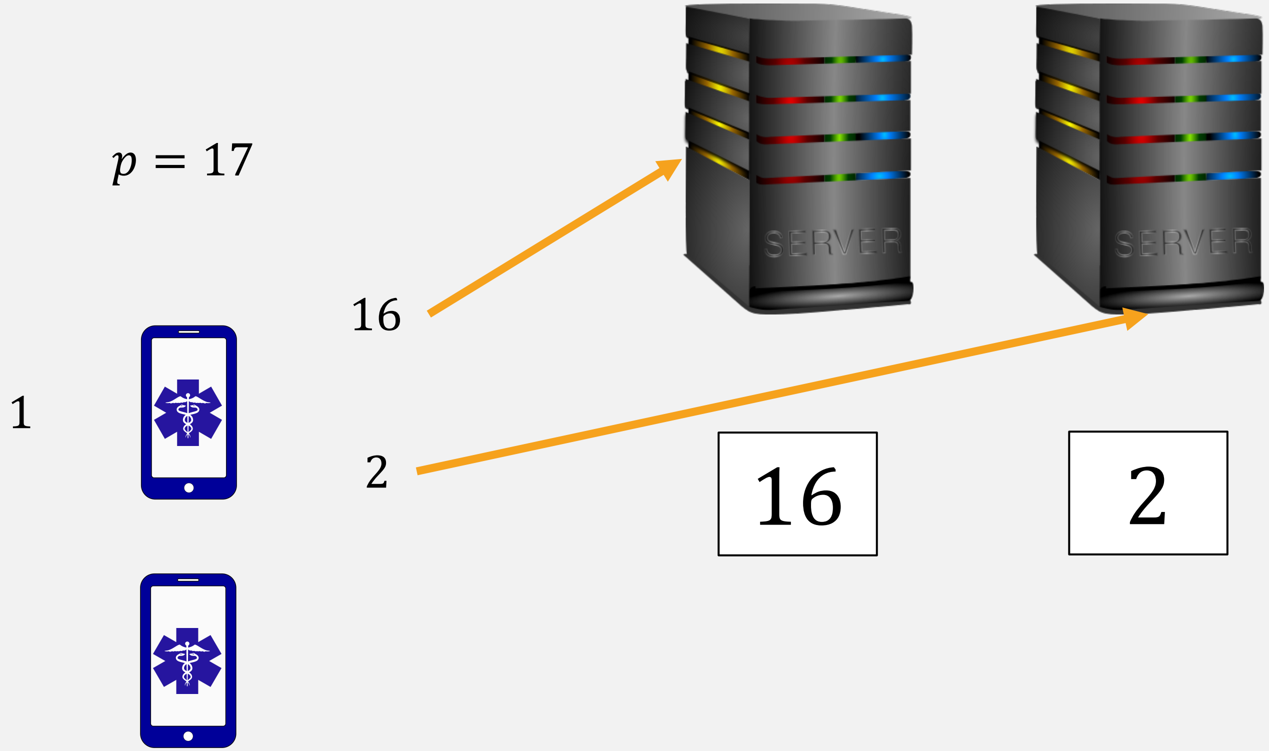
1

2

Example of a Simple Scheme



Example of a Simple Scheme



Example of a Simple Scheme

$p = 17$

16

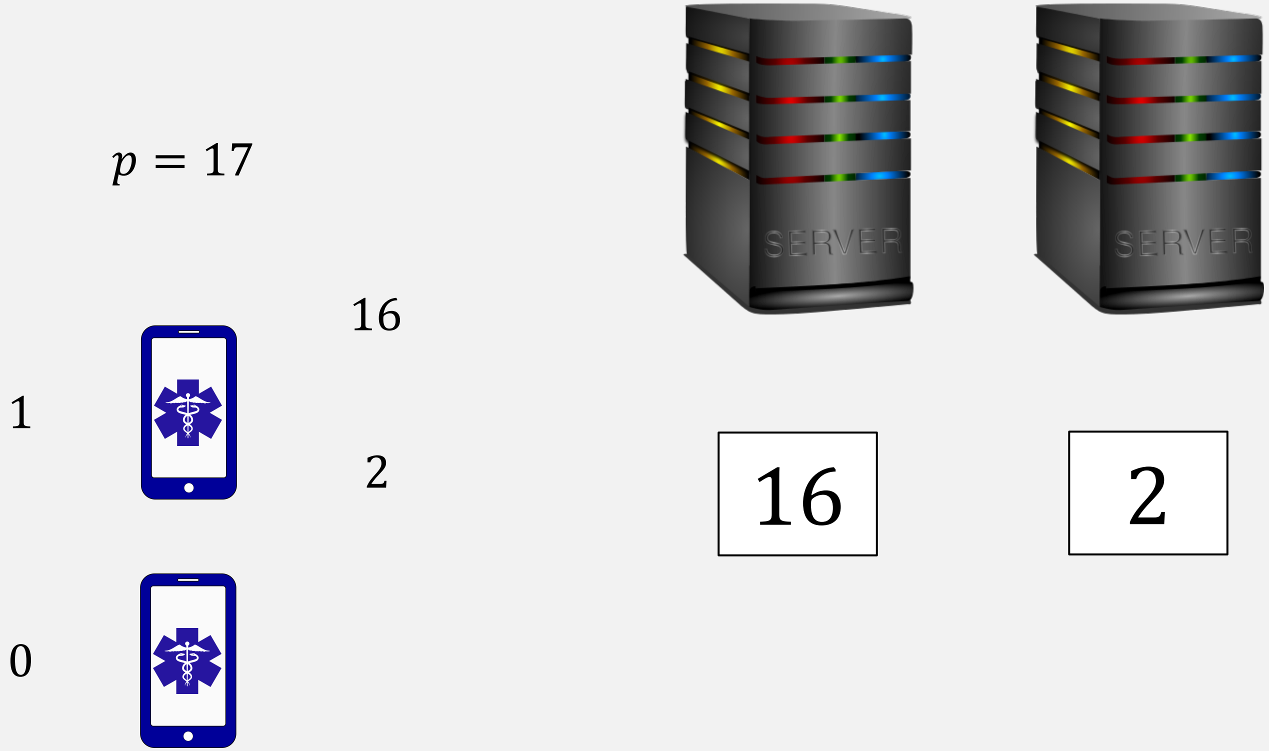
1

2

16

2

Example of a Simple Scheme



Example of a Simple Scheme

$p = 17$

16

1 

2

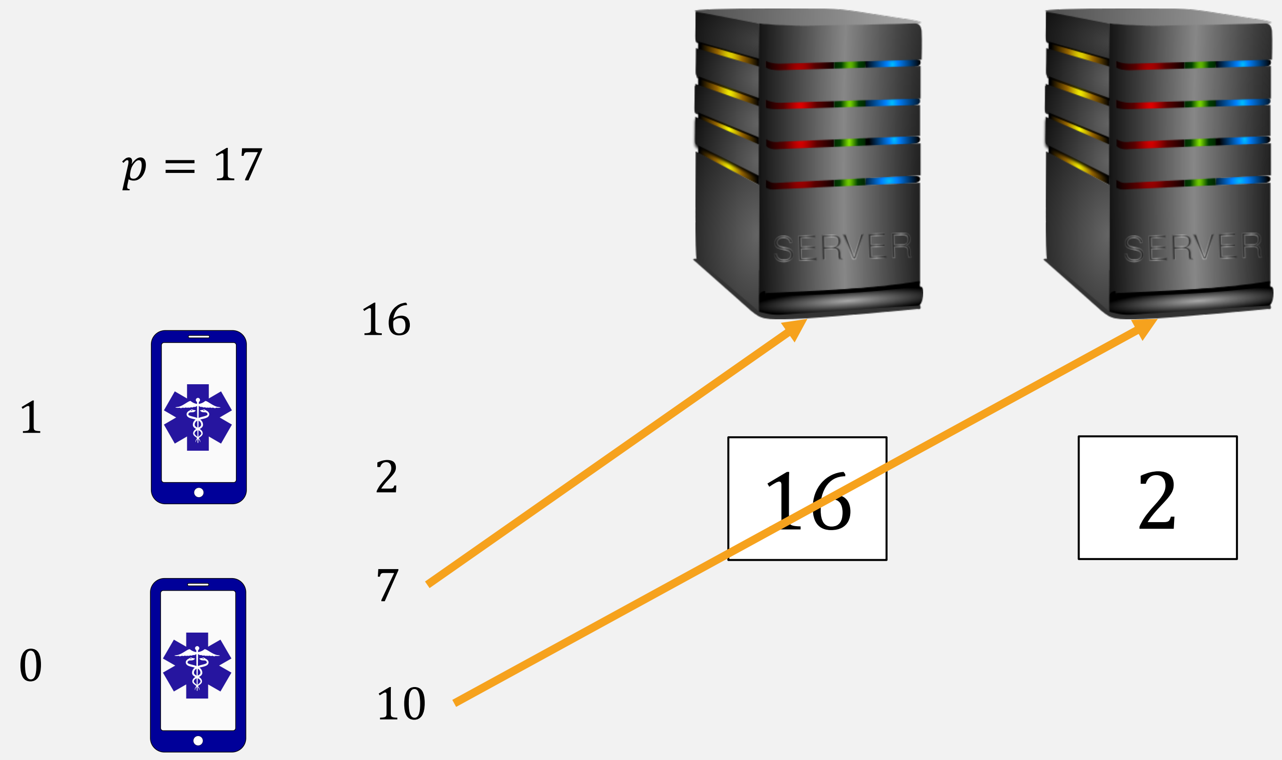
7

0 

10

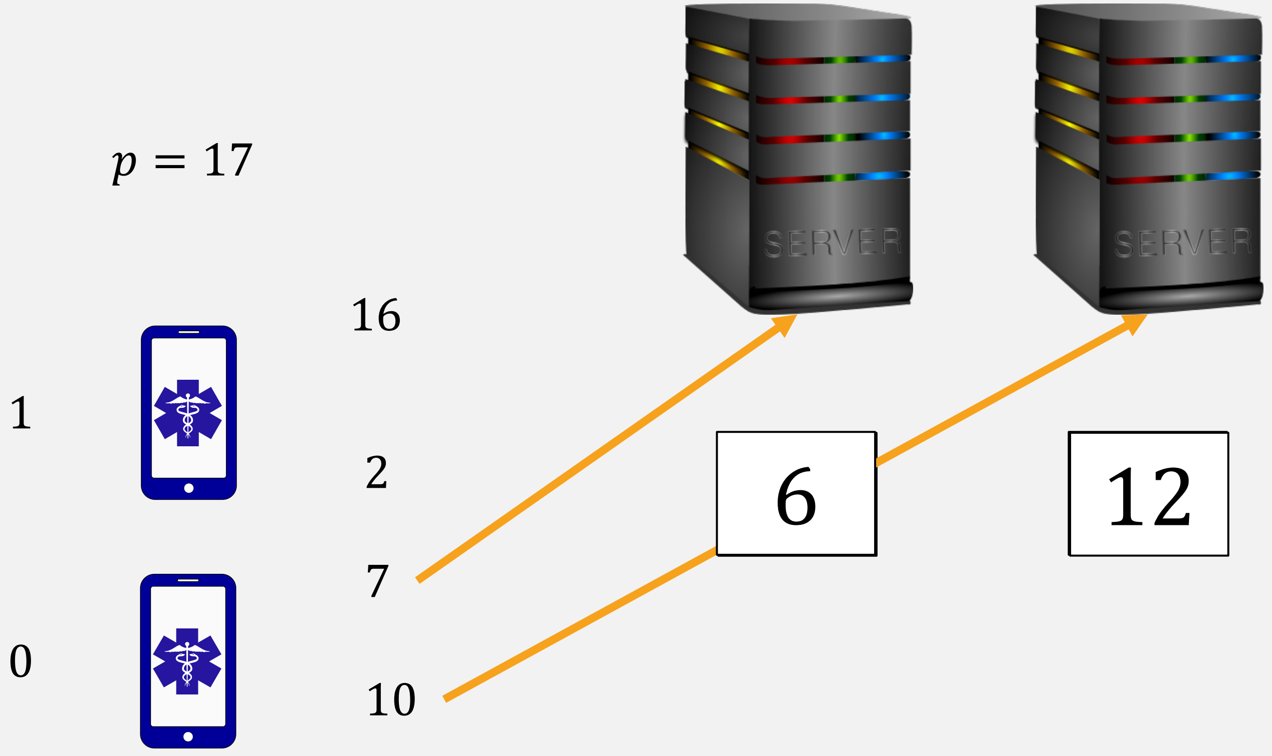
$\boxed{16}$ $\boxed{2}$

Example of a Simple Scheme



Example of a Simple Scheme

$p = 17$



Example of a Simple Scheme

$p = 17$

16

1

2

6

12

7

0

10

Example of a Simple Scheme

$p = 17$

16

1 

2

7

0 

10

6	12

1

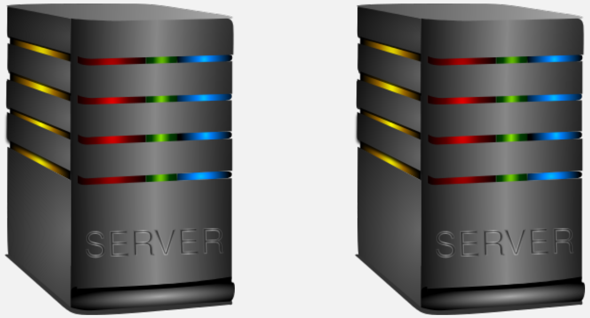
Example of a Simple Scheme

$p = 17$

16

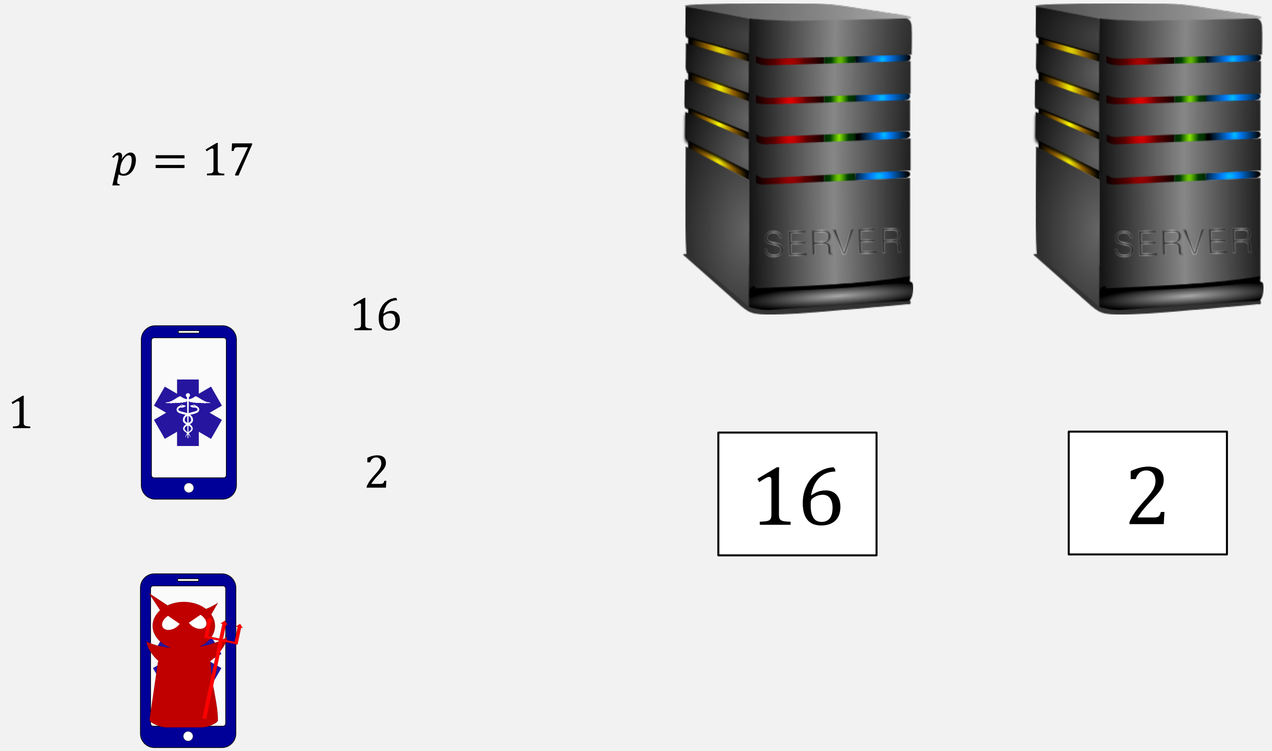
1

2



$\boxed{16}$ $\boxed{2}$

Example of a Simple Scheme



Example of a Simple Scheme

$p = 17$

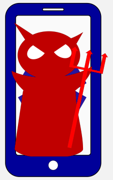
1

16

2

16

2



15

Example of a Simple Scheme

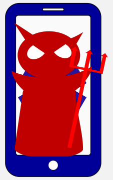
$p = 17$

16

1 

2

9

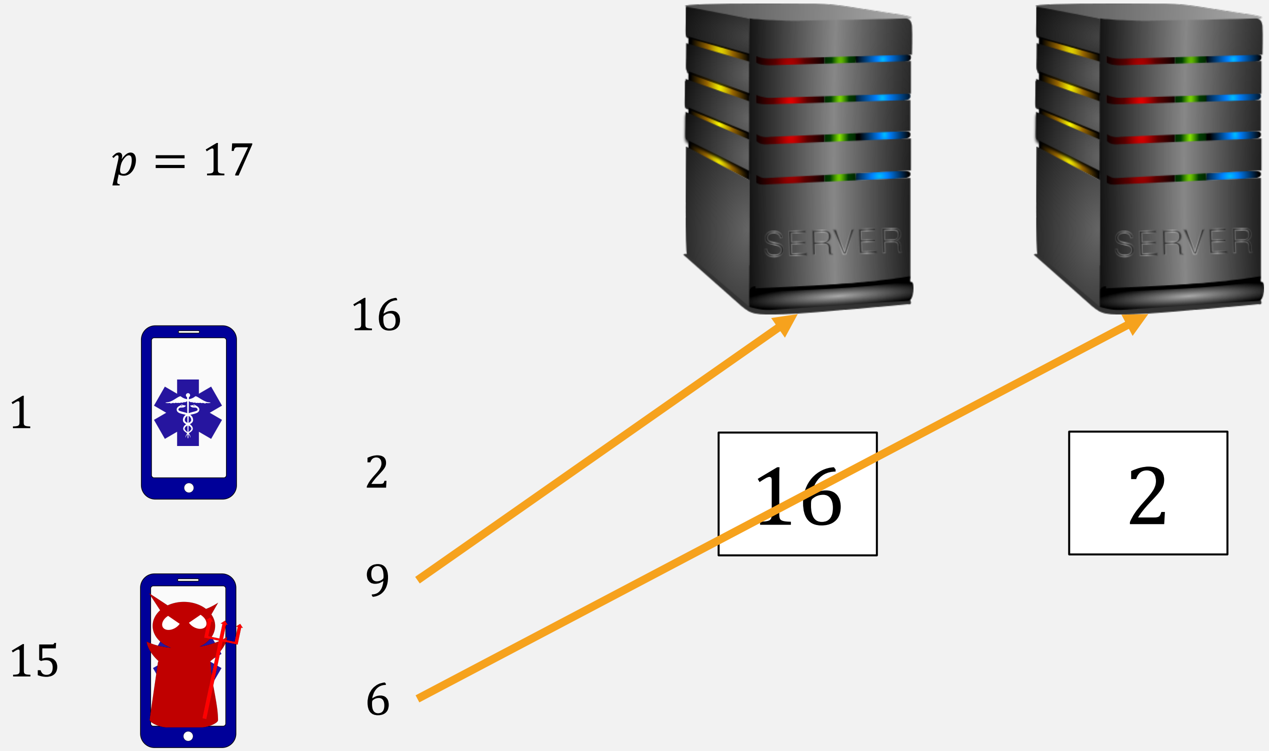
15 

6

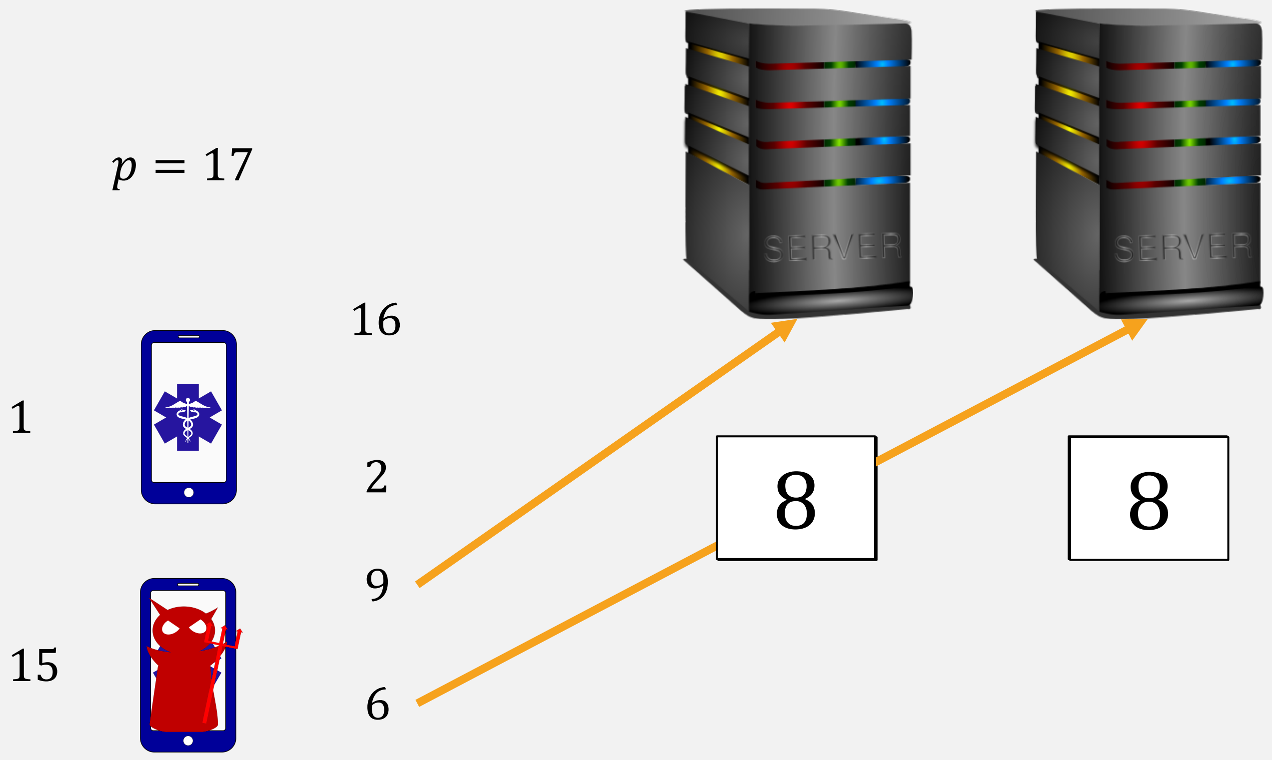
$\boxed{16}$ $\boxed{2}$

Example of a Simple Scheme

$p = 17$



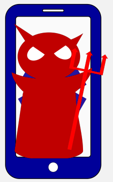
Example of a Simple Scheme



Example of a Simple Scheme

$p = 17$

1  16, 2

15  9, 6

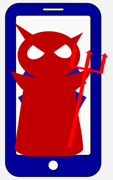
8	8

Example of a Simple Scheme

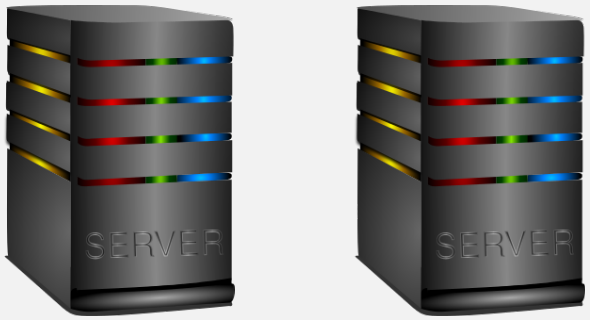
$p = 17$

1  16

2

15  9

6



8 8

16

Adding Robustness: SNIPs

Secret-shared non-interactive proofs (SNIPs).

For some circuit $Valid$ a SNIP proves that secret-shared data x is such that $Valid(x) = 1$.

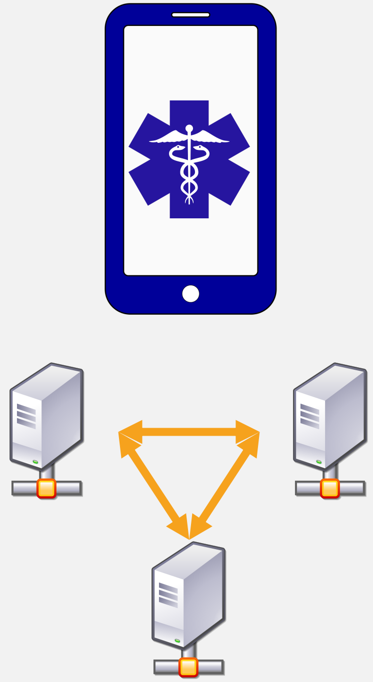
- Correctness: If all parties are honest, the servers will accept x.
- Soundness: If all servers are honest, and if $Valid(x) \neq 1$ then for all malicious clients the servers will reject x.
- Zero knowledge: If the client and at least one server are honest, the servers learn nothing about x except that $Valid(x) = 1$.

SNIPs at a High Level: Think ZKBoo

SNIPs at a High Level: Think ZKBoo

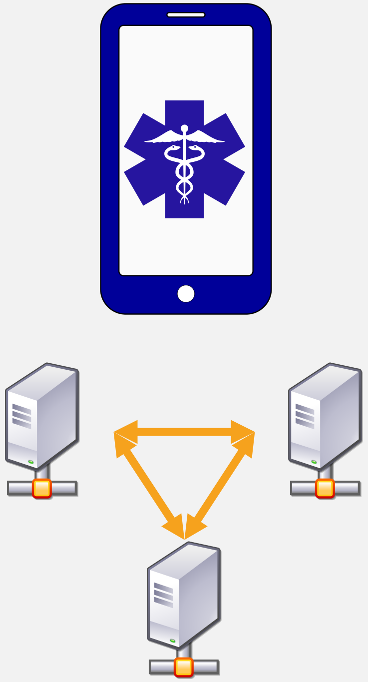
SNIPs at a High Level: Think ZKBoo

SNIPs at a High Level: Think ZKBoo



$Valid(x)$

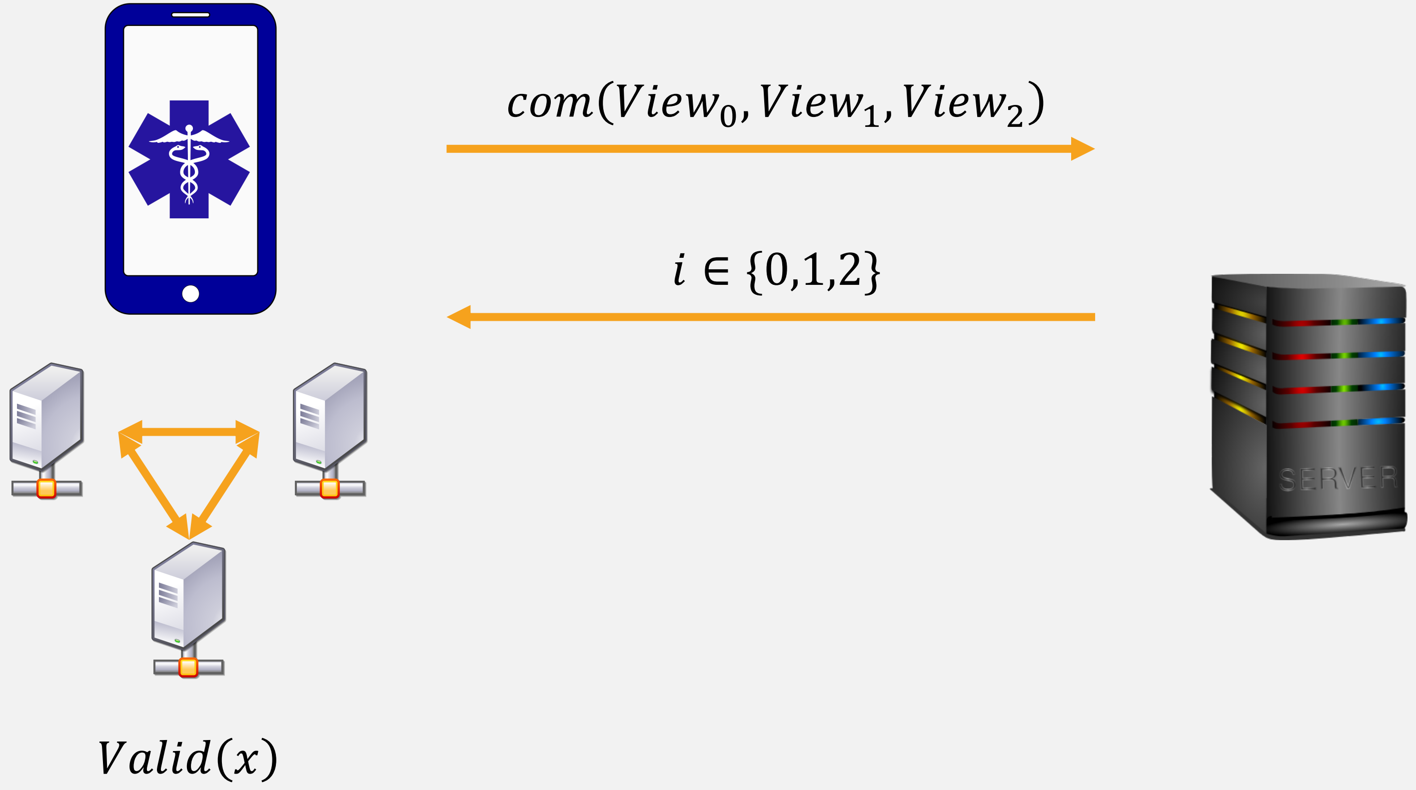
SNIPs at a High Level: Think ZKBoo



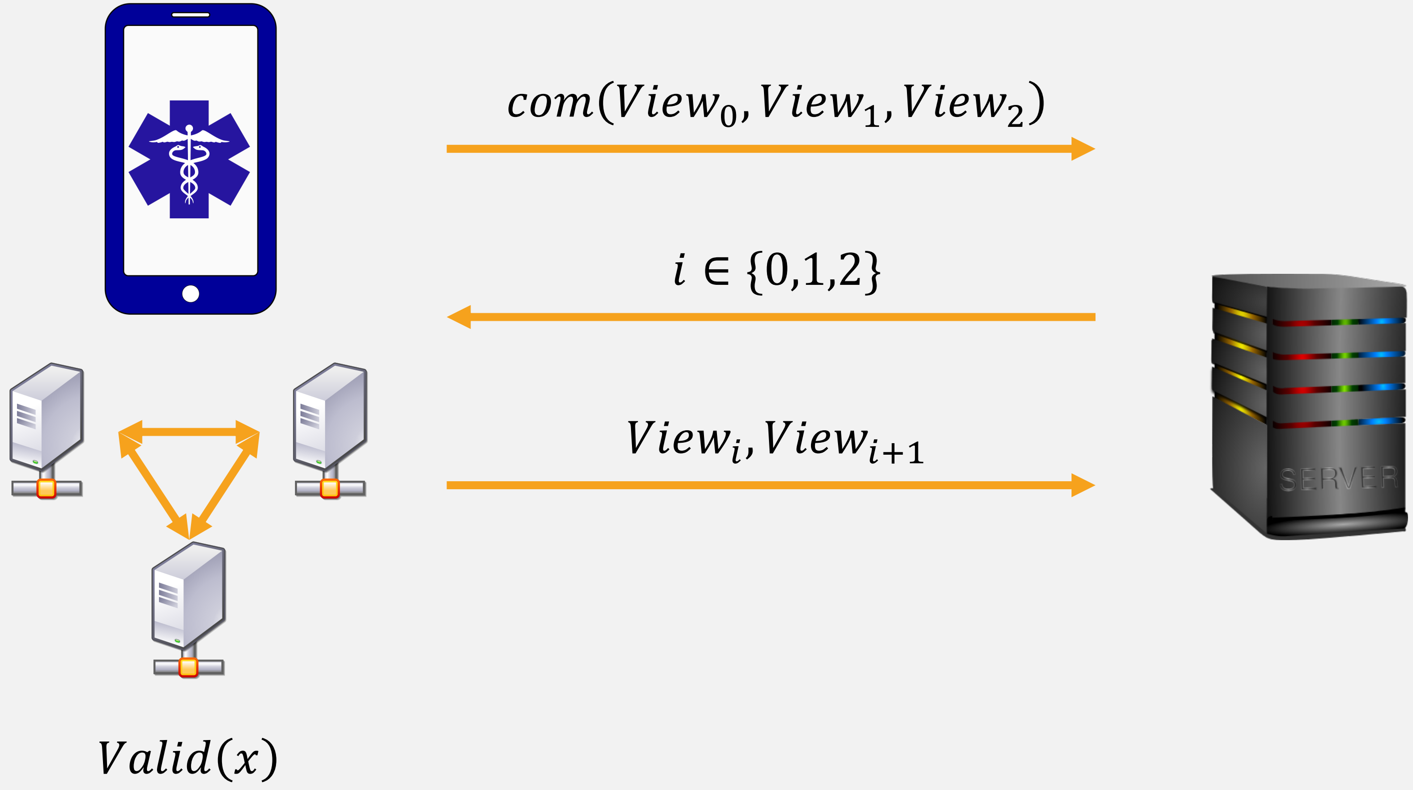
$com(View_0, View_1, View_2)$

$Valid(x)$

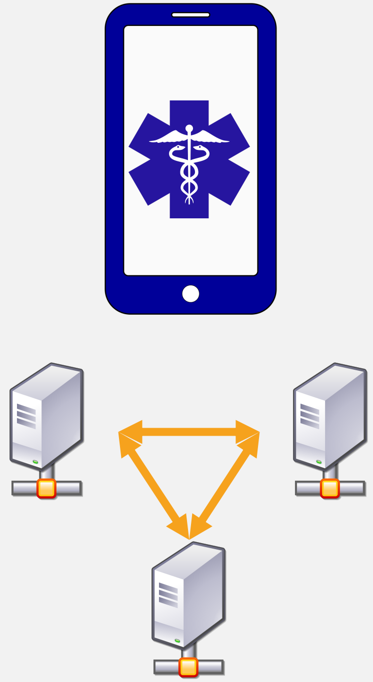
SNIPs at a High Level: Think ZKBoo



SNIPs at a High Level: Think ZKBoo

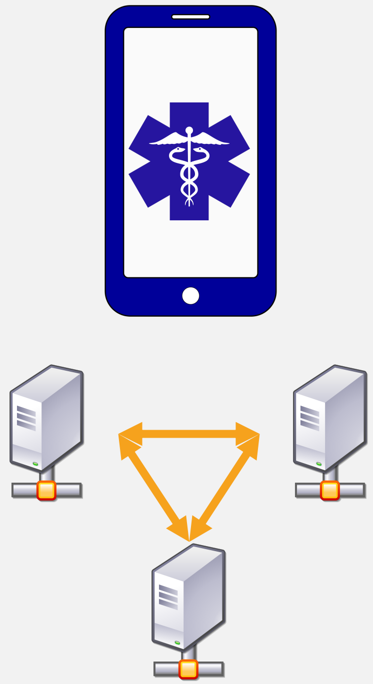


SNIPs at a High Level: Think ZKBoo



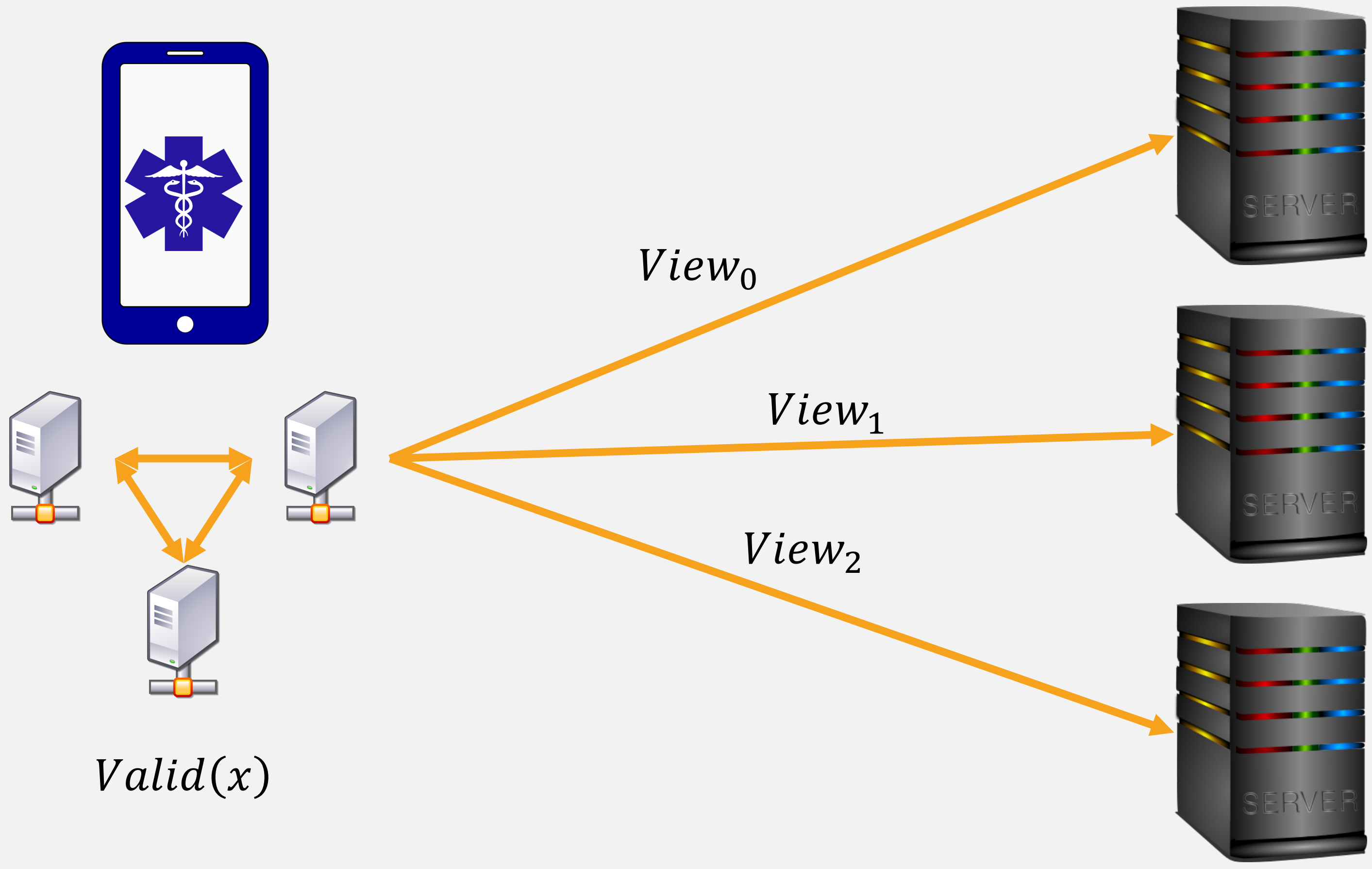
$Valid(x)$

SNIPs at a High Level: Think ZKBoo

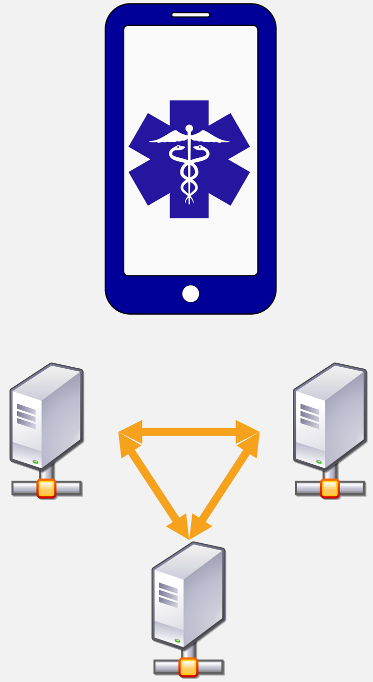


$Valid(x)$

SNIPs at a High Level: Think ZKBoo

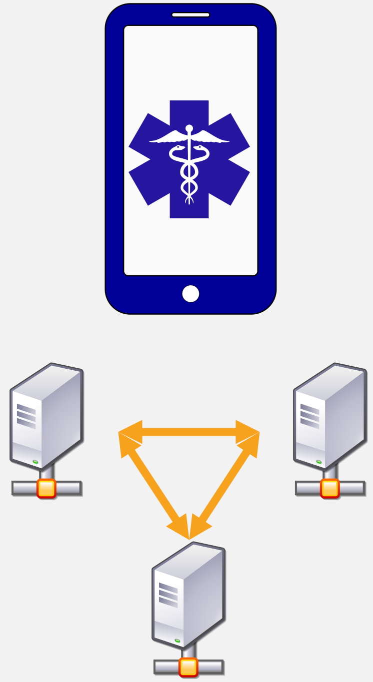


SNIPs at a High Level: Think ZKBoo



$Valid(x)$

SNIPs at a High Level: Think ZKBoo



$Valid(x)$

SNIPs in detail

Step 1: Client

Executes $Valid(x)$.

Uses polynomial interpolation to construct polynomials f and g and multiplies to get $h = f \cdot g$.

Split and send to each server $[f(0)]_i$ and $[g(0)]_i$ and $[h]_i$.

Polynomial Interpolation

For M multiplication gates in $Valid$

Where u_t and v_t are the left and right input of the t-th multiplication gate.

$$h(t) = f(t) \cdot g(t) = u_t \cdot v_t \; \forall \, t \in \{1, \dots, M\}$$

SNIPs in detail

Step 2: Server i

Constructs shares of $[f_i], [g_i], [h_i]$.

Perform polynomial identity test to prove that $[f(t)] \cdot [g(t)] = [h(t)]$.

Multiplication of shares between servers using constant round MPC protocol

Publish shares of the output wire from MPC protocol and check if sum is 1.

What kind of statistics can this system gather?

From computing private sums some of the aggregates you can compute using known techniques:

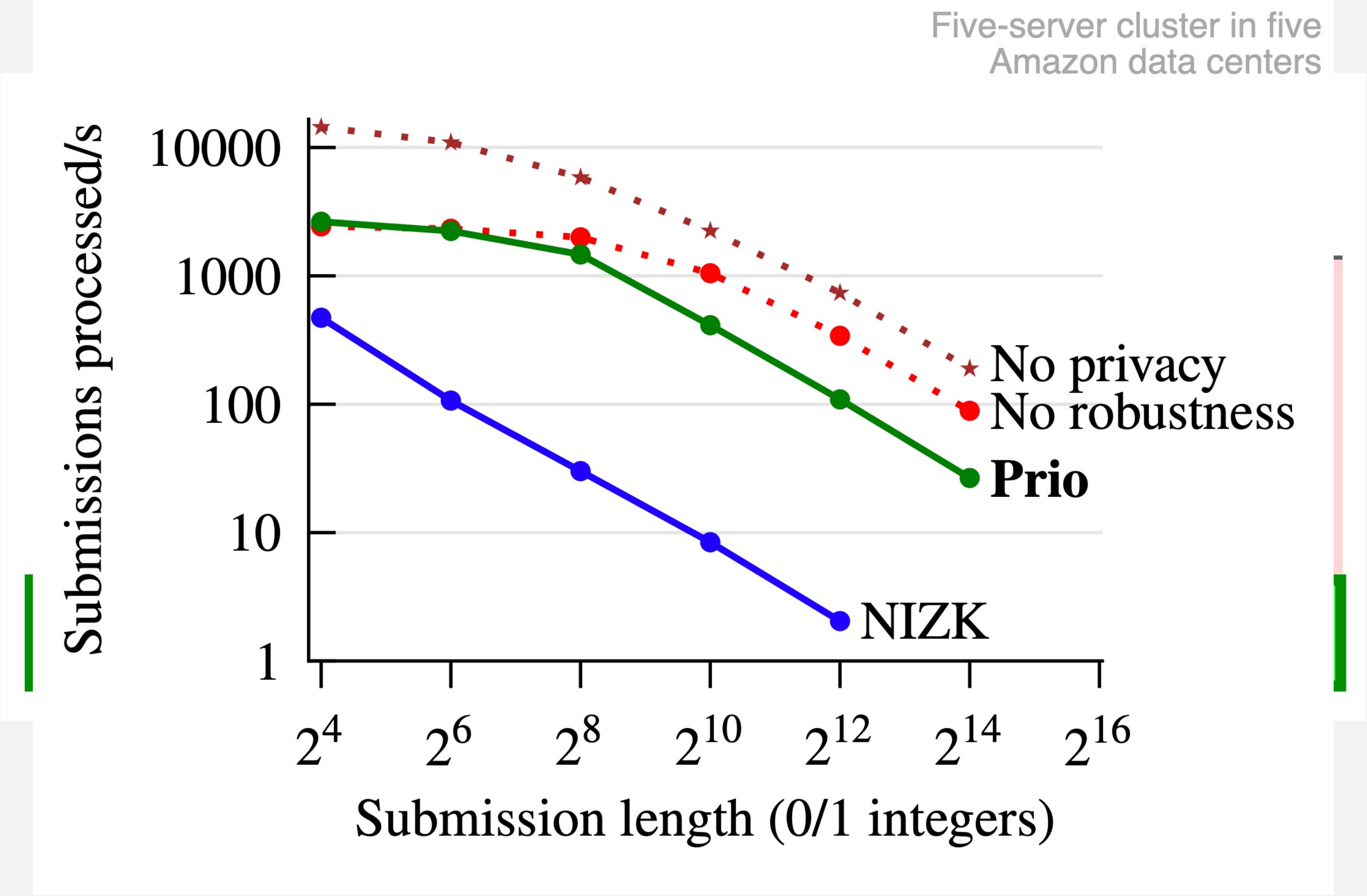
- Average
- Variance
- Standard Deviation
- Most Popular (approx.)
- “Heavy Hitters” (approx.)
- Min and max (approx.)
- Privately train linear models (machine learning)
- Least-squares regression
- Stochastic gradient descent

Efficiency Results

M = # of multiplication gates in Valid(·) circuit	Public-key ops. Client	Public-key ops. Server	Communication C-to-S	Communication S-to-S	Slow-down
Dishonest-maj. MPC [CLOS02], [DPSZ12], …	0	Θ(M)	0	Θ(M)	5,000x at server
Commits + NIZKs [FS86], [CP92], [CS97], …	Θ(M)	Θ(M)	Θ(M)	Θ(M)	50x at server
Commits + SNARKs [GGPR13], [BCGTV13], …	Θ(M)	O(1)	O(1)	O(1)	500x at client
This work: SNIPs	0	0	Θ(M)	O(1)	1x

[Corrigan-Gibbs 2017]

Efficiency Results



[Corrigan-Gibbs 2017]