Higher-Order Glitches Free Implementation of the AES using Secure Multi-Party Computation Protocols

Thomas Roche and Emmanuel Prouff

ANSSI, France — Oberthur Technologies, France

CHES'11, Nara, Japan
26 December, 2011

Overview

Introduction

Framework to prove Higher-Order SCA Resistance and Glitches freeness

From BGW's protocol to secure masking scheme w.r.t. HO-SCA in presence of glitches

Conclusions and Future Directions

SCA Attacks

x, y are sensitive variables: dependent on the secret and on a known value.

SCA Attacks

x, y are sensitive variables: dependent on the secret and on a known value.

SCA attacks

- DPA [Kocher 98]
- CPA [Brier *et al.* 04]
- MIA, Stochastic, ...

1^{st}-order Masking Schemes

Masking/Sharing Function

$$
\begin{aligned}
x_0 &\leftarrow RNG \\
x_1 &\leftarrow x_0 \oplus x
\end{aligned}
$$

$\Pr(x|x_i) = \Pr(x)$ $\Pr(y|y_i) = \Pr(y)$

1^{st}-order Masking Schemes

Masking/Sharing Function

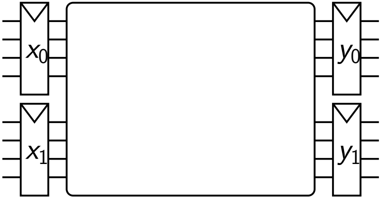
$$\begin{array}{lcl} x_0 & \leftarrow & RNG \\ x_1 & \leftarrow & x_0 \oplus x \end{array}$$

Works well for Homomorphic functions (*w.r.t.* $\oplus$).

$\Pr(x|x_i) = \Pr(x)$ $\Pr(y|y_i) = \Pr(y)$

1^{st}-order Masking Schemes

Non-Homomorphic functions:
combinations of the inputs are necessary.

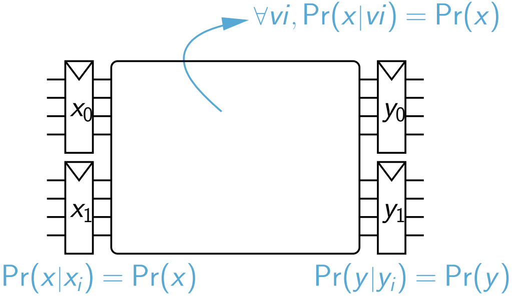


$\Pr(x|x_i) = \Pr(x)$ $\Pr(y|y_i) = \Pr(y)$

1st-order Masking Schemes

Non-Homomorphic functions:

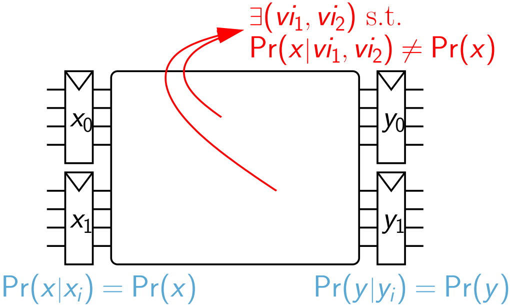
- Table re-computation methods [Kocher *et al.* 99]
- Tower Field computations (AES) [Oswald *et al.* 05]
- ...



1^{st}-order Masking Schemes

Non-Homomorphic functions:

- Table re-computation methods [Kocher *et al.* 99]
- Tower Field computations (AES) [Oswald *et al.* 05]
- ...



2^{nd}-order SCA attacks [Messerges *et al.* 00, ...]

HO-SCA Masking Schemes

dth-order schemes

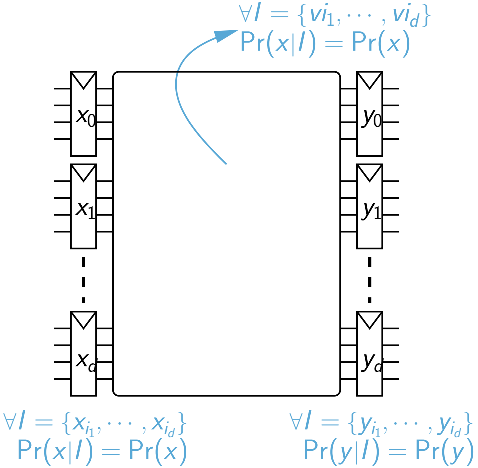
HW [Ishai *et al.* 03]

SW [Rivain *et al.* 10, Faust *et al.* 10, Genelle *et al.* 11]

Relaxed SMC Protocol

Probing Model

Here the order d relates to the # of observed data, w/o notion of time or space location.



HO-SCA Masking Schemes

dth-order schemes

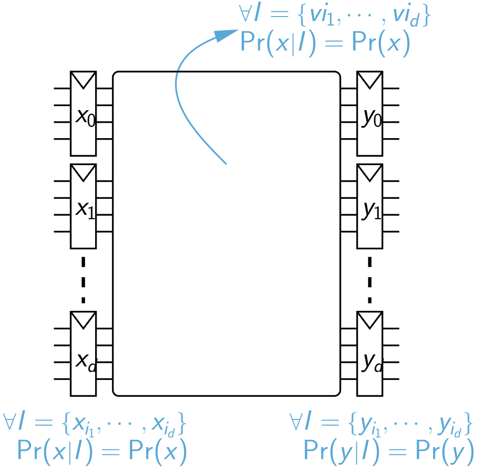
HW [Ishai *et al.* 03]

SW [Rivain *et al.* 10, Faust *et al.* 10, Genelle *et al.* 11]

Relaxed SMC Protocol

\# of shares

$$n = d + 1$$



HO-SCA Masking Schemes

dth-order schemes

HW [Ishai *et al.* 03]

SW [Rivain *et al.* 10, Faust *et al.* 10, Genelle *et al.* 11]

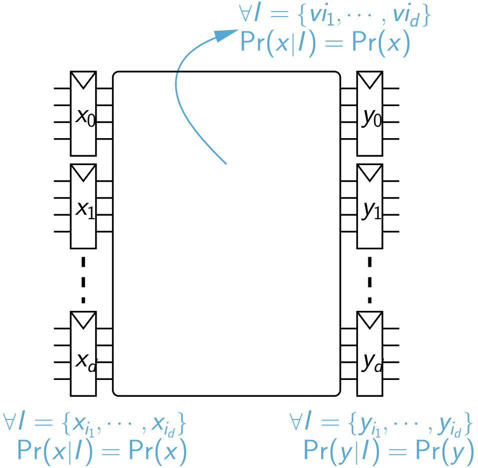
Relaxed SMC Protocol

\# of shares

$$n = d + 1$$

Soundness [Chari *et al.*]

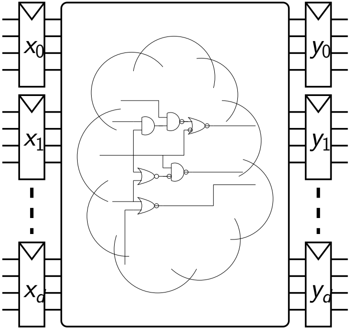
Complexity: $O(\sigma^d)$



Glitches Attacks

Transition Energy in a clock cycle

E_T



$\forall I = \{x_{i_1}, \cdots, x_{i_d}\}$
$\Pr(x|I) = \Pr(x)$

$\forall I = \{y_{i_1}, \cdots, y_{i_d}\}$
$\Pr(y|I) = \Pr(y)$

Glitches Attacks

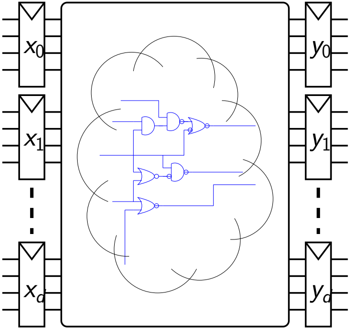
Transition Energy in a clock cycle

E_T

Idealized model

Each gate switches only once.
$\hookrightarrow \Pr(x|E_T^{\text{Ideal}}) = \Pr(x)$

E_T^{Ideal}



$\forall I = \{x_{i_1}, \cdots, x_{i_d}\}$
$\Pr(x|I) = \Pr(x)$

$\forall I = \{y_{i_1}, \cdots, y_{i_d}\}$
$\Pr(y|I) = \Pr(y)$

Glitches Attacks

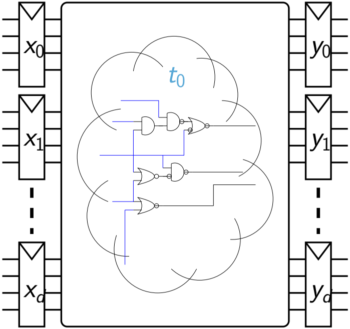
Transition Energy in a clock cycle

E_T

More realistic model

Propagation delays.

$\hookrightarrow E_T \neq E_T^{Ideal}$



$\forall I = \{x_{i_1}, \cdots, x_{i_d}\}$
$\Pr(x|I) = \Pr(x)$

$\forall I = \{y_{i_1}, \cdots, y_{i_d}\}$
$\Pr(y|I) = \Pr(y)$

Glitches Attacks

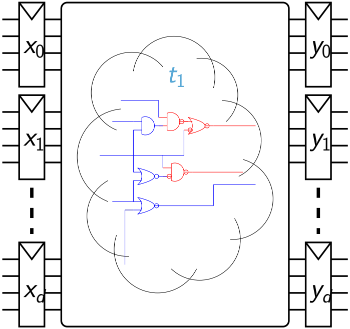
Transition Energy in a clock cycle

E_T

More realistic model

Propagation delays.

$\hookrightarrow E_T \neq E_T^{Ideal}$



$\forall I = \{x_{i_1}, \cdots, x_{i_d}\}$ $\Pr(x|I) = \Pr(x)$

$\forall I = \{y_{i_1}, \cdots, y_{i_d}\}$ $\Pr(y|I) = \Pr(y)$

Glitches Attacks

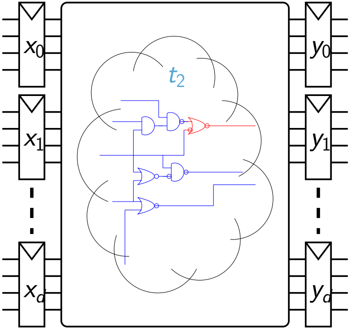
Transition Energy in a clock cycle

E_T

More realistic model

Propagation delays.

$\hookrightarrow E_T \neq E_T^{Ideal}$



$\forall I = \{x_{i_1}, \cdots, x_{i_d}\}$
$\Pr(x|I) = \Pr(x)$

$\forall I = \{y_{i_1}, \cdots, y_{i_d}\}$
$\Pr(y|I) = \Pr(y)$

Glitches Attacks

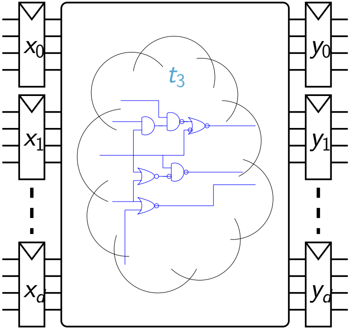
Transition Energy in a clock cycle

E_T

More realistic model

Propagation delays.

$\hookrightarrow E_T \neq E_T^{Ideal}$



$\forall I = \{x_{i_1}, \cdots, x_{i_d}\}$ $\Pr(x|I) = \Pr(x)$

$\forall I = \{y_{i_1}, \cdots, y_{i_d}\}$ $\Pr(y|I) = \Pr(y)$

Glitches Attacks

Transition Energy in a clock cycle

E_T

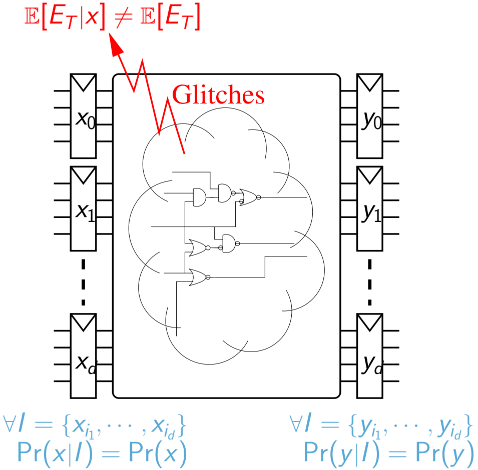
More realistic model

Propagation delays.

$\hookrightarrow E_T \neq E_T^{Ideal}$

[Mangard *et al.* 05]

$$\mathbb{E}[E_T|x] \neq \mathbb{E}[E_T]$$



Glitches Attacks

Transition Energy in a clock cycle

E_T

More realistic model

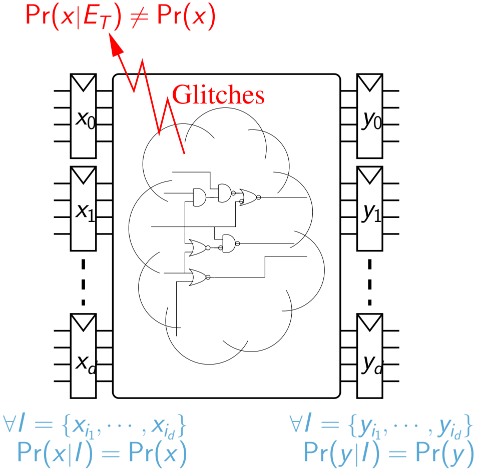
Propagation delays.

$\hookrightarrow E_T \neq E_T^{Ideal}$

[Mangard *et al.* 05]

$$\mathbb{E}[E_T|x] \neq \mathbb{E}[E_T]$$

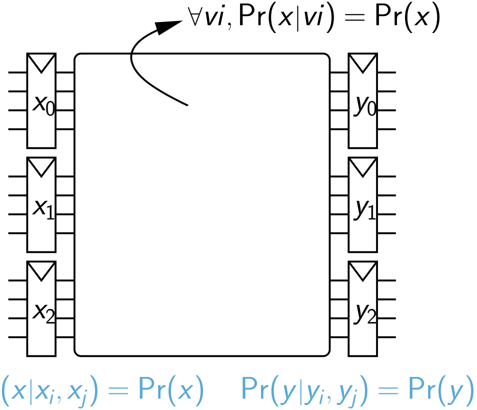
Glitches effects relate power consumption to a combination of the circuit inputs.



1st-order Glitches Free Scheme

[Nikova *et al.* 06,08,10]

Back to classical constraints of SMC
e.g. $n \geq 2d + 1$



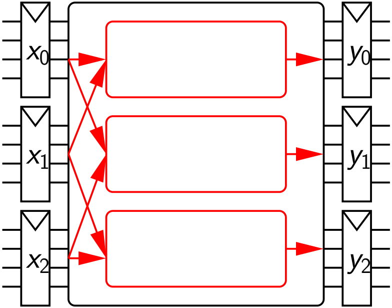
1^{st}-order Glitches Free Scheme

[Nikova *et al.* 06,08,10]

Back to classical constraints of SMC
e.g. $n \geq 2d + 1$

Necessary Condition

The overall leakage is a linear combination of the sub-leakage.



$\Pr(x|x_i, x_j) = \Pr(x)$ $\Pr(y|y_i, y_j) = \Pr(y)$

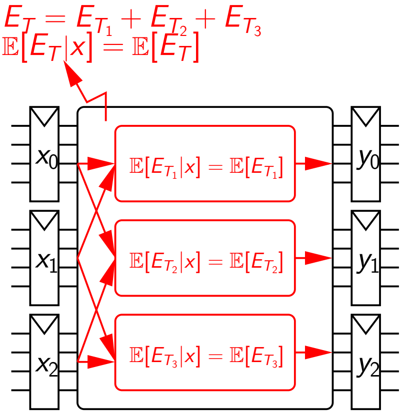
1^{st}-order Glitches Free Scheme

[Nikova *et al.* 06,08,10]

Back to classical constraints of SMC
e.g. $n \geq 2d + 1$

Necessary Condition

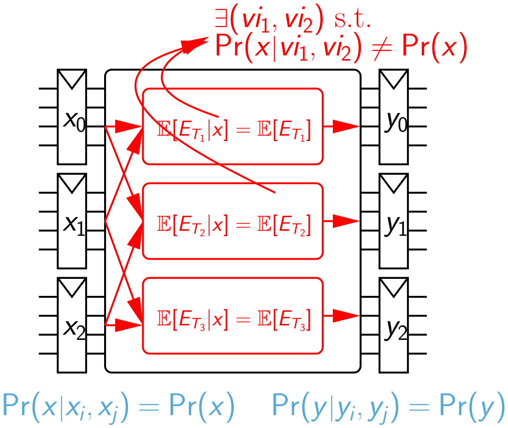
The overall leakage is a linear combination of the sub-leakage.



$\Pr(x|x_i, x_j) = \Pr(x)$ $\quad$ $\Pr(y|y_i, y_j) = \Pr(y)$

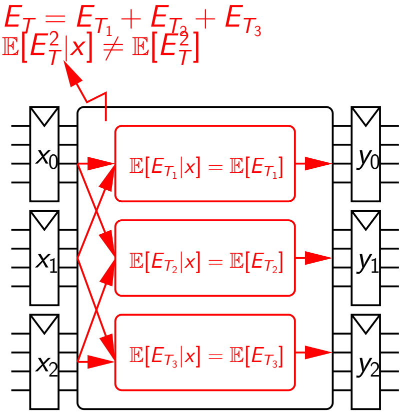
1st-order Glitches Free Scheme

But still susceptible to 2nd-order SCA.



1^{st}-order Glitches Free Scheme

But still susceptible to 2^{nd}-order SCA.
Also susceptible to 2^{nd}-order Glitches attacks.



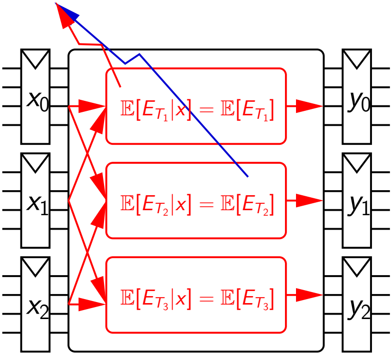
$\Pr(x|x_i, x_j) = \Pr(x)$ $\quad$ $\Pr(y|y_i, y_j) = \Pr(y)$

1^{st}-order Glitches Free Scheme

But still susceptible to 2^{nd}-order SCA.

Also susceptible to 2^{nd}-order Glitches attacks.

$\mathbb{E}[E_{T_1} \times E_{T_2} | x] \neq \mathbb{E}[E_{T_1} \times E_{T_2}]$

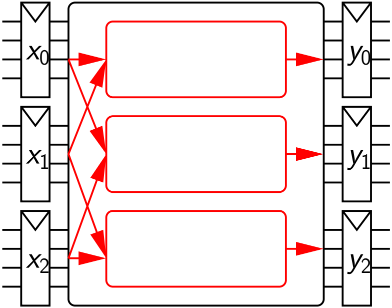


$\Pr(x|x_i, x_j) = \Pr(x) \quad \Pr(y|y_i, y_j) = \Pr(y)$

1^{st}-order Glitches Free Scheme

No simple generalisation of Nikova *et al.* scheme.

↪ Sub-Optimal in the nb of shares when $d > 1$



$\Pr(x|x_i, x_j) = \Pr(x) \quad \Pr(y|y_i, y_j) = \Pr(y)$

Building HO-Masking HO-Glitches Free Scheme

Constraints to prove d^{th}-order security

- Independence of Sub-Circuits leakage.

- Side-channel information (from Probing and/or Glitches) from any family of d Sub-Circuits executions is independent of sensitive variables.

Building HO-Masking HO-Glitches Free Scheme

Constraints to prove d^{th}-order security

- Independence of Sub-Circuits leakage.

  Temporal or Spatial Separation.

- Side-channel information (from Probing and/or Glitches) from any family of d Sub-Circuits executions is independent of sensitive variables.

Building HO-Masking HO-Glitches Free Scheme

Constraints to prove d^{th}-order security

- Independence of Sub-Circuits leakage.

  Temporal or Spatial Separation.

- Side-channel information (from Probing and/or Glitches) from any family of d Sub-Circuits executions is independent of sensitive variables.

  ↪ (Glitches Effects) Any family of d Sub-circuits Inputs are independent of sensitive variables.

Building HO-Masking HO-Glitches Free Scheme

Constraints to prove d^{th}-order security

- Independence of Sub-Circuits leakage.

  Temporal or Spatial Separation.

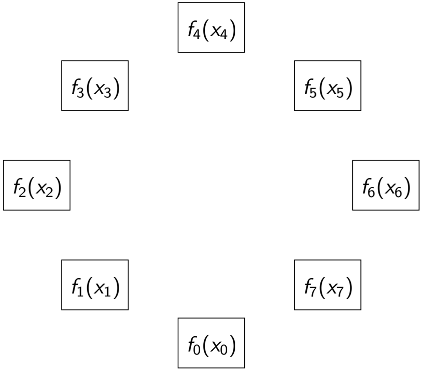
- Side-channel information (from Probing and/or Glitches) from any family of d Sub-Circuits executions is independent of sensitive variables.

  ↪ (Glitches Effects) Any family of d Sub-circuits Inputs are independent of sensitive variables.

Secure Multi-Party Computation Protocols

Secure Multi-Party Computation Protocol

How to securely compute a function f over the n shares?

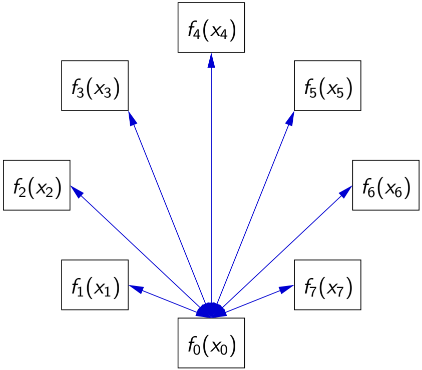


Secure Multi-Party Computation Protocol

How to securely compute a function f over the n shares?

[BGW 88]

- Secure communication b/t the Players.
- $n \geq 2d + 1$.



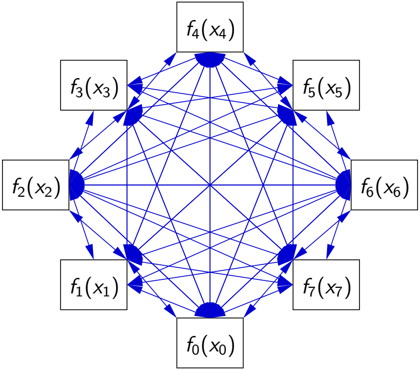
Secure Multi-Party Computation Protocol

How to securely compute a function f over the n shares?

[BGW 88]

- Secure communication b/t the Players.
- $n \geq 2d + 1$.

Data is re-shared before going through the secure channels.

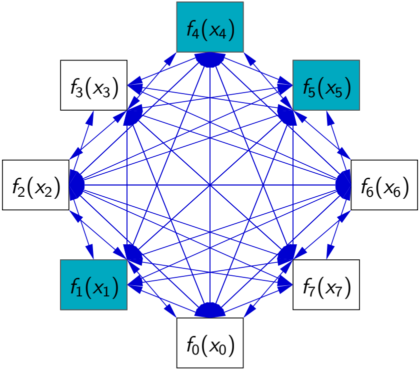


Secure Multi-Party Computation Protocol

How to securely compute a function f over the n shares?

[BGW 88]

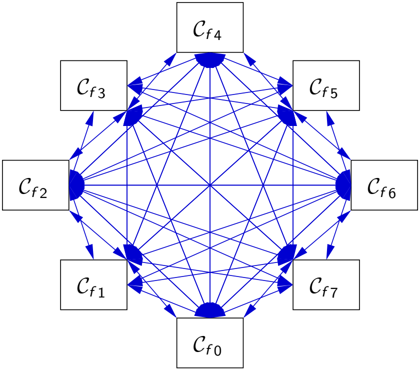
- Secure communication b/t the Players.
- $n \geq 2d + 1$.



Multi-Party Circuits

The Multi-Party Circuit $\mathcal{C}_f$ verifies the SMC constraints

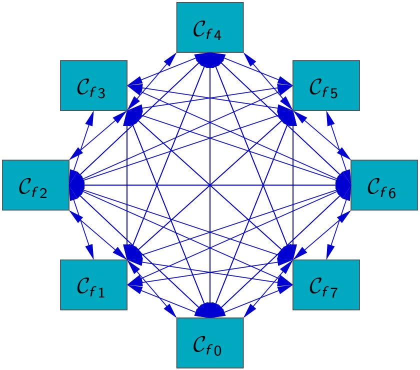
Players $\equiv$ Sub-Circuits $\mathcal{C}_{f\,i}$.



Multi-Party Circuits

The Multi-Party Circuit $\mathcal{C}_f$ verifies the SMC constraints
Players $\equiv$ Sub-Circuits $\mathcal{C}_{f\,i}$.

In our context, there is no limit in the number of observed Sub-Circuits



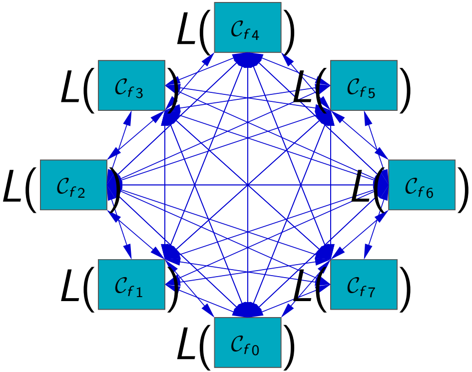
Multi-Party Circuits

The Multi-Party Circuit $\mathcal{C}_f$ verifies the SMC constraints
Players $\equiv$ Sub-Circuits $\mathcal{C}_{f\,i}$.

In our context, there is no limit in the number of observed Sub-Circuits

[Chari et al. 99]

the complexity of the HO-SCA attack is exponential in the order w.r.t. the noise.



BGW's SMC protocol

Shamir's Secret Sharing Scheme ($n \geq d+1$) [Shamir 79]

$$(Z, RNG) \rightarrow P_Z[X] : Z + a_1 X + \cdots + a_d X^d$$
$$(P_Z, \alpha_1, \cdots, \alpha_n) \rightarrow \{P_Z(\alpha_1), \cdots, P_Z(\alpha_n)\}$$

BGW's SMC Protocol ($n \geq 2d+1$) [Ben-Or *et al.* 88]

$\mathcal{C}_{f_i}'s$ inputs: $\{P_A(\alpha_i), P_B(\alpha_i)\}$

- $A + B$: $P_A(\alpha_i) + P_B(\alpha_i)$
- $xA + y$: $xP_A(\alpha_i) + y$
- $A \times B$: $P_A(\alpha_i) \times P_B(\alpha_i)$

BGW's SMC protocol

Shamir's Secret Sharing Scheme ($n \geq d+1$) [Shamir 79]

$$(Z, RNG) \rightarrow P_Z[X] : Z + a_1 X + \cdots + a_d X^d$$

$$(P_Z, \alpha_1, \cdots, \alpha_n) \rightarrow \{P_Z(\alpha_1), \cdots, P_Z(\alpha_n)\}$$

BGW's SMC Protocol ($n \geq 2d+1$) [Ben-Or *et al.* 88]

$\mathcal{C}_{f_i}'s$ inputs: $\{P_A(\alpha_i), P_B(\alpha_i)\}$

- $A+B$: $P_A(\alpha_i) + P_B(\alpha_i)$
- $xA+y$: $xP_A(\alpha_i) + y$
- $A \times B$: $P_A(\alpha_i) \times P_B(\alpha_i)$

BGW's SMC protocol

Shamir's Secret Sharing Scheme ($n \geq d + 1$) [Shamir 79]

$$(Z, RNG) \to P_Z[X] : Z + a_1 X + \cdots + a_d X^d$$

$$(P_Z, \alpha_1, \cdots, \alpha_n) \to \{P_Z(\alpha_1), \cdots, P_Z(\alpha_n)\}$$

BGW's SMC Protocol ($n \geq 2d + 1$) [Ben-Or *et al.* 88]

$\mathcal{C}_{f\,i}'s$ inputs: $\{P_A(\alpha_i), P_B(\alpha_i)\}$

- $A + B$: $P_A(\alpha_i) + P_B(\alpha_i)$
- $xA + y$: $xP_A(\alpha_i) + y$
- $A \times B$: $P_A(\alpha_i) \times P_B(\alpha_i)$

BGW's SMC protocol

Shamir's Secret Sharing Scheme ($n \geq d+1$) [Shamir 79]

$$(Z, RNG) \rightarrow P_Z[X] : Z + a_1 X + \cdots + a_d X^d$$

$$(P_Z, \alpha_1, \cdots, \alpha_n) \rightarrow \{P_Z(\alpha_1), \cdots, P_Z(\alpha_n)\}$$

BGW's SMC Protocol ($n \geq 2d+1$) [Ben-Or *et al.* 88]

$\mathcal{C}_{f'_i}s$ inputs: $\{P_A(\alpha_i), P_B(\alpha_i)\}$

- $A + B$: $P_A(\alpha_i) + P_B(\alpha_i)$
- $xA + y$: $xP_A(\alpha_i) + y$
- $A \times B$: $P_A(\alpha_i) \times P_B(\alpha_i)$

BGW's SMC protocol

Shamir's Secret Sharing Scheme ($n \geq d + 1$) [Shamir 79]

$$(Z, RNG) \to P_Z[X] : Z + a_1 X + \cdots + a_d X^d$$

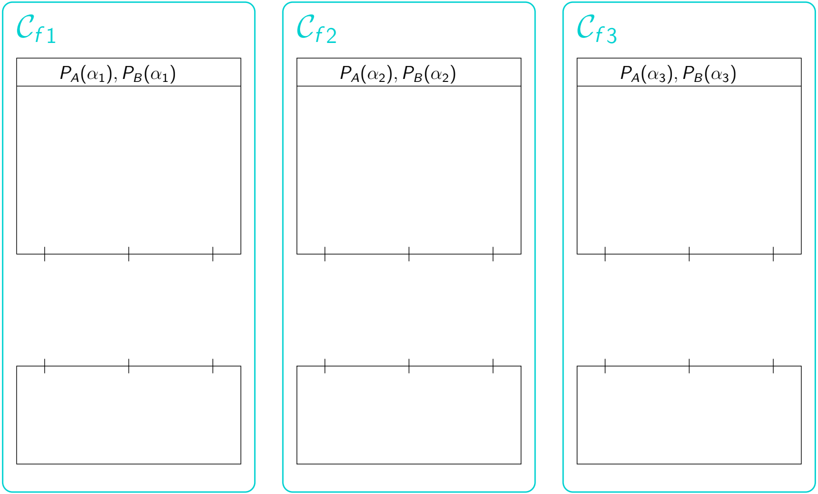
$$(P_Z, \alpha_1, \cdots, \alpha_n) \to \{P_Z(\alpha_1), \cdots, P_Z(\alpha_n)\}$$

BGW's SMC Protocol ($n \geq 2d + 1$) [Ben-Or *et al.* 88]

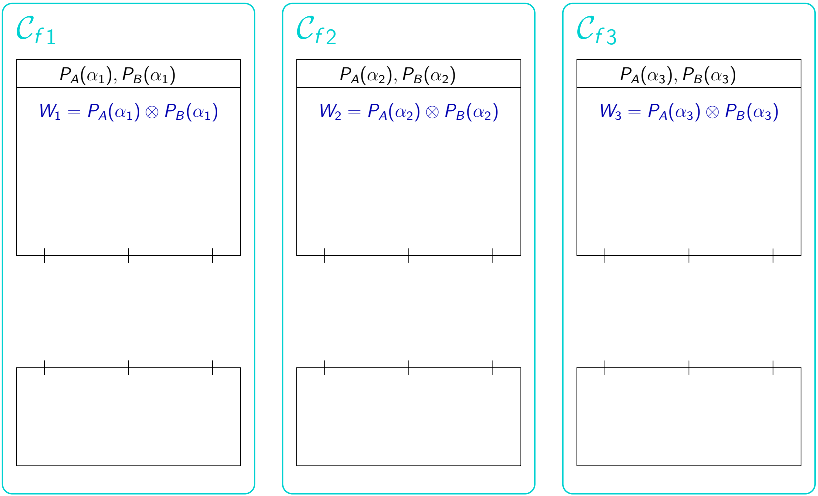
$\mathcal{C}_{f}{}'_{i}s$ inputs: $\{P_A(\alpha_i), P_B(\alpha_i)\}$

- $A + B$: $P_A(\alpha_i) + P_B(\alpha_i)$
- $xA + y$: $xP_A(\alpha_i) + y$
- $A \times B$: $P_A(\alpha_i) \times P_B(\alpha_i)$

BWG's scheme for the product in $\mathrm{GF}(2)^n$, $n = 3, d = 1$



BWG's scheme for the product in $\mathrm{GF}(2)^n$, $n = 3, d = 1$



BWG's scheme for the product in $\mathrm{GF}(2)^n$, $n = 3, d = 1$

$\mathcal{C}_{f1}$

$P_A(\alpha_1), P_B(\alpha_1)$

$W_1 = P_A(\alpha_1) \otimes P_B(\alpha_1)$

$P_{W_1} \leftarrow (W_1, RNG)$

$\mathcal{C}_{f2}$

$P_A(\alpha_2), P_B(\alpha_2)$

$W_2 = P_A(\alpha_2) \otimes P_B(\alpha_2)$

$P_{W_2} \leftarrow (W_2, RNG)$

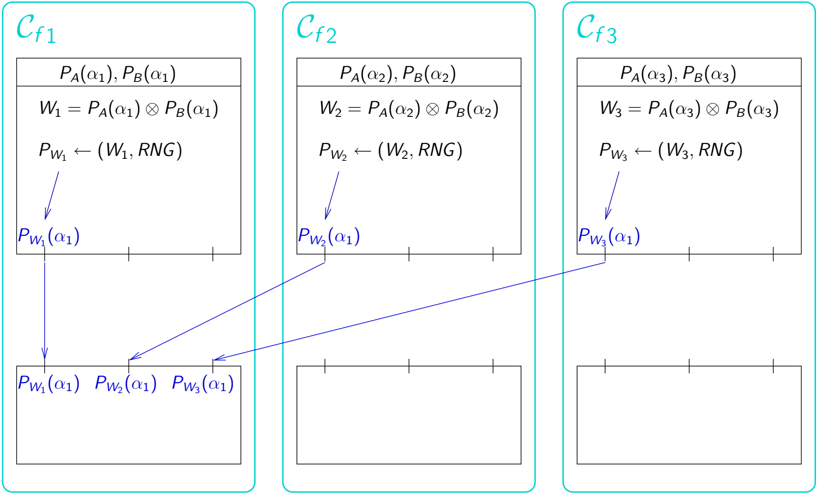
$\mathcal{C}_{f3}$

$P_A(\alpha_3), P_B(\alpha_3)$

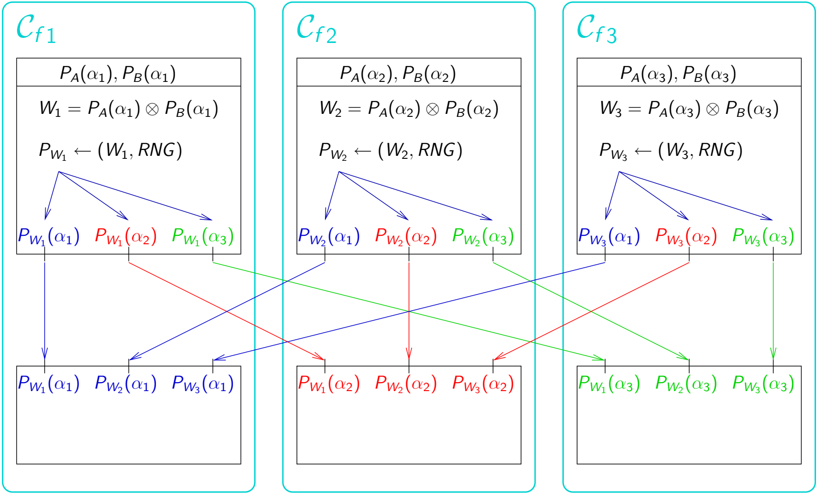
$W_3 = P_A(\alpha_3) \otimes P_B(\alpha_3)$

$P_{W_3} \leftarrow (W_3, RNG)$

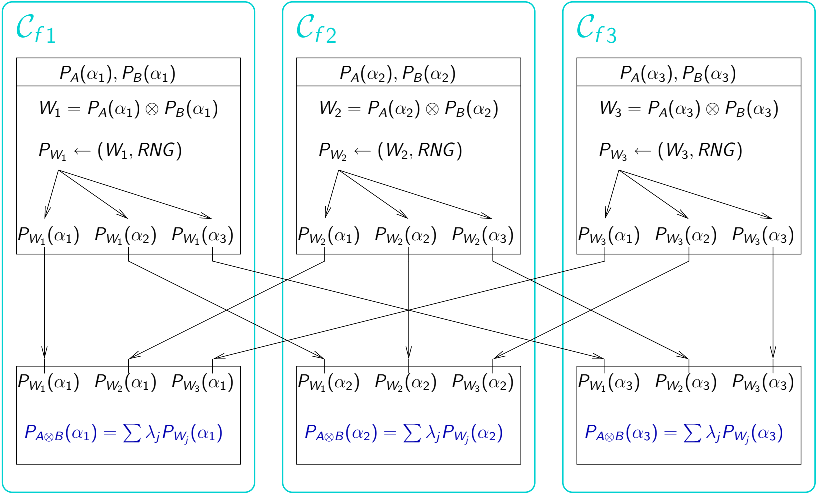
BWG's scheme for the product in $\mathrm{GF}(2)^n$, $n = 3, d = 1$



BWG's scheme for the product in $\mathrm{GF}(2)^n$, $n = 3, d = 1$



BWG's scheme for the product in $\mathrm{GF}(2)^n$, $n = 3, d = 1$



Comparison with Rivain and Prouff's Scheme [CHES 2010]

Method	multiplications	additions
This paper	$4d^3 + 8d^2 + 3d$	$4d^3 + 8d^2 + 7d + 2$
[Rivain *et al.* 10]	$2d^2 + 2d$	$d^2 + d + 1$

Method	random bytes
This paper	$d(2d + 1)$
[Rivain *et al.* 10]	$d(d + 1)/2$

Conclusions and Future Directions

What has been achieved

- First glitches free HO-masking scheme.
- New Masking Function: Shamir's secret sharing scheme.

Next Steps

- How to satisfy the separation of sub-circuits.
- Efficient implementations.
- Relaxations *w.r.t.* leakages models.
  (*e.g.* reduce random bytes, cf. Nikova *et al.*)