Masking schemes: evaluation

Oscar Reparaz
COSIC/KU Leuven

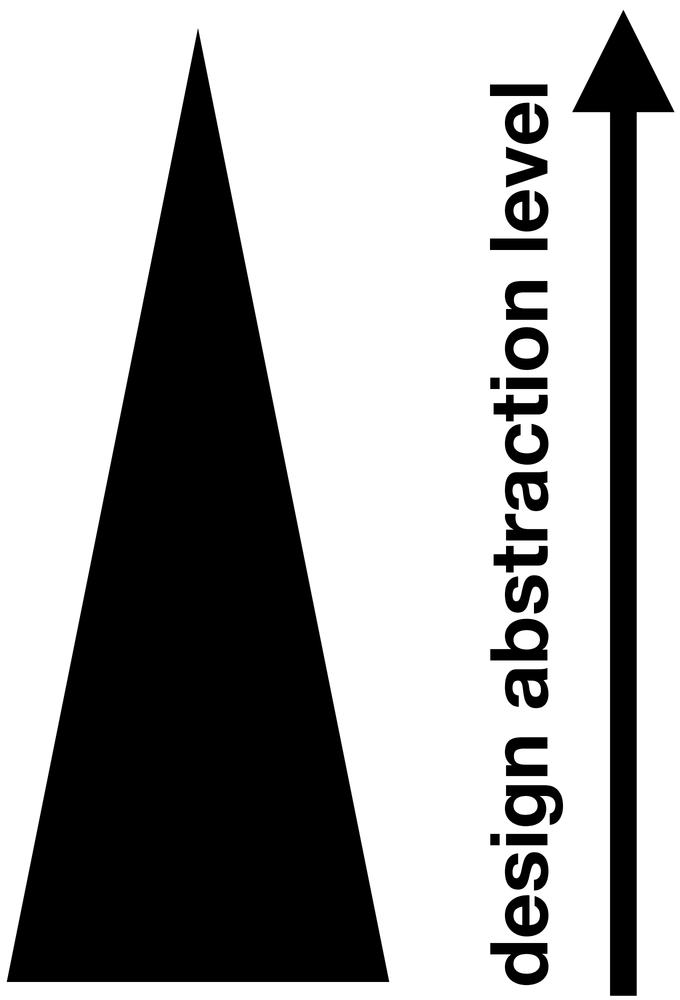
PROOFS
Taipei (Taiwan)
2017-09-29

quick intro to masking

- masking = countermeasure against DPA
- idea: secret sharing $b = b_1 + b_2$
- individual shares tell you nothing about the intermediate
  - power consumption tells you nothing about the intermediate
- main difficulty: compute on masked data
  - AES / RSA / …
- **not as easy as it sounds**

masking common problems

- masking is hard to implement…
  - delicate to implement in SW, delicate to implement in HW
- …but sometimes the scheme is structurally flawed
- …especially tricky in higher-order scenario

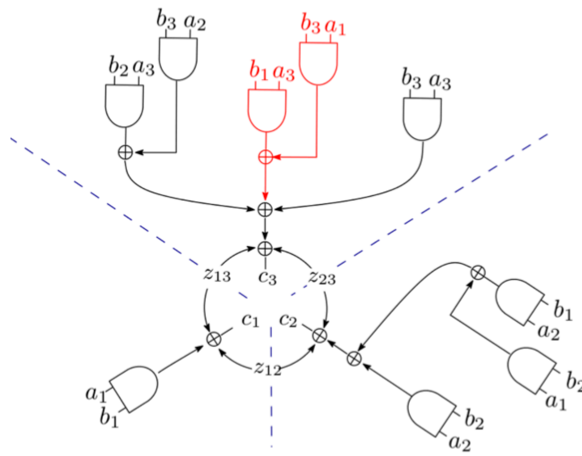


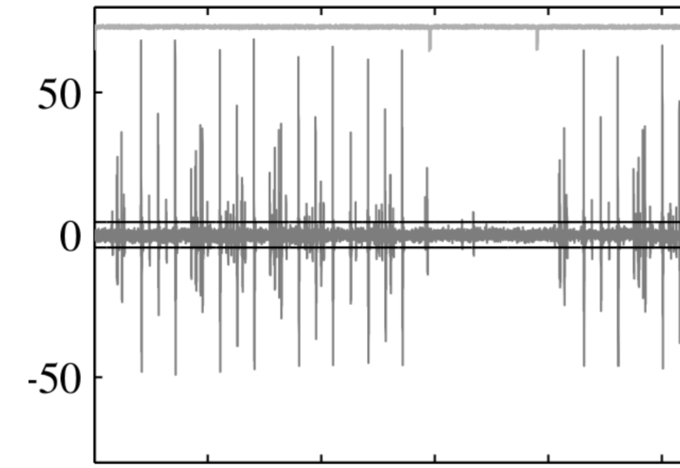
Protocol
* Algorithm
* Architecture: co-design, HW/SW, SoC
Micro-architecture: buses, registers, …
Circuit

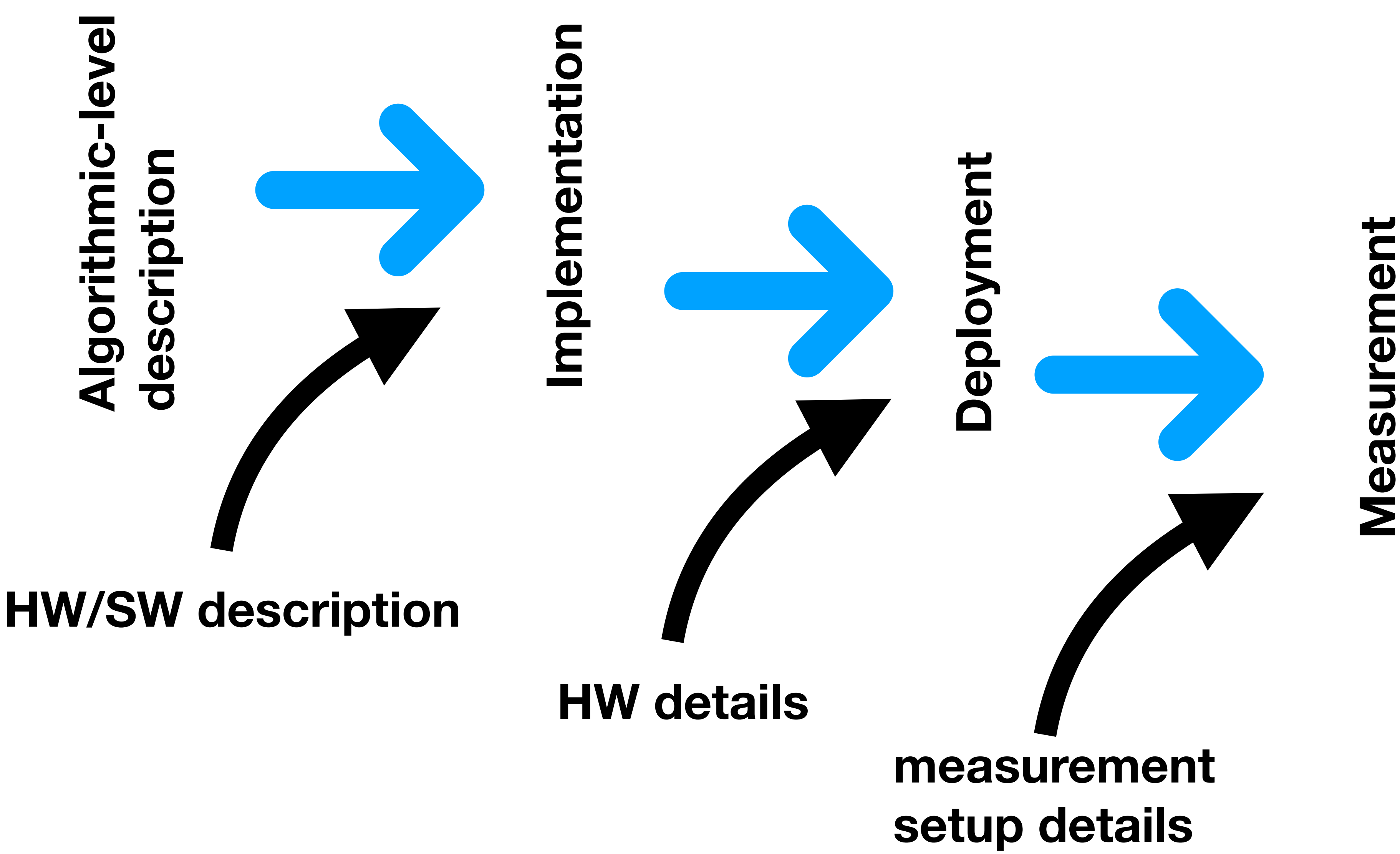
[IEEE Computer 2005]

Require: s-shares $\mathbf{a}$ and $\mathbf{b}$
Ensure: s-shares $\mathbf{c}$ satisfying $c = ab$

```
for i from 1 to s do
    for j from i+1 to s do
        z_ij ← rnd()
        z_ji ← (z_ij ⊕ a_i b_j) ⊕ a_j b_i
    end for
end for
for i from 1 to s do
    c_i ← a_i b_i
    for j from 1 to s, j ≠ i do
        c_i ← c_i ⊕ z_ij
    end for
end for
```

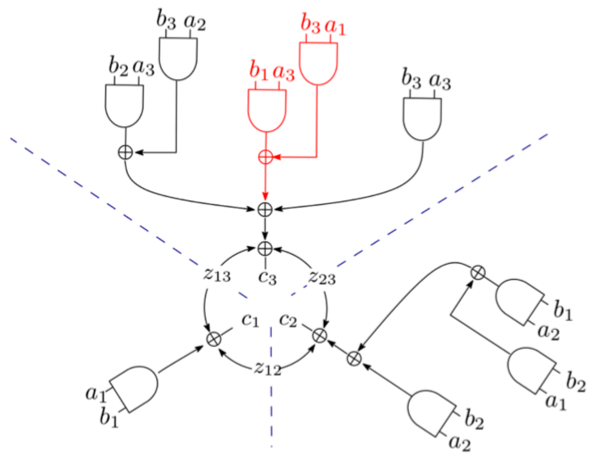


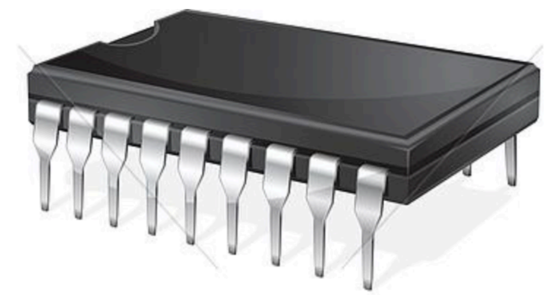


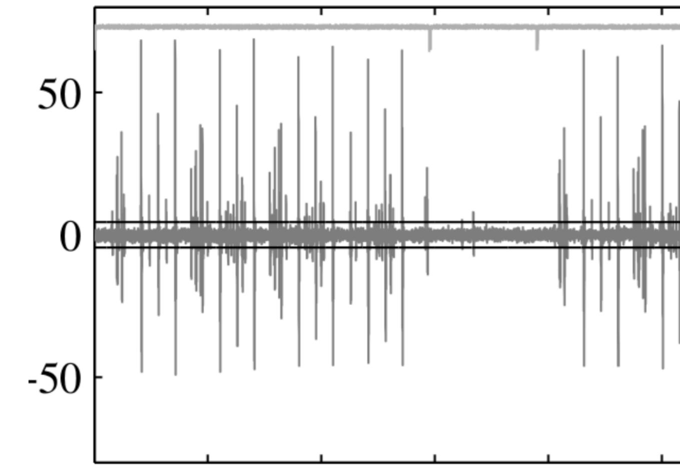


Require: s-shares $\mathbf{a}$ and $\mathbf{b}$
Ensure: s-shares $\mathbf{c}$ satisfying $c = ab$

```
for i from 1 to s do
    for j from i+1 to s do
        z_ij ← rnd()
        z_ji ← (z_ij ⊕ a_i b_j) ⊕ a_j b_i
    end for
end for
for i from 1 to s do
    c_i ← a_i b_i
    for j from 1 to s, j ≠ i do
        c_i ← c_i ⊕ z_ij
    end for
end for
```







“golden standard”,
but maybe too late

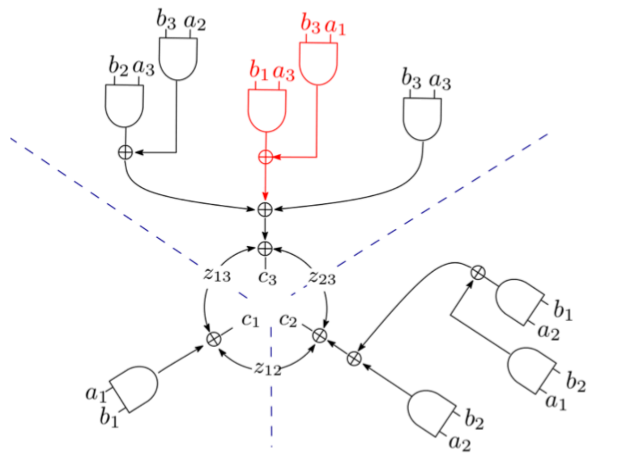
Algorithmic-level description → Implementation → Deployment → Measurement

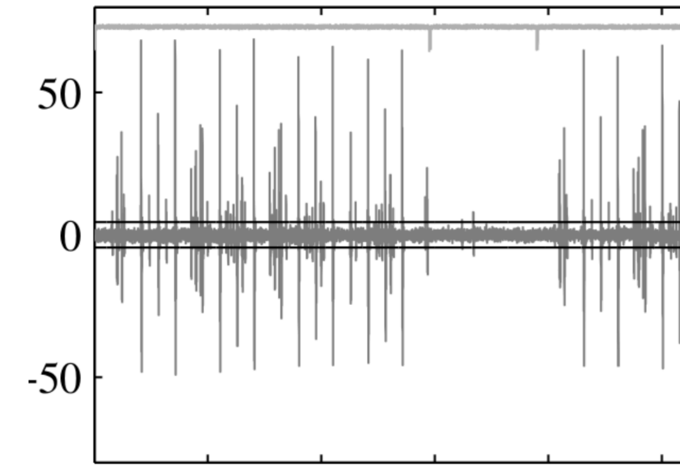
HW/SW description

HW details

measurement setup details

```
Require: s-shares a and b
Ensure: s-shares c satisfying c = ab
    for i from 1 to s do
        for j from i + 1 to s do
            z_ij ← rnd()
            z_ji ← (z_ij ⊕ a_i b_j) ⊕ a_j b_i
        end for
    end for
    for i from 1 to s do
        c_i ← a_i b_i
        for j from 1 to s, j ≠ i do
            c_i ← c_i ⊕ z_ij
        end for
    end for
```





too abstract
very early

Algorithmic-level description → Implementation → Deployment → Measurement

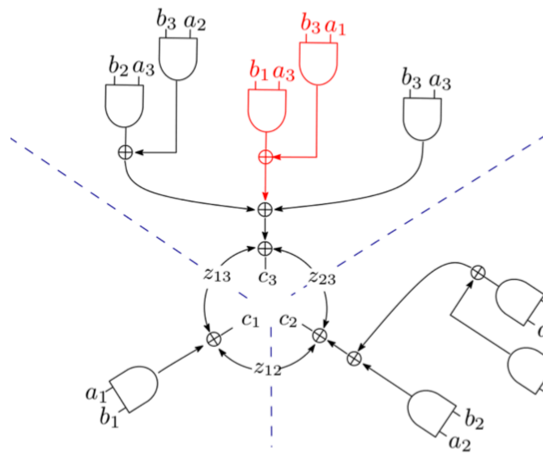
HW/SW description

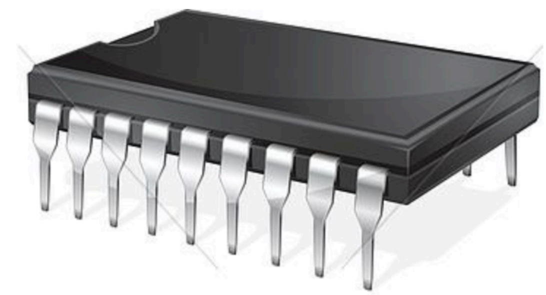
HW details

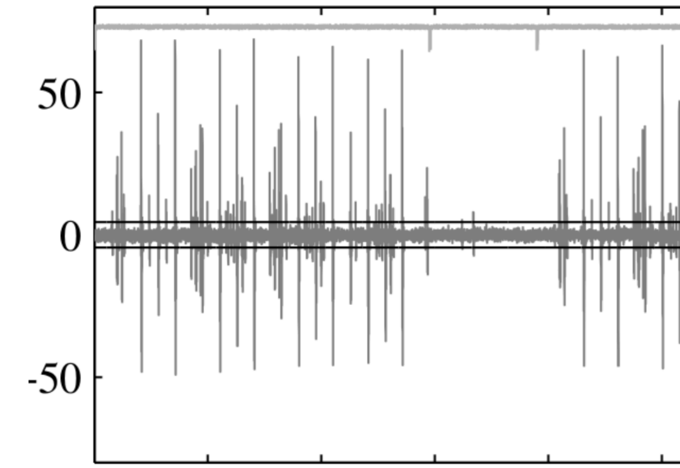
measurement
setup details

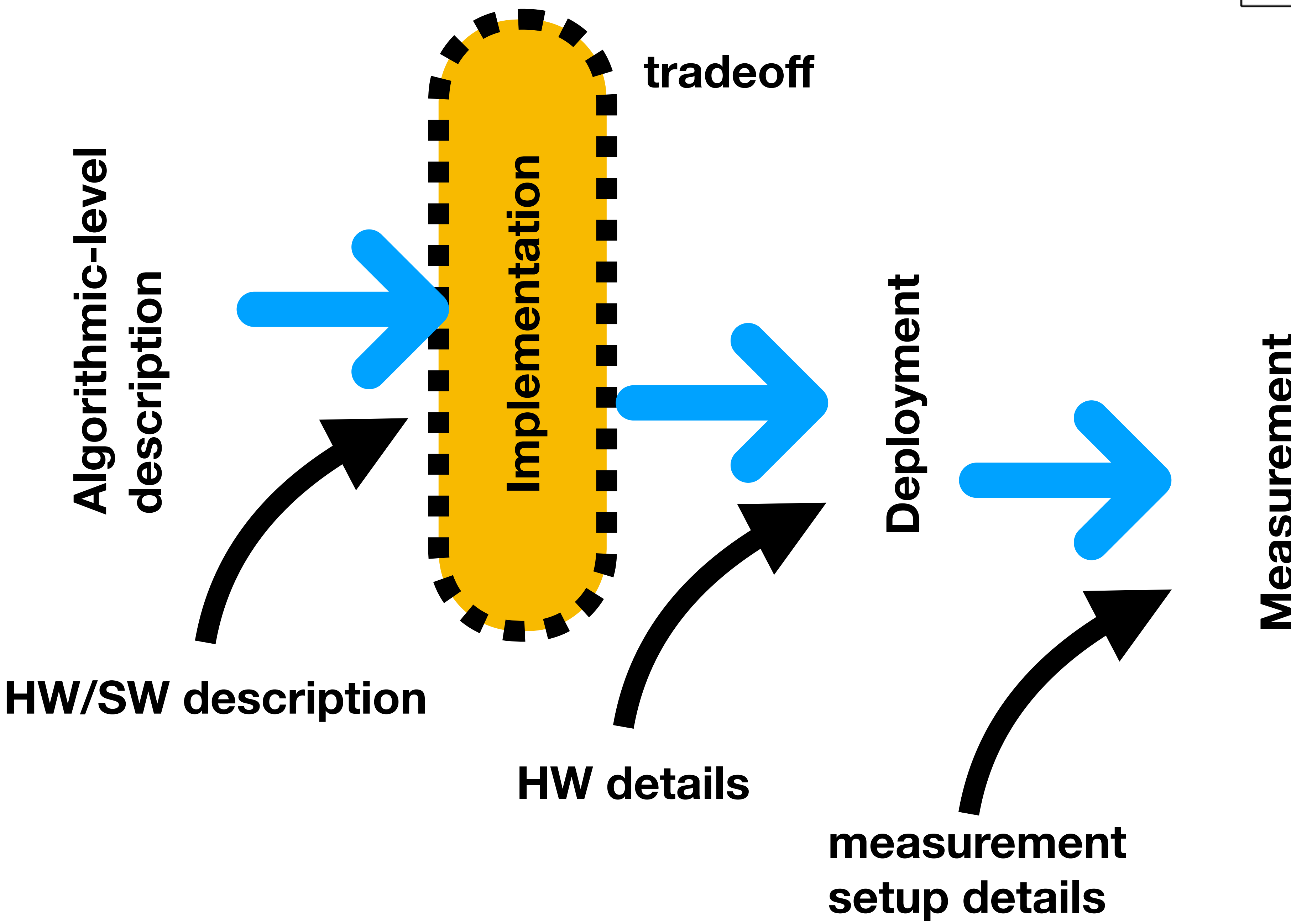
Require: s-shares $\mathbf{a}$ and $\mathbf{b}$
Ensure: s-shares $\mathbf{c}$ satisfying $c = ab$

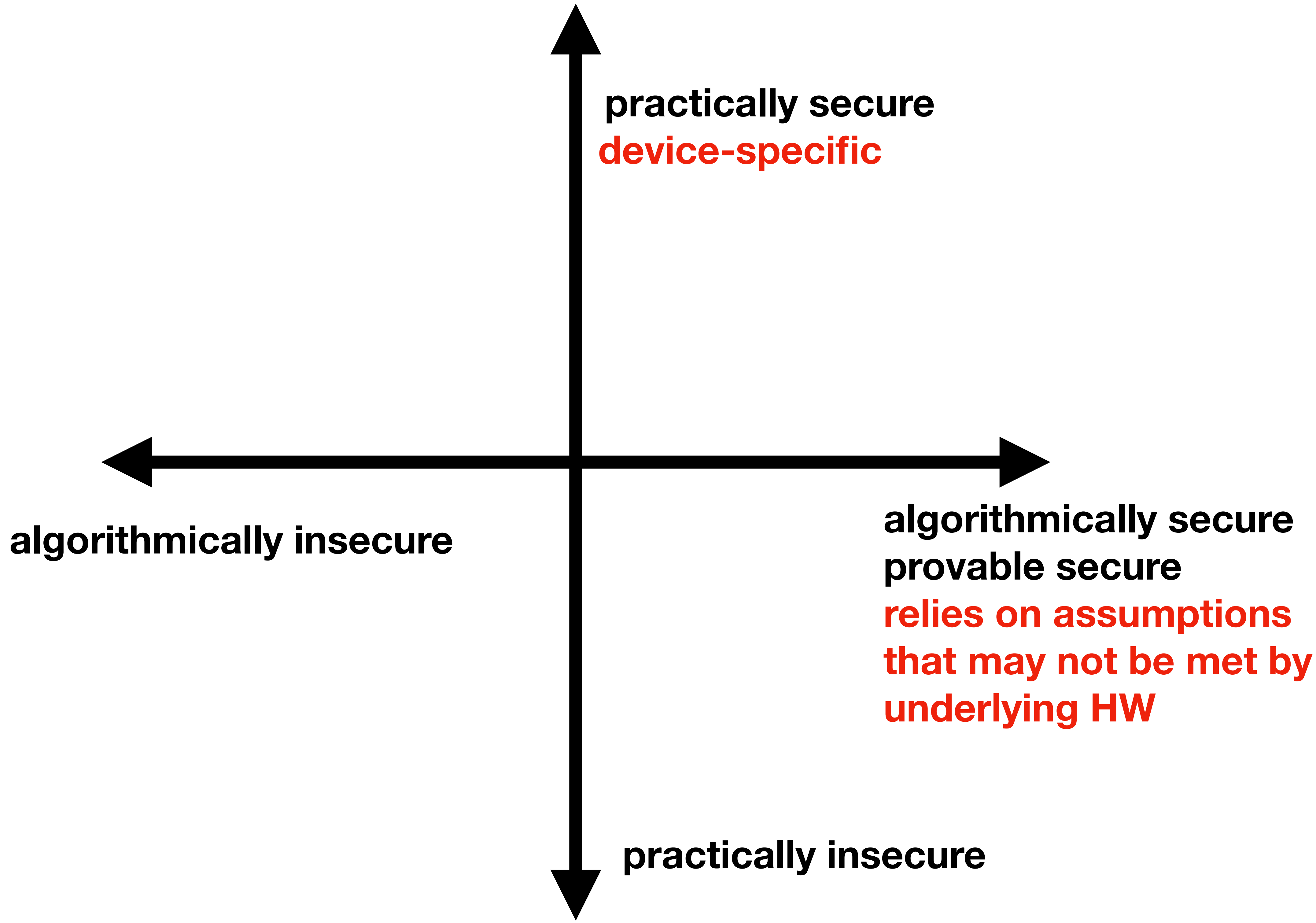
```
for i from 1 to s do
    for j from i+1 to s do
        z_ij ← rnd()
        z_ji ← (z_ij ⊕ a_i b_j) ⊕ a_j b_i
    end for
end for
for i from 1 to s do
    c_i ← a_i b_i
    for j from 1 to s, j ≠ i do
        c_i ← c_i ⊕ z_ij
    end for
end for
```










practically secure
device-specific
algorithmically insecure
algorithmically secure
provable secure
relies on assumptions
that may not be met by
underlying HW
practically insecure

Garden of Eden

practically secure
device-specific

algorithmically insecure

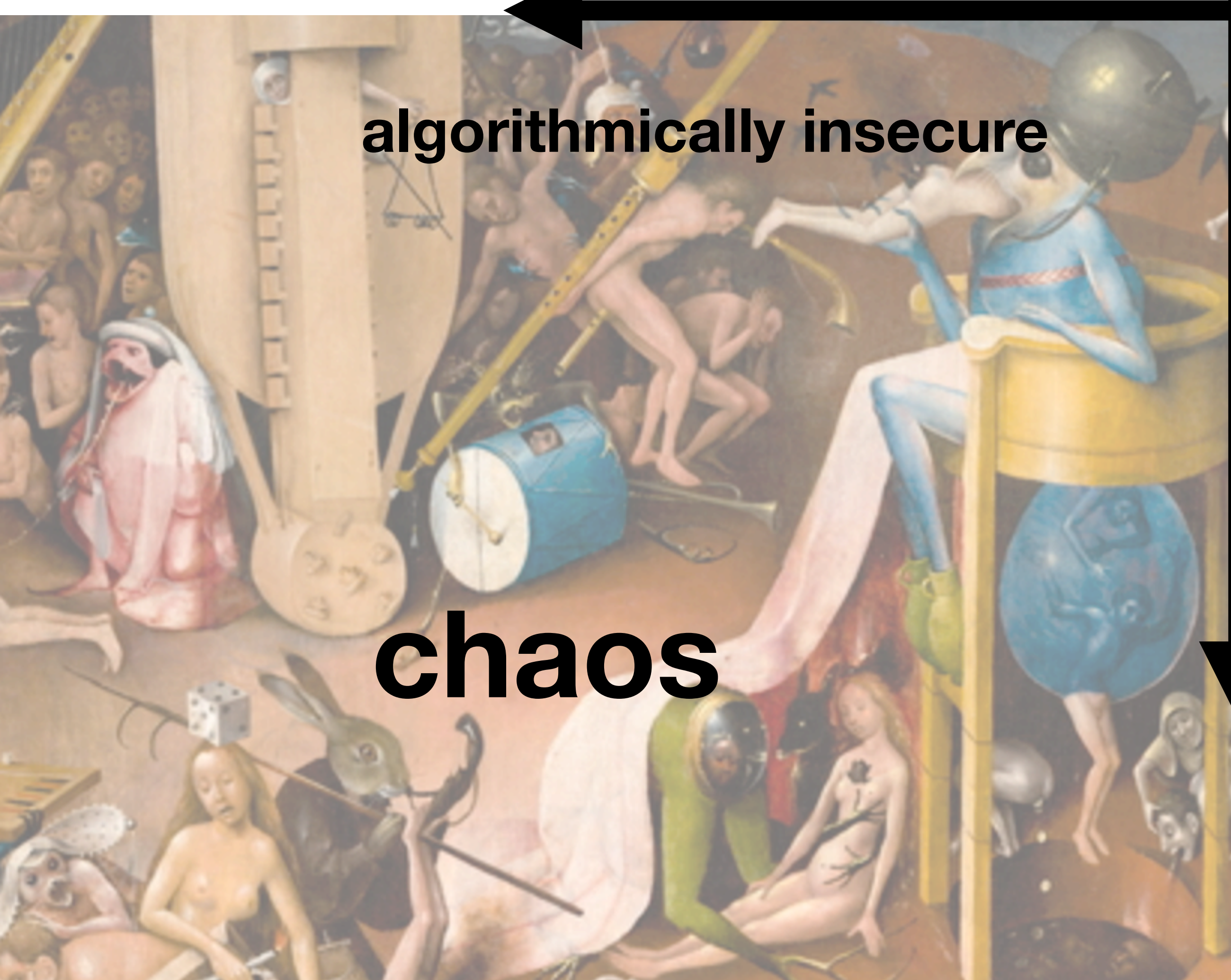
algorithmically secure
provable secure
relies on assumptions

practically insecure

practically secure
device-specific

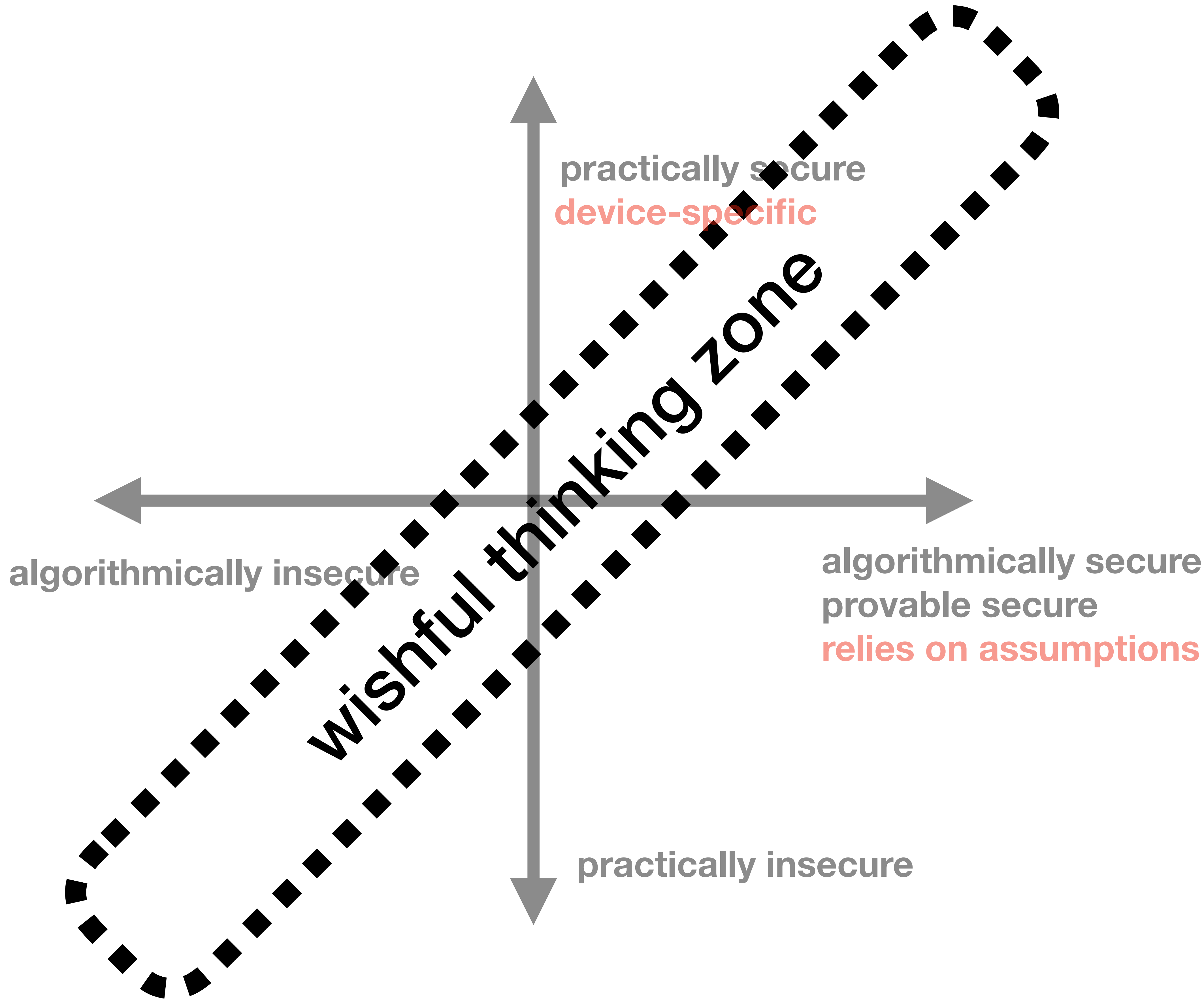
algorithmically insecure

algorithmically secure
provable secure
relies on assumptions



chaos

practically insecure


practically secure
device-specific
wishful thinking zone
algorithmically insecure
algorithmically secure
provable secure
relies on assumptions
practically insecure

practically secure
device-specific

algorithmically insecure

algorithmically secure
provable secure
relies on assumptions

practically insecure

despair

practically secure
device-specific

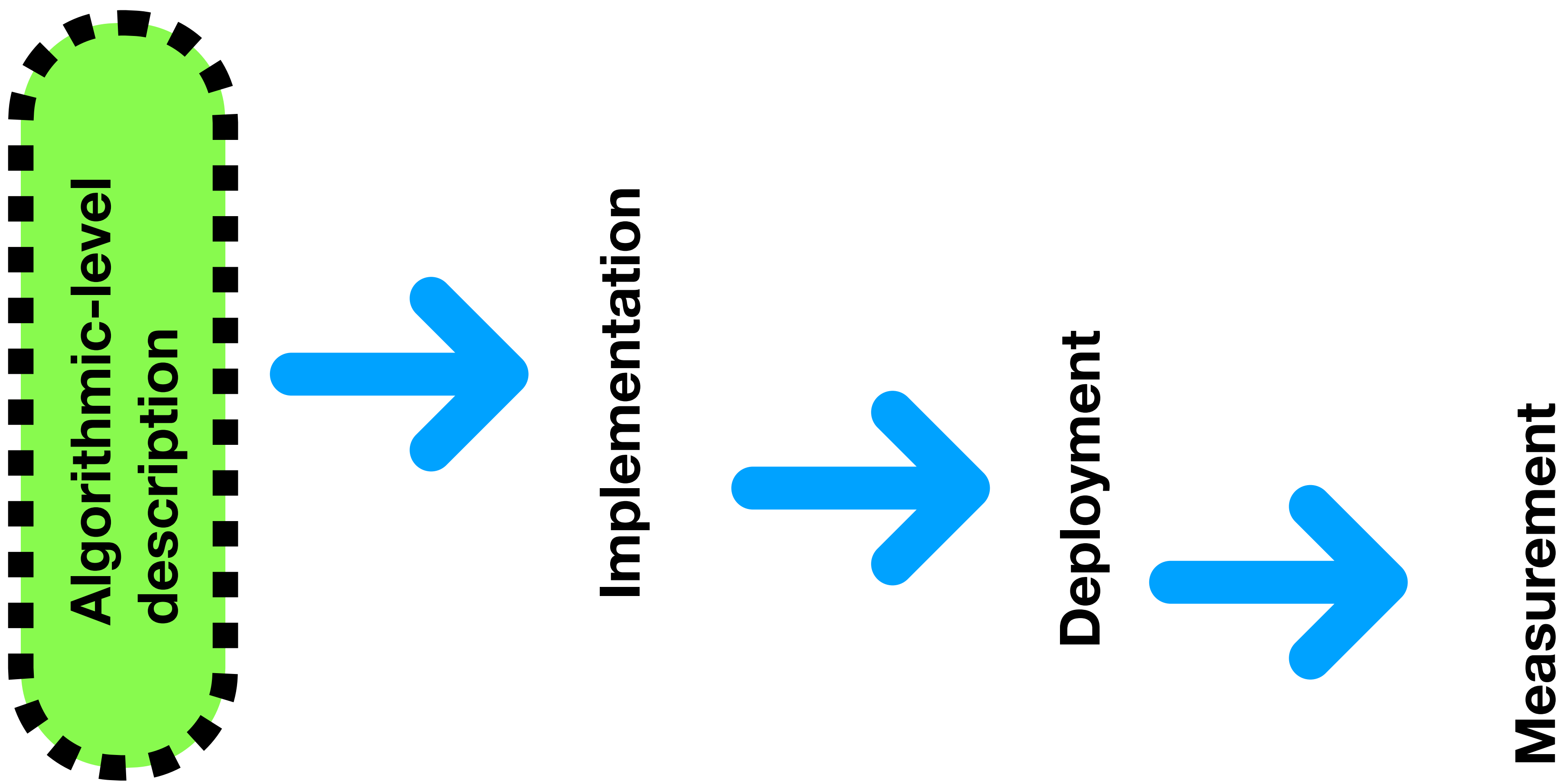
algorithmically insecure

algorithmically secure
provable secure
relies on assumptions

practically insecure

Evaluating masking at design time

[FSE 2016]



timeline/history

Schramm-Paar
Higher-order tables
CT-RSA 2006

Coron-Prouff-Rivain
CHES 2007

Prouff-Giraud-Aumonier
"**provably secure**"
CHES 2006

Coron-Giraud-Prouff-Rivain
CHES 2008

Rivain-Prouff
"**Provably secure**"
CHES 2010

Coron-Prouff-Rivain-Roche
FSE 2013

Balasch-Faust
Gierlichs-Verbauwhede
ASIACRYPT 2012

Prouff-Rivain-Roche
CT-RSA 2014.

Bilgin-Gierlichs-Nikova
Nikov-Rijmen
ASIACRYPT 2014

Reparaz-Bilgin-Nikova
Gierlichs-Verbauwhede
CRYPTO 2015

Algorithm 4 Masked Multiplication: $(\boldsymbol{X},\boldsymbol{Y}) \leftarrow \texttt{IPMult}((\boldsymbol{L},\boldsymbol{R}),(\boldsymbol{K},\boldsymbol{Q}))$

INPUT: Two Masked variables $(\boldsymbol{L},\boldsymbol{R})$ and $(\boldsymbol{K},\boldsymbol{Q})$

OUTPUT: New masked variable $(\boldsymbol{X},\boldsymbol{Y})$ such that $\langle \boldsymbol{X},\boldsymbol{Y}\rangle = \langle \boldsymbol{L},\boldsymbol{R}\rangle \otimes \langle \boldsymbol{K},\boldsymbol{Q}\rangle$

for $i = 0$ **to** $n-1$ **do**
    for $j = 1$ **to** n **do**
        $\tilde{U}_{i*n+j} \leftarrow L_{i+1} \otimes K_j$
        $\tilde{V}_{i*n+j} \leftarrow R_{i+1} \otimes Q_j$
$(\boldsymbol{U},\boldsymbol{V}) \leftarrow \texttt{IPRefresh}(\tilde{\boldsymbol{U}},\tilde{\boldsymbol{V}})$
$\boldsymbol{A} \leftarrow (U_1,\cdots,U_n);\quad \boldsymbol{C} \leftarrow (U_{n+1},\cdots,U_{n^2})$
$\boldsymbol{B} \leftarrow (V_1,\cdots,V_n);\quad \boldsymbol{D} \leftarrow (V_{n+1},\cdots,V_{n^2})$
$Z \leftarrow \langle \boldsymbol{C},\boldsymbol{D}\rangle$
$\boldsymbol{Y} \leftarrow \texttt{IPHalfMask}(Z,\boldsymbol{A})$
$\boldsymbol{X} \leftarrow \boldsymbol{A}$
$\boldsymbol{Y} \leftarrow \boldsymbol{Y} \oplus \boldsymbol{B}$
return $(\boldsymbol{X},\boldsymbol{Y})$

3 A First-Order Flaw

Balasch *et al.* claim that the above IP masking scheme is secure against any side-channel attack of order $d = n-1$, or equivalently, that any family of $n-1$ intermediate variables is independent of any sensitive variable. We contradict this claim hereafter by showing that for any fixed parameter n, there always exists a first-order side-channel attack on the IP masking scheme. To this end, we will exhibit an intermediate variable that is statistically dependent on some sensitive variable in both the `IPRefresh` and `IPAdd` procedures (Algorithms 2 and 3 above).

Let $\boldsymbol{A} = (A_1, A_2, \ldots, A_n)$ and $\boldsymbol{B} = (B_1, B_2, \ldots, B_n)$ be random vectors uniformly distributed over $(\mathbb{F}_q^*)^n$, and let $\boldsymbol{R} = (R_1, R_2, \ldots, R_n)$ be a random vector uniformly distributed over $\mathbb{F}_q^n$, such that $\boldsymbol{A}$, $\boldsymbol{B}$ and $\boldsymbol{R}$ are mutually independent. Consider the function f_n defined by:

$$f_n(a,b) = \Pr[\langle \boldsymbol{A},\boldsymbol{R}\rangle = a \wedge \langle \boldsymbol{B},\boldsymbol{R}\rangle = b] \ . \tag{1}$$

We first study f_n with respect to n before exhibiting the IP masking flaw.

3.1 Study of f_n

The study of f_n developed in this section is recursive. First, in Lemma 1, we give an explicit expression to f_1. Then, in Lemma 2, we exhibit a recursive relationship for f_n. Both lemmas are eventually involved to provide an explicit expression to f_n (Theorem 1).

Lemma 1. *The function f_1 satisfies*

$$f_1(a,b) = \begin{cases} \frac{1}{q} & \text{if } (a,b) = (0,0) \\ 0 & \text{if } (a,b) \in (\{0\}\times\mathbb{F}_q^*) \cup (\mathbb{F}_q^* \times \{0\}) \\ \frac{1}{q(q-1)} & \text{if } (a,b) \in \mathbb{F}_q^* \times \mathbb{F}_q^* \end{cases}$$

Proof. First, since both A_1 and B_1 are non-zero, we have

$$f_1(0,0) \ = \ \Pr[A_1 \otimes R_1 = 0 \wedge B_1 \otimes R_1 = 0] \ = \ \Pr[R_1 = 0] \ = \ \frac{1}{q} \ .$$

Moreover, for any $a \neq 0$, we have

$$f_1(a,0) \ = \ \Pr[R_1 = a \otimes A_1^{-1} \wedge R_1 = 0] \ = \ 0 \ .$$

Similarly, we also have $f(0,b) = 0$ if $b \neq 0$.
Eventually, the total probability law together with the mutual independence between A_1, B_1 and R_1, imply

$$f_1(a,b) \ = \ \sum_{a_1 \in \mathbb{F}_q^*} \Pr[A_1 = a_1] \times \Pr[R_1 = a \otimes a_1^{-1} \wedge B_1 \otimes R_1 = b] \ ,$$

which for $a \neq 0$ and $b \neq 0$ gives

$$f_1(a,b) \ = \ \sum_{a_1 \in \mathbb{F}_q^*} \Pr[A_1 = a_1] \times \Pr[R_1 = a \otimes a_1^{-1} \wedge B_1 = b\,(a^{-1} \otimes a_1)] \ = \ \frac{1}{q(q-1)} \ .$$

□

Lemma 2. *For every $n \geqslant 1$, there exist $f_n^{00}, f_n^{01}, f_n^{11} \in \mathbb{R}$ such that*

$$f_n(a,b) = \begin{cases} f_n^{00} & \text{if } (a,b) = (0,0) \\ f_n^{01} & \text{if } (a,b) \in (\{0\}\times\mathbb{F}_q^*) \cup (\mathbb{F}_q^* \times \{0\}) \\ f_n^{11} & \text{if } (a,b) \in \mathbb{F}_q^* \times \mathbb{F}_q^* \end{cases}$$

Moreover, we have

$$\begin{aligned} f_{n+1}^{00} &= \frac{1}{q} f_n^{00} + \frac{q-1}{q} f_n^{11} \ , \\ f_{n+1}^{01} &= \frac{2}{q} f_n^{01} + \frac{q-2}{q} f_n^{11} \ , \\ f_{n+1}^{11} &= \frac{1}{q(q-1)} f_n^{00} + \frac{2(q-2)}{q(q-1)} f_n^{01} + \frac{(q-1)+(q-2)^2}{q(q-1)} f_n^{11} \ . \end{aligned}$$

Proof. The first statement is true for $n = 1$ by Lemma 1. It is then implied by recurrence from the second statement. Therefore, we only need to show the latter statement.

For every $n > 1$, the total probability law implies

$$f_{n+1}(a,b) = \sum_{(a_0,b_0)\in\mathbb{F}_q^2} f_n(a \oplus a_0, b \oplus b_0) f_1(a_0,b_0) \ . \tag{2}$$

1. For $(a,b) = (0,0)$, the terms in the sum (2) are of the form $f_n(a_0,b_0)f_1(a_0,b_0)$. Then by Lemma 1, we get

$$f_n(a_0,b_0)f_1(a_0,b_0) = \begin{cases} \frac{1}{q} f_n(0,0) & \text{if } (a_0,b_0) = (0,0) \\ 0 & \text{if } (a_0,b_0) \in (\{0\}\times\mathbb{F}_q^*) \cup (\mathbb{F}_q^* \times \{0\}) \\ \frac{1}{q(q-1)} f_n(a_0,b_0) & \text{if } (a_0,b_0) \in \mathbb{F}_q^* \times \mathbb{F}_q^* \end{cases}$$

We deduce

$$f_{n+1}(a,b) = \frac{1}{q} f_n^{00} + (q-1)^2 \frac{1}{q(q-1)} f_n^{11} \ . \tag{3}$$

2. For $(a,b) \in \{0\} \times \mathbb{F}_q^*$, the terms in the sum (2) are of the form $f_n(a_0, b \oplus b_0) f_1(a_0, b_0)$, with $b \neq 0$. Then by Lemma 1, we get

$$f_n(a_0, b \oplus b_0) f_1(a_0, b_0) = \begin{cases} \frac{1}{q} f_n(0,b) & \text{if } (a_0,b_0) = (0,0) \\ 0 & \text{if } (a_0,b_0) \in (\{0\} \times \mathbb{F}_q^*) \cup (\mathbb{F}_q^* \times \{0\}) \\ \frac{1}{q(q-1)} f_n(a_0,0) & \text{if } (a_0,b_0) \in \mathbb{F}_q^* \times \{b\} \\ \frac{1}{q(q-1)} f_n(a_0,b_0) & \text{if } (a_0,b_0) \in \mathbb{F}_q^* \times (\mathbb{F}_q^* \backslash \{b\}) \end{cases}$$

We deduce

$$f_{n+1}(a,b) = \frac{1}{q} f_n^{01} + (q-1) \frac{1}{q(q-1)} f_n^{01} + (q-1)(q-2) \frac{1}{q(q-1)} f_n^{11} \ . \tag{4}$$

For $(a,b) \in \mathbb{F}_q^* \times \{0\}$, we have the same equality by symmetry of the function f_n.

3. For $(a,b) \in \mathbb{F}_q^* \times \mathbb{F}_q^*$, the terms in the sum (2) are of the form $f_n(a \oplus a_0, b \oplus b_0) f_1(a_0, b_0)$, with $a \neq 0$ and $b \neq 0$. Then by Lemma 1, we get

$$f_n(a \oplus a_0, b \oplus b_0) f_1(a_0, b_0) = \begin{cases} \frac{1}{q} f_n(a,b) & \text{if } (a_0,b_0) = (0,0) \\ \frac{1}{q(q-1)} f_n(0,0) & \text{if } (a_0,b_0) = (a,b) \\ 0 & \text{if } (a_0,b_0) \in (\{0\} \times \mathbb{F}_q^*) \cup (\mathbb{F}_q^* \times \{0\}) \\ \frac{1}{q(q-1)} f_n(a \oplus a_0, 0) & \text{if } (a_0,b_0) \in (\mathbb{F}_q^* \backslash \{a\}) \times \{b\} \\ \frac{1}{q(q-1)} f_n(0, b \oplus b_0) & \text{if } (a_0,b_0) \in \{a\} \times (\mathbb{F}_q^* \backslash \{b\}) \\ \frac{1}{q(q-1)} f_n(a \oplus a_0, b \oplus b_0) & \text{if } (a_0,b_0) \in (\mathbb{F}_q^* \backslash \{a\}) \times (\mathbb{F}_q^* \backslash \{b\}) \end{cases}$$

We deduce

$$f_{n+1}(a,b) = \frac{1}{q} f_n^{11} + \frac{1}{q(q-1)} f_n^{00} + 2\Big((q-2) \frac{1}{q(q-1)} f_n^{01}\Big) + (q-2)^2 \frac{1}{q(q-1)} f_n^{11} \ . \tag{5}$$

Equations (3), (4) and (5) directly yield the second statement. ☐

Theorem 1. *For every $n \geqslant 1$ we have*

$$f_n(a,b) = \begin{cases} \frac{1}{q^2} + \frac{1}{q^2(q-1)^{n-2}} & \text{if } (a,b) = (0,0) \\ \frac{1}{q^2} - \frac{1}{q^2(q-1)^{n-1}} & \text{if } (a,b) \in (\{0\} \times \mathbb{F}_q^*) \cup (\mathbb{F}_q^* \times \{0\}) \\ \frac{1}{q^2} + \frac{1}{q^2(q-1)^{n}} & \text{if } (a,b) \in \mathbb{F}_q^* \times \mathbb{F}_q^* \end{cases}$$

Proof. From Lemma 2, we have

$$\begin{pmatrix} f_{n+1}^{00} \\ f_{n+1}^{01} \\ f_{n+1}^{11} \end{pmatrix} = \begin{pmatrix} \frac{1}{q} & 0 & \frac{q-1}{q} \\ 0 & \frac{2}{q} & \frac{q-2}{q} \\ \frac{1}{q(q-1)} & \frac{2(q-2)}{q(q-1)} & \frac{(q-1)+(q-2)^2}{q(q-1)} \end{pmatrix} \cdot \begin{pmatrix} f_n^{00} \\ f_n^{01} \\ f_n^{11} \end{pmatrix} = P \cdot \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & \frac{1}{q-1} \end{pmatrix} \cdot P^{-1} \cdot \begin{pmatrix} f_n^{00} \\ f_n^{01} \\ f_n^{11} \end{pmatrix} \tag{6}$$

where P is the matrix of eigenvectors which satisfies

$$P = \begin{pmatrix} 1 & 1-q & q^2-2q+1 \\ 1 & \frac{1}{2}(2-q) & 1-q \\ 1 & 1 & 1 \end{pmatrix}$$

By recursively applying (6), we can express $(f_n^{00}, f_n^{01}, f_n^{11})$ with respect to $(f_1^{00}, f_1^{01}, f_1^{11})$ as

$$\begin{pmatrix} f_n^{00} \\ f_n^{01} \\ f_n^{11} \end{pmatrix} = P \cdot \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & \frac{1}{(q-1)^{n-1}} \end{pmatrix} \cdot P^{-1} \cdot \begin{pmatrix} f_1^{00} \\ f_1^{01} \\ f_1^{11} \end{pmatrix}$$

Finally, by Lemma 1 we have $(f_1^{00}, f_1^{01}, f_1^{11}) = \big(\frac{1}{q}, 0, \frac{1}{q(q-1)}\big)$, which together with the above equation yields the theorem statement. ☐

3.2 Application to the IP Masking Scheme

The flaw occurs in the mask-refreshing procedure `IPRefresh` and in the addition procedure `IPAdd` (see in Algorithm 2 and Algorithm 3). For the sake of clarity, we first detail it in the `IPRefresh` setting and then show it occurs as well in the `IPAdd` procedure.

Flaw in mask-refreshing procedure. The `IPRefresh` procedure takes an IP masking $(\boldsymbol{L}, \boldsymbol{R})$ of some sensitive variable V (*i.e.* such that $V = \langle \boldsymbol{L}, \boldsymbol{R} \rangle$), and it returns a fresh masking $(\boldsymbol{L}', \boldsymbol{R}')$ such that $V = \langle \boldsymbol{L}', \boldsymbol{R}' \rangle$. The first step of the procedure consists in randomly picking some vector $\boldsymbol{A} \in \mathbb{F}_q^n$ such that $A_i \neq L_i$ for every i. Then one computes $\boldsymbol{L}' = \boldsymbol{L} \oplus \boldsymbol{A}$ and $X = \langle \boldsymbol{A}, \boldsymbol{R} \rangle$. Note that $\boldsymbol{L}$ and $\boldsymbol{L}'$ are mutually independent and both uniformly distributed over $(\mathbb{F}_q^*)^n$. We show hereafter that X leaks information on the sensitive variable V. Indeed we have

$$\Pr[X = x \mid V = v] = \frac{\Pr[V = v \wedge X = x]}{\Pr[V = v]} = \frac{\Pr[V = v \wedge X \oplus V = x \oplus v]}{\Pr[V = v]} \ .$$

Then from

$$\Pr[V = v \wedge X \oplus V = x \oplus v] = \Pr[\langle \boldsymbol{L}, \boldsymbol{R} \rangle = v \wedge \langle \boldsymbol{L}', \boldsymbol{R} \rangle = x \oplus v] = f_n(v, x \oplus v) \ ,$$

we get

$$\Pr[X = x \mid V = v] = \frac{f_n(v, x \oplus v)}{\Pr[V = v]} \ . \tag{7}$$

By Theorem 1 and given that $\Pr[V = v] = \frac{1}{q}$, (7) gives

$$\Pr[X = x \mid V = v] = \begin{cases} \frac{1}{q} + \frac{1}{q(q-1)^{n-2}} & \text{if } x = 0 \\ \frac{1}{q} - \frac{1}{q(q-1)^{n-1}} & \text{if } x \neq 0 \end{cases}$$

for $v = 0$, and

$$\Pr[X = x \mid V = v] = \begin{cases} \frac{1}{q} - \frac{1}{q(q-1)^{n-1}} & \text{if } x = v \\ \frac{1}{q} + \frac{1}{q(q-1)^{n}} & \text{if } x \neq v \end{cases}$$

otherwise.

We see that when the sensitive variable V equals 0, then the intermediate variable X is more likely to equal 0 than another value in $\mathbb{F}_q$. On the other hand, when V does not equal 0, the sensitive variable X is more likely to be any value of $\mathbb{F}_q$ but v. Although the bias is exponentially small in n, for small values of n it may induce a significant information leakage.

leakage assessment

Security notions

key recovery

Can an adversary extract the key?

“pragmatic” security notion

≈ DPA

Security notions

key recovery

😈

Can an adversary
extract the key?

“pragmatic” security notion

≈ DPA

(in)distinguishability

😈

Can an adversary
tell the two devices apart?

“stronger” security notion

≈ leakage assessment

Statistics and Secret Leakage

FC 2000

JEAN-SEBASTIEN CORON and DAVID NACCACHE
Gemplus
and
PAUL KOCHER
Cryptography Research, Inc.

Leakage assessment review

measurement setup

A. Take N measurements for each plaintext class

distribution statistic

B. For each class, describe the trace distribution

  A. normally use some descriptive statistic: mean, variances, skewness, kurtosis, …

statistical test

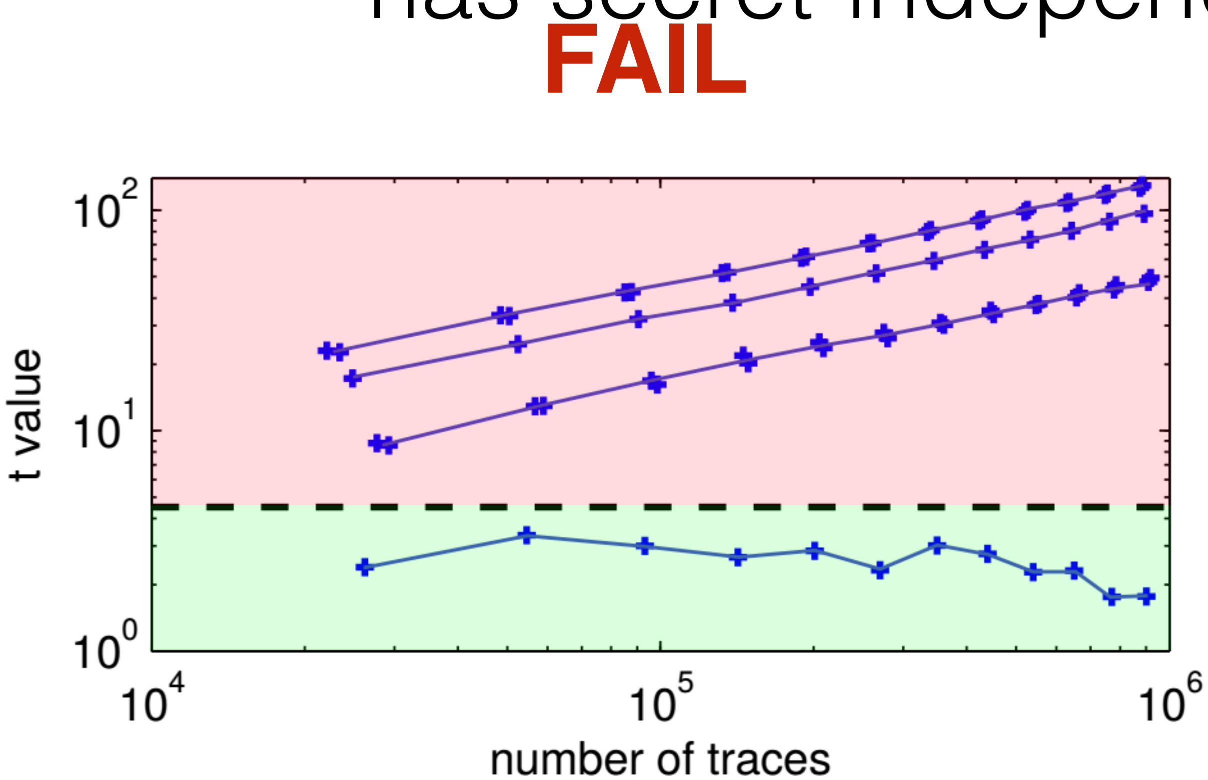
C. Compare the class-dependent statistics

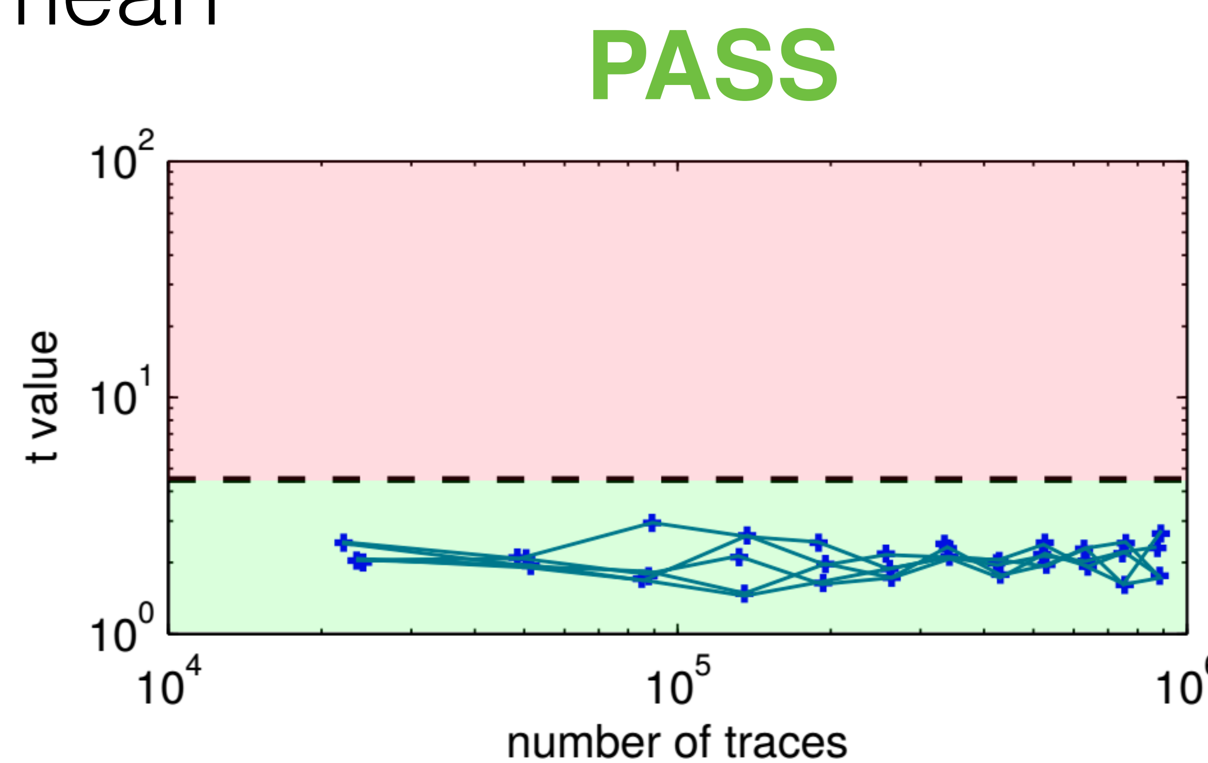
  A. If significant difference -> fail test

  B. Otherwise: “pass”

principle of operation

- Leakage detection test on simulated measurements
    - Statistically test if the distribution of each variable has secret-independent mean





Statistics and Secret Leakage

JEAN-SEBASTIEN CORON and DAVID NACCACHE
Gemplus
and
PAUL KOCHER
Cryptography Research, Inc.

In addition to its usual complexity assumptions, cryptography silently assumes that information can be physically protected in a single location. As one can easily imagine, real-life devices are not ideal and information may leak through different physical channels.

This paper gives a rigorous definition of leakage immunity and presents several leakage detection tests. In these tests, failure *confirms* the probable existence of secret-correlated emanations and indicates how likely the leakage is. Success *does not refute* the existence of emanations but indicates that significant emanations were not detected *on the strength of the evidence presented*, which of course, leaves the door open to reconsider the situation if further evidence comes to hand at a later date.

More heuristics

- Scale down algorithm
  - test first small instances: smaller bit-width, smaller fields. Biases normally more pronounced in smaller fields
  - smaller rounds, combine components
- Deactivate parts of plaintext
- Carefully pick input texts: fixed points, or inputs that are specially handled
  - AES sbox input 0

```
70 void MaskRefresh(u8 *s) {
71  u8 r;
72  for (int i = 1; i < number_shares; i++) {
73   r = rnd ();
74   s[0] ^= r;
75   s[i] ^= r;
76  }
77 }
...
110 void SecMult (u8 *out, u8 *a, u8 *b) {
111  u8 aibj,ajbi;
...
114  for (int i = 0; i < number_shares; i++) {
115   for (int j = i + 1; j < number_shares; j++) {
...
119    aibj = mult(a[i], b[j]);
120    ajbi = mult(a[j], b[i]);
----------------------------------------------------
$ ./run
entering fixed_vs_fixed(00,01)
> leakage detected with 1.20k traces
  higher order leakage between
    line 74 and
    line 120
  with tvalue of -7.03
```

results

- reproduced previous attacks

Schramm-Paar
Higher-order tables
CT-RSA 2006

Coron-Prouff-Rivain
CHES 2007

Rivain-Prouff
"**Provably secure**"
CHES 2010

Coron-Prouff-Rivain-Roche
FSE 2013

Balasch-Faust
Gierlichs-Verbauwhede
ASIACRYPT 2012

Prouff-Rivain-Roche
CT-RSA 2014.

Bilgin-Gierlichs-Nikova
Nikov-Rijmen
ASIACRYPT 2014

Reparaz-Bilgin-Nikova
Gierlichs-Verbauwhede
CRYPTO 2015

- new second-order flaw on Schramm-Paar when unbalanced sboxes

first-, second- and third-order attacks

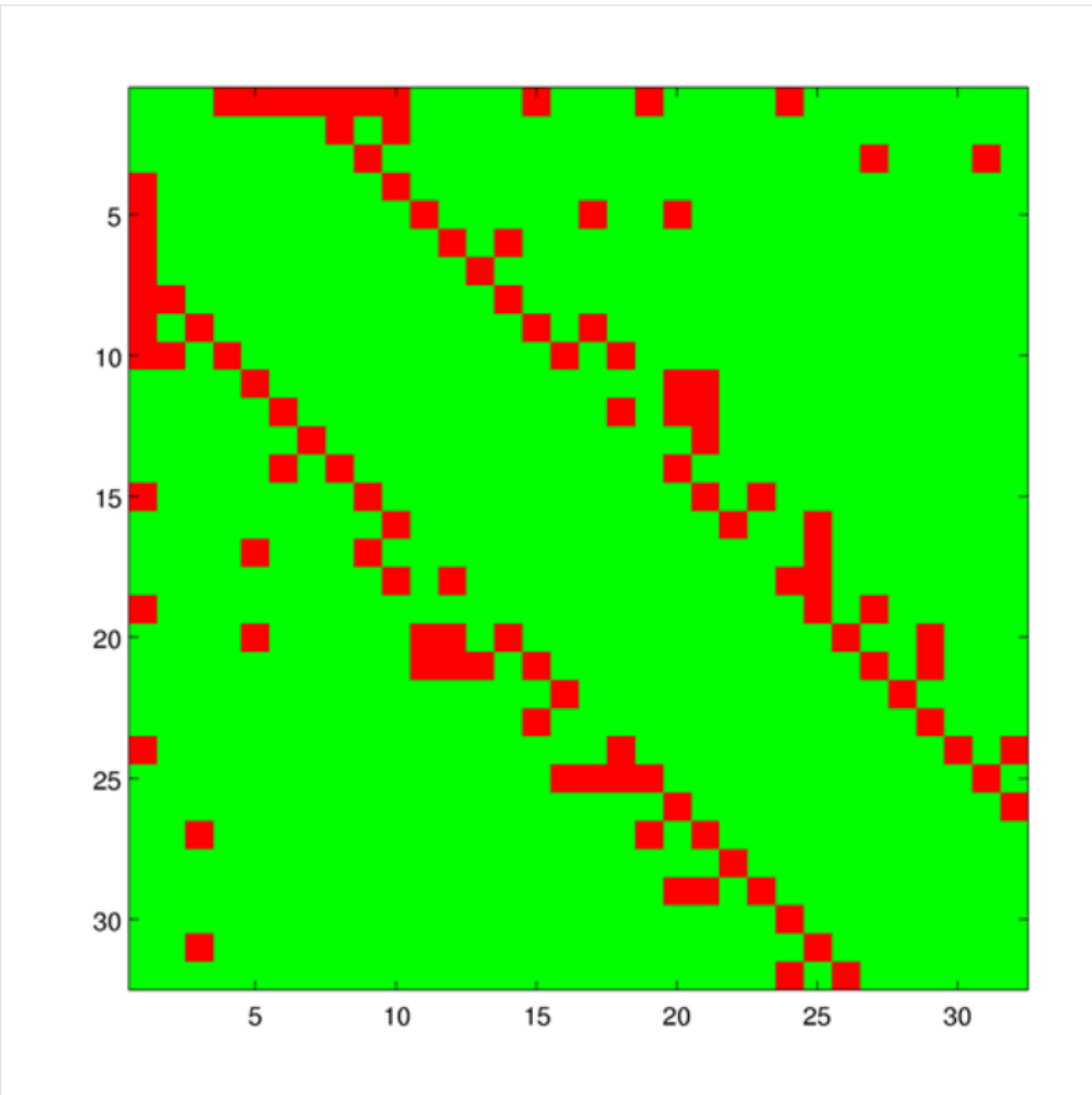


Fig. 5: Pairs of rounds with $|t| > 80$

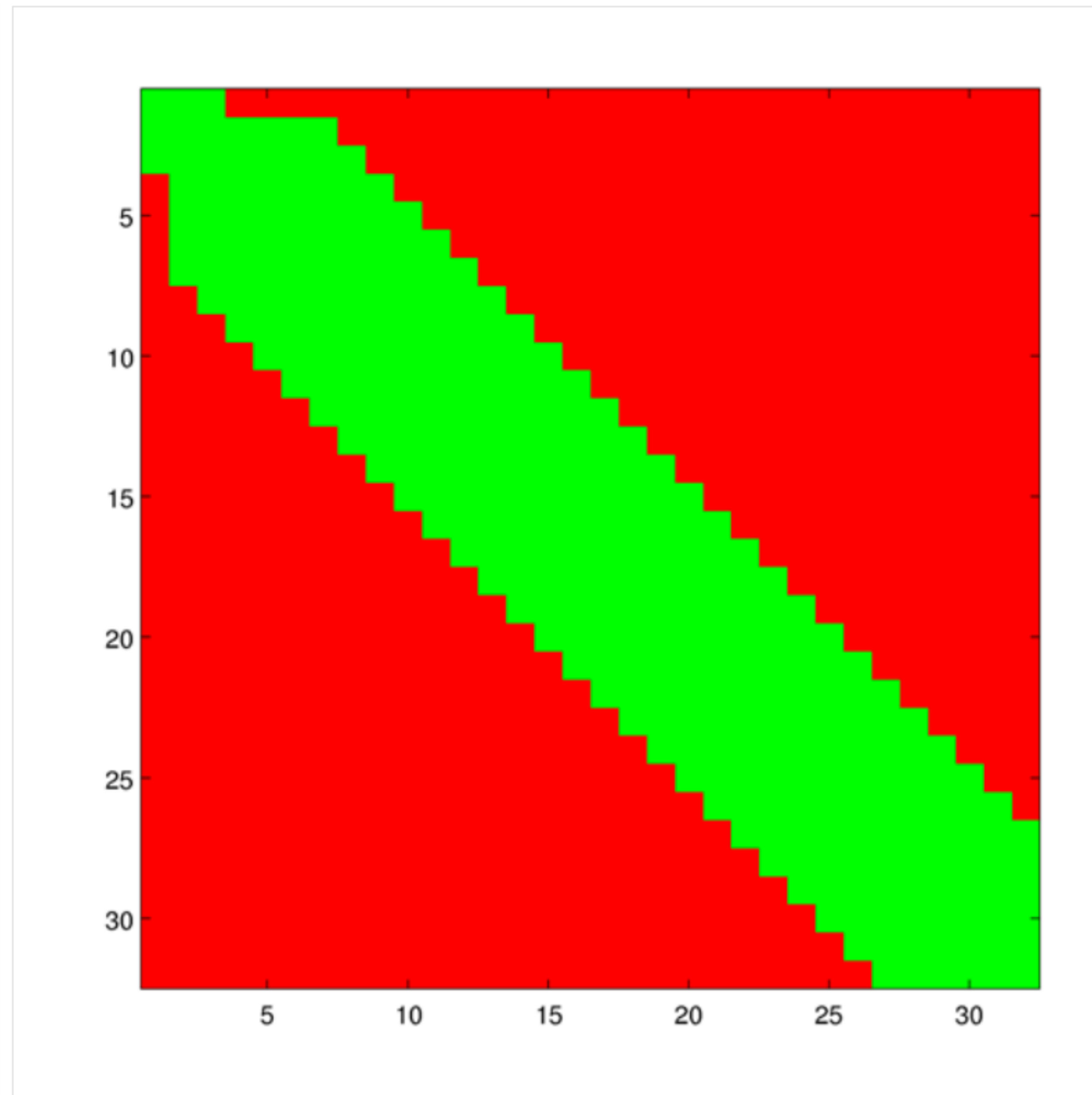


Fig. 6: Pairs of rounds with $|t| > 5$

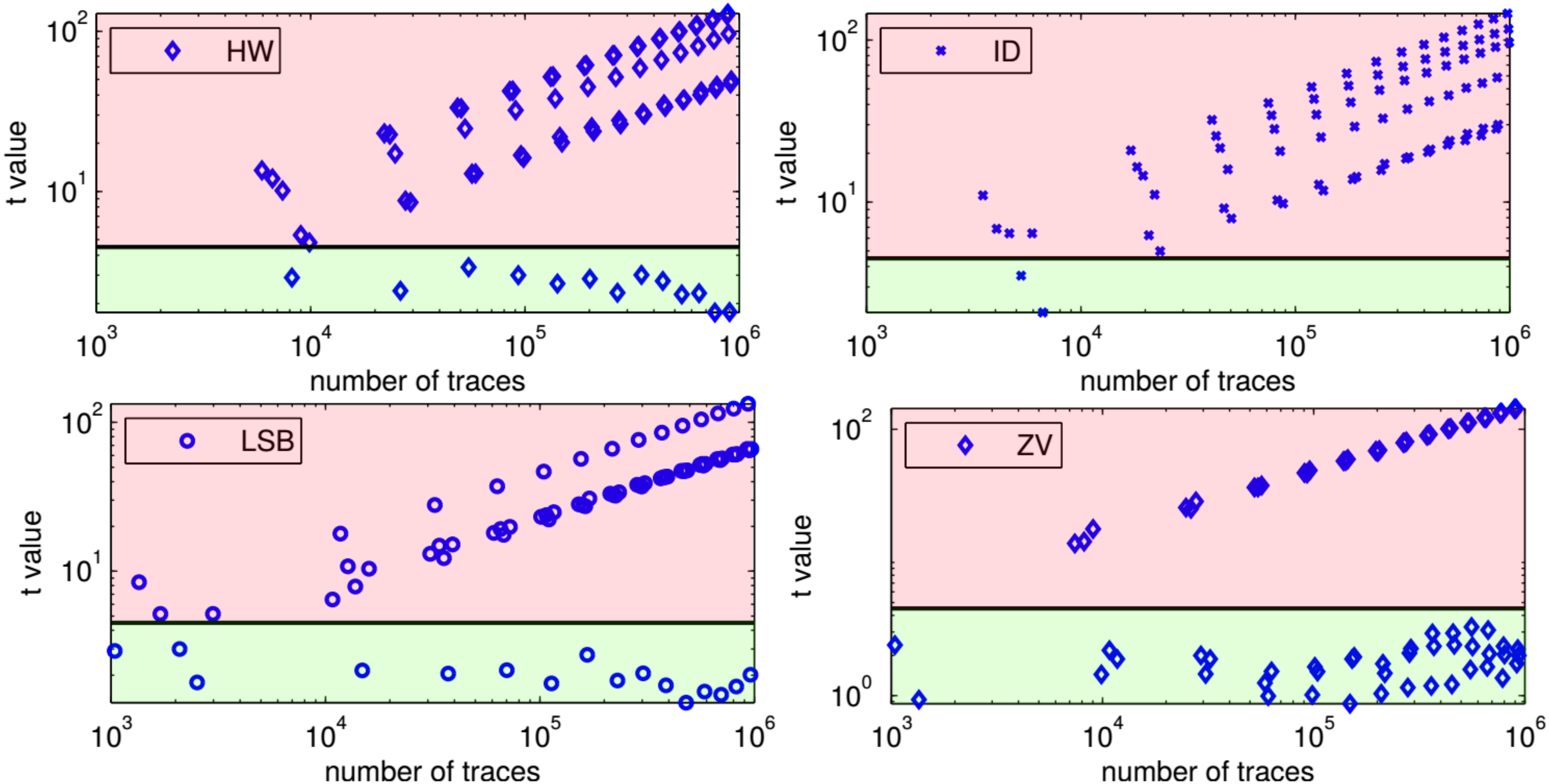


Fig. 9: Influence of leakage function.

Software and hardware

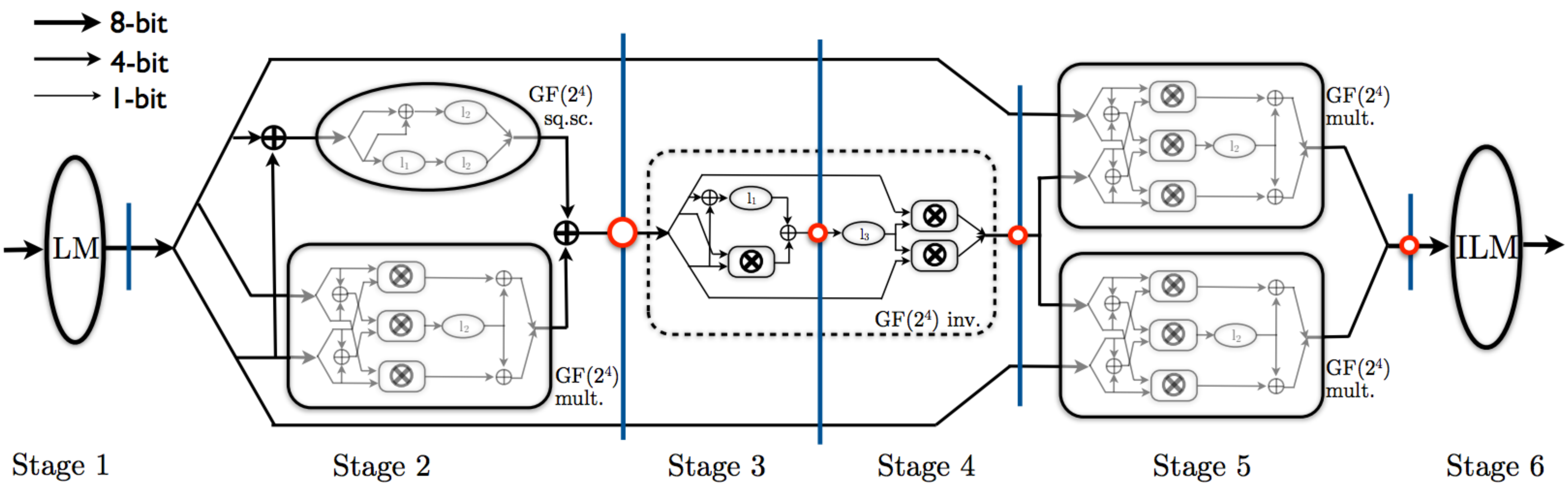


Fig. 7: Higher-order masked AES sbox from de Cnudde et al.

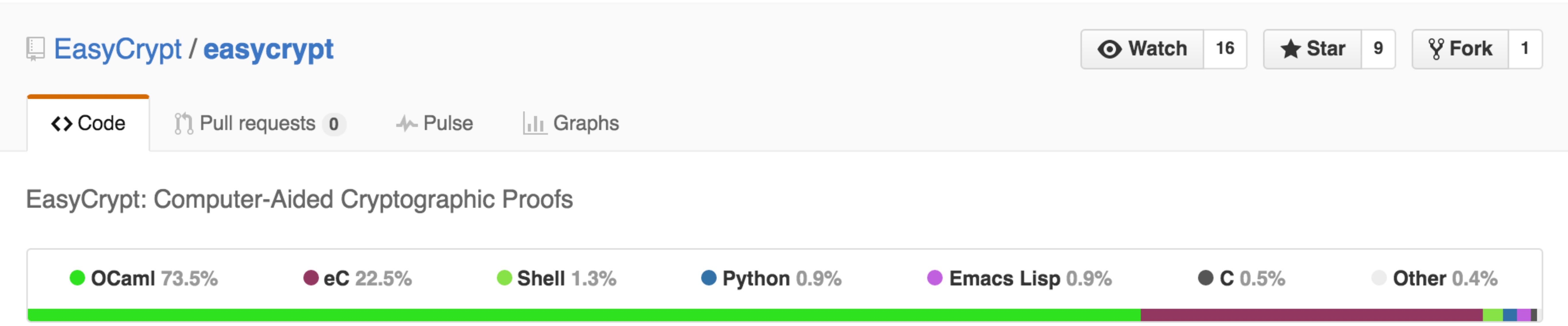
performance

Scheme	Flaw order	Field size	Time	Traces needed
IP	1	4	0.04s	1k
RP	2	4	5s	14k
SP	3	4	0.2s	2k

Fig. 8: Running time to discover flaw in the studied schemes, and number of traces needed to detect the bias.

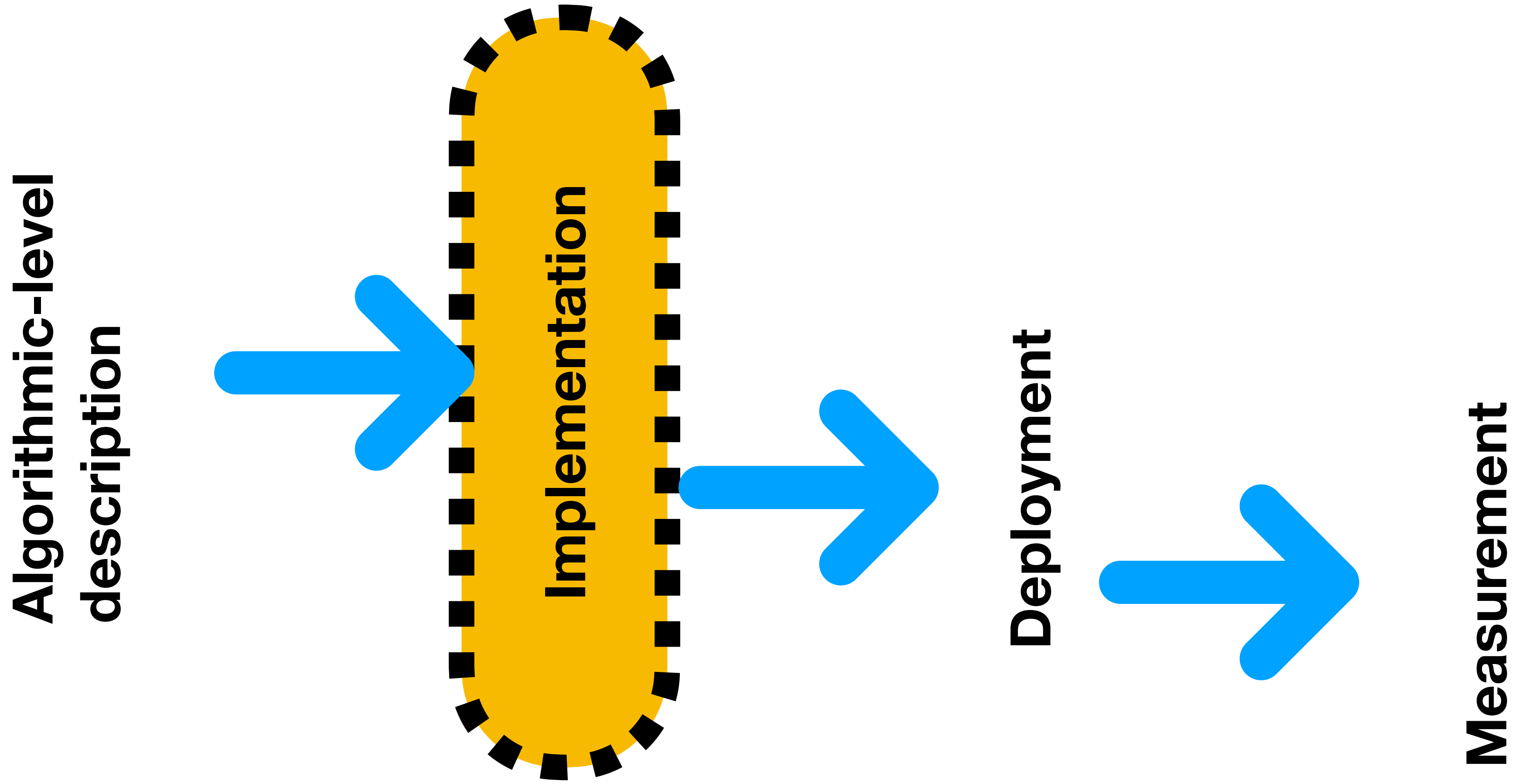
comparision with other approaches

- easycrypt: impressive scientific + engineering achievement



- **188k** lines of code

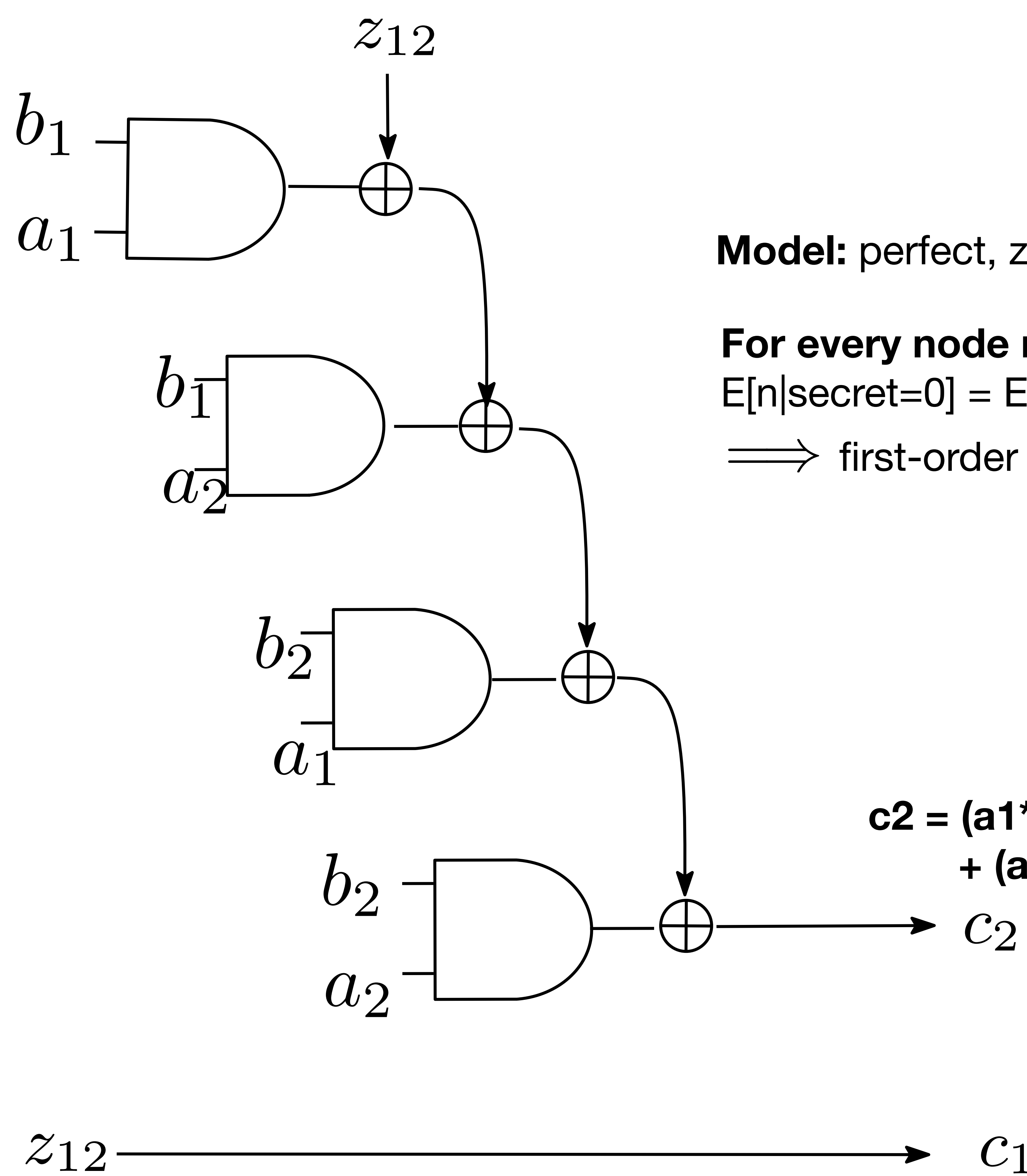
Evaluating masking in HW circuits



⚠️ **not yet peer reviewed**

Work in progress: verification of HW circuits

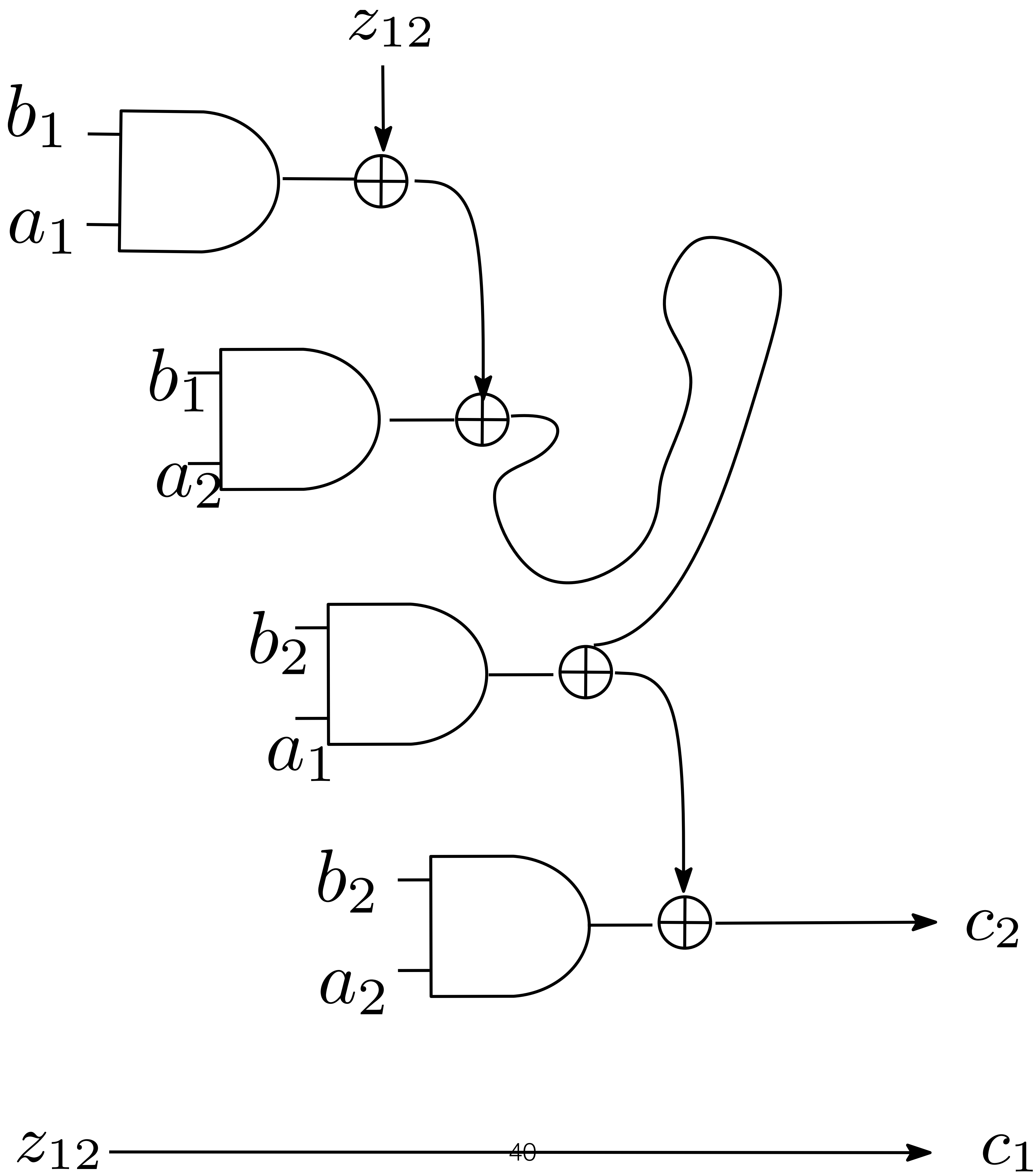
- Glitches: unintentional, spurious signal transitions. Signals go thru different state changes till they stabilise.
- A headache for many people:
  - glitches consume unnecessary power, energy
  - security implications: can make masking insecure [Mangard et al. 2005]
- Mitigation:
  - manually
  - or by using techniques: TI, CMS, DOM, …
- Next: verifying HW circuits, taking into consideration glitches.

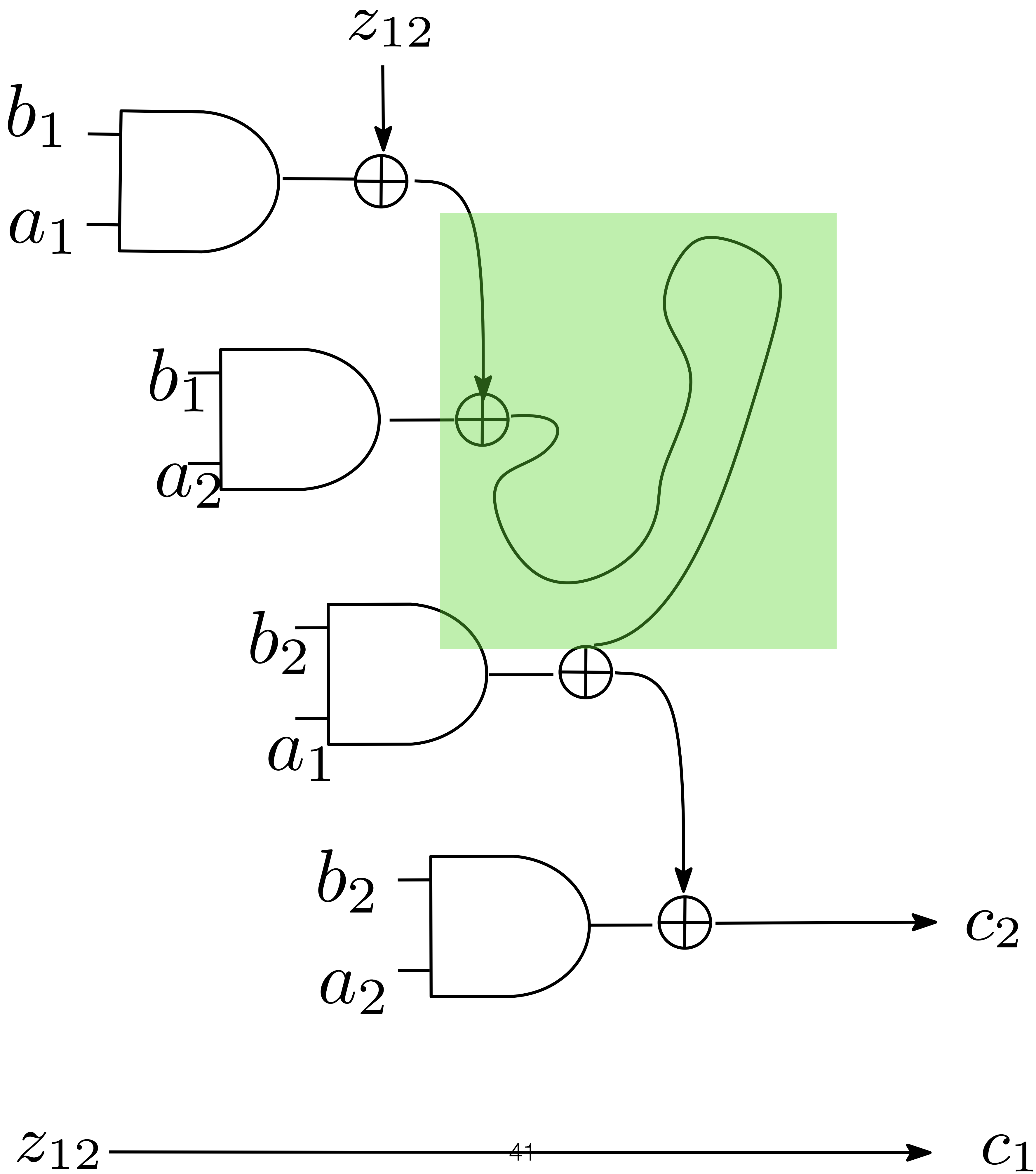


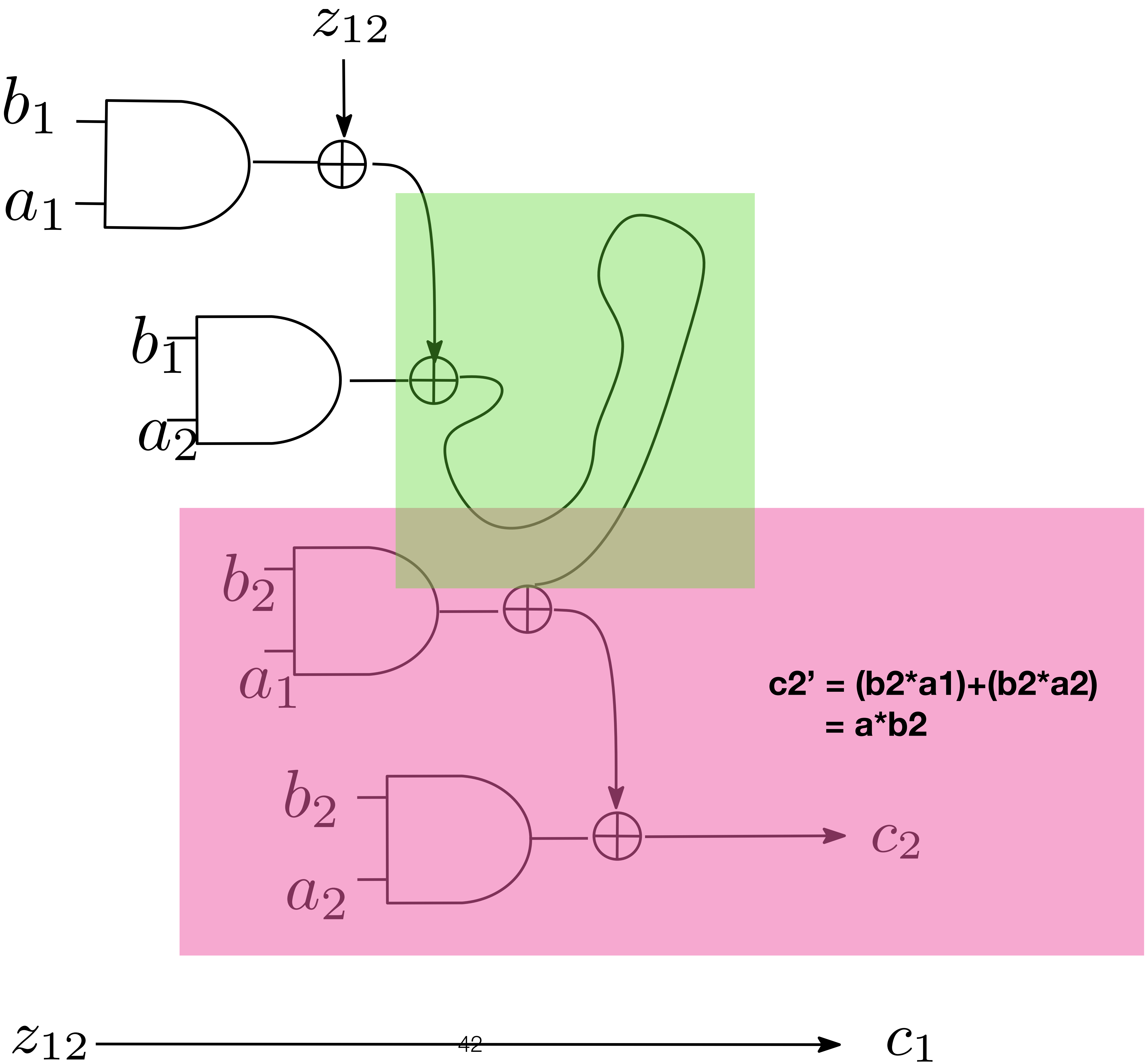
Model: perfect, zero-delay gates

For every node n:
E[n|secret=0] = E[n|secret=1]
$\Longrightarrow$ first-order security

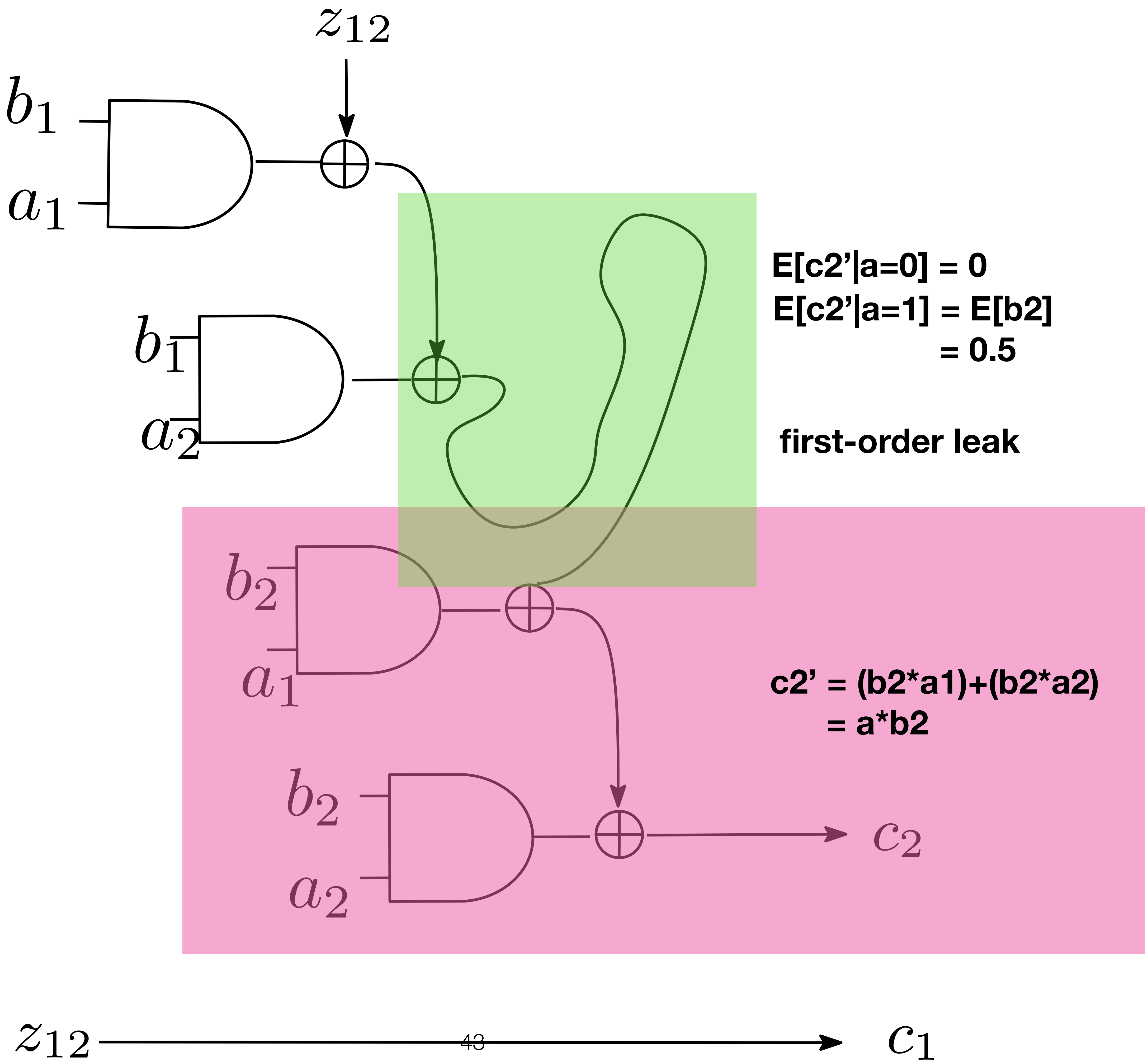
c2 = (a1*b1) + (a1*b2) +
+ (a2*b1)+(a2*b2) + z


z_{12}
b_1
a_1
b_1
a_2
b_2
a_1
b_2
a_2
c_2
z_{12}
c_1


z_{12}
b_1
a_1
b_1
a_2
b_2
a_1
b_2
a_2
c_2
z_{12}
c_1



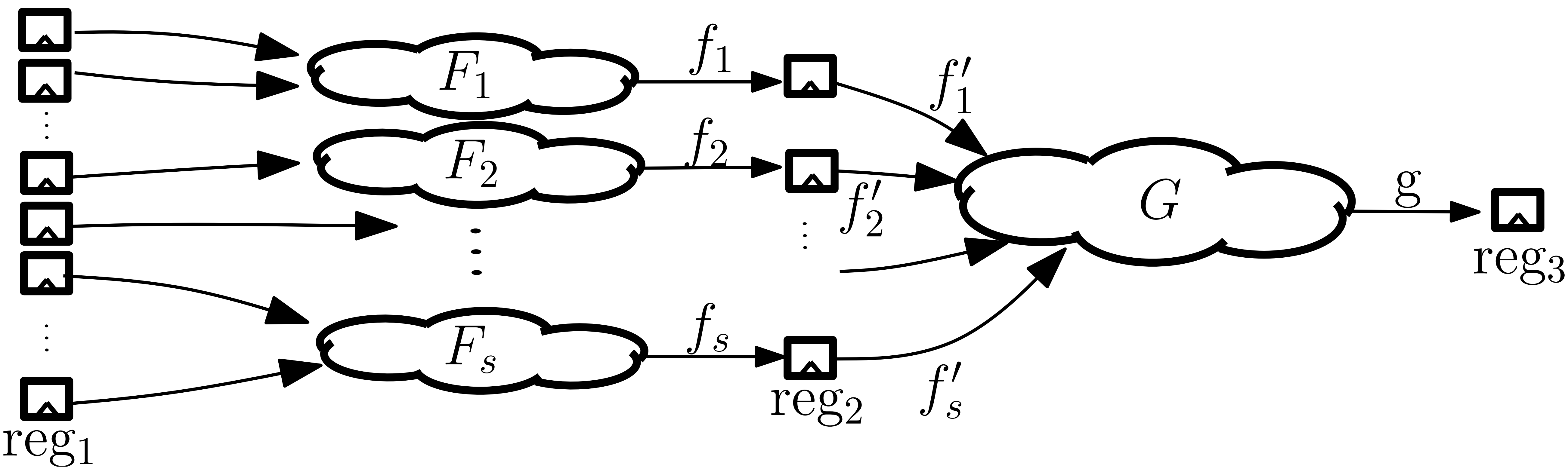
z_{12}
b_1
a_1
b_1
a_2
b_2
a_1
c2' = (b2*a1)+(b2*a2)
= a*b2
b_2
a_2
c_2
z_{12}
c_1


z_{12}
b_1
a_1
b_1
a_2
E[c2'|a=0] = 0
E[c2'|a=1] = E[b2]
= 0.5
first-order leak
b_2
a_1
c2' = (b2*a1)+(b2*a2)
= a*b2
b_2
a_2
c_2
z_{12}
c_1

The “glitch function”

- The “glitch function” is a fictitious function. It is actually a family of functions.
  - Definition: the circuit computes “glitch functions” before getting a stable output
  - The “glitch function” is often very difficult to completely determine (need to have very careful characterisation of logic gate library, routing details). We assume it is unknown.
  - But we know certain properties!

Leakage behaviour induced



Can work at the RTL level:

- **no timing information, no library characterisation needed**
- **at the expense of more false positives (overkill evaluation)**

Key (obvious) observation

- glitch function depends only on input nodes!
- If input nodes are (jointly) secret-independent, then no glitch function can make the node leak
  - in other words, I(input nodes; secret) = 0

Testing for glitch-security

- “One probe”: for each circuit node n
  - verify that I(inputs to n; secret) = 0
    - boils down to verifying distribution of inputs conditioned on secret are the same
- “Two probes”: For each pair or circuit nodes (n1,n2)
  - Verify that I(inputs to n1 || inputs to n2 ; secret) = 0


Thank you for your attention
Questions?