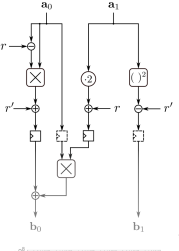
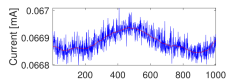
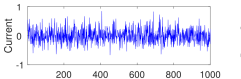
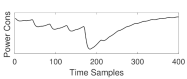
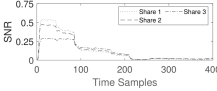
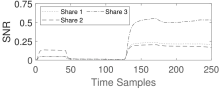


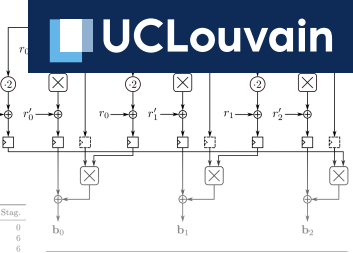








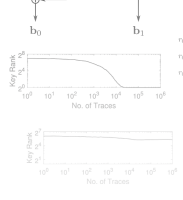


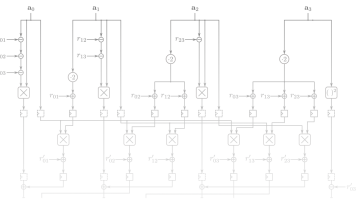


S-box	Shares d	FFs	LUTs	Slic.	DSPs	Crit. Path [ns]	Reg. Stag.
AES	1	0	87	19	0	3.63	0
	2	616	513	101	0	5.08	6
	3	1362	1217	212	0	5.53	6
	4	2400	2203	523	0	6.79	6
AES-prime	1	0	72	24	3	6.99	0
	2	133	495	123	10	5.95	3
	3	364	1207	246	21	6.69	4
	4	1022	2462	487	36	7.02	5
AES-prime*	1	0	204	39	0	5.76	0
	2	133	1095	202	0	5.55	3
	3	364	2589	464	0	6.26	4
	4	1022	4774	834	0	7.13	5

*Use of DSP slices prohibited

Gadget	d	Multiplications	Squarings	Randomness	Reg. stages
HPC1	2	4	0	2	2
HPC1	3	9	0	5	2
HPC1	d	d^2	0	$d^2/2 + O(d \log d)$	2
Algorithm 2	2	2	1	3	1
Algorithm 3	3	6	0	5	1
Algorithm 1	d	$d(d+1)/2-1$	1	$d(d-1)$	2





UCLouvain Institute of Information and Communication Technologies, Electronics and Applied Mathematics (ICTEAM)

Prime-Field Masking in Hardware and its Soundness against Low-Noise SCA Attacks

G. Cassiers, L. Masure, C. Momin, T. Moos, F.-X. Standaert

Crypto Group, ICTEAM Institute, UCLouvain, Louvain-la-Neuve, Belgium.

Graz University of Technology, Graz, Austria.

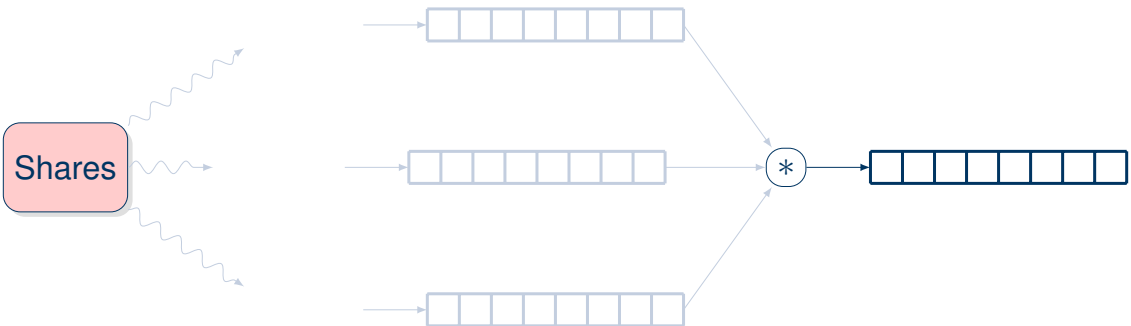
September 11th, 2023

European Research Council

Why Masking?

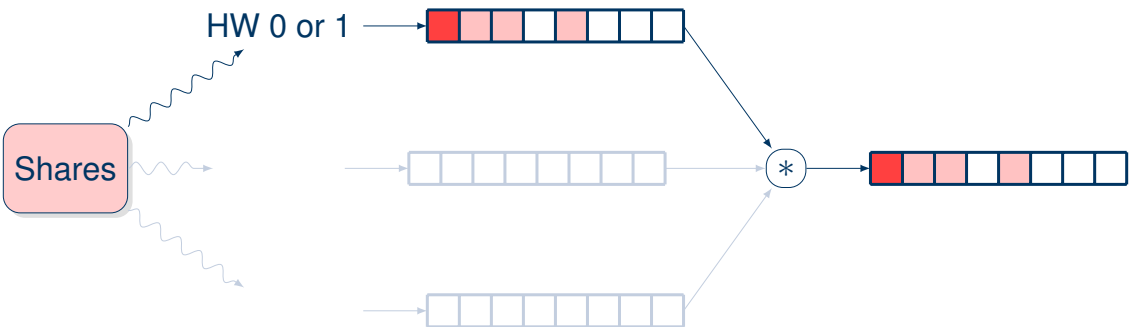
UCLouvain

Adversaries Make Imprecise Observations, Masking Amplifies Imprecision:



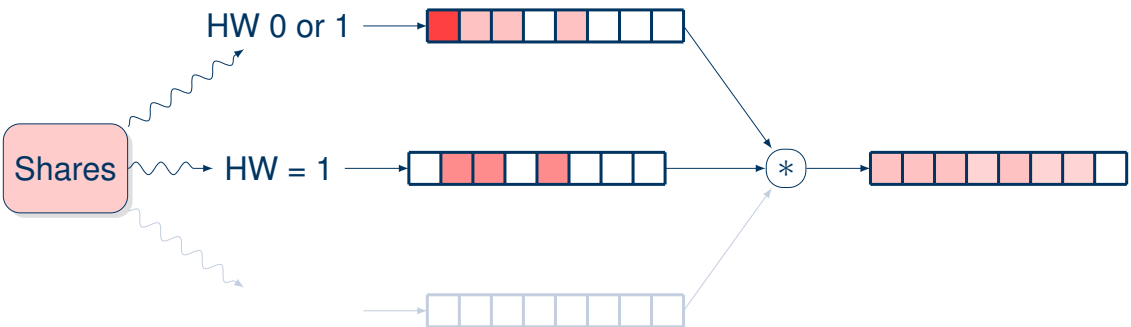
Why Masking?

Adversaries Make Imprecise Observations, Masking Amplifies Imprecision:



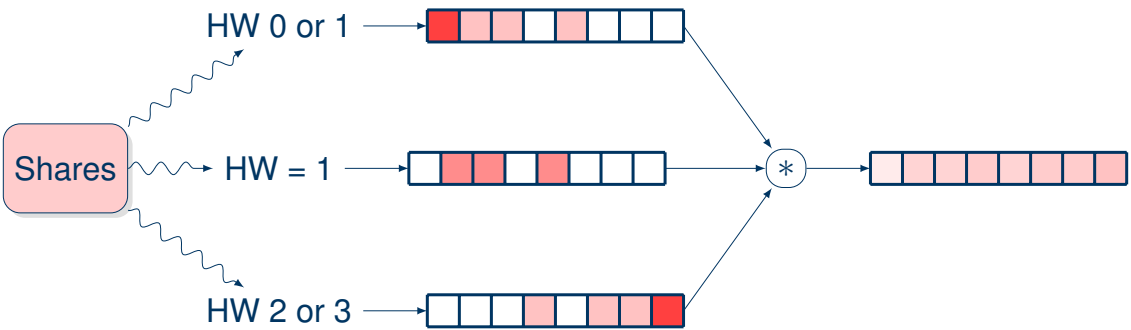
Why Masking?

Adversaries Make Imprecise Observations, Masking Amplifies Imprecision:



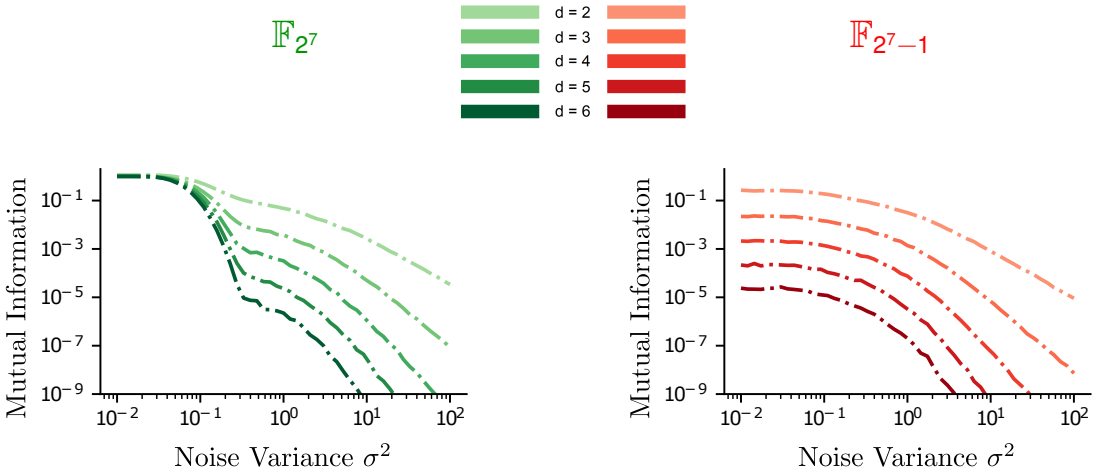
Why Masking?

Adversaries Make Imprecise Observations, Masking Amplifies Imprecision:



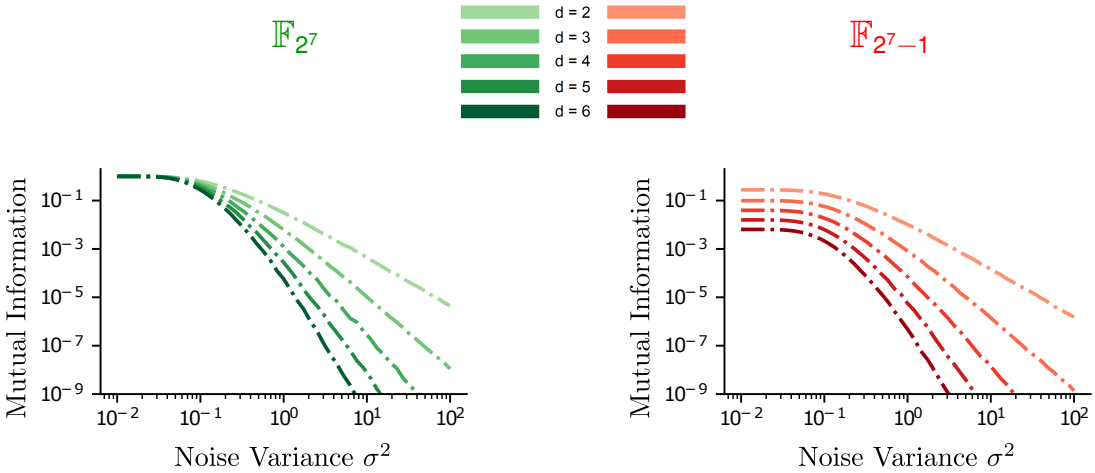
Why Masking?

But Does it Always? An Example Based on Hamming Weight Leakage:



Why Masking?

But Does it Always? An Example Based on Bit Leakage (LSB):



Why Prime-Field Masking?

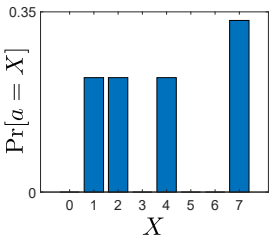
Noise-Free Hamming Weight Leakage of 2 Shares:

$\mathbb{F}_{2^3}$

$a = a_0 \oplus a_1$

$\mathrm{HW}(a_0) = 2$
$\mathrm{HW}(a_1) = 1$

$\Rightarrow$

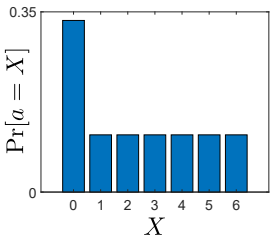


$\mathbb{F}_{2^3-1}$

$a = a_0 + a_1 \bmod 7$

$\mathrm{HW}(a_0) = 2$
$\mathrm{HW}(a_1) = 1$

$\Rightarrow$



Why Prime-Field Masking?

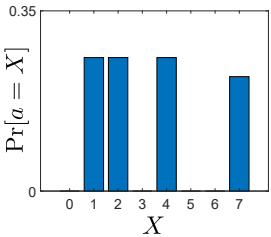
Noise-Free Hamming Weight Leakage of 3 Shares:

$\mathbb{F}_{2^3}$

$a = a_0 \oplus a_1 \oplus a_2$

$\mathrm{HW}(a_0) = 2$
$\mathrm{HW}(a_1) = 1$
$\mathrm{HW}(a_2) = 2$

$\Rightarrow$

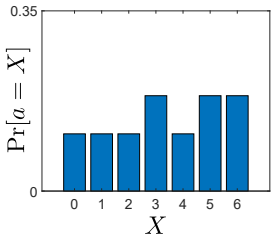


$\mathbb{F}_{2^3-1}$

$a = a_0 + a_1 + a_2 \bmod 7$

$\mathrm{HW}(a_0) = 2$
$\mathrm{HW}(a_1) = 1$
$\mathrm{HW}(a_2) = 2$

$\Rightarrow$



Why Prime-Field Masking?

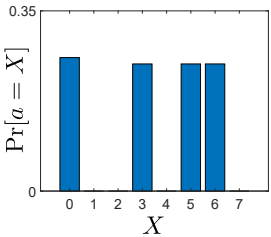
Noise-Free Hamming Weight Leakage of 4 Shares:

$\mathbb{F}_{2^3}$

$$a = a_0 \oplus a_1 \oplus a_2 \oplus a_3$$

$\mathrm{HW}(a_0) = 2$
$\mathrm{HW}(a_1) = 1$
$\mathrm{HW}(a_2) = 2$
$\mathrm{HW}(a_3) = 1$

$\Rightarrow$

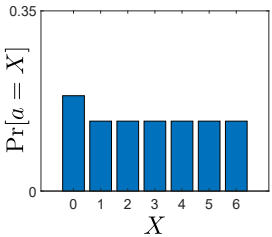


$\mathbb{F}_{2^3-1}$

$$a = a_0 + a_1 + a_2 + a_3 \bmod 7$$

$\mathrm{HW}(a_0) = 2$
$\mathrm{HW}(a_1) = 1$
$\mathrm{HW}(a_2) = 2$
$\mathrm{HW}(a_3) = 1$

$\Rightarrow$



Why Prime-Field Masking?

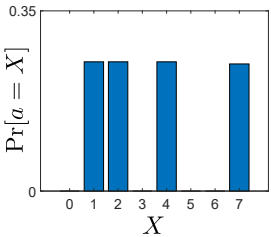
Noise-Free Hamming Weight Leakage of 5 Shares:

$\mathbb{F}_{2^3}$

$a = a_0 \oplus a_1 \oplus a_2 \oplus a_3 \oplus a_4$

$\mathrm{HW}(a_0) = 2$
$\mathrm{HW}(a_1) = 1$
$\mathrm{HW}(a_2) = 2$
$\mathrm{HW}(a_3) = 1$
$\mathrm{HW}(a_4) = 1$

$\Rightarrow$

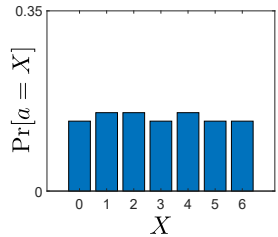


$\mathbb{F}_{2^3-1}$

$a = a_0 + a_1 + a_2 + a_3 + a_4 \bmod 7$

$\mathrm{HW}(a_0) = 2$
$\mathrm{HW}(a_1) = 1$
$\mathrm{HW}(a_2) = 2$
$\mathrm{HW}(a_3) = 1$
$\mathrm{HW}(a_4) = 1$

$\Rightarrow$



Why Prime-Field Masking?

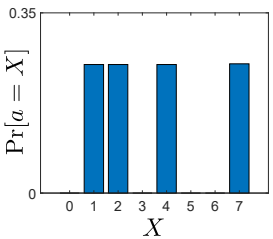
Noise-Free Hamming Weight Leakage of 6 Shares:

$\mathbb{F}_{2^3}$

$a = a_0 \oplus a_1 \oplus a_2 \oplus a_3 \oplus a_4 \oplus a_5$

$\mathrm{HW}(a_0) = 2$
$\mathrm{HW}(a_1) = 1$
$\mathrm{HW}(a_2) = 2$
$\mathrm{HW}(a_3) = 1$
$\mathrm{HW}(a_4) = 1$
$\mathrm{HW}(a_5) = 2$

$\Rightarrow$

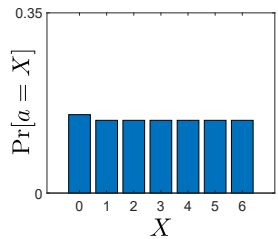


$\mathbb{F}_{2^3-1}$

$a = a_0 + a_1 + a_2 + a_3 + a_4 + a_5 \bmod 7$

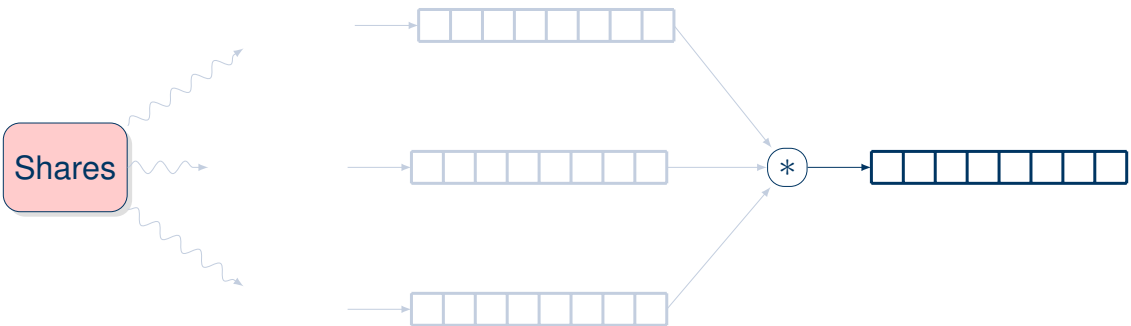
$\mathrm{HW}(a_0) = 2$
$\mathrm{HW}(a_1) = 1$
$\mathrm{HW}(a_2) = 2$
$\mathrm{HW}(a_3) = 1$
$\mathrm{HW}(a_4) = 1$
$\mathrm{HW}(a_5) = 2$

$\Rightarrow$



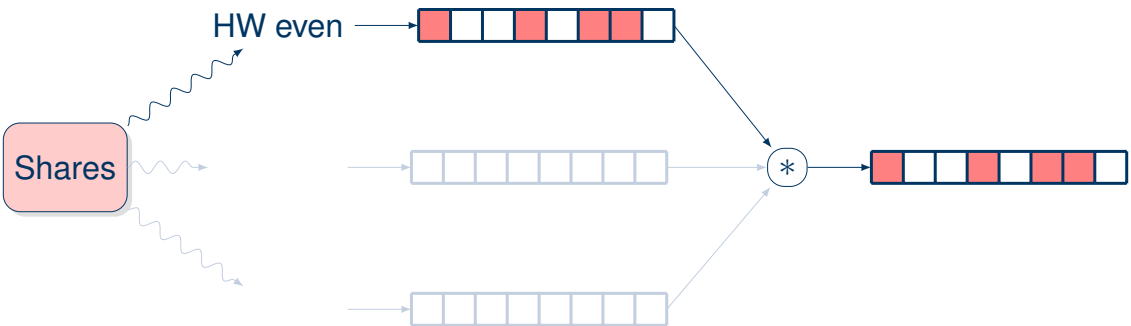
Why Prime-Field Masking?

Parity of HWs $\rightarrow$ Subgroup of $\mathbb{F}_{2^n}$:



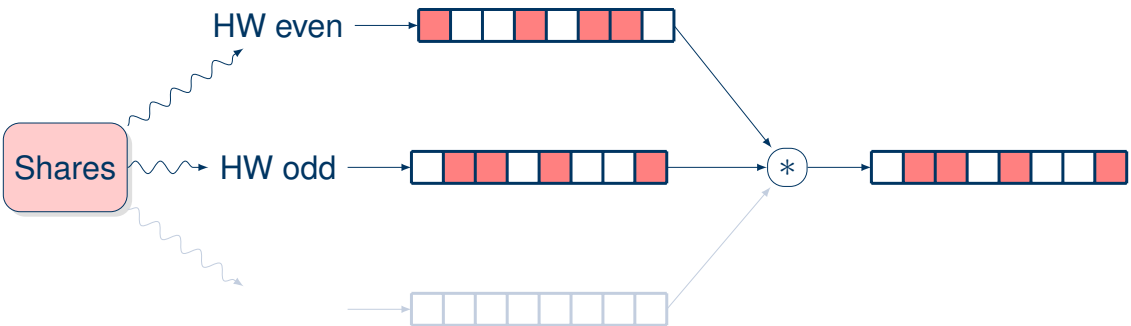
Why Prime-Field Masking?

Parity of HWs $\rightarrow$ Subgroup of $\mathbb{F}_{2^n}$:



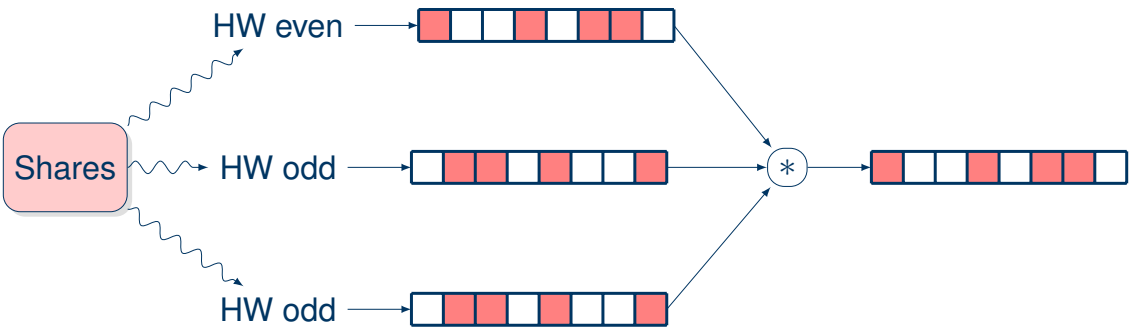
Why Prime-Field Masking?

Parity of HWs $\rightarrow$ Subgroup of $\mathbb{F}_{2^n}$:



Why Prime-Field Masking?

Parity of HWs $\rightarrow$ Subgroup of $\mathbb{F}_{2^n}$:



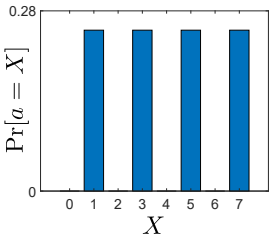
Why Prime-Field Masking?

Noise-Free Bit Leakage (LSB) of 2 Shares:

$\mathbb{F}_{2^3}$

$a = a_0 \oplus a_1$

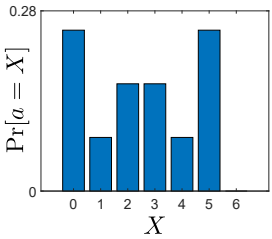
$\mathrm{LSB}(a_0) = 0$
$\mathrm{LSB}(a_1) = 1$ $\Rightarrow$



$\mathbb{F}_{2^3-1}$

$a = a_0 + a_1 \bmod 7$

$\mathrm{LSB}(a_0) = 0$
$\mathrm{LSB}(a_1) = 1$ $\Rightarrow$



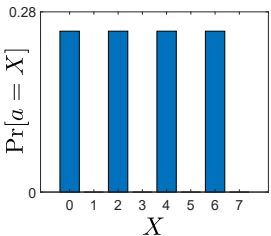
Why Prime-Field Masking?

Noise-Free Bit Leakage (LSB) of 3 Shares:

$\mathbb{F}_{2^3}$

$$a = a_0 \oplus a_1 \oplus a_2$$

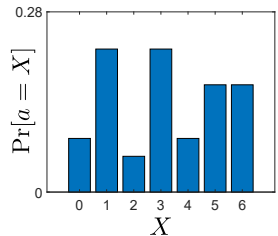
$$\begin{aligned} \mathrm{LSB}(a_0) &= 0 \\ \mathrm{LSB}(a_1) &= 1 \\ \mathrm{LSB}(a_2) &= 1 \end{aligned} \quad \Rightarrow$$



$\mathbb{F}_{2^3-1}$

$$a = a_0 + a_1 + a_2 \bmod 7$$

$$\begin{aligned} \mathrm{LSB}(a_0) &= 0 \\ \mathrm{LSB}(a_1) &= 1 \\ \mathrm{LSB}(a_2) &= 1 \end{aligned} \quad \Rightarrow$$



Why Prime-Field Masking?

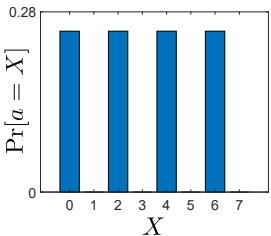
Noise-Free Bit Leakage (LSB) of 4 Shares:

$\mathbb{F}_{2^3}$

$a = a_0 \oplus a_1 \oplus a_2 \oplus a_3$

$\mathrm{LSB}(a_0) = 0$
$\mathrm{LSB}(a_1) = 1$
$\mathrm{LSB}(a_2) = 1$
$\mathrm{LSB}(a_3) = 0$

$\Rightarrow$

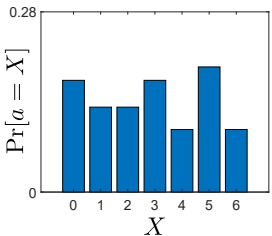


$\mathbb{F}_{2^3-1}$

$a = a_0 + a_1 + a_2 + a_3 \bmod 7$

$\mathrm{LSB}(a_0) = 0$
$\mathrm{LSB}(a_1) = 1$
$\mathrm{LSB}(a_2) = 1$
$\mathrm{LSB}(a_3) = 0$

$\Rightarrow$



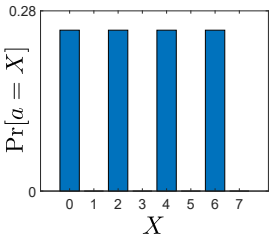
Why Prime-Field Masking?

Noise-Free Bit Leakage (LSB) of 5 Shares:

$\mathbb{F}_{2^3}$

$$a = a_0 \oplus a_1 \oplus a_2 \oplus a_3 \oplus a_4$$

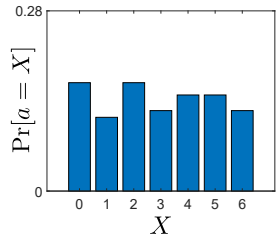
$$\begin{aligned} &\mathrm{LSB}(a_0) = 0 \\ &\mathrm{LSB}(a_1) = 1 \\ &\mathrm{LSB}(a_2) = 1 \\ &\mathrm{LSB}(a_3) = 0 \\ &\mathrm{LSB}(a_4) = 0 \end{aligned} \Rightarrow$$



$\mathbb{F}_{2^3-1}$

$$a = a_0 + a_1 + a_2 + a_3 + a_4 \bmod 7$$

$$\begin{aligned} &\mathrm{LSB}(a_0) = 0 \\ &\mathrm{LSB}(a_1) = 1 \\ &\mathrm{LSB}(a_2) = 1 \\ &\mathrm{LSB}(a_3) = 0 \\ &\mathrm{LSB}(a_4) = 0 \end{aligned} \Rightarrow$$



Why Prime-Field Masking?

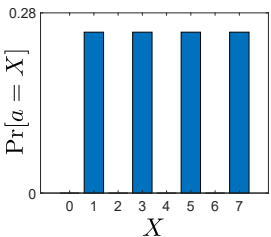
Noise-Free Bit Leakage (LSB) of 6 Shares:

$\mathbb{F}_{2^3}$

$a = a_0 \oplus a_1 \oplus a_2 \oplus a_3 \oplus a_4 \oplus a_5$

$\mathrm{LSB}(a_0) = 0$
$\mathrm{LSB}(a_1) = 1$
$\mathrm{LSB}(a_2) = 1$
$\mathrm{LSB}(a_3) = 0$
$\mathrm{LSB}(a_4) = 0$
$\mathrm{LSB}(a_5) = 1$

$\Rightarrow$

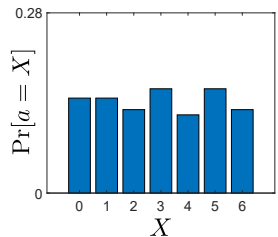


$\mathbb{F}_{2^3-1}$

$a = a_0 + a_1 + a_2 + a_3 + a_4 + a_5 \bmod 7$

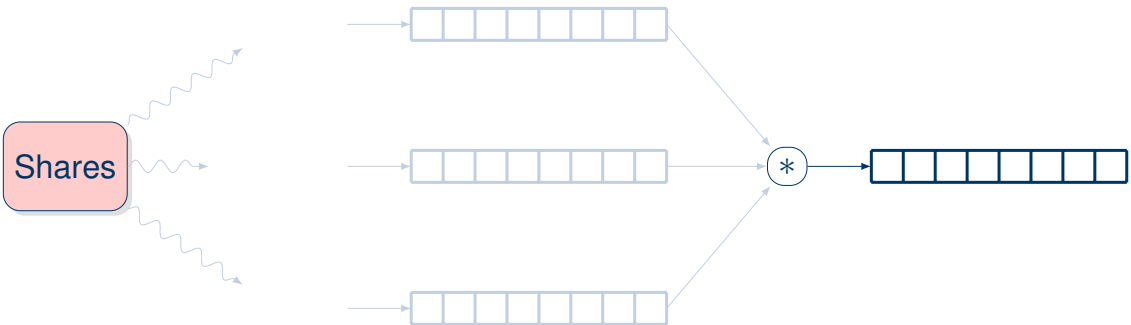
$\mathrm{LSB}(a_0) = 0$
$\mathrm{LSB}(a_1) = 1$
$\mathrm{LSB}(a_2) = 1$
$\mathrm{LSB}(a_3) = 0$
$\mathrm{LSB}(a_4) = 0$
$\mathrm{LSB}(a_5) = 1$

$\Rightarrow$



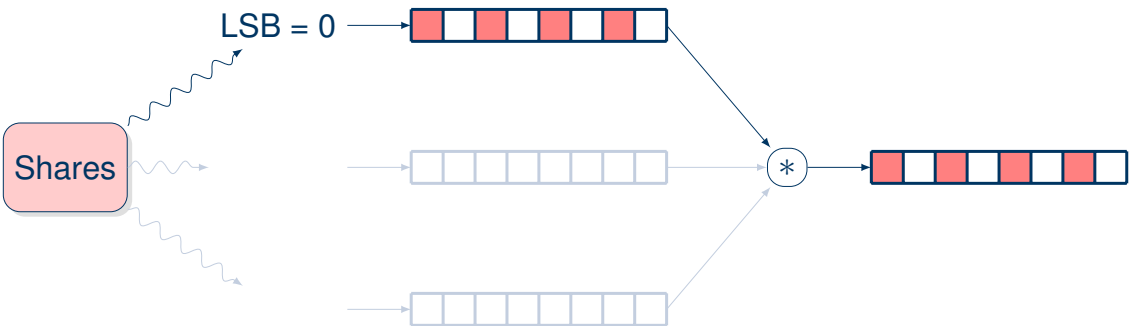
Why Prime-Field Masking?

Parity of LSBs $\rightarrow$ Subgroup of $\mathbb{F}_{2^n}$:



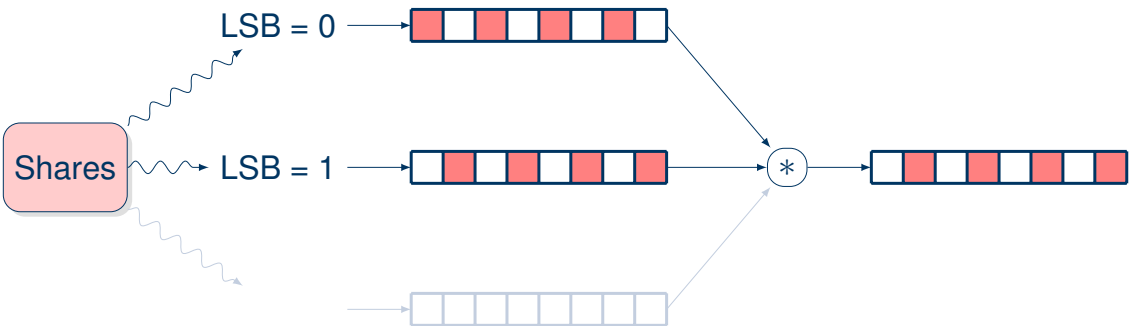
Why Prime-Field Masking?

Parity of LSBs $\rightarrow$ Subgroup of $\mathbb{F}_{2^n}$:



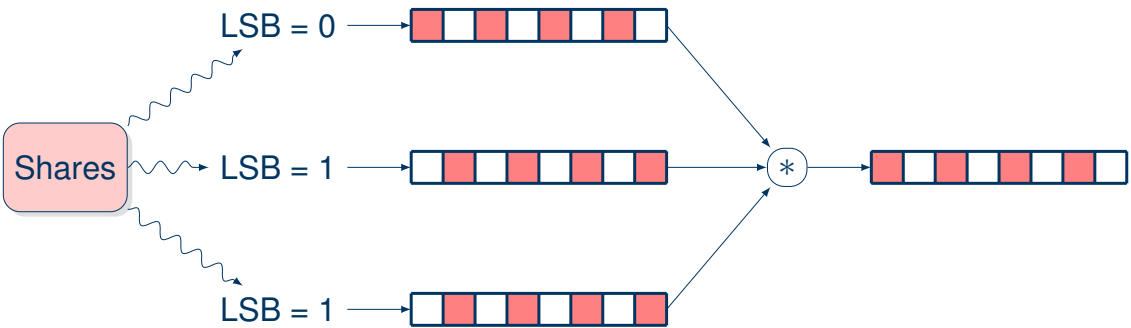
Why Prime-Field Masking?

Parity of LSBs $\rightarrow$ Subgroup of $\mathbb{F}_{2^n}$:



Why Prime-Field Masking?

Parity of LSBs $\rightarrow$ Subgroup of $\mathbb{F}_{2^n}$:



That's Why!

- Advanced multivariate SCA attacks aim to extract and combine as much information as possible about a targeted intermediate to reduce the effective noise level
- "Algebraic compatibility" between leakage functions and field arithmetic can become a problem in practice when the effective noise level is too low
- Parallelism may not save binary-field masking either, parities are still visible
- Whenever $\geqslant 1$ bit of information per share is leaked, masking in binary fields may not guarantee security amplification

- $\mathbb{F}_p$ with p a prime has no non-trivial subgroups, i.e., no "algebraic compatibility"
- **Amplification is guaranteed for any non-injective leakage model!**

How to leverage?

Q: How can we make use of masking in $\mathbb{F}_p$ to effectively and efficiently protect crypto implementations?

A: Ideally, we need algorithms that work in implementation-friendly prime fields, such as small-Mersenne-prime fields ($\mathbb{F}_{2^n-1}$), and use only simple field arithmetic $(+,\ -,\ \cdot)$

AES-prime: An AES-like toy cipher adapted for prime-field masking

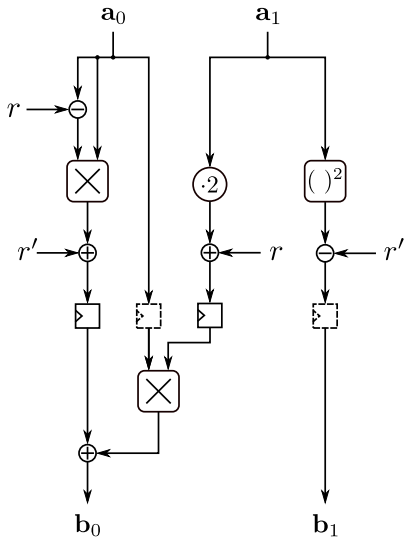
- Based on arithmetic addition/multiplication modulo a prime, applied to 4×4 state
- Small Mersenne prime, i.e., $p = 2^7 - 1$, for efficient reduction (and constant mult.)
- Sbox is based on a small power map in $\mathbb{F}_p$ (bijection without fixed point)
- MixColumns is a 4×4 MDS matrix over $\mathbb{F}_p$
- Security claim: Attack complexity $\geqslant 2^{7 \cdot 16}$ with 14 cipher rounds

$$S(x) = x^5 + 2 \bmod p \qquad M = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 2 & 4 & 16 \\ 1 & 4 & 16 & 2 \\ 1 & 16 & 2 & 4 \end{bmatrix}$$

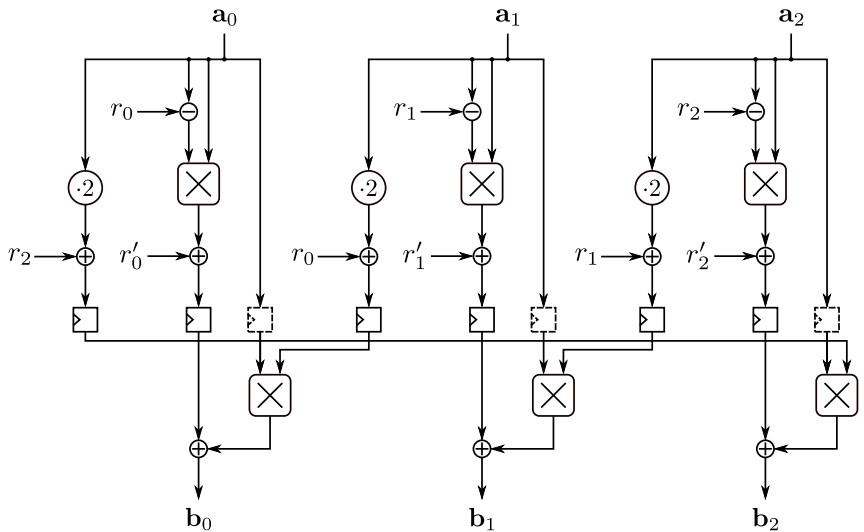
One Problem!

- To implement masked power maps securely in hardware we ideally need robust probing secure and composable multiplications and squarings
- Secure and composable masked multiplication gadgets are well-known, some are even field agnostic (e.g., ISW, DOM-indep, HPC1)
- The squaring operation, due to its linearity in Boolean masking, has not received much attention in that regard (typically no gadget needed)
- However, as squaring is non-linear in prime fields we either have to use multiplication gadgets or develop secure squaring gadgets
- We opted for the second solution!

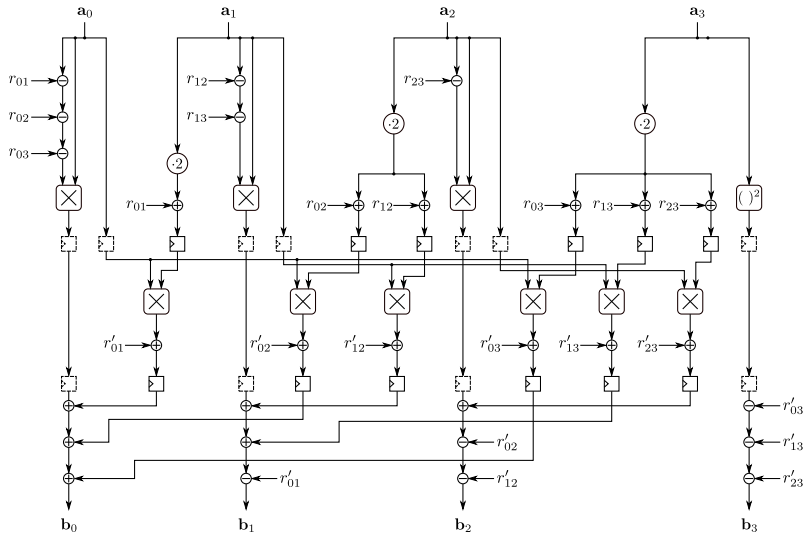
Glitch-robust 1-PINI Squaring Gadget (any field)



Glitch-robust 2-PINI Squaring Gadget (any field)



Glitch-robust 3-PINI Squaring Gadget (any field)



Glitch-robust (d-1)-PINI Squaring Gadget (any field)

UCLouvain

Algorithm 1 Masked squaring (glitch-robust PINI) with d shares.

Input: Sharing $\mathbf{a}$.
Output: Sharing $\mathbf{b}$ such that $b = a^2$.

1: **for** $i = 0$ to $d-1$ **do**
2: **for** $j = i+1$ to $d-1$ **do**
3: $r_{ij} \xleftarrow{\$} \mathbb{F}_p$
4: $r'_{ij} \xleftarrow{\$} \mathbb{F}_p$
5: $\alpha_{ij} \leftarrow 2\mathbf{a}_j + r_{ij}$
6: $\beta_{ij} \leftarrow \mathbf{a}_i \operatorname{Reg}(\alpha_{ij}) + r'_{ij}$
7: **for** $i = 0$ to $d-1$ **do**
8: $\gamma_i \leftarrow \mathbf{a}_i \left(\mathbf{a}_i - \sum_{j=i+1}^{d-1} r_{ij}\right)$
9: $\mathbf{b}_i = \operatorname{Reg}(\gamma_i) + \sum_{j=i+1}^{d-1} \operatorname{Reg}(\beta_{ij}) - \sum_{j=0}^{i-1} r'_{ji}$ ▷ Blue Reg not needed for PINI.

Comparison to PINI multiplications (any field)

Gadget	d	Multiplications	Squarings	Randomness	Reg. stages
HPC1	2	4	0	2	2
HPC1	3	9	0	5	2
HPC1	d	d^2	0	$d^2/2 + \mathcal{O}(d \log d)$	2
Algorithm 2	2	2	1	2	1
Algorithm 3	3	6	0	5	1
Algorithm 1	d	$d(d+1)/2 - 1$	1	$d(d-1)$	2

Optimized PINI S-boxes

- Using those gadgets we built 3 optimized order-specific PINI `AES-prime` S-boxes
- 2 Shares: 2 x 1-cycle SQ + DOM-indep = 3-cycle masked S-box
- 3 Shares: 2 x 1-cycle SQ + Reg + DOM-indep = 4-cycle masked S-box
- 4 Shares: 2 x 2-cycle SQ + DOM-indep = 5-cycle masked S-box
- Constructions and proofs in the paper!

Mersenne Prime Field Operations in Hardware

UCLouvain

Field Addition in $\mathbb{F}_{2^n-1}$ in VHDL ($c = a + b \bmod p$)

```
ab   <= ('0' & a) + ('0' & b);
c    <= ab(n-1 downto 0) + ('0' & ab(n));
```

Field Multiplication in $\mathbb{F}_{2^n-1}$ in VHDL ($c = a \cdot b \bmod p$)

```
ab   <= a * b;
ab_r <= ('0' & ab(n-1 downto 0)) + ('0' & ab(2*n-1 downto n));
c    <= ab_r(n-1 downto 0) + ('0' & ab_r(n));
```

- Works with `NUMERIC_STD` package as well as the proprietary `STD_LOGIC_ARITH` & `STD_LOGIC_UNSIGNED` packages
- If $c < p$ is strictly needed for the addition result, then $c \stackrel{?}{=} p$ needs to be checked after reduction

S-box Cost and Performance Comparison (ASIC)

UCLouvain

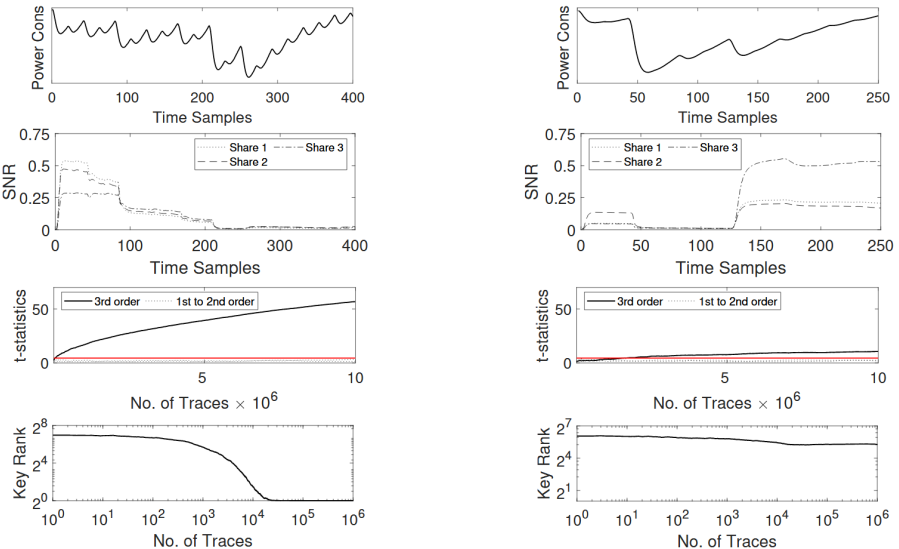
S-box	Shares d	Area [GE]	Crit. Path [ns]	Reg. Stages	Randomn. [bits]
AES	1	697.75	0.83	0	0
	2	4980.25	0.40	6	34
	3	10788.00	0.44	6	102
	4	18485.00	0.51	6	204
`AES-prime`	1	2165.75	1.81	0	0
	2	6449.50	1.93	3	35
	3	16493.75	2.21	4	91
	4	31992.75	1.50	5	210

S-box Cost and Performance Comparison (FPGA)

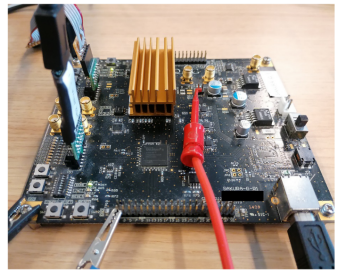
S-box	Shares d	FFs	LUTs	Slic.	DSPs	Crit. Path [ns]	Reg. Stag.
AES	1	0	87	19	0	3.63	0
	2	616	513	101	0	5.08	6
	3	1362	1217	212	0	5.53	6
	4	2400	2203	523	0	6.79	6
AES-prime	1	0	72	24	3	6.99	0
	2	133	495	123	10	5.95	3
	3	364	1207	246	21	6.69	4
	4	1022	2462	487	36	7.02	5
AES-prime*	1	0	204	39	0	5.76	0
	2	133	1095	202	0	5.55	3
	3	364	2589	464	0	6.26	4
	4	1022	4774	834	0	7.13	5

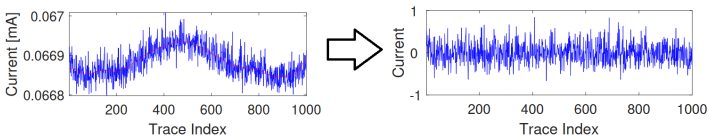
*Use of DSP slices prohibited

S-box SCA Security Comparison (Dynamic Power)

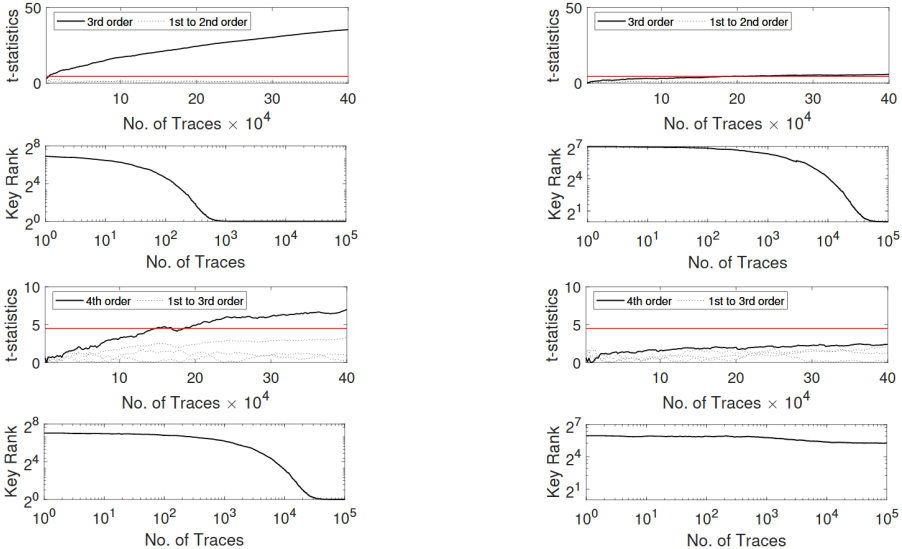


Static Power Measurement Setup





S-box SCA Security Comparison (Static Power)



S-box SCA Security Attack Comparison

Cipher	Shares d	Dynamic Power		Static Power	
		Absolute	Factor	Absolute	Factor
AES	1	2		2	
	2	385		10	
	3	35 084		1 186	
	4	944 390		59 281	
AES-prime	1	3	1.50	3	1.50
	2	2 992	7.77	123	12.30
	3	$> 1\,000\,000$	> 28.50	88 921	74.98
	4	$> 1\,000\,000$	-	$> 100\,000$	-

Conclusion

- Additive masking in small and implementation-friendly prime fields seems promising for efficient physically secure cryptography
- We can mask securely without the need to guarantee a notable amount of noise
- Secure and composable squaring gadgets enable efficient hardware designs
- We demonstrated: Security advantages over Boolean masking can reach orders of magnitude against low-noise attacks in practical experiments
- New dedicated ciphers for efficient masking in prime fields are needed to explore the interest of this design space
- Code: `https://github.com/uclcrypto/prime_field_masking_hardware`