

MPC-Friendly Symmetric Key Primitives

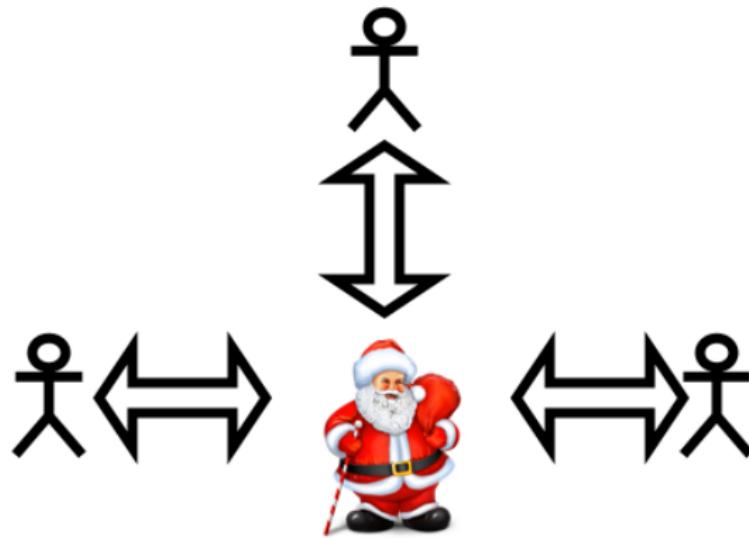
Lorenzo Grassi ¹ Christian Rechberger ¹ **Dragoș Rotaru** ²
Peter Scholl ² Nigel P. Smart ²

¹Graz University of Technology

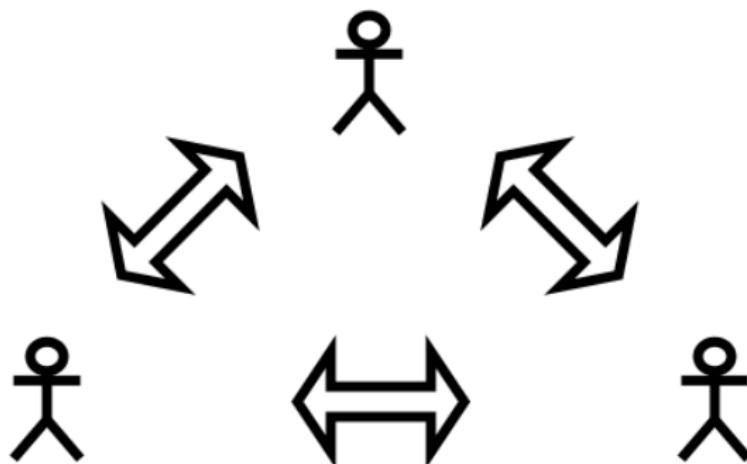
²University of Bristol

October 25, 2016

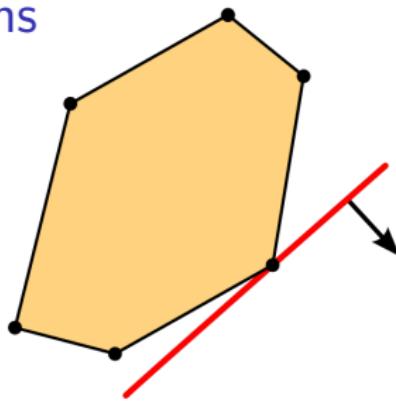
What is Multiparty Computation?



What is Multiparty Computation?

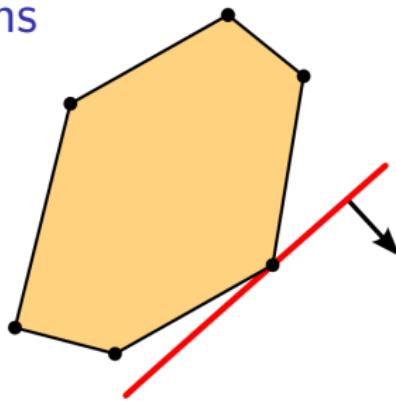


Interesting problems



Linear Programming

Interesting problems

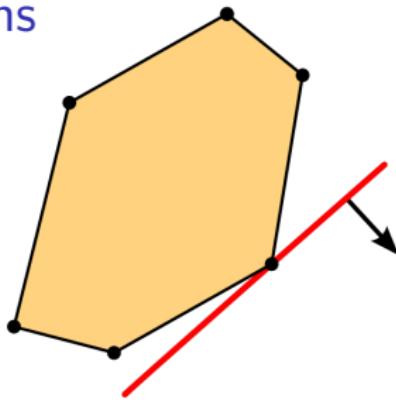


Linear Programming



Integer Comparison

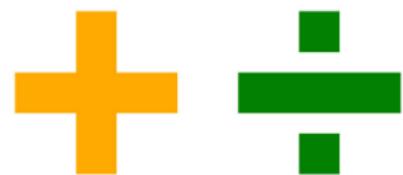
Interesting problems



Linear Programming



Integer Comparison

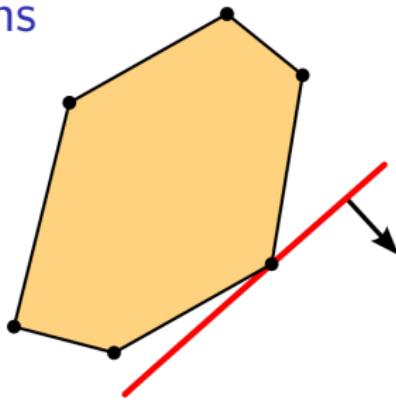


3.141592653589793



Fixed Point Arithmetic

Interesting problems



Linear Programming



Integer Comparison



3.141592653589793



Fixed Point Arithmetic

Interesting problems

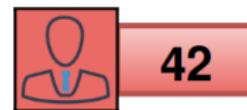
Easy to implement via
arithmetic circuits mod p

There is a problem.

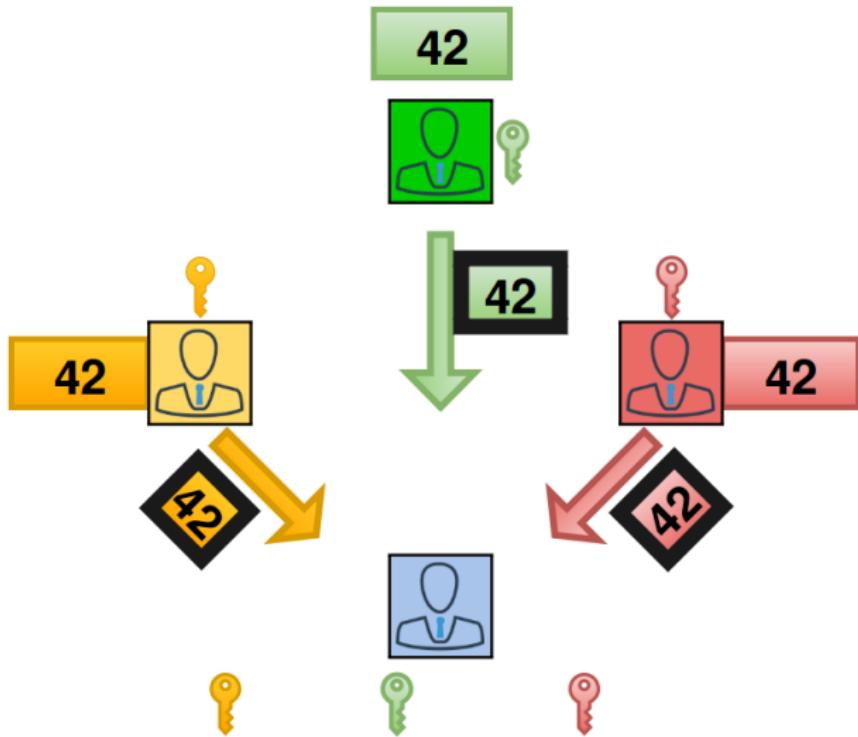
42



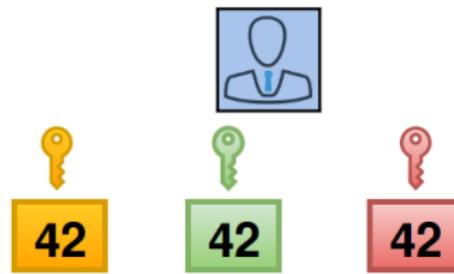
There is a problem.



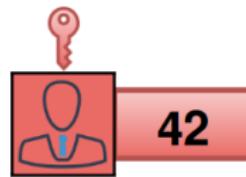
There is a problem.



There is a problem.



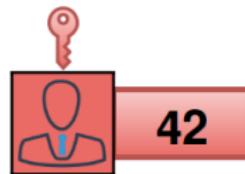
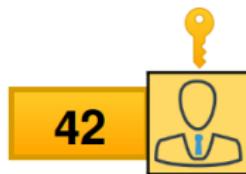
There is a problem.



$$42 + 42 + 42 = 42$$

The equation consists of three terms: a yellow box with "42", a green box with "42", and a red box with "42", followed by an equals sign and another red box with "42". Above the first term is a yellow key icon. Above the second term is a green key icon. Above the third term is a red key icon.

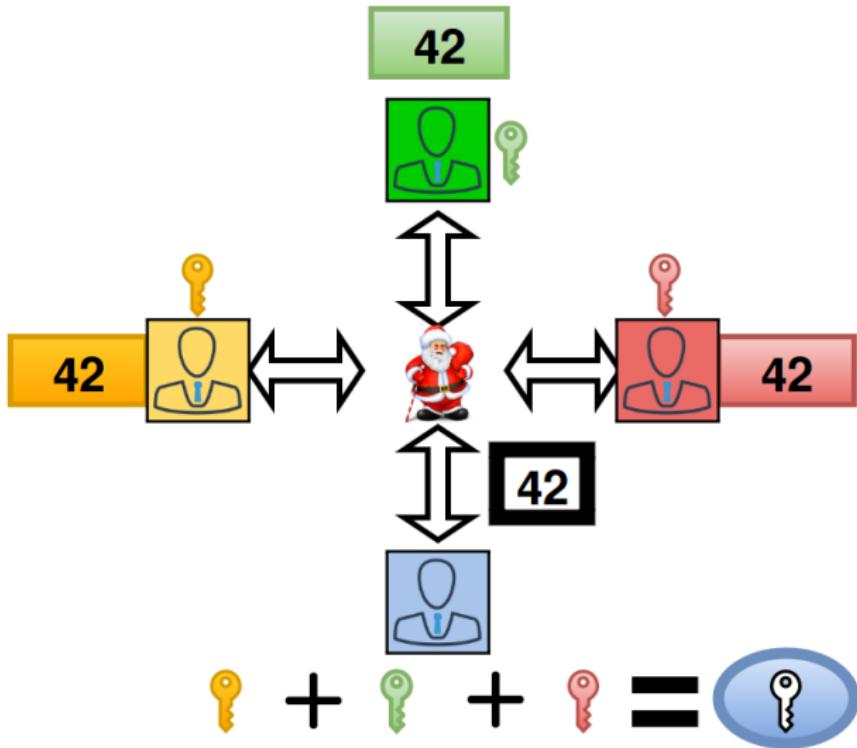
There is a problem.



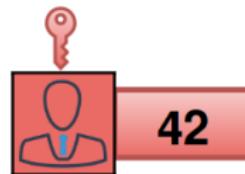
$$\text{Yellow Key} + \text{Green Key} + \text{Red Key} = \text{Large Blue Oval Key}$$

A mathematical equation illustrating a problem. It shows three individual key icons (yellow, green, and red) being added together to equal one large blue oval key icon. This likely represents the concept of combining multiple individual components to achieve a single, larger result.

There is a problem.



There is a problem.

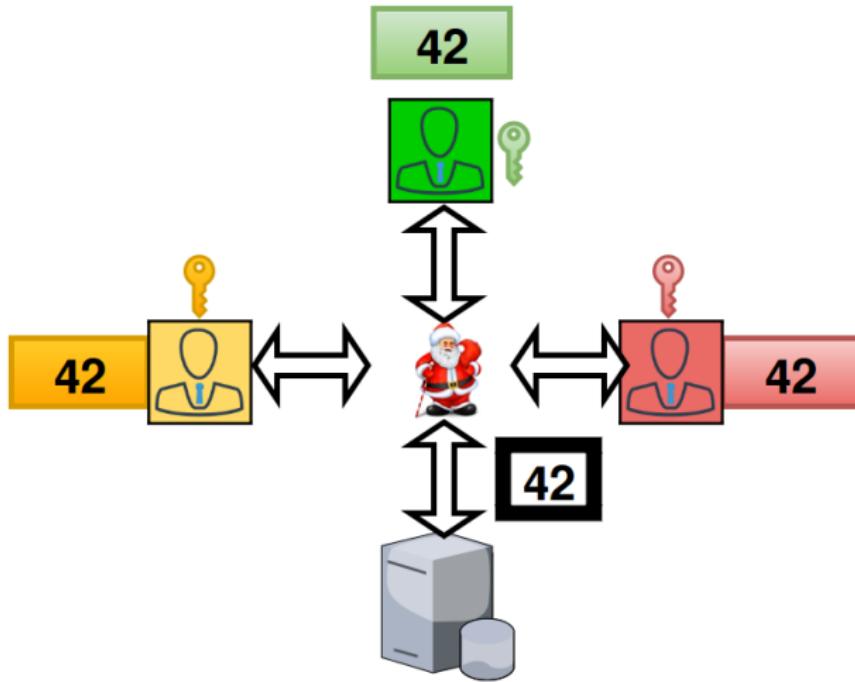


$$\text{Yellow Key} + \text{Green Key} + \text{Red Key} = \text{Blue Oval Key}$$

42

A diagram illustrating a mathematical concept. At the bottom left is a black-bordered box containing the black text "42". To its right is a blue oval icon containing a white key icon. Above the oval key are three smaller key icons: a yellow one on the left, a green one in the middle, and a red one on the right. A plus sign is placed between the yellow and green keys, and another plus sign is placed between the green and red keys. An equals sign is placed to the right of the red key. To the left of the equals sign is a blue square icon with a white user silhouette.

There is a problem.



Take home message

Move data **securely** between
clients and MPC engines.

Need a PRF mod p

- ▶ Enc / Dec in CTR mode use only PRF calls.
- ▶ Avoid the n fold database/key blowup by secret share the key and use a PRF mod p in MPC!
- ▶ Why mod p ? Conversion between binary and arithmetic shares is expensive.

Other use cases for PRF's in MPC

- ▶ Secure database joins [LTW13].
- ▶ Oblivious RAM [LO13].
- ▶ Searchable symmetric encryption, order-revealing encryption [BCO'N11, BLRSZZ15, CLWW16, BBO'N07, CJJKRS13].

What we have done

Benchmark and **create new protocols** using PRF's within SPDZ protocol.

Why SPDZ?

- ▶ MPC protocol with active security.
- ▶ 200 times faster pre-processing phase [KOS16].
- ▶ It is open source!

<https://github.com/bristolcrypto/SPDZ-2>.

MPC with secret sharing 101

- ▶ Each party P_i has $[a] \leftarrow a_i$
s.t. $a = \sum_{i=1}^n a_i$.
- ▶ Triples generation:
 $[a] = [b] \cdot [c]$
- ▶ Random bits and squares:
 $[b]$, $[s^2]$.



Preprocessing Phase

MPC with secret sharing 101

- ▶ Use 1 triple for each multiplication gate.
- ▶ Number of communication rounds is given by the multiplicative depth.



Online Phase

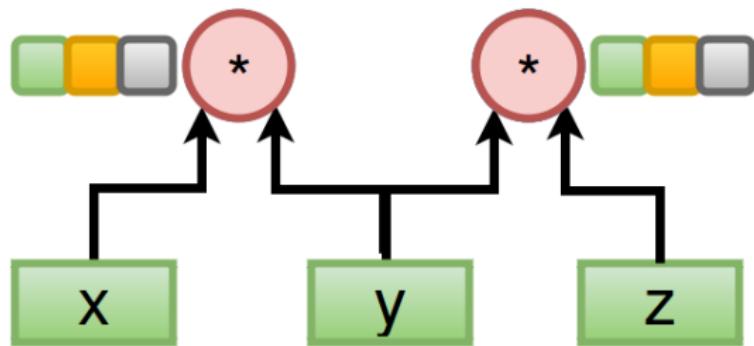
Circuit Evaluation in SPDZ

x

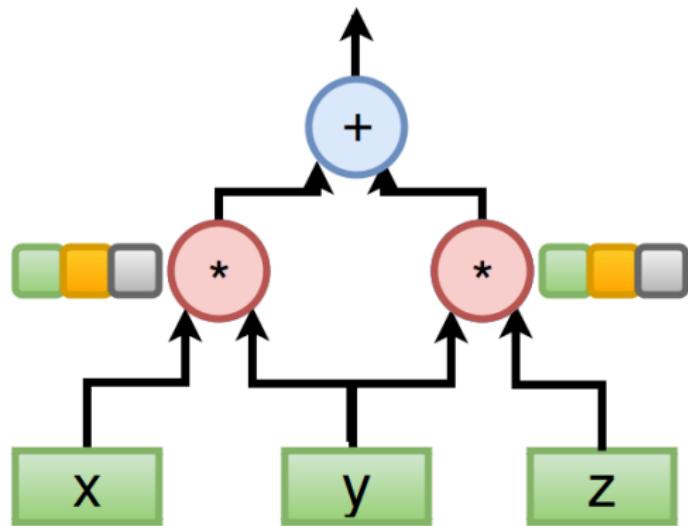
y

z

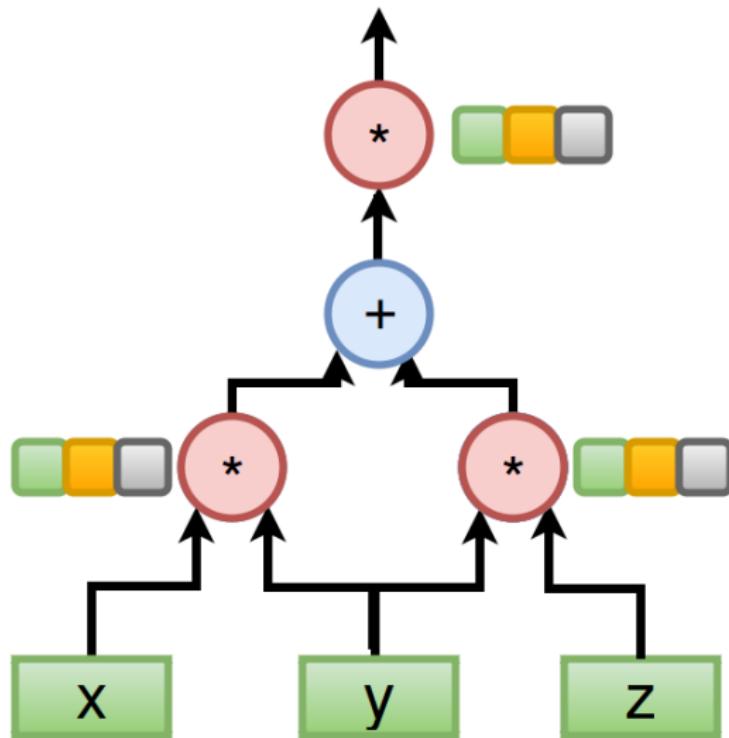
Circuit Evaluation in SPDZ



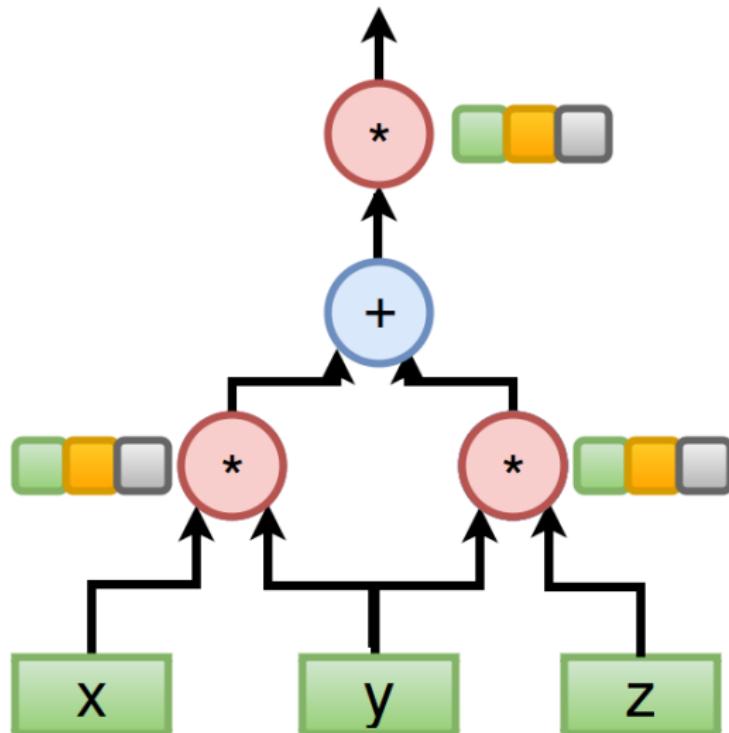
Circuit Evaluation in SPDZ



Circuit Evaluation in SPDZ



Circuit Evaluation in SPDZ



3 triples; 2 rounds.

What PRF's have we looked at?

- ▶ AES [DR01].
- ▶ LowMC (Low Multiplicative Complexity) [ARS⁺15].
- ▶ Naor-Reingold PRF [NR04].
- ▶ MiMC (Minimum Multiplicative Complexity) [AGR⁺16].
- ▶ Legendre PRF [Dam88].

What PRF's have we looked at?

- ▶ AES [DR01].
- ▶ LowMC (Low Multiplicative Complexity) [ARS⁺15].
- ▶ Naor-Reingold PRF [NR04].
- ▶ MiMC (Minimum Multiplicative Complexity) [AGR⁺16].
- ▶ Legendre PRF [Dam88].

Let's play a game



Let's play a game



AES - de-facto benchmark

- ▶ 960 multiplications
- ▶ 50 rounds
- ▶ Operations done in $\mathbb{F}_{2^{40}}$.



PRF on blocks

AES - de-facto benchmark

- ▶ 960 multiplications
- ▶ 50 rounds
- ▶ Operations done in $\mathbb{F}_{2^{40}}$.



PRF on blocks



5 blocks/s

AES - de-facto benchmark

- ▶ 960 multiplications
- ▶ 50 rounds
- ▶ Operations done in $\mathbb{F}_{2^{40}}$.



PRF on blocks



8ms latency

AES - de-facto benchmark

- ▶ 960 multiplications
- ▶ 50 rounds
- ▶ Operations done in $\mathbb{F}_{2^{40}}$.



PRF on blocks



530 blocks/s throughput

AES - de-facto benchmark

- ▶ Compare the PRF's mod p with AES only for benchmarking purposes.
- ▶ In real world we want to keep all data in \mathbb{F}_p .

Naor-Reingold PRF

$$F_{\text{NR}(n)}(\mathbf{k}, \mathbf{x}) = g^{k_0 \cdot \prod_{i=1}^n k_i^{x_i}}$$

where $\mathbf{k} = (k_0, \dots, k_n) \in \mathbb{F}_p^{n+1}$ is the key.

Naor-Reingold PRF

$$F_{\text{NR}(n)}(\mathbf{k}, \mathbf{x}) = g^{k_0 \cdot \prod_{i=1}^n k_i^{x_i}}$$

where $\mathbf{k} = (k_0, \dots, k_n) \in \mathbb{F}_p^{n+1}$ is the key.

Fortunately, in some applications the output must be public!

Naor-Reingold PRF

- ▶ Active security version for public output.
- ▶ Why EC? Smaller modulus.
- ▶ $2 \cdot n$ multiplications.
- ▶ $3 + \log n + 1$ rounds.



EC based PRF

Naor-Reingold PRF

- ▶ Active security version for public output.
- ▶ Why EC? Smaller modulus.
- ▶ $4n + 2$ multiplications.
- ▶ 7 rounds [BB89, CH10].



EC based PRF in constant round

Naor-Reingold PRF

- ▶ Active security version for public output.
- ▶ Why EC? Smaller modulus.
- ▶ $4n + 2$ multiplications.
- ▶ 7 rounds [BB89, CH10].



EC based PRF in constant round



5 evals/s

Naor-Reingold PRF

- ▶ Active security version for public output.
- ▶ Why EC? Smaller modulus.
- ▶ $4n + 2$ multiplications.
- ▶ 7 rounds [BB89, CH10].



EC based PRF in constant round



4.3ms latency

Naor-Reingold PRF

- ▶ Active security version for public output.
- ▶ Why EC? Smaller modulus.
- ▶ $4n + 2$ multiplications.
- ▶ 7 rounds [BB89, CH10].



EC based PRF in constant round



370 blocks/s throughput

Naor-Reingold PRF

- ▶ Active security version for public output.
- ▶ Why EC? Smaller modulus.
- ▶ $4n + 2$ multiplications.
- ▶ 7 rounds [BB89, CH10].



EC based PRF in constant round

Results have shown that over 70% of the time was spent on EC computations.

Computation is the bottleneck, not communication!

MiMC - How does it work?

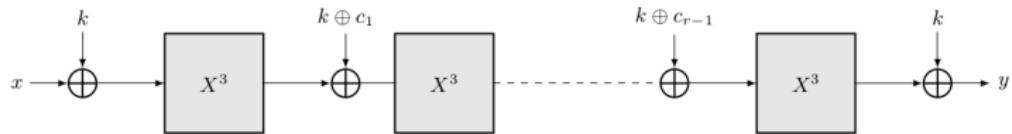


Fig. 1: r rounds of MiMC- n/n

[AGR⁺16]

MiMC PRF

- ▶ 146 multiplications
- ▶ 73 rounds
- ▶ 1 variant optimized for latency, other for throughput.



MiMC PRF - works in both worlds

MiMC PRF

- ▶ 146 multiplications
- ▶ 73 rounds
- ▶ 1 variant optimized for latency, other for throughput.



34 blocks/s

MiMC PRF - works in both worlds

MiMC PRF

- ▶ 146 multiplications
- ▶ 73 rounds
- ▶ 1 variant optimized for latency, other for throughput.



6ms latency

MiMC PRF - works in both worlds

MiMC PRF

- ▶ 146 multiplications
- ▶ 73 rounds
- ▶ 1 variant optimized for latency, other for throughput.



9000 blocks/s throughput - **16x AES**

MiMC PRF - works in both worlds

Legendre PRF

In 1988, Damgård conjectured that this sequence is pseudorandom starting from a random seed k .

$$\left(\frac{k}{p}\right), \left(\frac{k+1}{p}\right), \left(\frac{k+2}{p}\right), \dots$$

Legendre PRF - 1 bit output

- ▶ $\log p$ multiplications.
- ▶ $\log p$ rounds.



Legendre PRF - old version

Legendre PRF - 1 bit output

- ▶ $\log p$ 2 multiplications.
- ▶ $\log p$ 3 rounds.



Legendre PRF - new version

Legendre PRF - 1 bit output

- ▶ $\log p$ 2 multiplications.
- ▶ $\log p$ 3 rounds.



Legendre PRF - new version



1225 evals/s - **250x AES**

Legendre PRF - 1 bit output

- ▶ $\log p$ 2 multiplications.
- ▶ $\log p$ 3 rounds.



Legendre PRF - new version



0.3ms latency - **25x** faster AES

Legendre PRF - 1 bit output

- ▶ $\log p$ 2 multiplications.
- ▶ $\log p$ 3 rounds.



Legendre PRF - new version



202969 blocks/s throughput - **380x**
AES

How does it work?

Protocol Π_{Legendre}

Let α be a fixed, quadratic non-residue modulo p , i.e. $\left(\frac{\alpha}{p}\right) = -1$.

Eval: To evaluate $F_{\text{Leg(bit)}}$ on input $[x]$ with key $[k]$:

1. Take a random square $[s^2]$ and a random bit $[b]$
2. $[t] \leftarrow [s^2] \cdot ([b] + \alpha \cdot (1 - [b]))$
3. $u \leftarrow \text{Open}([t] \cdot ([k] + [x]))$
4. Output $[y] \leftarrow \left(\frac{u}{p}\right) \cdot (2[b] - 1)$

Securely computing the $F_{\text{Leg(bit)}}$ PRF with shared output

How does it work?

Protocol Π_{Legendre}

Let α be a fixed, quadratic non-residue modulo p , i.e. $\left(\frac{\alpha}{p}\right) = -1$.

Eval: To evaluate $F_{\text{Leg(bit)}}$ on input $[x]$ with key $[k]$:

1. Take a random square $[s^2]$ and a random bit $[b]$
2. $[t] \leftarrow [s^2] \cdot ([1] + \alpha \cdot (1 - [1]))$
3. $u \leftarrow \text{Open}([s^2] \cdot ([k] + [x]))$
4. Output $[y] \leftarrow \left(\frac{u}{p}\right) \cdot (2[1] - 1)$

Securely computing the $F_{\text{Leg(bit)}}$ PRF with shared output

How does it work?

Protocol Π_{Legendre}

Let α be a fixed, quadratic non-residue modulo p , i.e. $\left(\frac{\alpha}{p}\right) = -1$.

Eval: To evaluate $F_{\text{Leg(bit)}}$ on input $[x]$ with key $[k]$:

1. Take a random square $[s^2]$ and a random bit $[b]$
2. $[t] \leftarrow [s^2] \cdot ([0] + \alpha \cdot (1 - [0]))$
3. $u \leftarrow \text{Open}([s^2\alpha] \cdot ([k] + [x]))$
4. Output $[y] \leftarrow \left(\frac{u}{p}\right) \cdot (2[0] - 1)$

Securely computing the $F_{\text{Leg(bit)}}$ PRF with shared output

Security of Legendre PRF

Is it secure?



Security of Legendre PRF

Is it secure?



Yes, we give a reduction to the SLS problem: Given $(\frac{k+x}{p})$, find x .

Summary

- ▶ We have **efficiently** solved the problem of sending data between MPC engines.
- ▶ PRF's mod p in MPC are fast! Can you find other applications built on top of these?
- ▶ For proofs, WAN timings, other details, check out our paper!

Thank you!

LAN results

PRF	Best latency (ms/op)	Best throughput		Prep. (ops/s)	Cleartext (ops/s)
		Batch size	ops/s		
AES	8.378	2048	552	5.097	106268670
$F_{\text{NR}(128)}(\log)$	4.375	1024	370	4.787	1359
$F_{\text{NR}(128)}(\text{const})$	4.549	256	281	2.384	1359
$F_{\text{Leg}(\text{bit})}$	0.393	1024	163400	1225	17824464
$F_{\text{Leg}(1)}$	1.418	64	1331	9.574	115591
$F_{\text{MiMC}}(\text{basic})$	14.053	1024	6561	33.575	189525
$F_{\text{MiMC}}(\text{cube})$	7.267	2048	5536	33.575	189525

Table 3. Performance of the PRFs in a LAN setting

WAN results

PRF	Best latency (ms/op)	Best throughput		Prep. (ops/s)
		Batch size	ops/s	
AES	2640	1024	31.947	0.256
$F_{NR(128)}(\log)$	713	1024	59.703	0.2359
$F_{NR(128)}(\text{const})$	478	1024	30.384	0.1175
$F_{\text{Leg}(bit)}$	202	1024	2053	60.241
$F_{\text{Leg}(1)}$	210	512	68.413	0.4706
$F_{MiMC}(\text{basic})$	7379	512	59.04	1.650
$F_{MiMC}(\text{cube})$	3691	512	79.66	1.650

Table 4. Performance of the PRFs in a simulated WAN setting