WAHC 2019

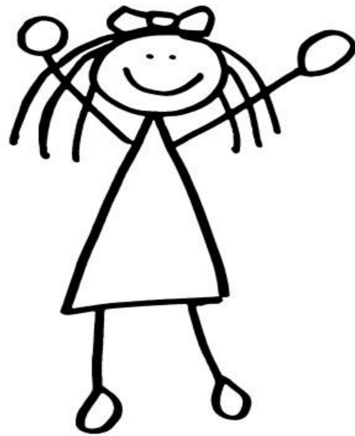
Zaphod: Efficiently Combining LSSS and Garbled Circuits in SCALE*

Abdelrahaman Aly
Emmanuela Orsini
Dragos Rotaru
Nigel P. Smart
Tim Wood

University of Bristol, KU Leuven

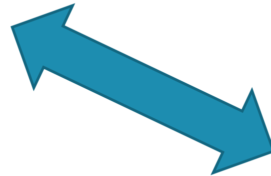
* **https://ia.cr/2019/974**

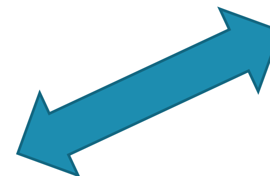
What is multiparty computation?



a

c

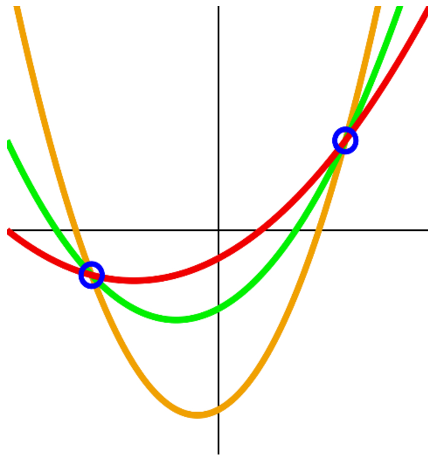




b

Goal: Compute F(a, b, c)

How can we achieve MPC?



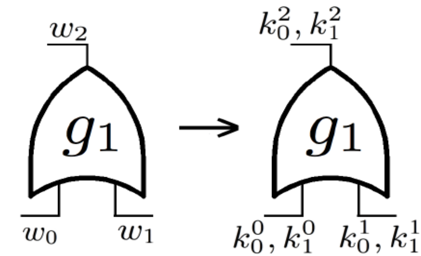


Figure 1: Garbling a single gate

w_0	w_1	w_2
0	0	0
0	1	1
1	0	1
1	1	1

(a) Original Values

w_0	w_1	w_2	garbled value
k_0^0	k_1^0	k_2^0	$H(k_0^0 \| k_1^0 \| g_1) \oplus k_2^0$
k_0^0	k_1^1	k_2^1	$H(k_0^0 \| k_1^1 \| g_1) \oplus k_2^1$
k_0^1	k_1^0	k_2^1	$H(k_0^1 \| k_1^0 \| g_1) \oplus k_2^1$
k_0^1	k_1^1	k_2^1	$H(k_0^1 \| k_1^1 \| g_1) \oplus k_2^1$

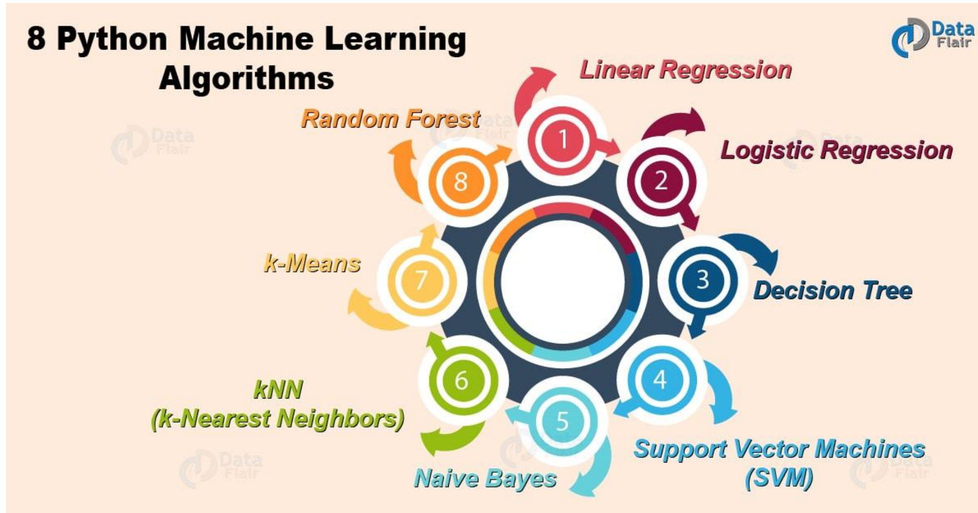
(b) Garbled Values

Figure 2: Computation table for g_1^{OR}

Secret Sharing	Garbled Circuits
Fast networks (LAN)	Slow Networks (WAN)
Arithmetic/Boolean circuits	Boolean circuits
Low depth, many AND gates*	Large depth, few AND gates*

Dragoș Rotaru
imec-Cosic, Dept. Electrical Engineering
KU LEUVEN

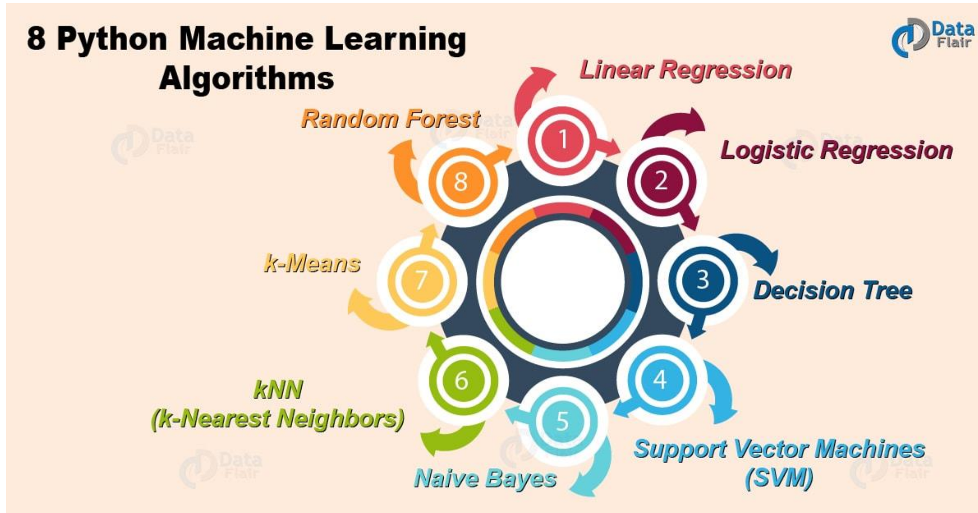
Why switch between?



Sint A, x, b
y = A * x + b
E = argmax(y)

Secret Sharing		Garbled Circuits
Fast networks (LAN)		Slow Networks (WAN)
Arithmetic/Boolean circuits	↔	Boolean circuits
Low depth, many AND gates*	↔	Large depth, few AND gates*

Why switch between?

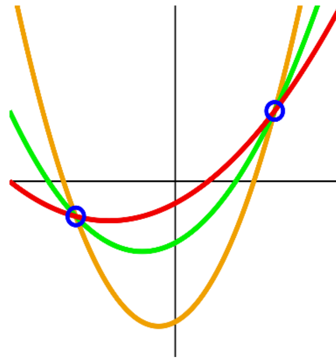


Sint A, x, b
y = A * x + b

E = argmax(y)

Secret Sharing		Garbled Circuits
Fast networks (LAN)		Slow Networks (WAN)
Arithmetic/Boolean circuits	⟷	Boolean circuits
Low depth, many AND gates*	⟷	Large depth, few AND gates*

Can we switch between?



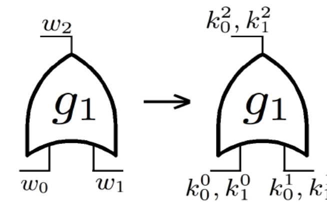


Figure 1: Garbling a single gate

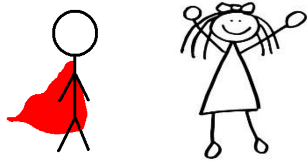
w_0	w_1	w_2
0	0	0
0	1	1
1	0	1
1	1	1

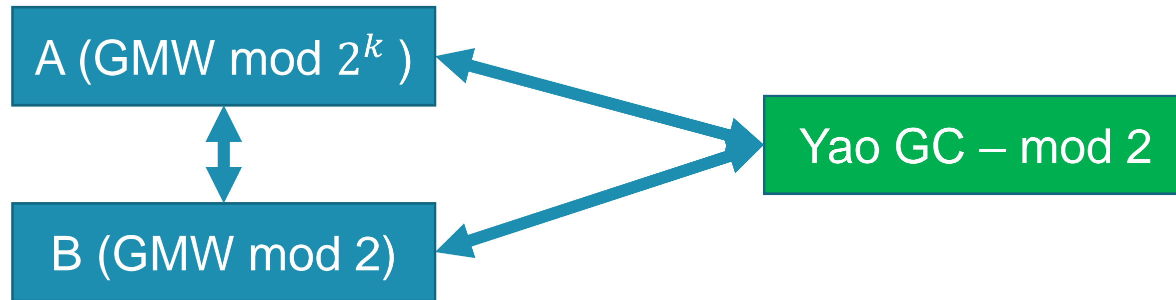
(a) Original Values

w_0	w_1	w_2	garbled value
k_0^0	k_1^0	k_2^0	$H(k_0^0 \| k_1^0 \| g_1) \oplus k_2^0$
k_0^0	k_1^1	k_2^1	$H(k_0^0 \| k_1^1 \| g_1) \oplus k_2^1$
k_0^1	k_1^0	k_2^1	$H(k_0^1 \| k_1^0 \| g_1) \oplus k_2^1$
k_0^1	k_1^1	k_2^1	$H(k_0^1 \| k_1^1 \| g_1) \oplus k_2^1$

(b) Garbled Values

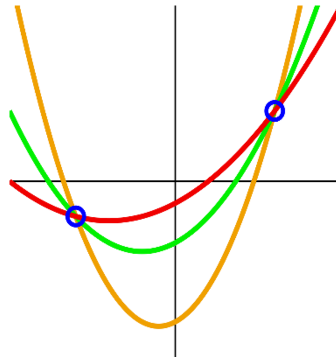
Figure 2: Computation table for g_1^{OR}





ABY [DSZ'15]

Can we switch between?



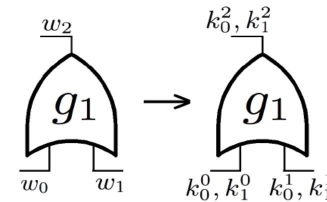


Figure 1: Garbling a single gate

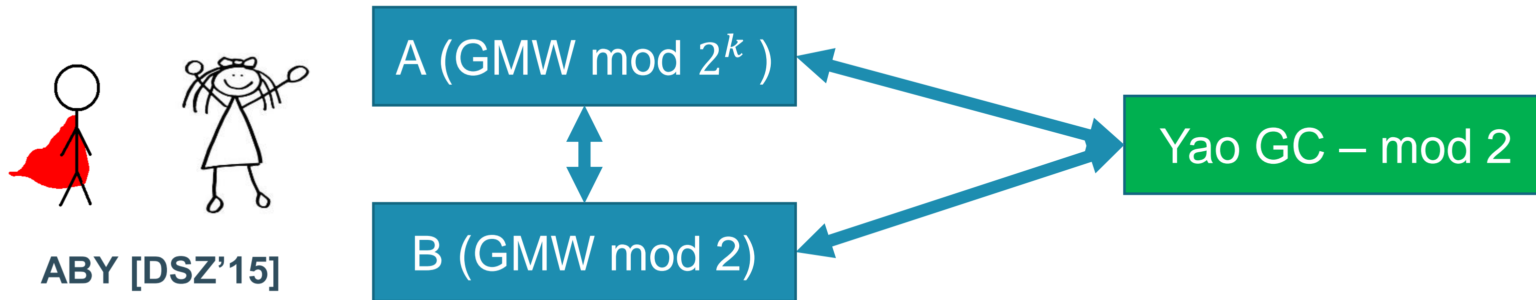
w_0	w_1	w_2
0	0	0
0	1	1
1	0	1
1	1	1

(a) Original Values

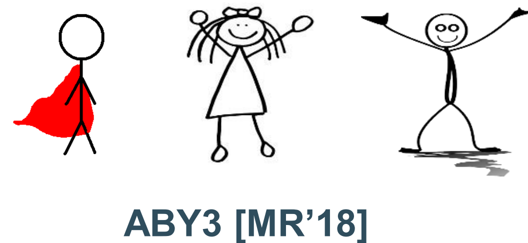
w_0	w_1	w_2	garbled value
k_0^0	k_1^0	k_2^0	$H(k_0^0 \| k_1^0 \| g_1) \oplus k_2^0$
k_0^0	k_1^1	k_2^1	$H(k_0^0 \| k_1^1 \| g_1) \oplus k_2^1$
k_0^1	k_1^0	k_2^1	$H(k_0^1 \| k_1^0 \| g_1) \oplus k_2^1$
k_0^1	k_1^1	k_2^1	$H(k_0^1 \| k_1^1 \| g_1) \oplus k_2^1$

(b) Garbled Values

Figure 2: Computation table for g_1^{OR}

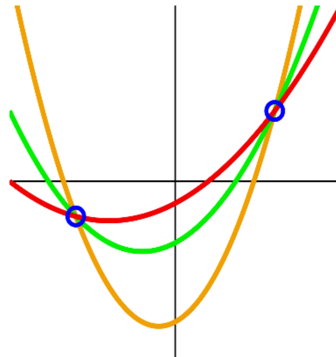


ABY [DSZ'15]



ABY3 [MR'18]

Can we switch between?



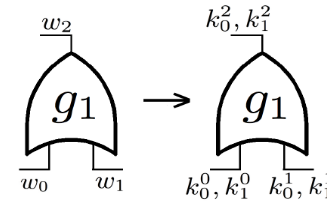


Figure 1: Garbling a single gate

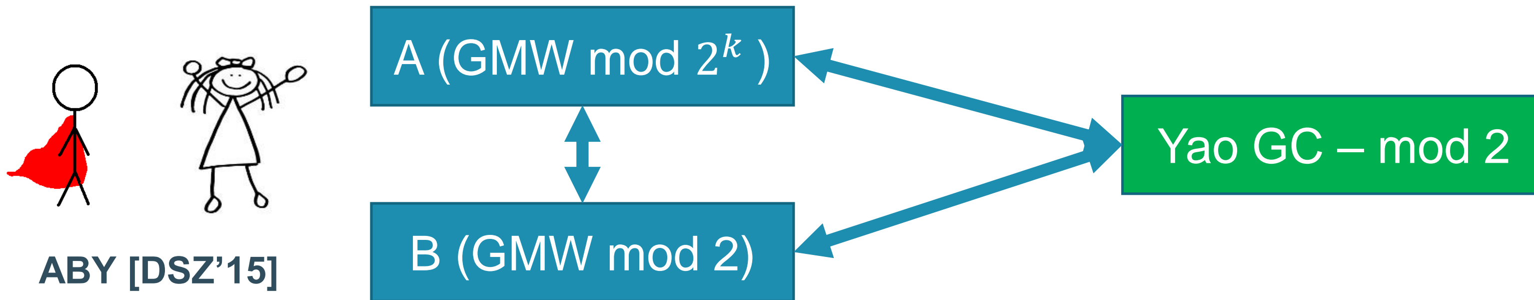
w_0	w_1	w_2
0	0	0
0	1	1
1	0	1
1	1	1

(a) Original Values

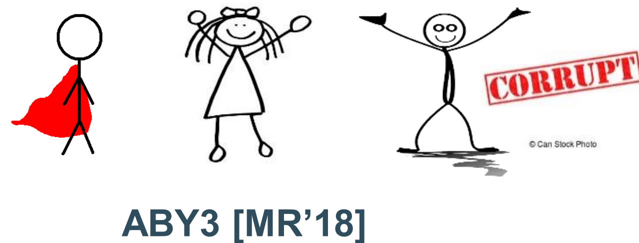
w_0	w_1	w_2	garbled value
k_0^0	k_1^0	k_2^0	$H(k_0^0 \| \| k_1^0 \| \| g_1) \oplus k_2^0$
k_0^0	k_1^1	k_2^1	$H(k_0^0 \| \| k_1^1 \| \| g_1) \oplus k_2^1$
k_0^1	k_1^0	k_2^1	$H(k_0^1 \| \| k_1^0 \| \| g_1) \oplus k_2^1$
k_0^1	k_1^1	k_2^1	$H(k_0^1 \| \| k_1^1 \| \| g_1) \oplus k_2^1$

(b) Garbled Values

Figure 2: Computation table for g_1^{OR}



Yao GC – mod 2



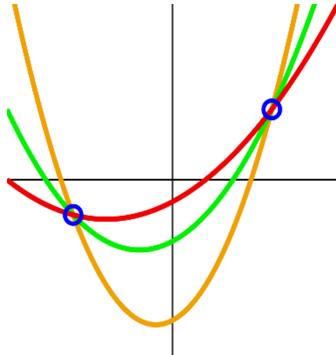
ABY3 [MR'18]

Dragoş Rotaru

imec-Cosic, Dept. Electrical Engineering

KU LEUVEN

What about dishonest majority?



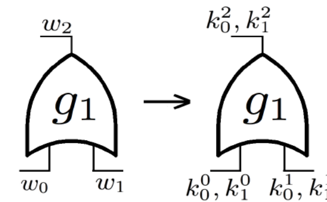


Figure 1: Garbling a single gate

w_0	w_1	w_2
0	0	0
0	1	1
1	0	1
1	1	1

(a) Original Values

w_0	w_1	w_2	garbled value
k_0^0	k_1^0	k_2^0	$H(k_0^0 \| k_1^0 \| g_1) \oplus k_2^0$
k_0^0	k_1^1	k_2^1	$H(k_0^0 \| k_1^1 \| g_1) \oplus k_2^1$
k_0^1	k_1^0	k_2^1	$H(k_0^1 \| k_1^0 \| g_1) \oplus k_2^1$
k_0^1	k_1^1	k_2^1	$H(k_0^1 \| k_1^1 \| g_1) \oplus k_2^1$

(b) Garbled Values

Figure 2: Computation table for g_1^{OR}

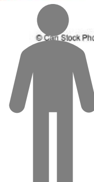
CORRUPT

© Can Stock Photo

CORRUPT

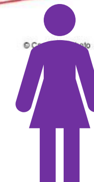
© Can Stock Photo

CORRUPT



© Can Stock Photo

CORRUPT

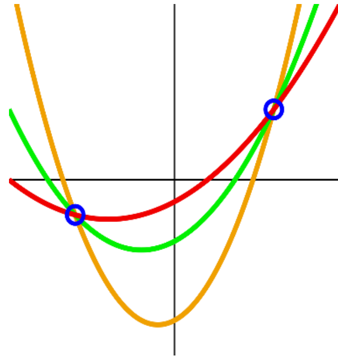


Dragoş Rotaru

imec-Cosic, Dept. Electrical Engineering

KU LEUVEN

What about dishonest majority?



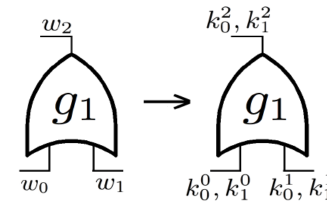


Figure 1: Garbling a single gate

w_0	w_1	w_2
0	0	0
0	1	1
1	0	1
1	1	1

(a) Original Values

w_0	w_1	w_2	garbled value
k_0^0	k_1^0	k_2^0	$H(k_0^0 \| k_1^0 \| g_1) \oplus k_2^0$
k_0^0	k_1^1	k_2^1	$H(k_0^0 \| k_1^1 \| g_1) \oplus k_2^1$
k_0^1	k_1^0	k_2^1	$H(k_0^1 \| k_1^0 \| g_1) \oplus k_2^1$
k_0^1	k_1^1	k_2^1	$H(k_0^1 \| k_1^1 \| g_1) \oplus k_2^1$

(b) Garbled Values

Figure 2: Computation table for g_1^{OR}

SPDZ

WRK'17

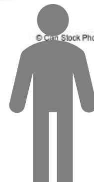
CORRUPT

© Can Stock Photo

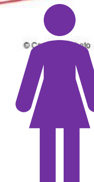
CORRUPT

© Can Stock Photo

CORRUPT



CORRUPT

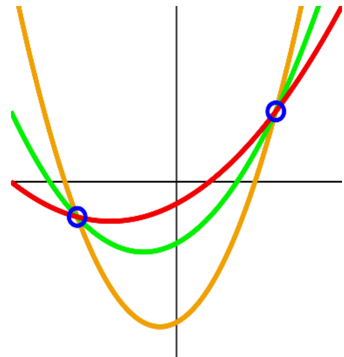


Dragoș Rotaru

imec-Cosic, Dept. Electrical Engineering

KU LEUVEN

What about dishonest majority?



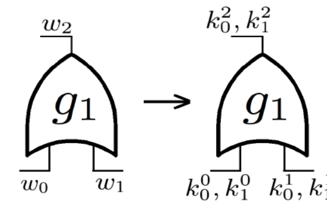


Figure 1: Garbling a single gate

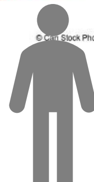
w_0	w_1	w_2
0	0	0
0	1	1
1	0	1
1	1	1

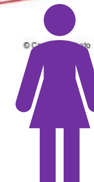
(a) Original Values

w_0	w_1	w_2	garbled value
k_0^0	k_1^0	k_2^0	$H(k_0^0 \| k_1^0 \| g_1) \oplus k_2^0$
k_0^0	k_1^1	k_2^1	$H(k_0^0 \| k_1^1 \| g_1) \oplus k_2^1$
k_0^1	k_1^0	k_2^1	$H(k_0^1 \| k_1^0 \| g_1) \oplus k_2^1$
k_0^1	k_1^1	k_2^1	$H(k_0^1 \| k_1^1 \| g_1) \oplus k_2^1$

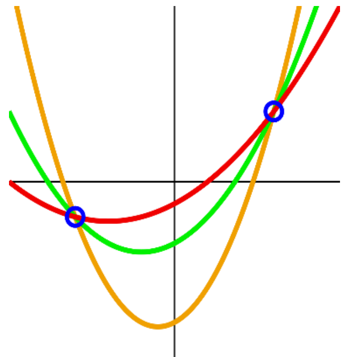
(b) Garbled Values

Figure 2: Computation table for g_1^{OR}





What about dishonest majority?



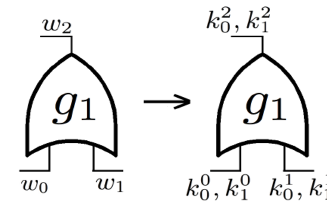


Figure 1: Garbling a single gate

w_0	w_1	w_2
0	0	0
0	1	1
1	0	1
1	1	1

(a) Original Values

w_0	w_1	w_2	garbled value
k_0^0	k_1^0	k_2^0	$H(k_0^0 \| k_1^0 \| g_1) \oplus k_2^0$
k_0^0	k_1^1	k_2^1	$H(k_0^0 \| k_1^1 \| g_1) \oplus k_2^1$
k_0^1	k_1^0	k_2^1	$H(k_0^1 \| k_1^0 \| g_1) \oplus k_2^1$
k_0^1	k_1^1	k_2^1	$H(k_0^1 \| k_1^1 \| g_1) \oplus k_2^1$

(b) Garbled Values

Figure 2: Computation table for g_1^{OR}

Naive

SPDZ

>110K ANDs

>110K ANDs

WRK'17

CORRUPT

CORRUPT

CORRUPT

CORRUPT

© Can Stock Photo

© Can Stock Photo

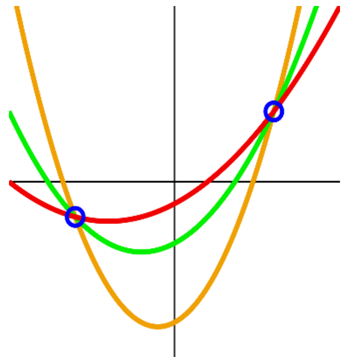
© Can Stock Photo

Dragoş Rotaru

imec-Cosic, Dept. Electrical Engineering

KU LEUVEN

What about dishonest majority?



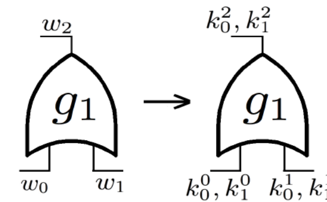


Figure 1: Garbling a single gate

w_0	w_1	w_2
0	0	0
0	1	1
1	0	1
1	1	1

(a) Original Values

w_0	w_1	w_2	garbled value
k_0^0	k_1^0	k_2^0	$H(k_0^0 \| \| k_1^0 \| \| g_1) \oplus k_2^0$
k_0^0	k_1^1	k_2^1	$H(k_0^0 \| \| k_1^1 \| \| g_1) \oplus k_2^1$
k_0^1	k_1^0	k_2^1	$H(k_0^1 \| \| k_1^0 \| \| g_1) \oplus k_2^1$
k_0^1	k_1^1	k_2^1	$H(k_0^1 \| \| k_1^1 \| \| g_1) \oplus k_2^1$

(b) Garbled Values

Figure 2: Computation table for g_1^{OR}

RW'19

<1K ANDs

SPDZ

WRK'17

0 ANDs

CORRUPT

© Can Stock Photo

CORRUPT

© Can Stock Photo

CORRUPT

© Can Stock Photo

CORRUPT

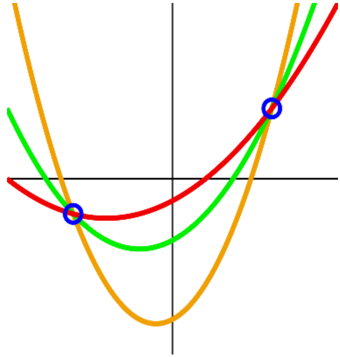
What we have done

State of affairs in RW'19 were unhappy (microbenchmarks) and expensive daBit generation using cut choose.

1. Cheaper daBit generation: 1 daBit = 1 random bit in SPDZ.
2. Complete online + preprocessing of WRK'17 and SPDZ in a single machine.
3. Extended compiler to support Z_{2^k} arithmetic using GC.

Document all the changes.

How general is this?



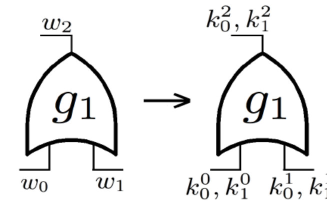


Figure 1: Garbling a single gate

w_0	w_1	w_2
0	0	0
0	1	1
1	0	1
1	1	1

(a) Original Values

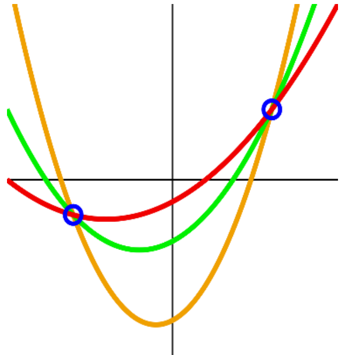
w_0	w_1	w_2	garbled value
k_0^0	k_1^0	k_2^0	$H(k_0^0 \| k_1^0 \| g_1) \oplus k_2^0$
k_0^0	k_1^1	k_2^1	$H(k_0^0 \| k_1^1 \| g_1) \oplus k_2^1$
k_0^1	k_1^0	k_2^1	$H(k_0^1 \| k_1^0 \| g_1) \oplus k_2^1$
k_0^1	k_1^1	k_2^1	$H(k_0^1 \| k_1^1 \| g_1) \oplus k_2^1$

(b) Garbled Values

Figure 2: Computation table for g_1^{OR}

Dragoș Rotaru
imec-Cosic, Dept. Electrical Engineering
KU LEUVEN

How general is this?



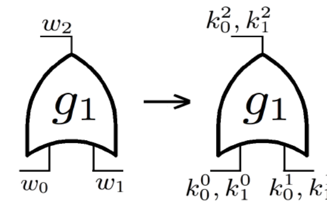


Figure 1: Garbling a single gate

w_0	w_1	w_2
0	0	0
0	1	1
1	0	1
1	1	1

(a) Original Values

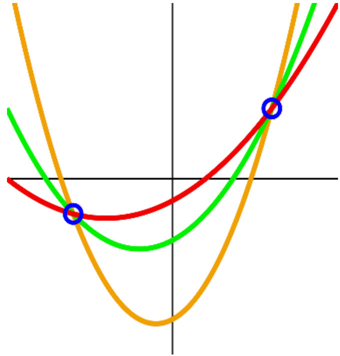
w_0	w_1	w_2	garbled value
k_0^0	k_1^0	k_2^0	$H(k_0^0 \| k_1^0 \| g_1) \oplus k_2^0$
k_0^0	k_1^1	k_2^1	$H(k_0^0 \| k_1^1 \| g_1) \oplus k_2^1$
k_0^1	k_1^0	k_2^1	$H(k_0^1 \| k_1^0 \| g_1) \oplus k_2^1$
k_0^1	k_1^1	k_2^1	$H(k_0^1 \| k_1^1 \| g_1) \oplus k_2^1$

(b) Garbled Values

Figure 2: Computation table for g_1^{OR}

Very fast using DEF+’19 (S&P’19) tricks

How general is this?



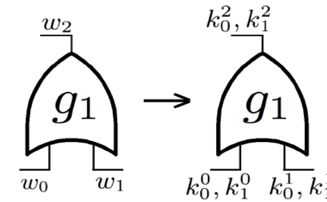


Figure 1: Garbling a single gate

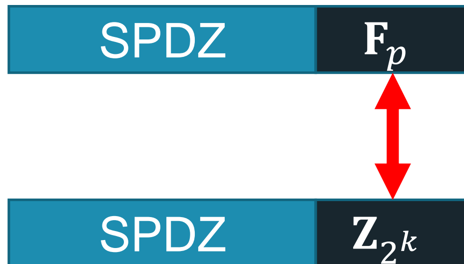
w_0	w_1	w_2
0	0	0
0	1	1
1	0	1
1	1	1

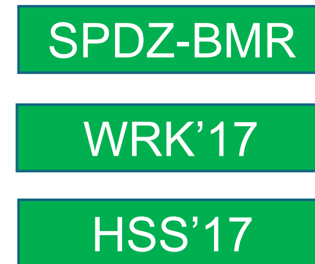
(a) Original Values

w_0	w_1	w_2	garbled value
k_0^0	k_1^0	k_2^0	$H(k_0^0 \| k_1^0 \| g_1) \oplus k_2^0$
k_0^0	k_1^1	k_2^1	$H(k_0^0 \| k_1^1 \| g_1) \oplus k_2^1$
k_0^1	k_1^0	k_2^1	$H(k_0^1 \| k_1^0 \| g_1) \oplus k_2^1$
k_0^1	k_1^1	k_2^1	$H(k_0^1 \| k_1^1 \| g_1) \oplus k_2^1$

(b) Garbled Values

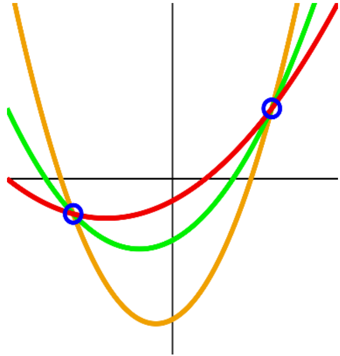
Figure 2: Computation table for g_1^{OR}





Dragoş Rotaru
imec-Cosic, Dept. Electrical Engineering
KU LEUVEN

How general is this?



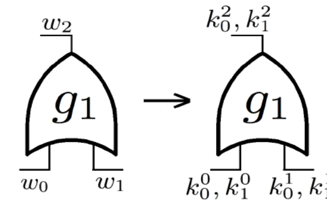


Figure 1: Garbling a single gate

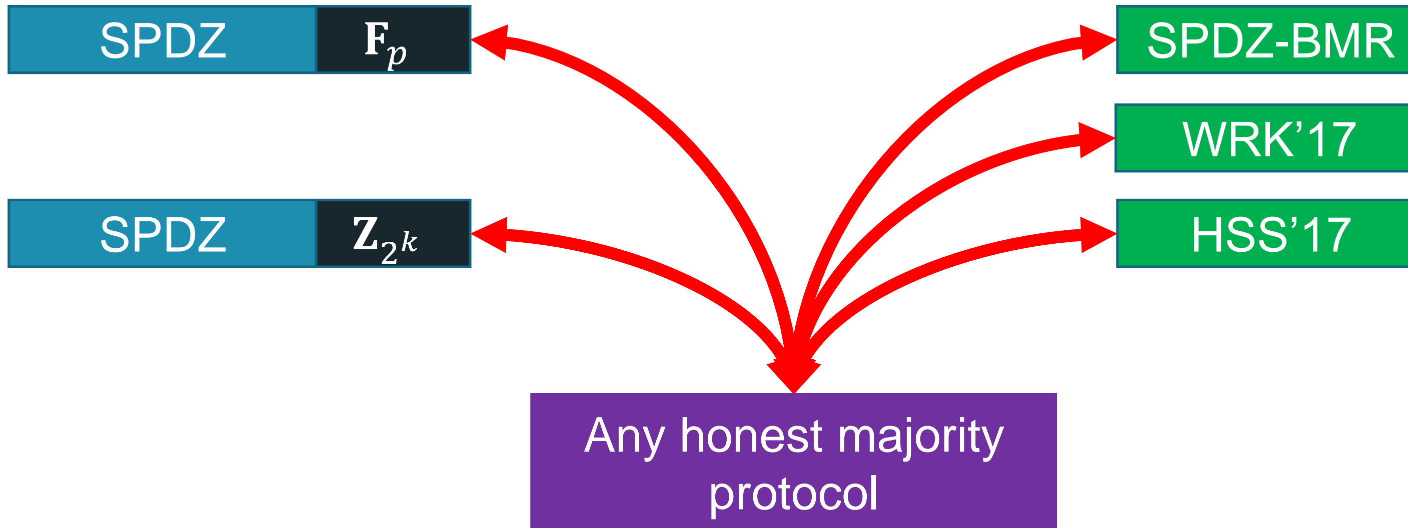
w_0	w_1	w_2
0	0	0
0	1	1
1	0	1
1	1	1

(a) Original Values

w_0	w_1	w_2	garbled value
k_0^0	k_1^0	k_2^0	$H(k_0^0 \| k_1^0 \| g_1) \oplus k_2^0$
k_0^0	k_1^1	k_2^1	$H(k_0^0 \| k_1^1 \| g_1) \oplus k_2^1$
k_0^1	k_1^0	k_2^1	$H(k_0^1 \| k_1^0 \| g_1) \oplus k_2^1$
k_0^1	k_1^1	k_2^1	$H(k_0^1 \| k_1^1 \| g_1) \oplus k_2^1$

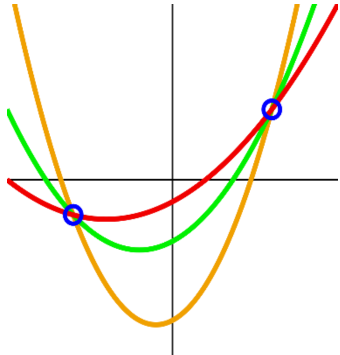
(b) Garbled Values

Figure 2: Computation table for g_1^{OR}



Dragoș Rotaru
imec-Cosic, Dept. Electrical Engineering
KU LEUVEN

Our focus



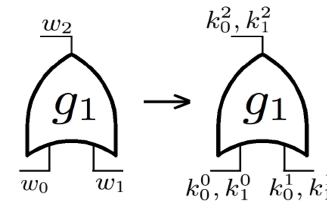


Figure 1: Garbling a single gate

w_0	w_1	w_2
0	0	0
0	1	1
1	0	1
1	1	1

(a) Original Values

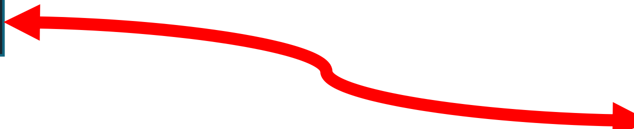
w_0	w_1	w_2	garbled value
k_0^0	k_1^0	k_2^0	$H(k_0^0 \| k_1^0 \| g_1) \oplus k_2^0$
k_0^0	k_1^1	k_2^1	$H(k_0^0 \| k_1^1 \| g_1) \oplus k_2^1$
k_0^1	k_1^0	k_2^1	$H(k_0^1 \| k_1^0 \| g_1) \oplus k_2^1$
k_0^1	k_1^1	k_2^1	$H(k_0^1 \| k_1^1 \| g_1) \oplus k_2^1$

(b) Garbled Values

Figure 2: Computation table for g_1^{OR}

SPDZ | $\mathbf{F}_p$

SPDZ | $\mathbf{Z}_{2^k}$



SPDZ-BMR

WRK'17

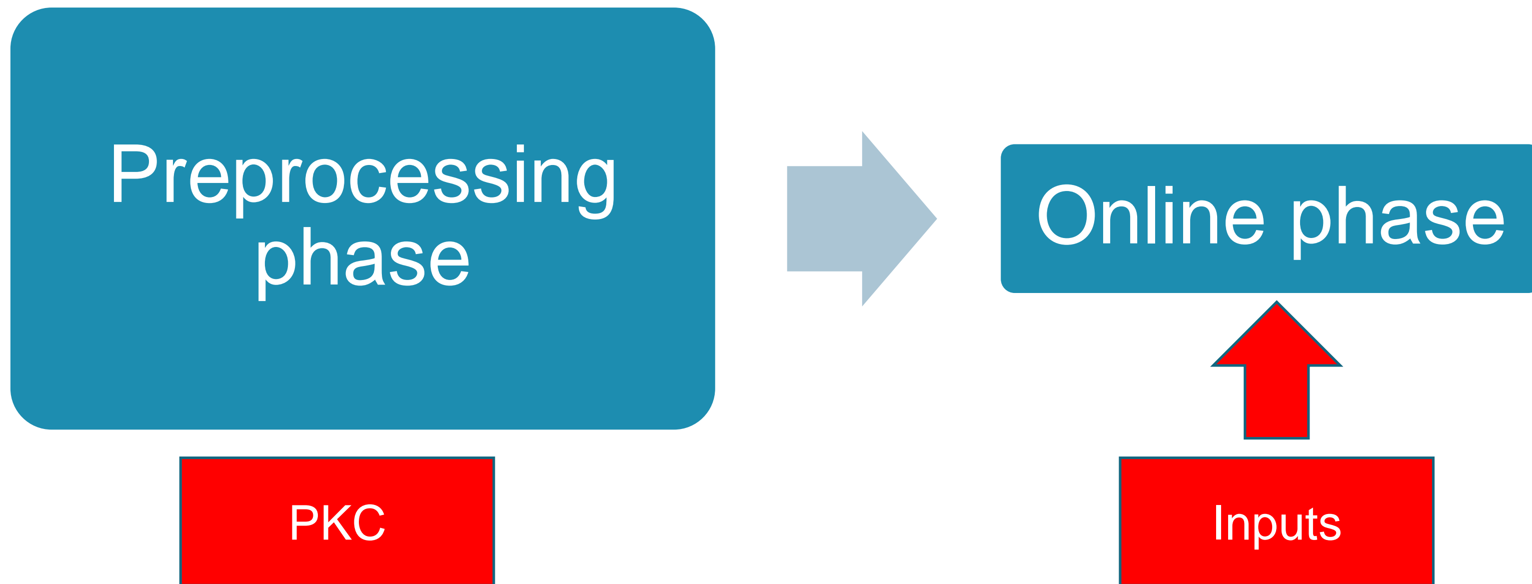
HSS'17

Dragoș Rotaru

imec-Cosic, Dept. Electrical Engineering

KU LEUVEN

Malicious MPC protocols

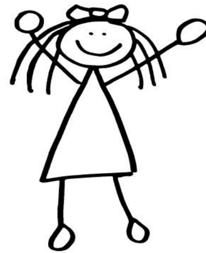


SPDZ, TinyOT, BDOZa, MASCOT, WRK’17, HSS’17, …

Let's talk about

SPDZ $\mathbf{F}_p$

SPDZ $\mathbf{F}_p$ online phase

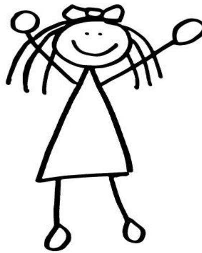


$$\alpha_1 + \alpha_2 + \alpha_3 = \alpha$$

$$x_1 + x_2 + x_3 = x$$

$$\gamma(x)_1 + \gamma(x)_2 + \gamma(x)_3 = \alpha x$$

SPDZ $\mathbf{F}_p$ online phase

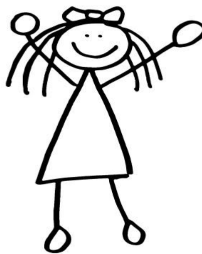


$$\alpha_1 + \alpha_2 + \alpha_3 = \alpha$$

$$x_1 + y_1 + x_2 + y_2 + x_3 + y_3 = x + y$$

$$\gamma(x)_1 + \gamma(y)_1 + \gamma(x)_2 + \gamma(y)_2 + \gamma(x)_3 + \gamma(y)_3 = \alpha(x + y)$$

SPDZ $\mathbf{F}_p$ online phase



Input

Retrieve a random mask

X_A X_A

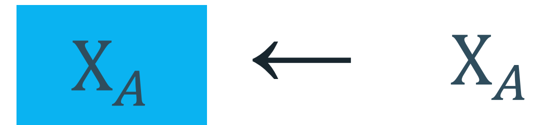
SPDZ $\mathbf{F}_p$ online phase

Input

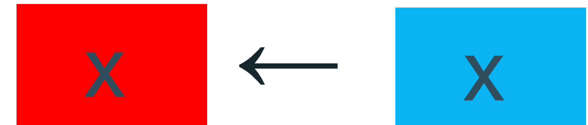
X_A $\leftarrow$ X_A

SPDZ $\mathbf{F}_p$ online phase

Input



Open



Dragoș Rotaru
imec-Cosic, Dept. Electrical Engineering
KU LEUVEN

SPDZ $\mathbf{F}_p$ online phase

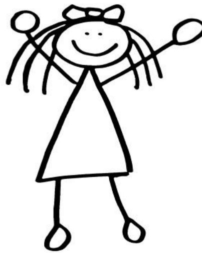
Input

X_A

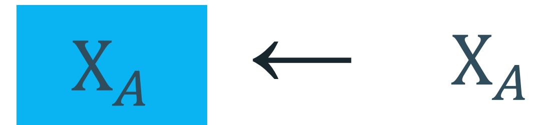
Open

MAC Check

SPDZ $\mathbf{F}_p$ online phase



Input



Open

Multiply Retrieve a Beaver triple

Dragoș Rotaru
imec-Cosic, Dept. Electrical Engineering
KU LEUVEN

SPDZ $\mathbf{F}_p$ online phase

Input			X_A	←	X_A	
Open	MAC Check		x	←	x	
Multiply		z ←	x	×	y	

Dragoș Rotaru
imec-Cosic, Dept. Electrical Engineering
KU LEUVEN

Let's talk about

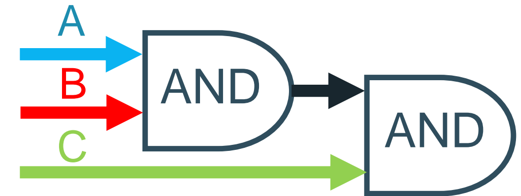
WRK'17	$\mathbf{F}_2$

Dragoș Rotaru

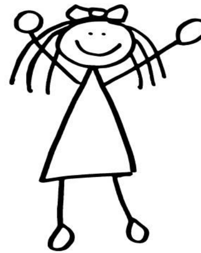
imec-Cosic, Dept. Electrical Engineering

KU LEUVEN

WRK'17 F_2 online phase



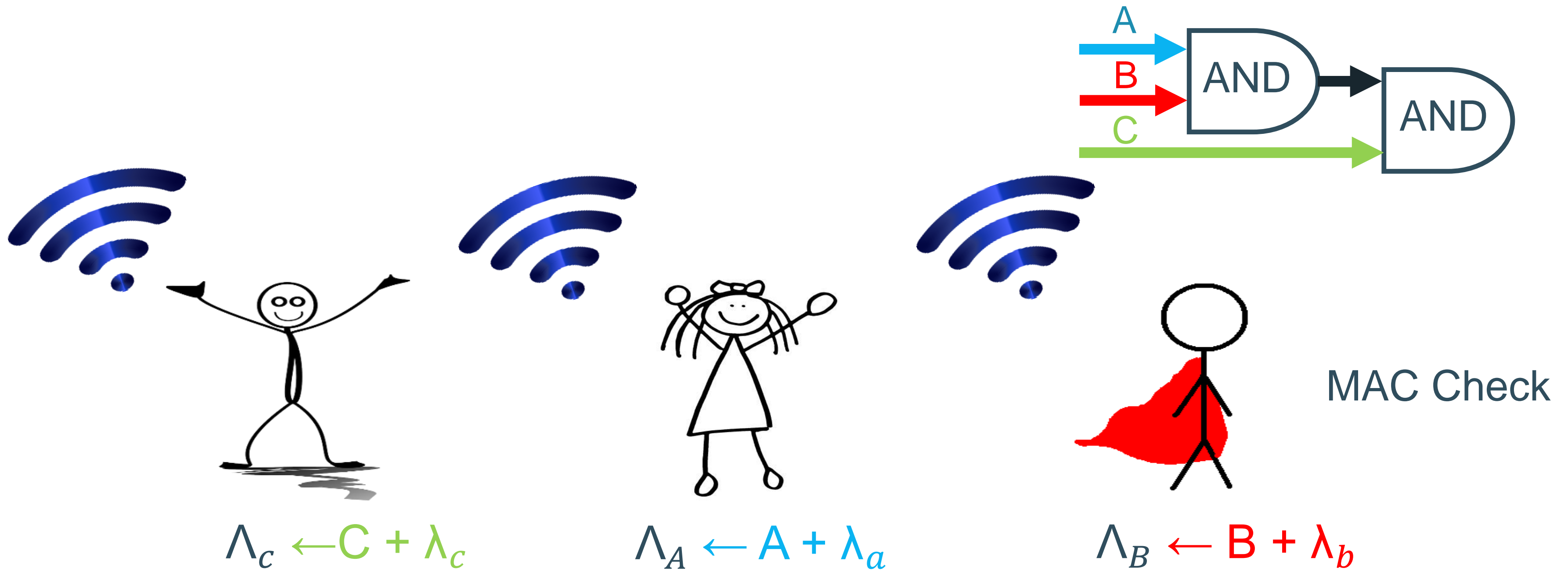
C



A

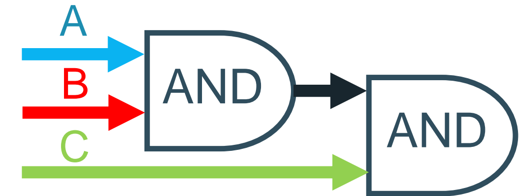
B

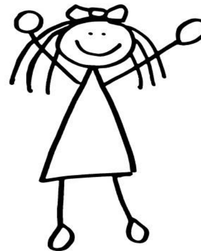
WRK'17 $\mathbb{F}_2$ online phase



Dragoș Rotaru
imec-Cosic, Dept. Electrical Engineering
KU LEUVEN

WRK'17 | $\mathbb{F}_2$ online phase





MAC Check

$\Lambda_c \leftarrow C + \lambda_c$ $\Lambda_A \leftarrow A + \lambda_a$ $\Lambda_B \leftarrow B + \lambda_b$

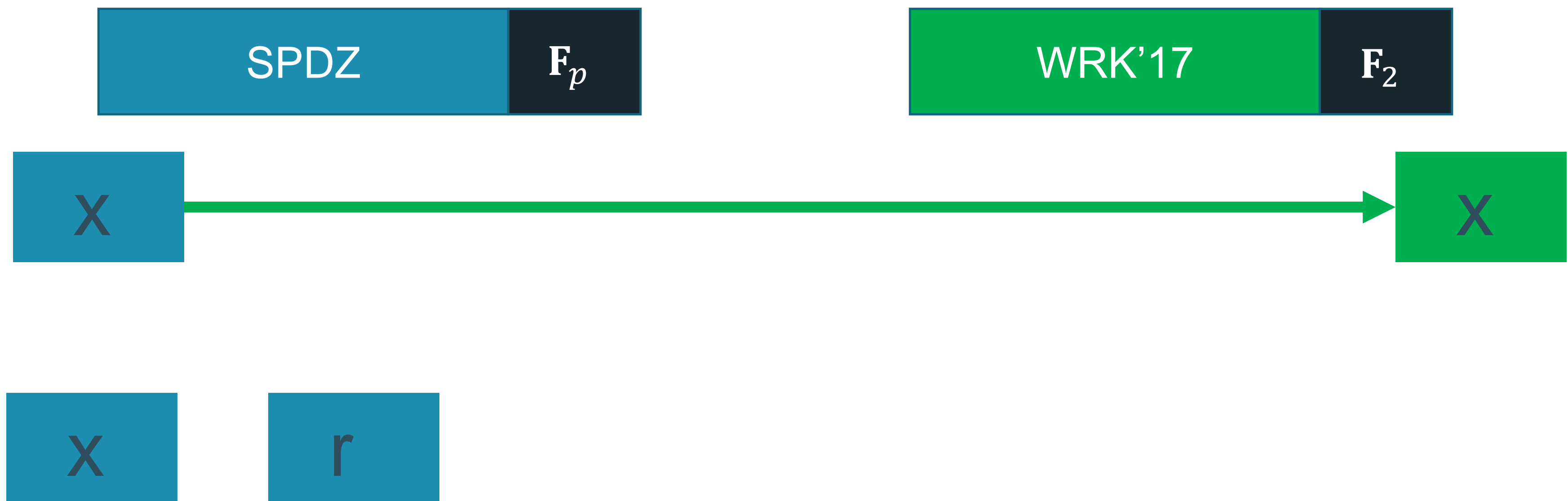
Inputs - cheap

XOR - free

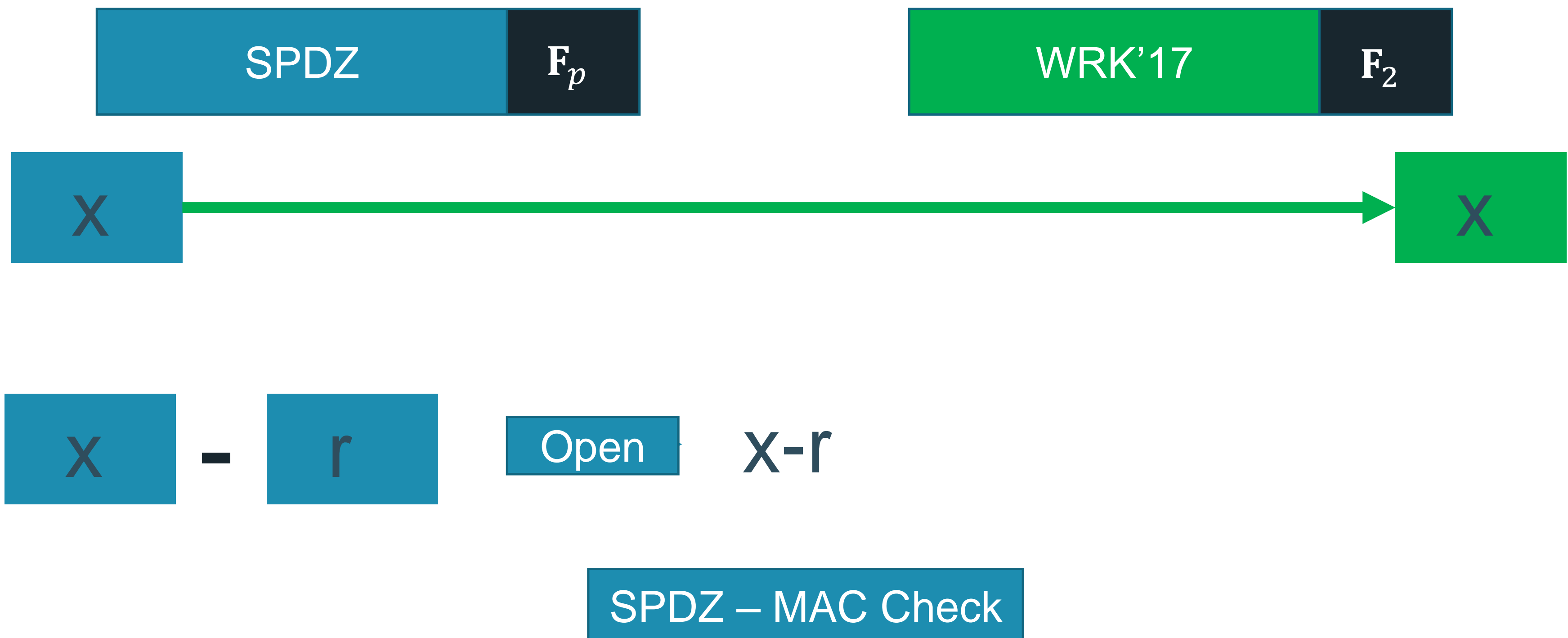
Mod p arithmetic - some AND gates

Main idea:

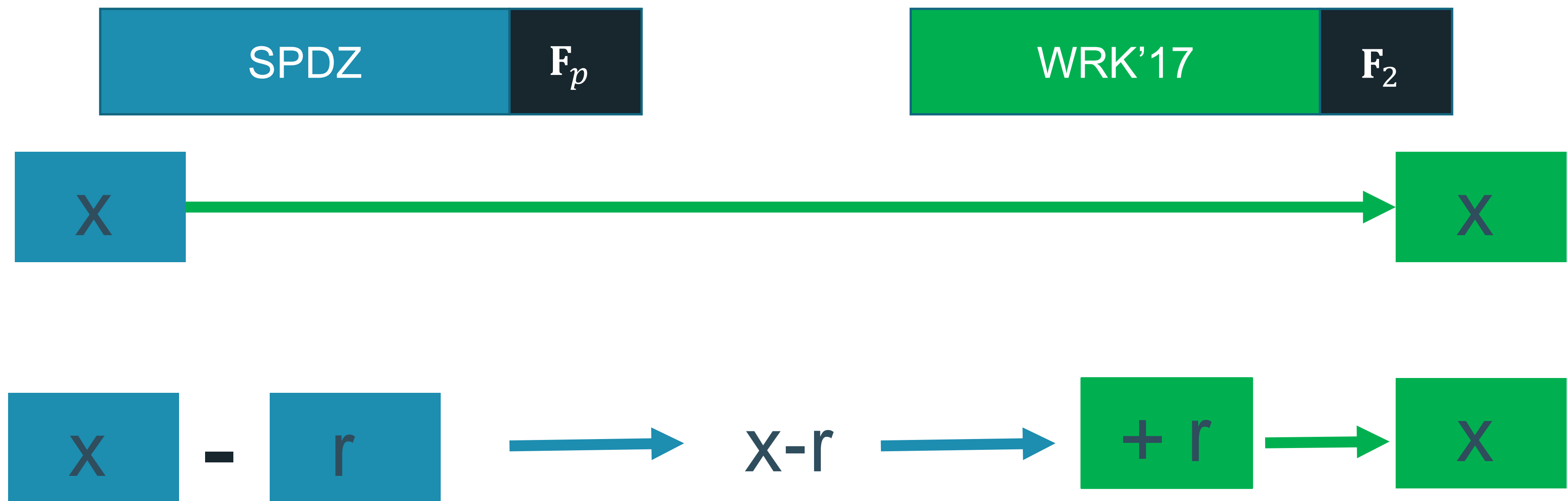
Main idea:



Main idea:



Main idea:



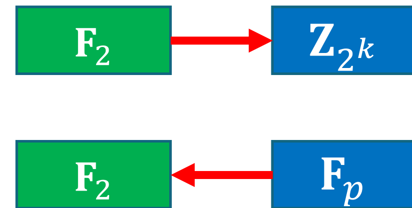
Introducing daBits 2.0

daBit 2.0

- Inspired from **DEFKSV'19**

daBit 2.0

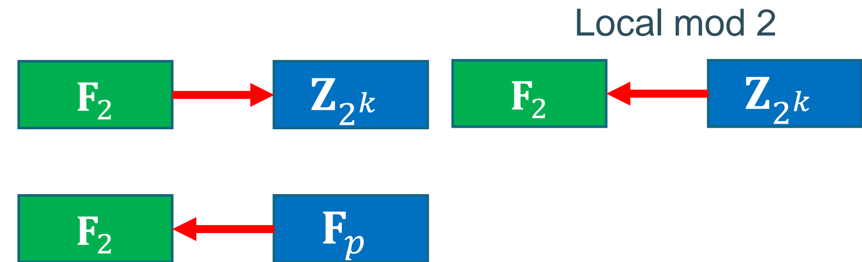
- Inspired from **DEFKSV'19**



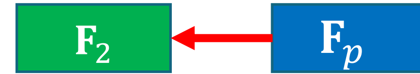
Dragoş Rotaru
imec-Cosic, Dept. Electrical Engineering
KU LEUVEN

daBit 2.0

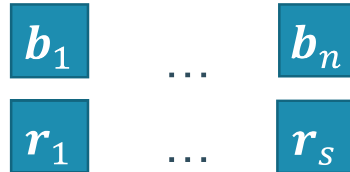
- Inspired from **DEFKSV'19**



daBit 2.0

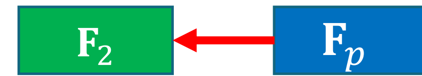


SPDZ[p].Random()

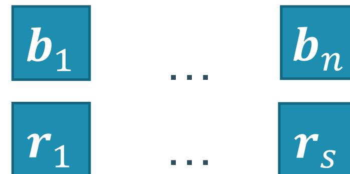


TinyOT.Input()

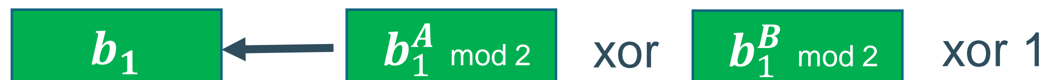
daBit 2.0



SPDZ[p].Random()

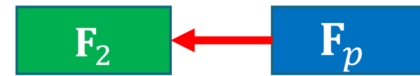


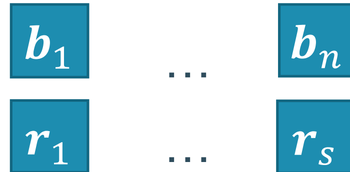
TinyOT.Input()



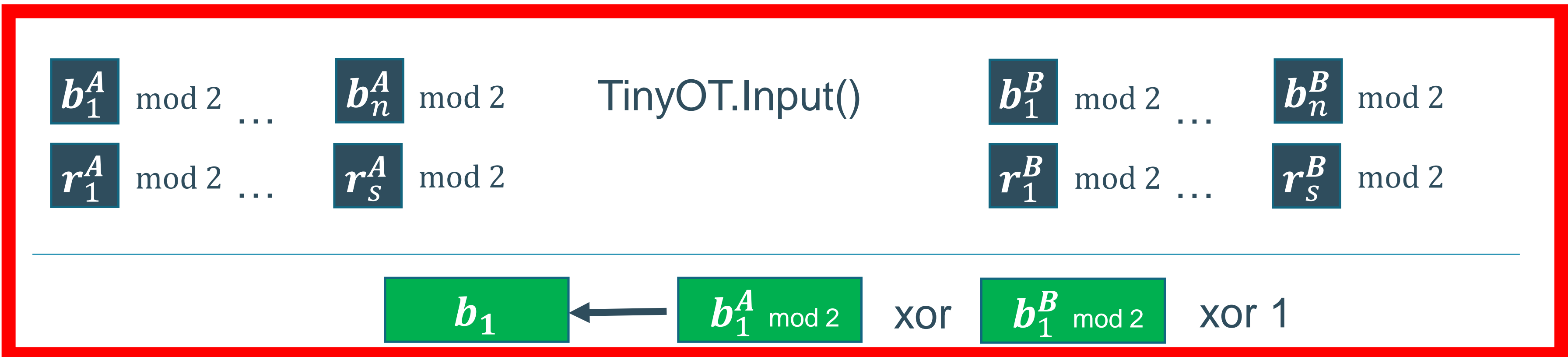
Take s linear combinations

daBit 2.0





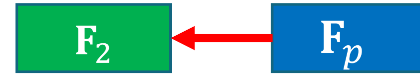
N-party case more tricky



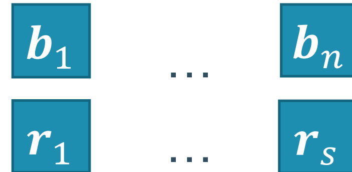
Take s linear combinations

Dragoș Rotaru
imec-Cosic, Dept. Electrical Engineering
KU LEUVEN

daBit 2.0



SPDZ[p].Random()

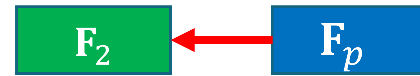


Take s linear combinations

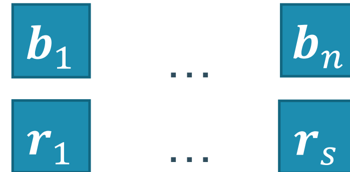
$r_i + \alpha_1 \times b_1 \ \ldots \ \alpha_n \times b_n$ mod p

$r_i + \alpha_1 \times b_1 \ \ldots \ \alpha_n \times b_n$ mod 2

daBit 2.0



SPDZ[p].Random()



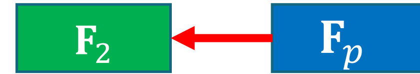
Take s linear combinations

$$\mathrm{LSB}(r_i + \alpha_1 \times b_1 \dots \alpha_n \times b_n) \quad \text{mod p}$$

$$\stackrel{?}{=}$$

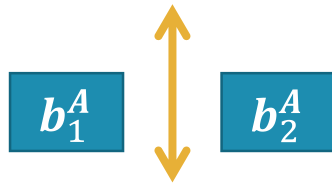
$$r_i + \alpha_1 \times b_1 \dots \alpha_n \times b_n \quad \text{mod 2}$$

daBit 2.0

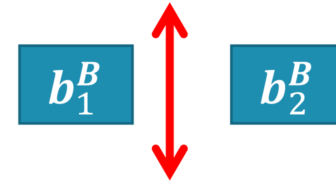


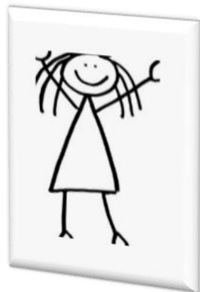
SPDZ[p].Random()

b_1

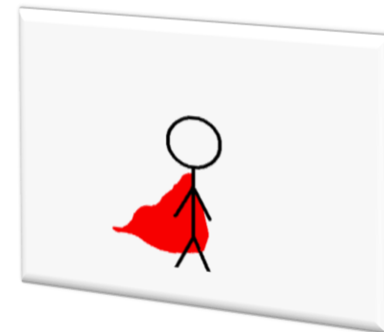


XOR

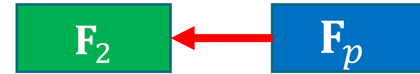




b_2

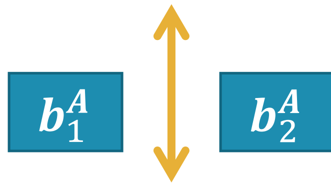


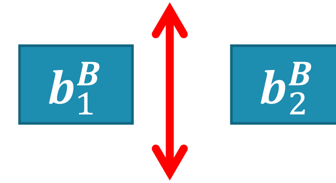
daBit 2.0

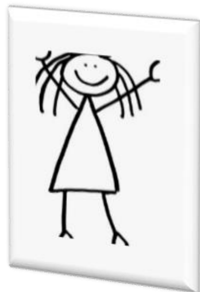


SPDZ[p].Random()

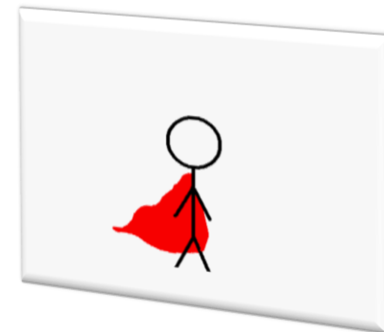
b_1





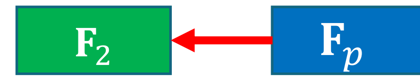


b_2

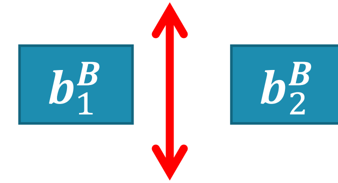


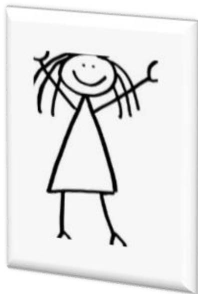
Dragoş Rotaru
imec-Cosic, Dept. Electrical Engineering
KU LEUVEN

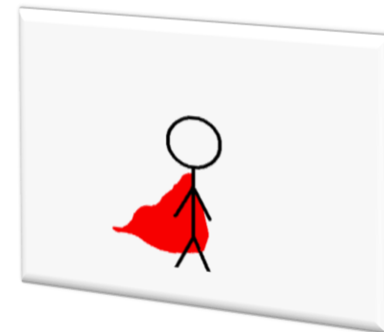
daBit 2.0



SPDZ[p].Random()



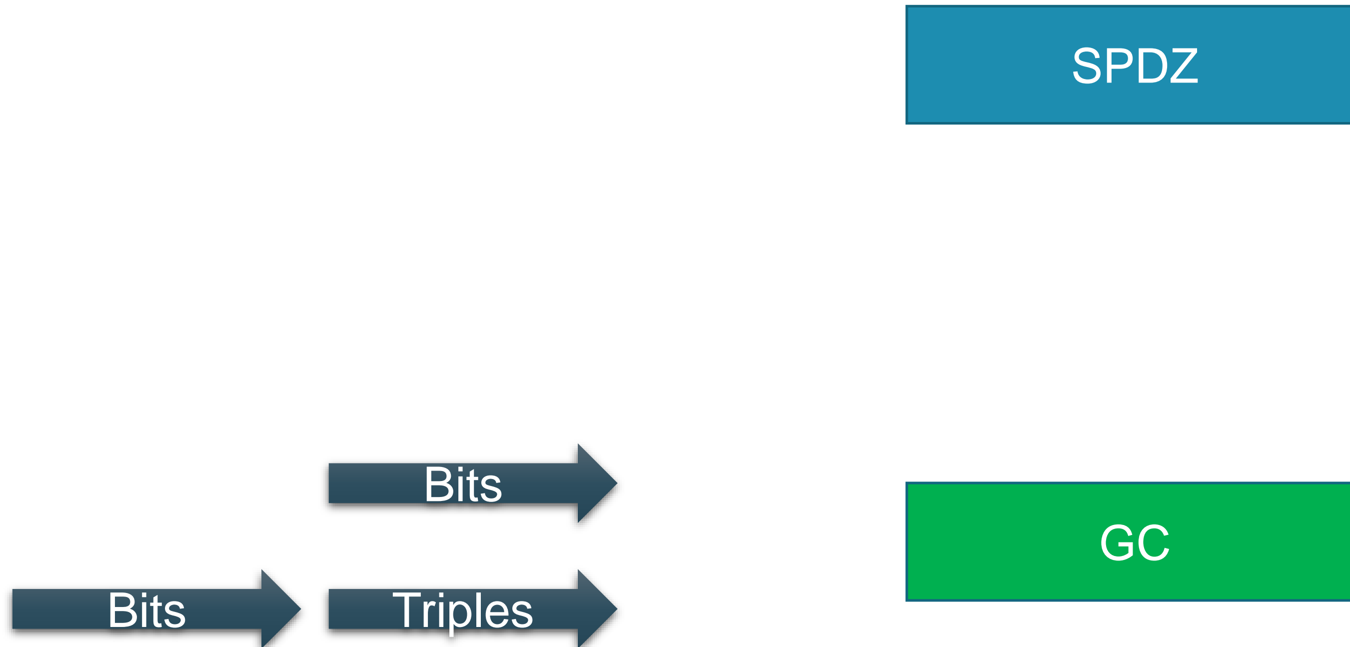




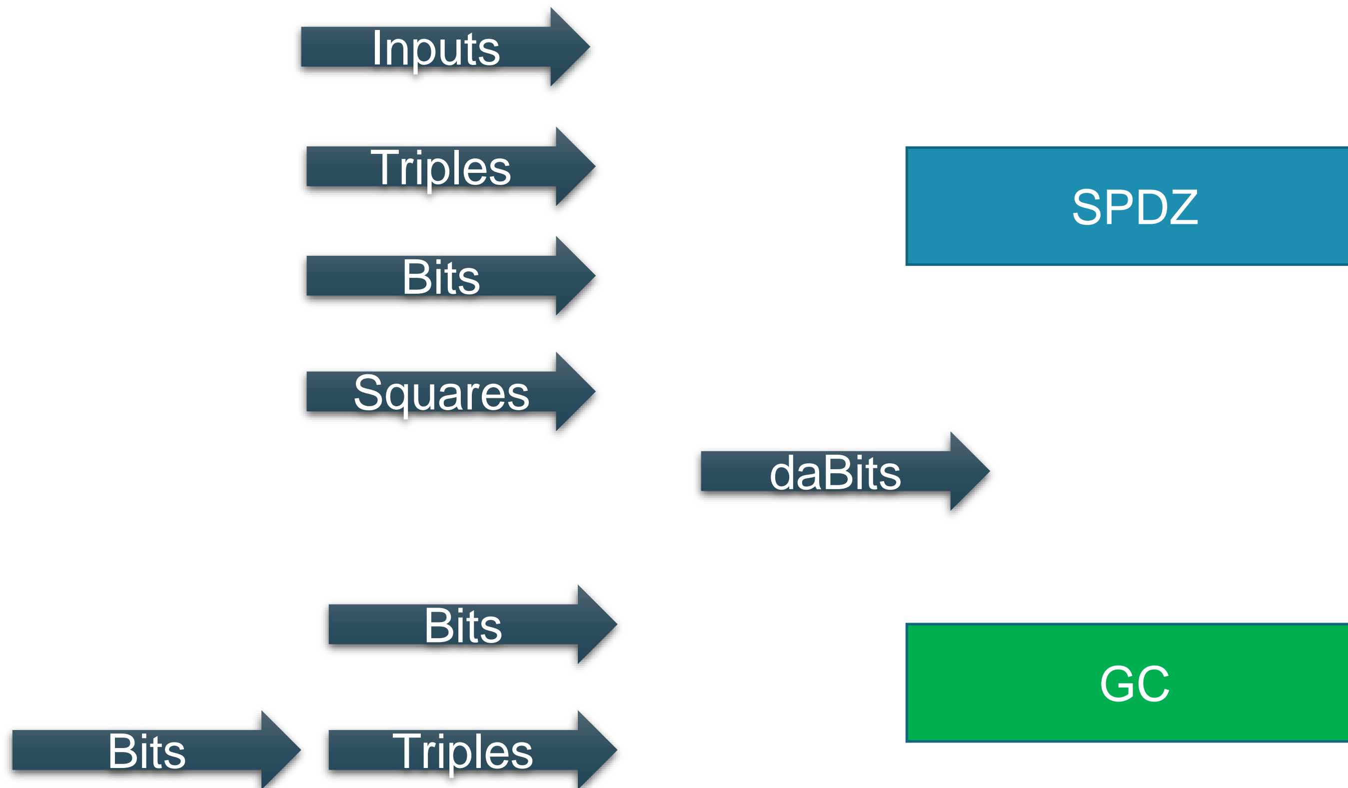
daBit production

Protocol	n	sec	Comm. (kb)	Throughput (ops/s)
daBit [RW19]	2	40	384	1008
dabit (ours)	2	40	94	2150
daBit [RW19]	3	40	1640	560
dabit (ours)	3	40	1104	650
daBit [RW19]	4	40	4781	306
dabit (ours)	4	40	2173	552

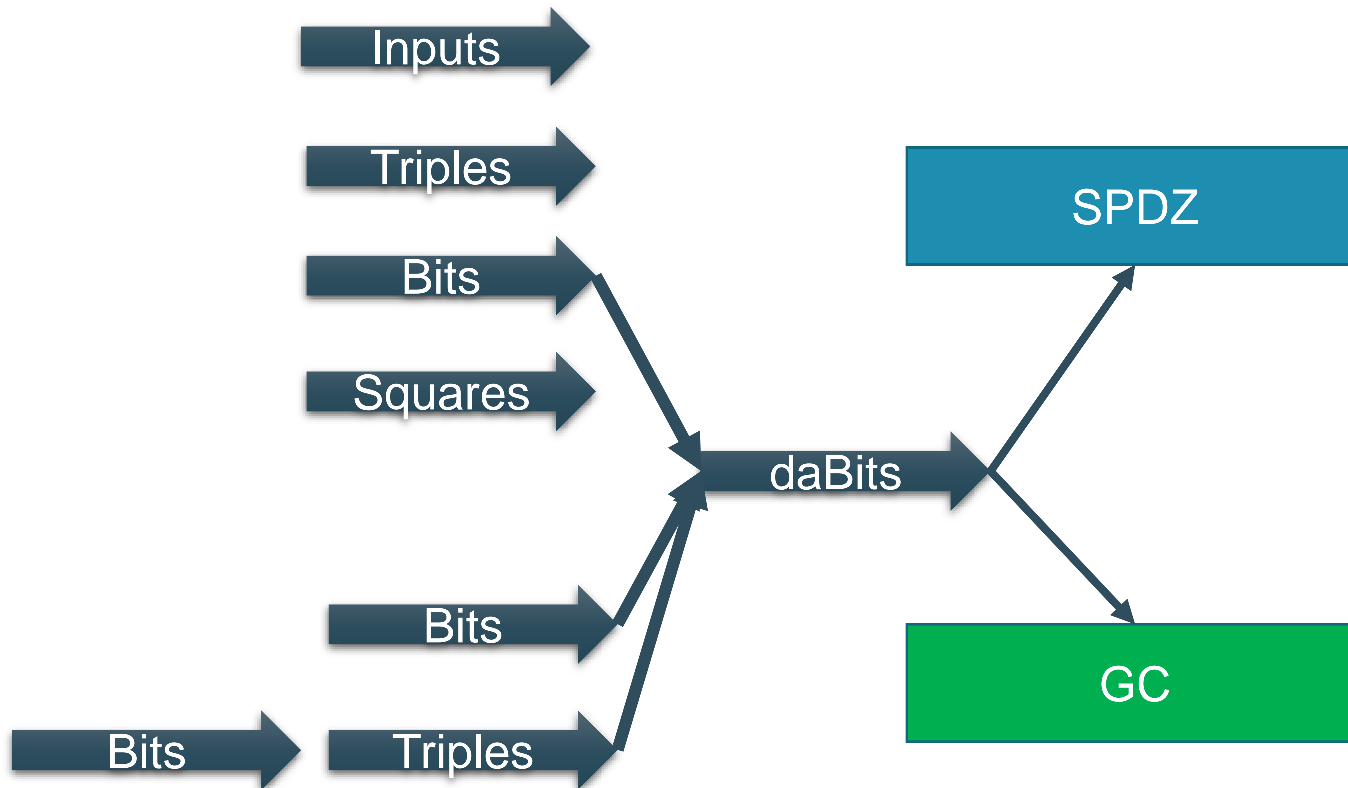
Full integration is challenging



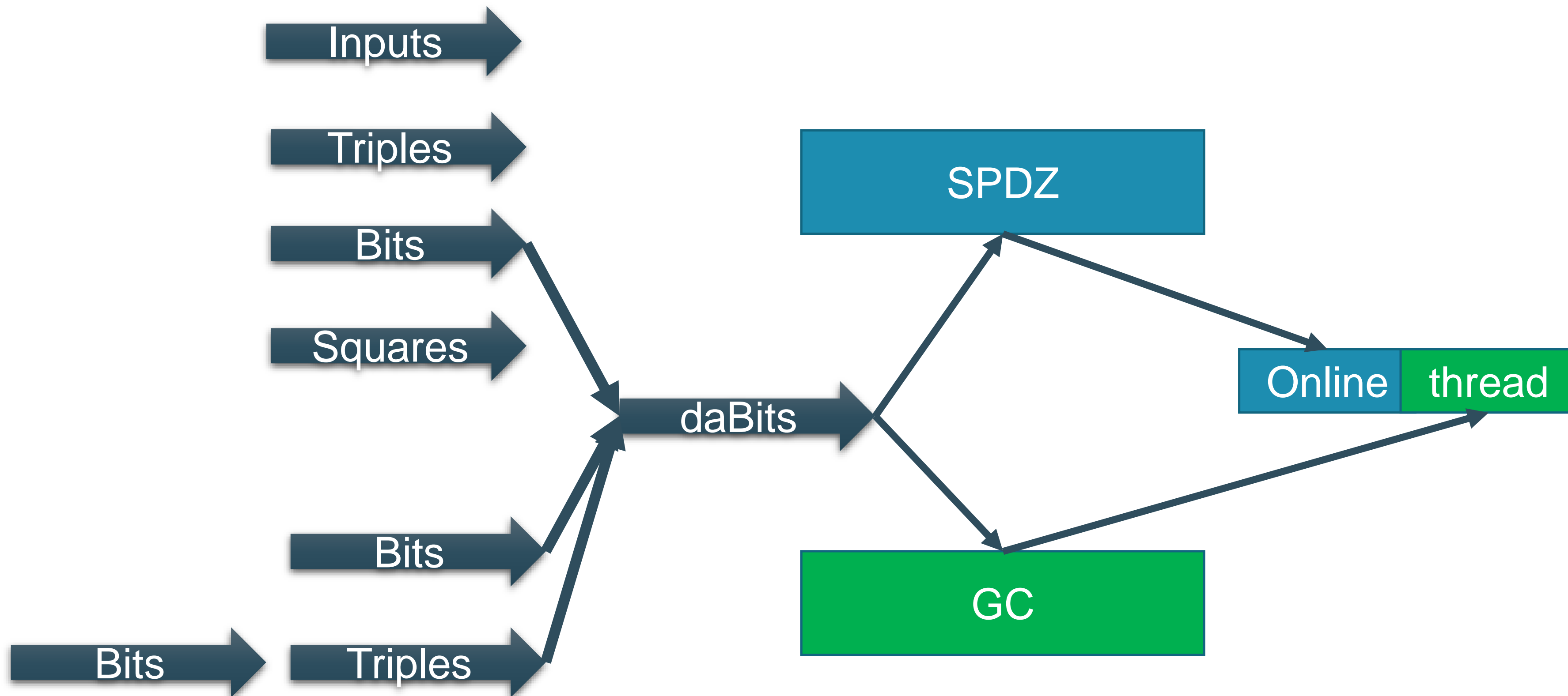
Full integration is challenging



Full integration is challenging



Full integration is challenging



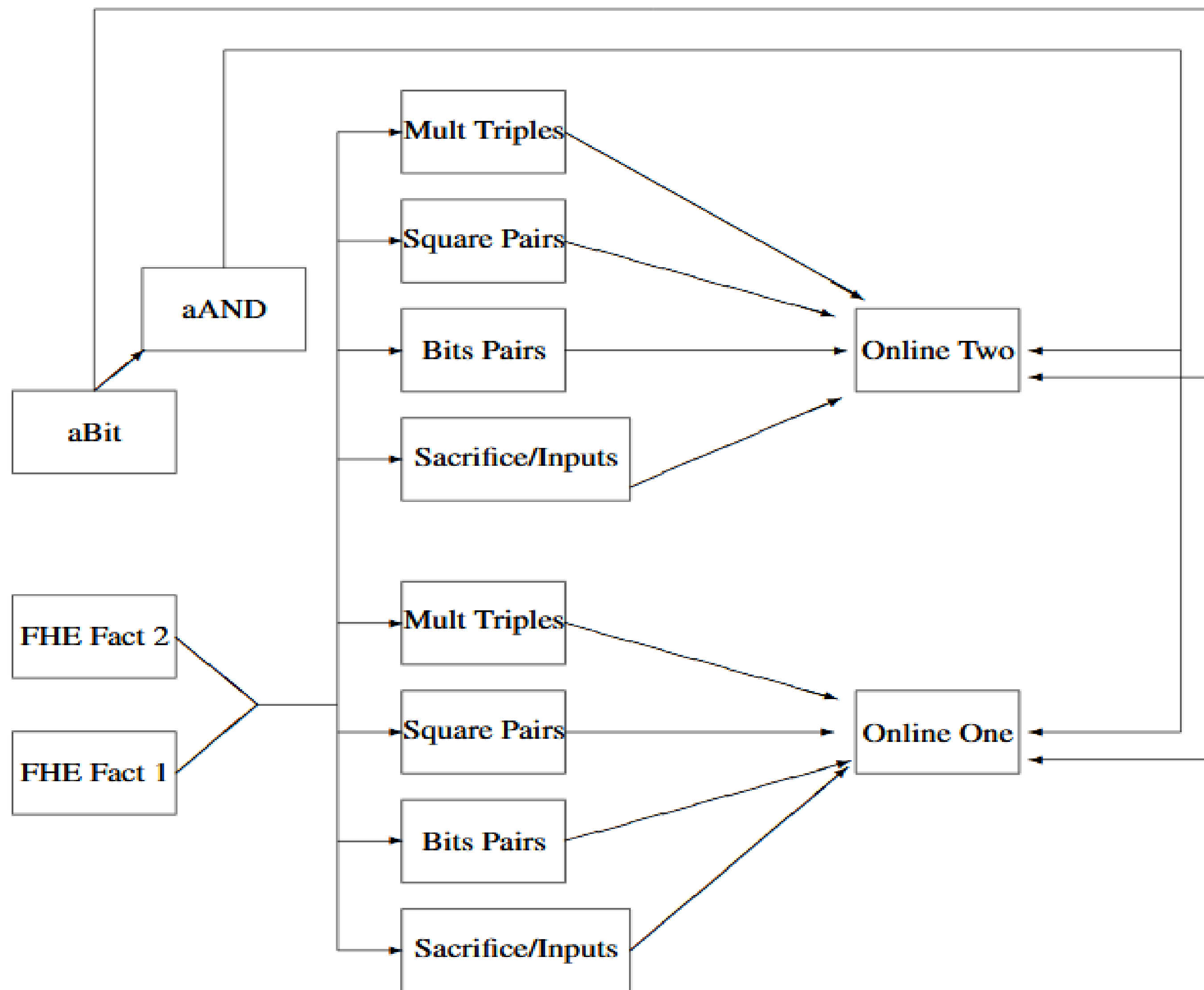


Figure 1: Pictorial View of a Players Threads: With Two Online Threads and Two FHE Factory Threads

Conclusions and future work

- Can we generate daBits faster? Answer is yes, stay tuned.
- More interesting examples where these conversions are good will come soon…

Thank you!

- Questions?
- **https://ia.cr/2019/974**