Fault Injection Resilience

Sylvain Guilley, Laurent Sauvage, Jean-Luc Danger, Nidhal Selmane

Fault Injection Resilience

Sylvain GUILLEY[1,2], Laurent SAUVAGE[1,2], Jean-Luc DANGER[1,2], Nidhal SELMANE[1]

[1]*Institut TELECOM, TELECOM ParisTech, CNRS LTCI (UMR 5141), 46 rue Barrault, 75 634 Paris, France.*
[2]*Secure-IC S.A.S., 37/39 rue Dareau, 75 014 Paris, France.*
{sylvain.guilley, laurent.sauvage, jean-luc.danger, nidhal.selmane}@telecom-paristech.fr

***Abstract*—Fault injections constitute a major threat to the security of embedded systems. The errors in the cryptographic algorithms have been shown to be extremely dangerous, since powerful attacks can exploit few of them to recover the full secrets. Most of the resistance techniques to fault attacks have relied so far on the detection of faults. We present in this paper another strategy, based on the resilience against fault attacks. The core idea is to allow an erroneous result to be outputted, but with the assurance that this faulty information conveys no information about the chip's secrets. We first underline the benefits of FIR: false positive are never raised, secrets are not erased uselessly in case of faults injections, which increases the card lifespan if the fault is natural and not malevolent, high potential of resistance even in the context of multiple faults. Then we illustrate two families of fault injection resilience (FIR) schemes suitable for symmetric encryption. The first family is a protocol-level scheme that can be formally proved resilient. The second family mobilizes a special logic level. We notably detail how a countermeasure of this later family, namely dual-rail with precharge logic style, can both protect both against active and passive attacks, thereby bringing a combined global protection of the device. The cost of this logic is evaluated as lower than detection schemes. Finally, we also give some ideas about the modalities of adjunction of FIR to some certification schemes.**

***Index Terms*—Fault Injection Attack (FIA), symmetric block encryption, Denial of Service (DoS), Fault Injection Resilience (FIR), Differential Fault Analysis (DFA), Side-Channel Attack (SCA).**

I. Introduction

Secure embedded systems such as smartcards must be tamper-resistant so as to defeat attacks that target directly their implementation. Three kinds of threats have been identified on these devices: perturbation, observation and manipulation. Perturbation attacks consist in covertly changing one data so as to either modify the chip's execution flow or force it to output incorrect results. Observation attacks specifically target the parts of the design that manipulate secrets; their goal is to exploit side-channel leakages so as to recover sensitive information. Manipulation is an invasive attack that gives to the attacker the power of modifying the chip's functionality or of directly probing signals [12].

Manipulation attacks are the most difficult to resist against, because of their intrusiveness: the device, expected to conceal data, is suddenly reduced into a white-box system. Fortunately, manipulation attacks involves expensive laboratory equipments and trained personnel. They are therefore not the more common ones. In addition, efficient countermeasures exist, such as active shield on top of the chip.

Observation attacks are less costly attacks, since some side-channels, such as the magnetic field, can be recorded at will without the chip even noticing it, in a non-invasive or semi-invasive manner. There also exists a wealth of counter-measures of different quality to make side-channel attacks (SCA) difficult.

Perturbation attacks require a means to alter the device's behavior, without triggering the countermeasures that continuously monitor the environment. Some low cost global countermeasures (such as overclocking, power underfeeding, clock and power transient, glitches or heating) can be used in weakly protected devices. Most expensive attacks rely on a local perturbation: for instance, laser or particle shots can avoid active shields and thus to surgically modify data in extremely well localized zones / dates. At the opposite, those tools can also be used to cause random and extremely spread faults in space / time. With little chance, those highly multiple faults are undetected and thus successfully alter the chip's state.

Observation attacks on cryptographic blocks usually require a couple of hundreds or thousands observations in absence of countermeasures. At the opposite, fault injection attacks can reveal the secret with a small number of measurements. For instance, RSA [32] computed with the Chinese Remainder Theorem (CRT) can be broken with as few as one faulty computation [7]. The last 128 bit of the key schedule of an AES block cipher can be retrieved with one single well-behaved faulty encryption [39]. These exploits motivate a special focus on fault attacks. This is all the more true as

theoretically sound countermeasures have been proposed for SCAs but that the coverage of fault attacks is lacunar: multiple faults, either spread in space or in time, are extremely difficult to withstand with the state-of-the-art countermeasures. We therefore focus on those attacks in the rest of this article.

Fault injections attacks (FIA) can basically attempt to deviate the device from its nominal functionality in two ways. Either the fault can directly profit to the attacker, such as allowing her to access unauthorized pieces of information, or the fault induces a corrupted computation that the attacker post-processes to recover secrets. The first case is an attack against security mechanisms, whereas the second one targets typically the cryptographic modules. We will not cover the first case, since known methods already exist to cross-check that a punctual valid is indeed correct. The second case is at the heart of this paper. Indeed, checking for the correctness of all the steps of a lengthy cryptographic computation is more costly. And above all, we notice that a cryptographic system can indeed remain secure even if it outputs incorrect results. We promote in this paper the idea that, in most cryptographic protocols, it suffices to make sure the fault does not depend on any secret to maintain a provable security level. We call this protection strategy "fault injection resilience", notion abridged as "FIR".

The rest of the paper is organized as follows. The benefit of the FIR over other techniques based on detection is discussed in Sec. II. In Sec. III, some suitable techniques to implement FIR are described. A case study of a register transfer level (RTL) implementation of FIR is detailed in Sec. IV. The impact of FIR in two security certification schemes is studied in Sec. V. Finally, conclusions and perspectives are given in Sec. VI.

II. Benefits of FIR

A. State-of-the-art of Detection Mechanisms

As already underlined, the detection of faults is traditionally the method of choice to prevent fault attacks.

In the early years of fault tolerance in secure embedded systems, analogue solutions were used. They consist in disseminating voltage, temperature, light sensors on the surface of the chip, and any miscellaneous combination thereof. The problem of this approach is that it requires a mixed design, which is much more complicated from a CAD perspective than a purely digital design. Also, the analogue parts are consuming a lot of power and area in the design. Those practical and economical reasons explain why the analogue solution is obsolescent.

Therefore modern designs resort to all-digital detection mechanisms. The generic ones exploit some artificial redundancy. It can be either implemented in time, space or information (code-based). All those strategies have been compared in [21], and shown to be roughly alike. Depending on the cryptographic scheme to protect, some dedicated countermeasures can also be implemented. The idea is to exploit some identities of the algorithm to protect so as to detect possible errors with a high probability. In a typical encryption: the encrypted message can be decrypted and tested against the original plaintext. The same applies to digital signatures: the signature can be verified before being outputted. We wish to underline that these very verifications can represent a weakness *per se*, notably in front of so-called *safe errors* attacks [40].

However, the resilience against faults attacks has seldom been proposed. At the opposite, resilience in observation attacks is a hot topic. Following the proposal of Paul C. Kocher at CHES 2006 to update the keys on a frequent and regular basis, ideas for side-channel resilient schemes have come up, as illustrated for instance by the "Provable Security against Physical Attacks" workshop [2]. But, to our best knowledge, no investigation about resilience against fault injection attacks has been published so far. Actually, many techniques of reliability have been ported *as such* to security applications. Nonetheless the objectives of reliability and security do differ:

- Reliability requires ideally that either the computations are correct or that an alarm is raised;
- Security requires that the computation result, if erroneous, carries no information about secret involved in the computation. This is a more flexible requirement than for reliability. On the one hand, it allows the system to output a false result C^* instead of the correct one C, as long as it reveals no information about the secret K. A formalization of security models under fault attacks can be done, and has actually already been initiated for instance by this paper [20]. From an information-theoretic perspective, the requirement can be stated as "*the mutual information between* (C, C^*) *and* K *is null*". On the other hand, rising an alarm can even be a vulnerability in some contexts [33].

B. Comparison between Detection and Resilience

Neither detection not resilient schemes are able to withstand all the faults. Indeed, whatever the protection

Table I
CLASSICAL FAULT DETECTION CHARACTERISTICS.

		Ciphertext incorrect?	
		Yes	**No**
Alarm raised?	**Yes**	Safe	Problem of **availability**
	No	Problem of **security**	Safe

mechanism, we can theoretically build an attacker (possibly adaptative) able to replace an authentic value with another one. The goal of the countermeasure is to make this substitution very chancy.

In this subsection, we investigate the side-effects of the countermeasures. The detection strategy suffers two drawbacks illustrated in Tab. I. First of all, the device can raise an alarm even if the result is correct. This is the case when the fault happens on a variable that does not impact the output. This situation is of course not true in general, otherwise the variable could have been removed from the implementation. However, in the course of a specific computation, this is possible. One trivial example is the result of an `AND` gate, that has zero for one input, and that is faulted on its second input. The fault will not be propagated and the result will be correct irrespective of the fault taking place or not. However, if a detection mechanism raises an alarm, then the whole computation will be stopped and adequate actions will be undertaken, thus causing a denial of service (DoS) despite the absence of security problem. Second, detection mechanisms do not cover all the possible faults, and some faults can propagate without being detected.

On the contrary, an ideal resilient scheme will feature:

- **an optimal availability**: false detections do not exist, since errors are not caught but propagated.
- **an optimal security**: the fault generates a wave of erroneous data independent of the previous pristine (and sensitive) values. Therefore no sensitive information is propagated.

Also, in terms of coding and deployment guidelines, the advantages of resilience as opposed to resistance (fault detection) are manifold. We can really claim that resilience is a new security approach to protect cryptography, because of these typical improvements:

- In traditional designs, miscellaneous checks are scattered in the code. For instance, ratification counters and baits are tricks to detect "blind attacks". No such extra operations are required in the context of fault resilience, since it is not catastrophic that the IC fails.
- When using detection, faults can also occur in the detection logic. But then, the problem becomes eventually insolvable, since more and more logic is necessary (by recursion, we need detection logic for the detection logic, itself being protected by detection, etc.)
- On top of that, the resilience relieves the designer from having to deal with the reactions to the threat. These features are all in one very annoying for the chip manufacturer ; if they are activated unexpectedly they possibly ruin the device, causing large costs to replace the defective card. Now, the secure chip manufacturers are often balancing between activating the maximum level of countermeasures and risking card auto-scuttling (false positive). Such a dilemma does not exist with fault resilience. The card starts to produce faulty results while under stress (either because of an attack or because of a natural hazard), but returns to its nominal operating conditions as soon as the stress disappears. Thus the risk of have a permanent damage due to a false alarm is thus nonexistent.

C. Merits of the FIR

One feature that gives to FIR a remarkable strength is its agnosticism with respect to attacks. By making any faults independent at its source and during its propagation independent of the previous values, it merely prevents any attack at their root. Therefore, new scenario schemes not envisioned yet are thwart proactively, which provides a forward security. Typically, most – if not all – attacks studied so far are differential: they assume the attacker knows couples of correct & faulted computations. Now, higher-order attacks could as well be possible: they would imply more than one faulty result. Additionally, faulted ciphertext-only attacks could also be devised. FIR fights all those future new threats that a pure DFA counter-measure would maybe fail to cover.

D. Related Works

Earlier publications have noticed the interest of allowing cryptographic devices to output faulty results, without jeopardizing their security. However, all those results focused on asymmetric cryptography, and more specifically on RSA. A fault tolerant RSA with CRT algorithm is given and formally proved in [41]. This article introduces the concept of *infective computation*:

the algorithm is designed to have the errors propagate everywhere, thus denying the Bellcore attack. The infective computation was crafted for RSA with CRT, whereas in our paper, the FIR is algorithm-agnostic. In [11], any implementation of RSA is proved immune to FIAs provided PSS is used on the input.

III. Some Practical Implementations of FIR

The purpose of this section is to provide with some resilient cryptographic schemes. For the sake of clarity, we focus on the protection of symmetric block encryption modules. Indeed, as they are deprived by construction from any algebraic properties, they are also the most difficult ones to protect. The state-of-the-art in asymmetrical algorithms protection is very well advanced and formally proved. An overview, on the example of RSA, can for instance be found in these papers [8], [9].

In the subsection III-A, we provide with a protocol-level resilient scheme. The subsection III-B rather introduces two gate-level solutions. In those two embodiments of FIR, we assume the cryptographic parameters are loaded securely, and thus that key alteration attacks (see for instance [13] or §III.C of [4]) are out of the scope. To sum up, our security goal is definitely the protection of symmetric cryptographic operations.

A. Formal Counter-Measures against Fault Injection Attacks

A differential fault analysis (DFA [6]) requires the same plaintext to be encrypted twice with the same key. Common attack scenarios consider the case where the attacker is able to inject one fault in only one of the encryptions. Then, she can deduce information about the key using a DFA. Thus, DFAs are made impossible if an attacker is not able to request twice the same encryption. It is possible to devise such a scheme, as typified by algorithm (1).

Algorithm 1: Probabilistic Encryption Algorithm built on top of AES.

Input : A plaintext x to be encrypted with the key k.

Output: A ciphertext along with a random number.

1 Determine a random number r of the same size as x; `/* This number will whiten x */`.
2 Return the couple $(y = \text{AES}_k(x \oplus r), r)$.

This algorithm (1) is considered as secure against DFA because the probability that two encryptions are generated with the same plaintext is roughly speaking $2^{n/2}$, where n is the entropy of x or r. Indeed, this is a classical instance of the birthday paradox.

We mention additionally that the scheme of algorithm (1) protects against a broader class of attacks than only the DFAs. It is a random encryption scheme, that has the remarkable property that the attacker cannot decide if the encryption is actually faulty or not.

Unfortunately, this scheme is not secure in decryption. As a matter of fact, the decryption algorithm corresponding to (1) is given in algorithm (2). This algorithm can be called repeatedly without the AES inputs being modified.

Algorithm 2: Deterministic Decryption Algorithm matching algorithm (1).

Input : A ciphertext under the form $(y = \text{AES}_k(x \oplus r), r)$ to be decrypted by the AES key k.

Output: The plaintext x.

1 Decrypt y with key k: $z = \text{AES}_k^{-1}(y)$.
2 Return the demasked input: $z \oplus r = x$.

This situation can however be exploited to protect low cost embedded systems, such as smartcards or RFID tags, that communicate with a larger device, such as a reader. It is fairly easy to protect the reader against fault attacks by "physical tamper-proof measures". For instance, the reader electronic circuits can be imprisoned into a mold, protected with a pasted metallic cover and sealed into a box equipped with intrusion detection sensors. The same level of sophistication is impossible for smartcard or tags modules, because their form factor is extremely constrained in size (due to stringent requirements about the mechanical strength edicted by ISO 7816-1). Therefore, the attacker will most certainly prefer to attack the embedded system to extract the shared secret key. Thus, if the reader plays the decryption (2) and the embedded system the encryption (1), the unbalance between the tamper-resistance of the two devices is made up by the opposite unbalance of the algorithm, in terms of resistance against DFA. This strategy of reinforcing the security by algorithmic means of the weakest element in the security chain is illustrated in Fig. 1.

Notice that if a handy homomorphous encryption algorithm HEA is available, a completely secure encryption/decryption scheme can be devised. Let us denote by $\text{HDA} = \text{HEA}^{-1}$ the corresponding decryption algorithm

Figure 1. Probabilistic encryption is performed on the most vulnerable device while the deterministic decryption is safely carried out within the most secure device.

and $\times$ the composition law in the group of homomorphy:

$$\forall y_1, y_2, \quad \mathrm{HDA}(y_1 \times y_2) = \mathrm{HDA}(y_1) \times \mathrm{HDA}(y_2)\,.$$

The encryption proceeds as per algorithm (1) using HEA instead of AES, whereas the decryption consists in algorithm (3). This scheme can use for instance Paillier's cryptosystem [26] as underlying encryption primitive. However care must be taken with RSA [7].

Algorithm 3: Probabilistic Decryption Algorithm matching (1) with HEA instead of AES as underlying cipher.

Input : A ciphertext under the form $(y = \mathrm{HEA}_k(x \oplus r), r)$ to be decrypted by the HEA key k.
Output: The plaintext x.

1 Determine a random number s of the same size as y or r.
2 Return $\mathrm{HDA}_k(y \times s)/\mathrm{HDA}_k(s) \oplus r = x$.

The resilient algorithms presented in this subsection III-A have the drawback that the size of the ciphertext is doubled. This can be a limitation for instance in contactless cards authentication, where the transmission time must remain short. Also in wireless sensor network

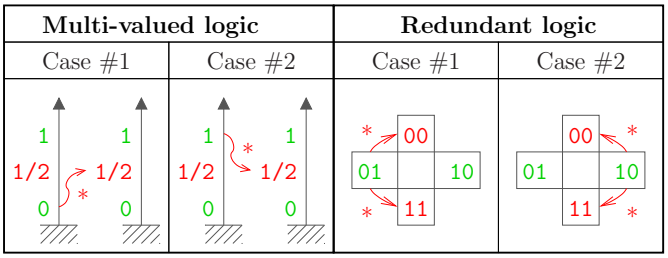

Figure 2. Two kinds of faults (in red), namely $\{0,1\} \xrightarrow{*} 1/2$ for 3-valued logic and $\{01,10\} \xrightarrow{*} \{00,11\}$ for DPL, after which the initial value (in green) has been forgotten.

the increase of the data transmitted means a very high cost in term of power.

Nonetheless the algorithm (1) can be made more bandwidth and power-efficient if the message x to encrypt is cut in several blocks. In this case, alternative encodings, such as the all-or-nothing probabilistic transform described in [23], can help reduce the number of blocks to be exchanged to the number of plaintext blocks plus one.

B. Multi-Valued and Redundant Representation Logics

Multi-valued logics allow to encode more than one bit with one electrical state. It is for instance used in some power-constant logic styles [3]. Let us consider the case of an equipotential holding three states, denoted 0, 1/2 and 1, amongst which only the two 0 and 1 are functional. Then, if a fault turns a valid value into 1/2, the provenance has been forgotten.

The same goes for redundant logics, such as the m-out-of-n representations (for $0 < m < n$). For instance, the 1-out-of-2 representation, also known as dual-rail with precharge logic (DPL), admits two valid states, denoted by 01 and 10, and two invalid states, denoted by 00 and 11. In the case one fault turns a valid token into an invalid one, the value before the fault is lost. The effect of faults on these two logic styles is summed up in Fig. 2. It clearly appears that the state after the fault is decorrelated from the initial state, thereby establishing the resilience, for the relevant cases where the data is sensitive.

Now, the resilience only works in the case the attacker fails to inject "valid false" faults, *i.e* $0 \xleftrightarrow{*} 1$ faults in multi-valued logic or $01 \xleftrightarrow{*} 10$ faults in DPL. Let us assume this situation is rare. It seems all the more difficult to achieve in DPL because the attacker must produce two antinomic concerted faults.

As will be exposed into greatest details in Sec. IV,

the resilience will build up each time a valid false is produced along with invalid faults. In this case, the two faults will propagate, and if the logic favors the generation of invalid instead of valid states, then the diffusion of the netlist will encourage the invalid states to hide the false valid states. This case is optimal if the logic meets this requirement: if any input is invalid, so is the output. This behavior is "saturating"; the faults will percolate in the netlist and the invalid values will saturate most of the nets, thereby absorbing all the false valids that are crossed. So the resilience is amplified by the diffusion in the netlist and the collaborative behavior of gates to favor invalid values propagation. This phenomenon of invalid values (dominant) suppressing false valid values (recessive) is further detailed in the next section IV.

IV. Dual-Rail with Precharge Logic as a Global Countermeasure against Implementation-Level Attacks

DPL styles are solutions primarily designed to protect a cryptographic implementation against side-channel attacks. However, it has been noticed that these styles can also natively withstand some perturbation attacks [34], [5]. It has already been underlined in Sec. II that, unlike traditional counter-measures against fault attacks, the DPL does not implement a protection, but is rather resilient. This means that faults are not caught, but rather left free to cascade their effect, knowing that eventually their observable consequences will not be harmful.

A. Requirements for Simultaneous SCA and FIA Protection

In order to better illustrate the close relationship between observation and perturbation attacks, we need to notice that security perimeters depend on the application. For instance, in an ISO/IEC 7816 compliant smartcard, several security violation situations can be encountered.

- The critical part is the memory in case of an external authentication. Indeed, if the memory can be corrupted, then any rogue reader can be forced to be seen as authentic. Here, there is no secret to retrieve, but simply an invalid state to be setup by force.
- However, during an internal authentication, the smartcard uses its cryptographic secret. Therefore, the risk for the smartcard is to have its key retrieved illegitimately. Differential fault attacks and side-channel attacks are two tools available to recover the key. In addition, as the protection against attacks is costly, the designer will try to partition the cryptographic block at risk. Typically, when it implements symmetrical encryption, this block can be split into:
  - a control part, subject to fault attacks, such as round reduction attacks [24], but leaking no sensitive information as the algorithm is supposed to be known by the attacker (common assumption with Kerckhoffs' law), and
  - a data processing part, subject to both fault attacks, such as DFAs [6], [27], and side-channel attacks, such as DPA [17].

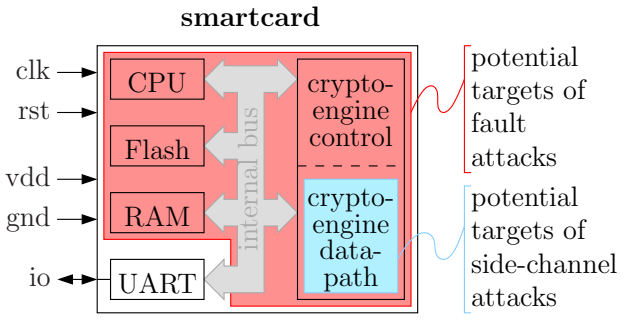

Figure 3. Susceptible organs of a smartcard in two representative sensitive operations (`EXTERNAL AUTHENTICATE` and `INTERNAL AUTHENTICATE`). Typically, the cryptography will be triple-DES or AES.

The overall requirement for security against implementation-level attacks in a smartcard is depicted in Fig. 3. This block-diagram shows in red the security boundary for fault attacks and in cyan that for SCAs. It appears clearly that some organs shall be protected only against fault attacks, but that all the organs that shall be protected against SCA must also be protected against FIA. This is an advanced question, all the more important as it is in this part of the design that the largest overheads are expected.

The countermeasures against SCA include:

- information masking, implemented with random splitting of data into shares,
- information hiding, implemented with DPL.

Amongst this array of possible protections, DPLs are of particular interest because they have native protections against DFAs. We will thus focus in this article on the combined DFA and SCA protection of the datapath of cryptographic modules; The type of fault attacks we consider are those described in [14], the two most famous of them being that of Biham & Shamir [6] (DES) or Piret & Quisquater [27] (AES), enhanced by Tunstall in [39]. Another motivation to focus on the crypto-

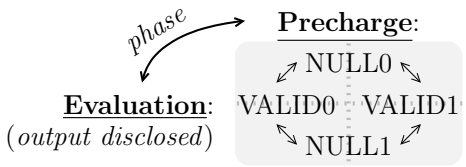

Figure 4. DPL protocol. Meaning values are computed during the evaluation stage; the precharge is meant to setup the nets to a known initial state.

datapath is that it is usually the most complex design part; therefore it represents the largest area and contains the longest critical timing paths. This explains that local faults are more likely to target the datapath because of its predominant surface, and that global faults also affect preferentially the datapath that is most tight in meeting the setup time constraint.

B. Previous Art about DPL in the Presence of Faults

We use the following notations for the DPL representation. Every logical variable a is represented by a couple (a_f, a_t) of wires, that carry two values. The semantic of the four possible combinations is detailed below.

- a is VALID if $\boxed{a_f \oplus a_t = 1}$. More precisely, VALID $\doteq \{\text{VALID0}, \text{VALID1}\}$ or, more explicitly, VALID $\doteq \{(1,0),(0,1)\}$.
- a is NULL if $\boxed{a_f \oplus a_t = 0}$. More precisely, NULL $\doteq \{\text{NULL0}, \text{NULL1}\}$ or, more explicitly, NULL $\doteq \{(0,0),(1,1)\}$.

The two NULL states are used alternatively with the VALID ones as precharge stage, so that the next evaluation starts afresh from a known state. The DPL protocol is recalled in Fig. 4.

There are two flavors of DPL, depending on if they feature the early propagation effect (named EPE in the literature, discovered independently by [36], [18]) or are protected against it. The definition of those variants can be summarized by the following conditions to be fulfilled by all the instances f:

- **DPL w/ EPE**: $\exists a$ VALID, $f(a, \text{NULL}) = \text{VALID}$;
- **DPL w/o EPE**: $\forall a$ VALID, $f(a, \text{NULL}) = \text{NULL}$.

In DPL, only results on *evaluation* are observable, because *return to precharge* faults are not outputted. We adopt the following faults typology on DPL:

- **Asymmetric faults**: {VALID0, VALID1} $\xrightarrow{\downarrow}$ NULL0, triggered by **global** perturbations (*e.g.* caused by a setup time violation due to power/clock glitch, overclocking or under-powering);
- **Symmetric faults**: {VALID0, VALID1} $\xrightarrow{\downarrow \text{ or } \uparrow}$ {NULL0, NULL1}, triggered by **local** perturbations (*e.g.* caused by injection of high energy laser light, electromagnetic field or particles beam).

1) DPL w/ EPE *is Protected against Multiple* Asymmetrical *Faults:* WDDL [38] is a typical DPL w/ EPE style. In this logic, the AND function is defined as: $(y_f, y_t) \doteq (a_f + b_f, a_t \cdot b_t)$. We use the following color code in Boolean truth tables:

- gray: the regular truth table in the absence of faults (*i.e.* the intended functionality),
- purple: anticipated values (evaluation even if not all inputs are valid).

Otherwise, green and red still represent respectively correct and incorrect behaviors or properties.

As shown below, WDDL can propagate correct valid results in the presence of asymmetrical faults.

b \ a	VALID0	VALID1	NULL0
VALID0	VALID0	VALID0	VALID0 (**EPE**)
VALID1	VALID0	VALID1	NULL0
NULL0	VALID0 (**EPE**)	NULL0	NULL0

This behavior is positively resilient. It is that of the Unitialized value in VHDL enumerated type `ieee.std_logic_1164.std_ulogic`, recalled below:

b \ a	'0'	'1'	'U'
'0'	'0'	'0'	'0'
'1'	'0'	'1'	'U'
'U'	'0'	'U'	'U'

where the tokens {VALID0, VALID1, NULL0} implement the items {'0', '1', 'U'}.

Actually, this FIA-resistance solution has already been sketched in [15]. However, in the context of "*reset faults*" (our "*asymmetric faults*"), this publication is overly conservative; invalid tokens are generated even if the data is not tainted. Therefore, the scheme we present is more user-friendly, in the sense it keeps the application up-and-running unless a fault is indeed influencing the result.

2) DPL w/ EPE *is not Protected against Multiple* Symmetric *Faults:* To start with, we assume neither $a \xrightarrow{*} \overline{a}$ nor $b \xrightarrow{*} \overline{b}$ happens. However, even in this favorable case, WDDL can generate incorrect false results. They are presented by skulls (symbol: ☠) in the following table.

b \ a	VALID0	VALID1	NULL0	NULL1
VALID0	VALID0	VALID0	VALID0 (**EPE**)	VALID0 (**EPE**)
VALID1	VALID0	VALID1	NULL0	NULL1
NULL0	VALID0 (**EPE**)	NULL0	NULL0	VALID0 (☠)
NULL1	VALID0 (**EPE**)	NULL1	VALID0 (☠)	NULL1

For instance, the twain simultaneous errors:

1) $a = \text{VALID1} \xrightarrow{*\uparrow} a = \text{NULL1}$ and
2) $b = \text{VALID1} \xrightarrow{*\downarrow} b = \text{NULL0}$

trigger a dreadful transformation: $\text{VALID1} \xrightarrow{*} \text{VALID0}$.

Therefore, because of EPE, logical inversions $f(a,b) \xrightarrow{*} \overline{f(a,b)}$ can occur, which makes FIAs possible.

3) DPL w/o EPE *is Protected in front of Multiple* Symmetric *Faults:* Now, the DPL w/o EPE styles are protected against multiple symmetric (hence asymmetric) faults. This is shown in the table below.

b \ a	VALID0	VALID1	NULL0	NULL1
VALID0	VALID0	VALID0	NULL0	NULL1
VALID1	VALID0	VALID1	NULL0	NULL1
NULL0	NULL0	NULL0	NULL0	NULL1
NULL1	NULL1	NULL1	NULL0	NULL1

Remark that if we call:

- `'0'`: VALID0,
- `'1'`: VALID1,
- `'X'`: NULL $= \{\text{NULL0}, \text{NULL1}\}$,

then we have the same behavior (*i.e.* "*propagate always*") as VHDL. This is illustrated below:

b \ a	`'0'`	`'1'`	`'X'`
`'0'`	`'0'`	`'0'`	`'X'`
`'1'`	`'0'`	`'1'`	`'X'`
`'X'`	`'X'`	`'X'`	`'X'`

Finally, we note that even if a few mutations $a \xrightarrow{*} \overline{a}$ exist for some variables a, it is very likely that the `'X'` wave caused by $a \xrightarrow{*}$NULL *eats* them. As detailed in the next sub-section, the recessivity of `'X'` over NULL, coupled with the avalanche of `'X'` caused by the diffusion property of the logic, accounts for that.

C. Revisiting the Comparison Resilience vs. Detection

One can argue that the DPL used as a FIR is in fact a very low-grain fault detection scheme. Indeed, FIR shares with the detection strategy the fact that redundancy is required. However, it is coupled to a diffusion that makes the detection at one stage take advantage of the rest of the stages. This detection is propagated in a wave, that constitutes a collaborative strategy that is absent from the pure detection schemes. There are two properties of DPL that help resilience:

- The **redundancy** of the netlist. At an n-bit output of a combinational block, only 2^n amongst the 2^{2n} possible ones are valid.
- The **diffusion** within the netlist, which is characteristic to the cryptography. This property is especially true for logics free from early evaluation [5]. Indeed, the fanout of each gate is double w.r.t. separable logics such as WDDL [38].

Current detections schemes work independently of the computation and in a non-collaborative way. At the opposite, FIR consists in intricating the detection agents with the computation and to tightly interconnect them.

D. Cost Estimation of FIR versus Traditional Approaches

The traditional approach to counteract implementation-level attacks is a composition:

- first use detection schemes, that can be inserted early at the RTL of the algorithm [19];
- then map this FIA-aware RTL description into a SCA-proof logic style. Indeed, the detection logic manipulates sensitive variables, and might itself leak secrets [31]. Therefore, it deserves a protection against SCAs.

This implies that the overhead of the FIA and SCA countermeasures get multiplied.

A typical overhead for FIA countermeasures can be found in [21]. Let us consider the case of a non-linear code, such as [16], that is suited to detect multiple faults. Its overhead is 77 % in area and 15 % in throughput.

As such, those performance losses are more affordable than those required to thwart SCAs. For instance, WDDL incures an increase of 3.1 in area and 3.9 in throughput [37].

The combination of [16] and [37] results in an increase of 5.5 in area and 4.5 in throughput.

Those results are to be contrasted with the FIR approach using an EPE-proof DPL style. This style already merges FIA and SCA countermeasures. The reported overheads for two of those logics are given in Tab. II. Those similar logics yield similar performances: iMDPL [28], STTL [35], WDDL w/o EPE [5] and BCDL [25]. It clearly appears that using a symbiotic SCA+FIA countermeasure is more efficient than combining two countermeasures one on top of each other.

Table II
PERFORMANCE OVERHEAD OF DIFFERENT SCA+FIA COUNTERMEASURES.

Strategy	**Detection + DPL**	**Resilience = DPL**	
Countermeasure	[16] + [37]	DRSL [10]	IWDDL [22]
Area	5.49 ×	2.56 ×	4.34 ×
Throughput	4.49 ×	2.00 ×	1.53 ×

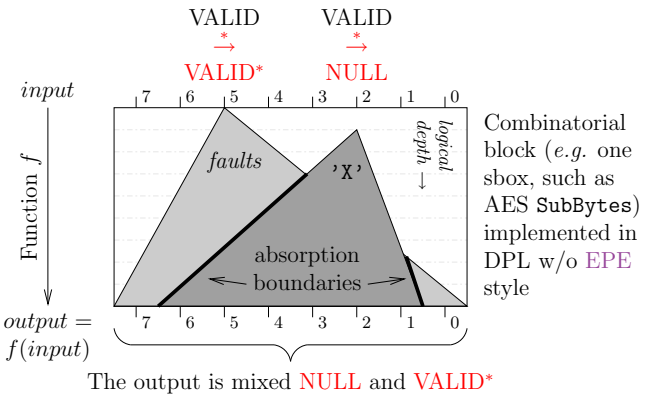

Figure 5. Multiple faults, where the false valid is not completely hidden by the 'X' wave.

E. Associating Three Protections to Reduce the Probability of a Successful FIA

Some faults in DPL circuits do not disclose any information about the faulted sensitive variable. However, in the case false valid are generated, the problem becomes different. This can happen in two problematic cases:

1) When the absorbing fault is too deep in the logic cone w.r.t. the false valid, as shown in Fig. 5, where f is a block with perfect diffusion, such as a substitution box implemented in logic. In this case, if the logic cone covered by the 'X' happen to yield a correct value, then a valid fault is generated; unless the 'X' are checked for at the output.
2) When a valid false occurs on one column alone, but that an 'X' is generated on another column (knowing the two columns are not interfering in AES last round). In this case also, the faulty behavior can be observed by checking the validity of all the output bits.

To fight these remaining risks, three protections can be associated so as to increase the security level:

1) DPL, as detailed in the previous section.
2) Detection of NULLs at the end of the computation.
3) Regular detection schemes, such as coding.

V. Applicability of Resilience with Certification Procedures

The two main certification schemes of security products are the FIPS 140 and the common criteria. We examine in this section if the resilience can be applied with the current version of those standard, or if the standards are too conservative.

A. NIST FIPS 140-3

The FIPS 140 [29], [30] formulates security requirements for cryptographic modules. It defines four levels of security, the highest of which is referred to as "security level 4". The functional security objectives of FIPS 140 are defined in §3. It includes those two requirements:

1) to detect errors in the operation of the cryptographic module and
2) to prevent the compromise or the modification of sensitive data and SSPs (Sensitive Security Parameters) resulting from these errors.

The "resilience" protection discussed in this article definitely fulfills the second requirement. However, not all resilient schemes comply with the first requirement. For instance, using the randomized homomorphic encryption (Algorithm 1), the errors cannot be detected. The partial resilience of dual-rail type countermeasure can allow a detection of the fault. However, the security of this scheme is ensured even if there is no detection. This means that FIPS-140 standards 2 & 3 are not resilience-ready, although they express this idea.

More precisely, the exact statement of the requirements is detailed in §4.5.5 (140-2 [29]) or §4.6.5 (140-3 [30]). For the security level 4, the cryptographic module shall either employ environmental failure protection (EFP) features or undergo environmental failure testing (EFT). The EFP consists in a constant monitoring of the environment (temperature and voltage) whereas EFP is an a priori characterization of the perturbation consequences. In both cases, the protection circuitry shall either (1) shutdown the module to prevent further operation or (2) immediately zeroize all plaintext secret and private cryptographic keys and SSPs.

Such authoritative and irremediable actions could have been prevented using a resilience scheme, without compromizing the device security. Therefore, we find that FIPS 140-{2,3} standards are too strict, resulting in potential inconveniences from the user perspective if non malicious faults causes the module shutdown or zeroization.

B. Common Criteria

The Common Criteria (CC) [1] is a framework that permits comparability between results of independent security evaluations. It is an international standard ISO/IEC 15408:2005. The CC in themselves do not specify security requirements. Instead, a "target of evaluation" (TOE) must meet "security targets" (ST). One or more "protection profiles" (PP) must be respected by the ST.

The security requirements are expressed in the PPs, whose structure is standardized but whose content is up to the designer. This flexibility allows a designer to tailor the PP to his (or that of his client) security objectives. Therefore, the CC readily accepts the resilience as a solution against fault attacks.

VI. Conclusions and Perspectives

In embedded devices, fault attacks are usually combated in software. The dominant strategy is their detection, which is costly and non-exhaustive. We present in this paper an approach based on resilience. The faults are not necessarily captured, but the information they contain about any secret is nullified. The benefits of this approach are the ergonomy and the cost. First of all, the resilience impose no destruction of the secrets in case of a fault attack; thus, in case of natural (non-malevolent faults) the user experience is a transient DoS, as opposed to a permanent DoS in traditional detection-based countermeasures. Symmetrically, when a fault is injected successfully but has no consequence in the computation, a card protected with a detection-based scheme may react, whereas this inconvenience is nonexistent in the resilience-based scheme. Several concrete methods to implement resilient symmetrical encryption are proposed, amongst which a random mode of operation that is suitable for low-cost (without expensive module-level protections) smartcards. When the designer can propose a hardware counter-measure, we suggest the use of multi-valued or DPL styles. Those logics simultaneously protect against observation and perturbation attacks, and are cheaper than detection based on codes.

As a perspective, we intend to quantify the optimal parameters of code-based detection schemes that can be added to a DPL logic (evoked in Sec. IV-E) to further reduce the number of faulty results outputted by the device. Also, we thrive to define a formal framework based on the information theory that could describe with commensurable metrics the resistance of a cryptographic implementations to both SCA and FIA.

Acknowledgments

The authors are very grateful to the five anonymous reviewers, that all contributed to improve the paper and to better place it in its scientific context. Novel ideas have also be suggested, that all open the door to efficient and formally proved countermeasures against active and passive attacks.

References

[1] Common Criteria. http://www.commoncriteriaportal.org/.

[2] Workshop on "Provable Security against Physical Attacks", February 10-19 2010. Amsterdam, Netherlands. http://www.lorentzcenter.nl/lc/web/2010/383/program.php3?wsid=383.

[3] Yuichi Baba, Atsushi Miyamoto, Naofumi Homma, and Takafumi Aoki. Multiple-Valued Constant-Power Adder for Cryptographic Processors. In *ISMVL*, pages 239–244. IEEE Computer Society, May 21-23 2009. Naha, Okinawaw, Japan.

[4] Hagai Bar-El, Hamid Choukri, David Naccache, Michael Tunstal, and Claire Whelan. The Sorcerer's Apprentice Guide to Fault Attacks. *Proceedings of the IEEE*, 94(2):370–382, 2006. DOI: 10.1109/JPROC.2005.862424.

[5] Shivam Bhasin, Jean-Luc Danger, Florent Flament, Tarik Graba, Sylvain Guilley, Yves Mathieu, Maxime Nassar, Laurent Sauvage, and Nidhal Selmane. Combined SCA and DFA Countermeasures Integrable in a FPGA Design Flow. In *ReConFig*, pages 213–218. IEEE Computer Society, December 9–11 2009. Cancún, Quintana Roo, México.

[6] Eli Biham and Adi Shamir. Differential Fault Analysis of Secret Key Cryptosystems. In *CRYPTO*, volume 1294 of *LNCS*, pages 513–525. Springer, August 1997. Santa Barbara, CA, USA.

[7] Dan Boneh, Richard A. DeMillo, and Richard J. Lipton. On the Importance of Checking Cryptographic Protocols for Faults. In *Proceedings of Eurocrypt'97*, volume 1233 of *LNCS*, pages 37–51. Springer, May 11-15 1997. Konstanz, Germany.

[8] Arnaud Boscher, Helena Handschuh, and Elena Trichina. Blinded Fault Resistant Exponentiation Revisited. In *FDTC*, pages 3–9. IEEE Computer Society, September 6 2009. Lausanne, Switzerland.

[9] Arnaud Boscher, Robert Naciri, and Emmanuel Prouff. CRT RSA Algorithm Protected Against Fault Attacks. In *WISTP*, volume 4462 of *LNCS*, pages 229–243. Springer, May 9-11 2007. Heraklion, Crete, Greece.

[10] Zhimin Chen and Yujie Zhou. Dual-Rail Random Switching Logic: A Countermeasure to Reduce Side Channel Leakage. In *CHES*, volume 4249 of *LNCS*, pages 242–254. Springer, 2006. Yokohama, Japan, http://dx.doi.org/10.1007/11894063_20.

[11] Jean-Sébastien Coron and Avradip Mandal. PSS Is Secure against Random Fault Attacks. In *ASIACRYPT*, volume 5912 of *Lecture Notes in Computer Science*, pages 653–666. Springer, December 6-10 2009. Tokyo, Japan.

[12] Berndt M. Gammel and Stefan Mangard. On the duality of probing and fault attacks. Cryptology ePrint Archive, Report 2009/352, 2009. http://eprint.iacr.org/.

[13] Rosario Gennaro, Anna Lysyanskaya, Tal Malkin, Silvio Micali, and Tal Rabin. Algorithmic Tamper-Proof (ATP) Security: Theoretical Foundations for Security against Hardware Tampering. In *TCC*, volume 2951 of *Lecture Notes in Computer Science*, pages 258–277. Springer, February 19-21 2004. Cambridge, MA, USA.

[14] Christophe Giraud and Hugues Thiebeauld. A Survey on Fault Attacks. In Kluwer, editor, *CARDIS*, pages 159–176, 2004. Toulouse, France.

[15] Yuval Ishai, Manoj Prabhakaran, Amit Sahai, and David Wagner. Private Circuits II: Keeping Secrets in Tamperable Circuits. In *EUROCRYPT*, volume 4004 of *Lecture Notes in Computer Science*, pages 308–327. Springer, May 28 – June 1 2006. St. Petersburg, Russia.

[16] Mark G. Karpovsky, Konrad J. Kulikowski, and Alexander Taubin. Robust Protection against Fault Injection Attacks on

Smart Cards Implementing the Advanced Encryption Standard. In *DSN*, pages 93–101, June 2004. Florence, Italy.

[17] Paul C. Kocher, Joshua Jaffe, and Benjamin Jun. Differential Power Analysis. In *Proceedings of CRYPTO'99*, volume 1666 of *LNCS*, pages 388–397. Springer-Verlag, 1999. (PDF).

[18] Konrad J. Kulikowski, Mark G. Karpovsky, and Alexander Taubin. Power Attacks on Secure Hardware Based on Early Propagation of Data. In *IOLTS*, pages 131–138. IEEE Computer Society, 2006. Como, Italy.

[19] Régis Leveugle. Early Analysis of Fault-based Attack Effects in Secure Circuits. *IEEE Trans. Computers*, 56(10):1431–1434, 2007.

[20] Yang Li, Shigeto Gomisawa, Kazuo Sakiyama, and Kazuo Ohta. An Information Theoretic Perspective on the Differential Fault Analysis against AES. Cryptology ePrint Archive, Report 2010/032, 2010. http://eprint.iacr.org/.

[21] Tal Malkin, François-Xavier Standaert, and Moti Yung. A Comparative Cost/Security Analysis of Fault Attack Countermeasures. In *FDTC*, volume 4236 of *Lecture Notes in Computer Science*, pages 159–172. Springer, October 10 2006.

[22] Robert P. McEvoy, Colin C. Murphy, William P. Marnane, and Michael Tunstall. Isolated WDDL: A Hiding Countermeasure for Differential Power Analysis on FPGAs. *ACM Trans. Reconfigurable Technol. Syst. (TRETS)*, 2(1):1–23, 2009.

[23] Robert P. McEvoy, Michael Tunstall, Claire Whelan, Colin C. Murphy, and William P. Marnane. All-or-Nothing Transforms as a Countermeasure to Differential Side-Channel Analysis. Cryptology ePrint Archive, Report 2009/185, 2009. http://eprint.iacr.org/.

[24] Yannick Monnet, Marc Renaudin, Régis Leveugle, Christophe Clavier, and Pascal Moitrel. Case Study of a Fault Attack on Asynchronous DES Crypto-Processors. In *FDTC*, volume 4236 of *Lecture Notes in Computer Science*, pages 88–97. Springer, October 10 2006. Yokohama, Japan.

[25] Maxime Nassar, Shivam Bhasin, Jean-Luc Danger, Guillaume Duc, and Sylvain Guilley. BCDL: A high performance balanced DPL with global precharge and without early-evaluation. In *DATE'10*, pages 849–854. IEEE Computer Society, March 8-12 2010. Dresden, Germany.

[26] Pascal Paillier. Public-Key Cryptosystems Based on Composite Degree Residuosity Classes. In *EUROCRYPT*, volume 1592 of *Lecture Notes in Computer Science*, pages 223–238. Springer, May 2-6 1999. Prague, Czech Republic.

[27] Gilles Piret and Jean-Jacques Quisquater. A Differential Fault Attack Technique against SPN Structures, with Application to the AES and KHAZAD. In *CHES*, volume 2779 of *LNCS*, pages 77–88. Springer, September 2003. Cologne, Germany.

[28] Thomas Popp, Mario Kirschbaum, Thomas Zefferer, and Stefan Mangard. Evaluation of the Masked Logic Style MDPL on a Prototype Chip. In *CHES*, volume 4727 of *LNCS*, pages 81–94. Springer, Sept 2007. Vienna, Austria.

[29] NIST FIPS (Federal Information Processing Standards) publication 140-2. Security Requirements for Cryptographic Modules. page 69, May 25 2001.

[30] NIST FIPS (Federal Information Processing Standards) publication 140-3. Security Requirements for Cryptographic Modules (Draft, Revised). page 63, 09/11 2009.

[31] Francesco Regazzoni, Thomas Eisenbarth, Johann Großschädl, Luca Breveglieri, Paolo Ienne, Israel Koren, and Christof Paar. Power Attacks Resistance of Cryptographic S-Boxes with Added Error Detection Circuits. In *DFT*, pages 508–516. IEEE Computer Society, September 26-28 2007. Rome, Italy.

[32] Ronald L. Rivest, Adi Shamir, and Leonard M. Adleman. A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. *Commun. ACM*, 21(2):120–126, 1978.

[33] Bruno Robisson and Pascal Manet. Differential Behavioral Analysis. In *CHES*, volume 4727 of *LNCS*, pages 413–426. Springer, September 10-13 2007. Vienna, Austria.

[34] Nidhal Selmane, Shivam Bhasin, Sylvain Guilley, Tarik Graba, and Jean-Luc Danger. WDDL is Protected Against Setup Time Violation Attacks. In *FDTC*, pages 73–83. IEEE Computer Society, September 6th 2009. In conjunction with CHES'09, Lausanne, Switzerland. DOI: 10.1109/FDTC.2009.40; Online version: http://hal.archives-ouvertes.fr/hal-00410135/en/.

[35] Rafael Soares, Ney Calazans, Victor Lomné, Philippe Maurine, Lionel Torres, and Michel Robert. Evaluating the robustness of secure triple track logic through prototyping. In *SBCCI'08: Proceedings of the 21st symposium on Integrated circuits and system design*, pages 193–198, Gramado, Brazil, 2008. ACM.

[36] Daisuke Suzuki and Minoru Saeki. Security Evaluation of DPA Countermeasures Using Dual-Rail Pre-charge Logic Style. In *CHES*, volume 4249 of *LNCS*, pages 255–269. Springer, 2006. Yokohama, Japan. http://dx.doi.org/10.1007/11894063_21.

[37] Kris Tiri, David Hwang, Alireza Hodjat, Bo-Cheng Lai, Shenglin Yang, Patrick Schaumont, and Ingrid Verbauwhede. A side-channel leakage free coprocessor IC in 0.18 μm CMOS for Embedded AES-based Cryptographic and Biometric Processing. In *DAC*, pages 222–227. ACM, June 13-17 2005. San Diego, CA, USA.

[38] Kris Tiri and Ingrid Verbauwhede. A Logic Level Design Methodology for a Secure DPA Resistant ASIC or FPGA Implementation. In *DATE*, pages 246–251. IEEE Computer Society, February 2004. Paris, France.

[39] Michael Tunstall and Debdeep Mukhopadhyay. Differential Fault Analysis of the Advanced Encryption Standard using a Single Fault. Report 2009/575, 2009. http://eprint.iacr.org/2009/575.

[40] Sung-Ming Yen and Marc Joye. Checking Before Output May Not Be Enough Against Fault-Based Cryptanalysis. *IEEE Trans. Computers*, 49(9):967–970, 2000. DOI: 10.1109/12.869328.

[41] Sung-Ming Yen, Seungjoo Kim, Seongan Lim, and Sang-Jae Moon. RSA Speedup with Chinese Remainder Theorem Immune against Hardware Fault Cryptanalysis. *IEEE Trans. Computers*, 52(4):461–472, 2003. DOI: 10.1109/TC.2003.1190587.