High Precision EMFI Detector using Machine Learning and Sensor Fusion

Adrien Facon[1,2], Sylvain Guilley[1,2,3], Xuan-Thuy Ngo[1],
Robert Nguyen[1], Thomas Perianin[1], Ritu-Ranjan Shrivastwa[1,3]

[1] Secure-IC S.A.S., Rennes, FRANCE
{firstname.lastname@secure-ic.com}
[2] Département d'informatique de l'ENS, CNRS, PSL University, 75005 Paris, FRANCE
[3] LTCI, Télécom Paris, Institut Polytechnique de Paris, Paris, FRANCE

***Abstract*—As chips become more inter-connected, they are more exposed to both network and physical attacks rendering it pertinent to ensure a sufficient protection level to them. In this paper, we explain why it is worthwhile resorting to Artificial Intelligence (AI) for security event handling and present an experimental use-case of a crypto-accelerator protected by a massive fleet of digital sensors embedded on a Field-Programmable Gate Array (FPGA) board. The data from this fleet of sensors need to be aggregated and processed fast to produce exploitable information while maintaining a low false positive detection rate. We evaluate different Machine Learning (ML) techniques and conventional method of perturbation detection using sensor threshold. Analysis includes quantitative figures of merit regarding Electro-Magnetic Fault Injection (EMFI) detection comparing ML-based sensor teaming strategy and threshold-based individual sensor approach to establish significant gain in detection accuracy of the former. Upon carrying out detailed evaluation it is found that the best working ML method achieves $\sim$ 32% higher accuracy at fault detection (with 97% true positive rate) as compared to the conventional threshold based method (with only 65% true positive rate).**

***Index Terms*—Internet of Things (IoT), Artificial Intelligence (AI), Machine Learning (ML), Threat Detection, Cyber-Protection, Decision Making Process, Naive Bayes Classifier, Embedded Security, Cyber-Physical Attacks.**

I. Conceptual Approach & Motivations

Security chips have long been designed with primary goal to be tamper-proof. This strategy reflects the context of resistance by increasing the barriers between the secrets to conceal within the chip and an attacker. The chip was, thus, designed to be a "digital safe", with numerous security functions implemented following defense-in-depth methodology.

From a validation point of view, the device (named "TOE") needs to go through strict certification schemes such as "Common Criteria" (ISO/IEC 15408) which specify methods to grant a sufficient assurance level in those chips. However, this methodology relies on hypotheses on the "Operational Environment", which is expected to be "well-behaved". Addressing independently the inside and outside, security is relevant for unconnected devices. For instance, traditionally, the devices were designed to expose minimalistic Application Programming Interface (API) [1] and are seldom meant to be upgraded on-the-field after fabrication, since this is considered a weakness.

In the current context of connected chips, it is no longer realistic to grant security by reducing the possibilities of interaction with the user (that is the attacker). Indeed, chips take on more value if they can be operated in customized ways and be adapted to ones needs. We shall, thus, envision security in the context where the attacker is close to the secrets and has many degrees of freedom in attempting to trick the chip's defenses.

This situation is, however, not a net regression from the "golden age" of chips seen as shelves protecting a valuable pearl. Indeed, we, designers, can leverage on two innovations viz. firstly, security functions can be more complex than before, owing to greater possibility to integrate complex functions in silicon or in embedded firmware, and secondly, devices being connected consists in an increased attack surface, while at the same time allow for off-loaded security analysis within cloud (hence a still larger computational power required to investigate the security of the chip) and to cross-check security levels with other devices belonging to the same fleet. Indeed, the larger threats arise from cyber-attacks (hence, arising "over the top"), and simultaneously touch several devices, which can in response collude to detect collectively that they are in a dangerous situation, and, thus, take proactive actions to grant their own security. The Cloud can itself perform some AI treatments to detect simultaneous or similar incidents/attacks perpetrated at the same time on several devices, which is the sign of a distributed attack attempt (such as Mirai, Hajime, and other IoT botnet infections).

II. Sensing the Normal & the Pathological

Today, chips are equipped with multiple sensors, of different kinds, some with a primary goal of data acquisition functionality ([2], [3]), but also sensors for adaptation to the environment (like battery level, temperature, wireless activity

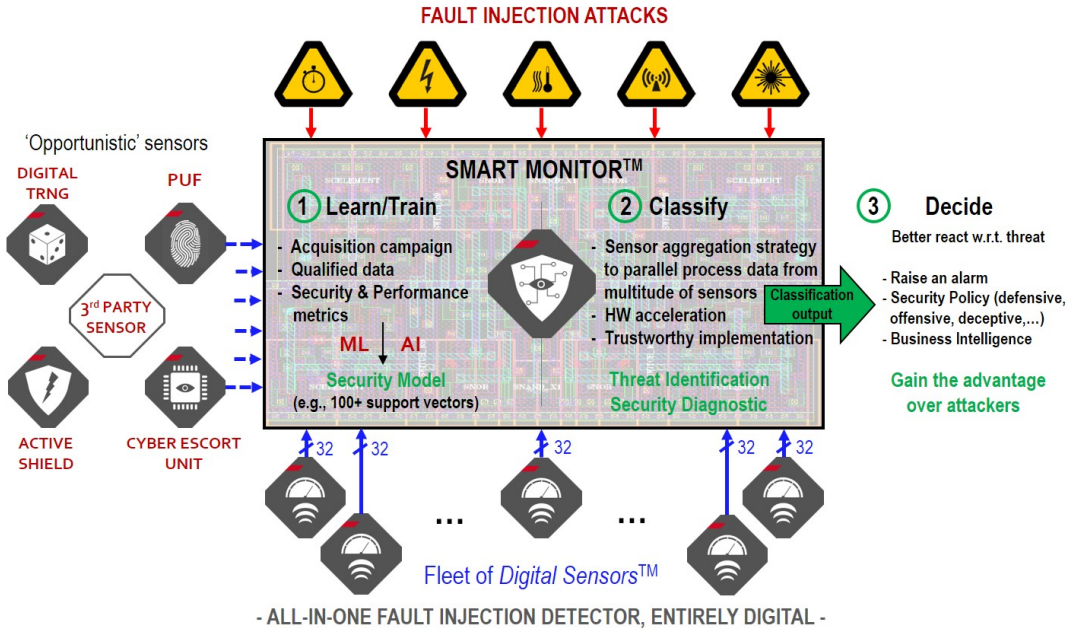

Figure 1: Smart Monitor is an AI-enabled on-chip security headquarter creating collective intelligence and coherence between IPs (analog or digital) and other whistleblowers and weak signals (software or hardware) improving both security event detection, analysis, diagnostic and decision-making process.

in the neighborhood, etc.). All those can advantageously complement security sensors. Security sensors watch events that would hint for attack conditions. Those events could be one of the following:

- abnormal physical operating conditions in terms of temperature, voltage, clock frequency, reset line stability, embedded health tests (on True Random Number Generators, Physically Unclonable Functions, etc.),
- abnormal activity (detection of port scanning, unexpected data flowing out the device, unusual load of processor, strange failure signals such as multiple segmentation violations detected by the kernel within a short period of time, etc.)

All these pieces of information can be processed to decide whether the device shall be considered in a *nominal* or in an *unsafe* environment. This is where AI comes into play. Indeed, AI is the solution to analyze fuzzy information arising from "big data" measurements collections. Aggregation of heterogeneous signals allow to leverage unexploited sources of information such as randomness quality, tiny clock modulation, noise statistic moments modification, etc. for security event detection.

Machine Learning (ML) techniques are widely used to diagnose various wireless sensor-based systems [4], [5], but such approaches have not yet been applied to hardware cybersecurity. Moreover for security chips, this AI shall run within the chips as an embedded hardware monitor [6], [7], as the rate and the volume of data collection is high, and because decisions must be taken fast. Indeed, a laser-induced or a malware-enabled attack requires few clock cycles to exploit the chip: installing a backdoor is a matter of kilobytes of payload. Therefore, instant detection is compulsory, which can only be achieved by a hardware approach.

AI is embedded under the form of a "Smart Monitor™". The rationale is the following:

- after tape-out, engineering samples are characterized in lab conditions: data from sensors are collected, and labeled by security condition (benign/malicious, attack discovery of payload injection phase, etc.)
- a model is derived and then saved in FLASH memory, protected by authentication
- at runtime, the Smart Monitor classifies the operation conditions according to the learned labels (security categories), and
- transmits the report to upper layers;
- in return, the security policy can be adapted from outside of the devices, if a presumption of an attack is computed from the correlation of the reports of several such devices.

This Intellectual Property (IP) finds application in smart devices (smartphones, laptops, smart home/office boxes), highly secure chipsets (hi-end smartcards, set-top boxes), cars (TCUs: Trusted Communication Units, ECUs: Electronic Control Units), and server security monitors for network appliances.

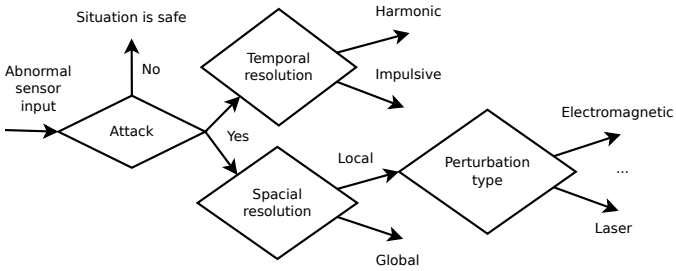

Figure 2: Various detection modalities in the "Smart Monitor™" IP.

The Smart Monitor not only improves the detection of a potential attack but also provides the diagnostic report. Additional information regarding the nature, temporality, locality, and intensity of the attack or even the attack phase can be derived (cf. illustration in Figure 2). This information allows to take the right decision at the right time in full knowledge of the situation. Security strategy can be adapted depending on the "anatomy" of the attack, either deceptive, defensive, offensive, analytical, etc. The ultimate goal is to be able to predict the threat and to stay ahead of it by gaining advantage over the attacker. All the features of this Machine Learning-enabled technology situated at the heart of silicon is depicted in Figure 1. There are three execution states viz. Learn/Train state, classification state, and decision state to apply security countermeasure (policy based on threat level) or simply raise an alarm.

Ultimately, this fruitful source of information allows to perform Business Intelligence. Indeed, Smart Monitor provides rich information regarding your devices post-deployment. This feedback, regarding attack typology and related statistics w.r.t device category, geographic area, technology nodes, etc. represent highly valuable information sourcing directly from the field and is made intelligible by powerful AI methods.

A part of this work is published in [7], where the main idea was to present a proof-of-concept about the Smart Monitor™ IP with a minimalistic implementation of the ML technique using sensor aggregation strategy. This work is, indeed, an extension and more accurate in-depth study on the same topic. We carried out extensive evaluation using different strategies to establish the quantitative gain of introducing AI in physical threat detection of EM fault injection at the hardware level.

III. Digital sensors

To illustrate the feasibility and the relevance of such an AI-driven approach, we present in this paper some selected results for improved detection of Electromagnetic Fault Injection attacks (in short EMFI) thanks to a plurality of dedicated digital sensors (DS).

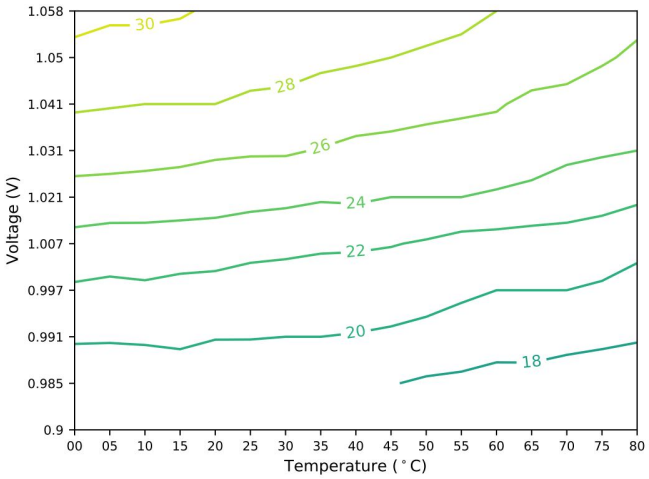

Figure 3: *Digital Sensor* iso-status curves. The outputted status of a DS, related to the propagation time of a signal along a delay chain, evaluates locally a threat level (here from 0 to 32) allowing threshold-based detection of FIA. The sensor's behavior regarding voltage and temperature variations has been theoretically studied and experimentally fully characterized as shown in this Figure.

Unlike analog sensors which are dedicated to the detection of a specific perturbation attack, *Digital Sensors* are delay chains which are longer than the critical path, thereby catching delay faults before any effect on the user logic. The rationale is, for instance, explained in [8, Fig. 14, page 189]. The digital sensor is designed to detect various threats belonging to the family of Fault Injection Attacks (FIA), such as clock glitch, overclocking, power glitch, underfeeding, heating, laser attack and EMFI. Individually, *Digital Sensor* (DS) converts all monitored stresses into a timing stress which is then measured. When a threat is detected, it provides the system with a measurement of the threat's level and it raises an alarm. These sensors are ultra-sensitive to temperature and voltage variations and beyond to internal on-chip activities (e.g., cryptographic hardware acceleration).

Primarily of security purpose, these digital sensors appear to be opportunistic temperature and voltage sensors as can be seen in Figure 3 which represents iso-status curves of one DS. The effect of increasing the temperature is to slow down the combinational logic, which results in an increase of the DS status. This increase can be compensated by an increase of the power supply, which accelerates the combinational gates when increased. Thus, the DS can advantageously be used as a correlated sensor, as recommended in the context of safety (cf. section D.2.10.2 of ISO 26262-5:2011).

This extreme sensitivity allows very accurate detection of FIA, but obligates the IP designer to set a precise threshold

(derived through simulations or empirically evaluated) which is far to be an easy task impacting directly the balance between false negative and false positive event detection. Additionally, sensors calibration are usually highly dependant of the target architecture and by essence hard to be transposed owing to technological dispersion. To illustrate the rich variations of the outputted status that we aim to exploit for improved detection, let us represent cartographies of values for a matrix of DS spread over on a Sakura–G FPGA board, also provided with a hardware crypto-accelerator.

Figure 4 shows snapshots of DS values in three situations viz. nominal, when the crypto-accelerator operates, and perturbed, when a very low-intensity EM injection is performed during AES (Advanced Encryption Standard) computation. Firstly, it can be observed that the output values of each identical DS, highly depends on its location. Secondly, the comparison between the three pictures allows to detect tiny "pixel–to–pixel" variations, revealing internal activity (AES computation) or perturbation attempt.

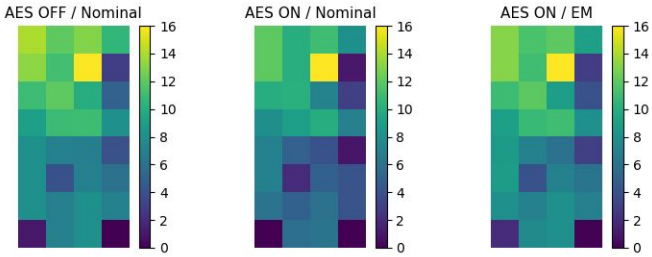

Figure 4: Cartography of a fleet of *Digital Sensors* spread over a Sakura–G FPGA board in a nominal situation (left), when the crypto-accelerator operates (center), and when a very low-intensity EM injection is performed during AES computation (right). Data obtained from DS with short status living in $[0, 16]$.

Nevertheless, DS–to–DS comparison contrast is low and gives rise to a poor detection signal, meaning in practice high false positive and false negative events. Moreover, effects can compensate giving rise to equal status in nominal situation and when AES computation is targeted by EM perturbation attacks. The need for sensor teaming —fusion of sensors— appears here to reduce the opportunity for and limit the damage of potential attacks. The ML-enhanced approach, that we propose here, allows to create collective intelligence between these individual IPs. By leveraging diversity of decalibrated sensors and complementarity, our goal is to combine the effort and show the benefits of multitude to gain assurance in threat detection: one may fool one sensor 1000 times, but may not fool 1000 sensors for once.

IV. Evaluation of EMFI Detection over different Machine Learning Models

In this section we discuss about the EMFI dataset that has been collected for both nominal and injected scenarios. To the dataset we apply different ML techniques to evaluate the performance of each in terms of detection accuracy. The process of preparing the dataset for training is also detailed below.

A. Dataset Information

The EMFI dataset is recorded from sixteen Digital Sensors (DS) on a chip executing AES encryption (shown in Figure 5). Same sized data is recorded for both Nominal (no EM Injection) and Injected states that are classified as classes 0 and 1 respectively (a binary classification problem). The same experiment is repeated with the EM probes placed at four different arbitrary locations on the chip. The choice of selecting the locations is based on the location of the Digital Sensors in the design i.e. from closest to farthest from the fleet of Digital Sensors, in order to have maximum coverage guarantee. Thus, there are four parts of the dataset with each having data for nominal and injected scenarios. Each part contains 1000 Test runs with each run comprising 13 cycles of sensor statuses from each DS. This can be understood from the directory tree below:

```
EMFI_Dataset/
├── Part1/ (Chip Location 0)
│   ├── Nominal (Class=0 : 1000 Tests)
│   └── Injected (Class=1 : 1000 Tests)
├── Part2/ (Chip Location 1)
│   ├── Nominal (Class=0 : 1000 Tests)
│   └── Injected (Class=1 : 1000 Tests)
├── Part3/ (Chip Location 2)
│   ├── Nominal (Class=0 : 1000 Tests)
│   └── Injected (Class=1 : 1000 Tests)
└── Part4/ (Chip Location 3)
    ├── Nominal (Class=0 : 1000 Tests)
    └── Injected (Class=1 : 1000 Tests)
```

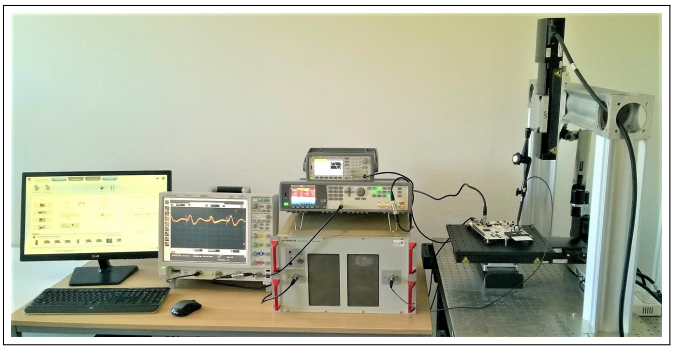

Figure 5: Apparatus setup for Electro-Magnetic Fault Injection at Secure-IC S.A.S. facility.

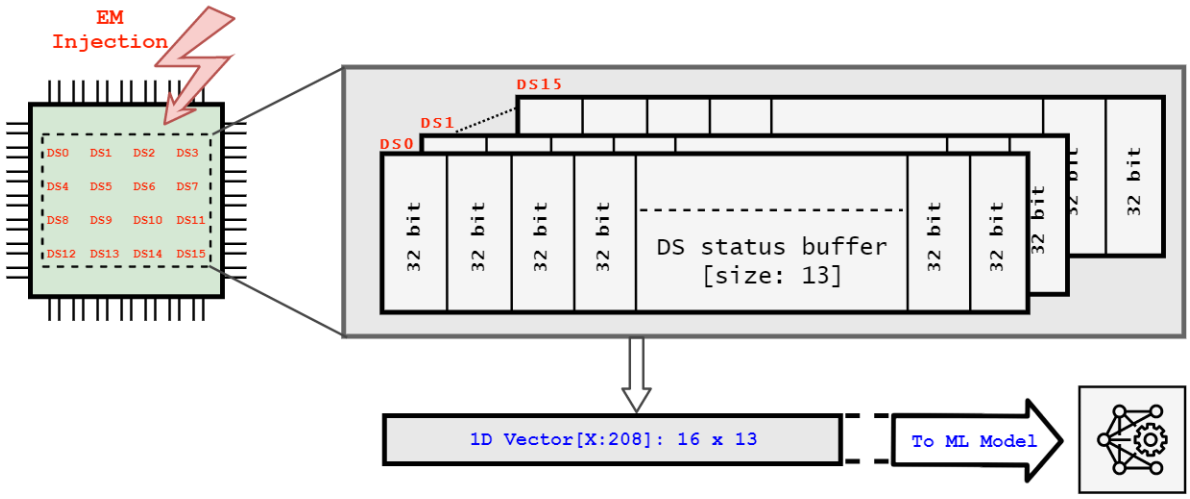

Figure 6: A total of 16 Digital Sensors™ (DS) are used for this experiment. The data aggregation is performed when the DS buffer is full, which has a depth of 13. Each 4 byte integer value from the buffer is linearly arranged for all the sensors to form a 1-D vector of size 208 values that is fed as an input to the ML models.

B. Preparing EMFI Data for ML Training/Inference

While performing train/test, the dataset is used part-wise as well as combined, in which all the four parts are combined into a single dataset and randomized for training. The significance of using separate and combined dataset is to evaluate the ML models' capacity in classifying between nominal and fault injected scenarios where the EM injection originates either from a single or multiple source locations, thus, making the model robust against locality of attack.

For each experiment, the ratio of *Train* and *Test* data is kept at $4 : 1$ (or 80% and 20%), respectively. Since the data representation is essential towards the accuracy of ML models, we formulate various methods of data pre-processing that are listed below:

1) Method 1: Each DS status per 13 cycles is fed sequentially into the model in the order DS0–DS12. Thus, input vector is of size 13×1.

2) Method 2: All the DS statuses per test (13 consecutive cycles), is arranged linearly and fed to the model. The input vector for this is of size 208×1 (13×16).

3) Method 3: A moving average of all the DS statuses in one test (13 cycles) is computed and fed to the model i.e. an average of 16 vectors each of size 13. The input vector for this method is same as Method 1 i.e. 13×1.

4) Method 4: In this method, for each DS per Test, the mean of the 13 status values is computed and all the values are stored in a vector of size 16×1 to feed the model.

5) Method 5: In this method, for each cycle, the status values of each DS is arranged together into one vector of size 16×1 before feeding the model.

Empirical analysis proves that Method **2** (IV-B2) is the best candidate for simplistic ML classification algorithms. Figure 6 shows the data aggregation methodology for Method 2. Each DS is equipped with a status buffer of size 13 with each memory element capable of storing a 4 bytes integer value. This is due to the fact that it holds a large number of spread-out parameters in single input. The averaging methods, also, contain same information but, for simple ML models it is difficult to unveil the features from them.

C. Accuracy Comparison of Tested ML Models

For our experiment we perform evaluation on four different ML models including Support Vector Machines (SVM), Logistic Regression Classifier (LRC), Naive Bayes Classifier (NBC), and Multi-Layered Perceptron (MLP). Each ML model is tested with all the five data representations to establish that the second method works best in all cases and is, thus, used in all further evaluations. A visual comparison over the performance of all the ML models is presented in figure 7.

V. Results: Comparison between Thresholding and ML Methods for EMFI Detection

In this section we try to establish the significant improvement in EM injection detection rate using ML models over a thresholding technique. For a better quantification of accuracy gain, an optimized thresholding technique is devised and then compared with the best performing ML model. This experiment comprises two parts viz. Optimizing the threshold of the DSs, and testing the performance of the thresholding technique over an unseen test dataset.

A. Threshold Optimization of every DS

For each digital sensor, firstly, the data is converted to the average form as shown below in equation 1, where X is the input vector (of size 13) and X' is the vector obtained after the averaging process.

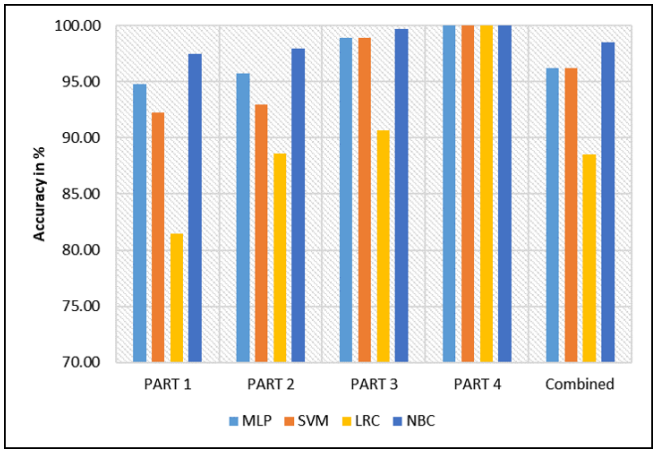

Figure 7: Performance comparison as accuracy in predicting EM Fault Injection from DS states over four different ML methods. Each method is tested separately over the four different parts as well as over the combined dataset (as detailed in section IV-A). Naive Bayes Classifier (NBC) outperforms other methods.

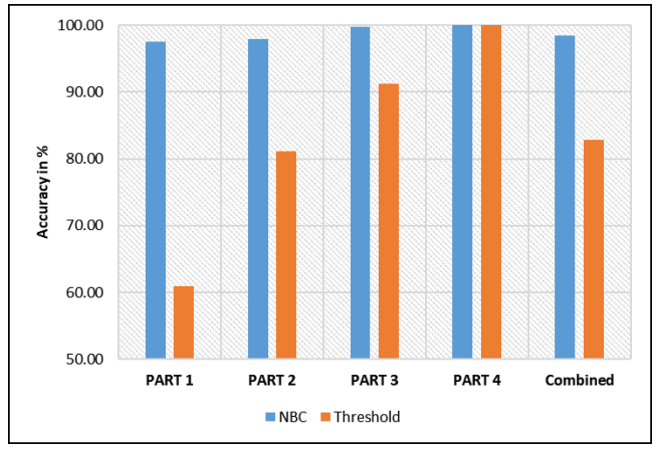

Figure 8: Performance comparison in accuracy of predicting EM Fault Injection, from aggregated DS states, of Naive Bayes Classifier (NBC) and Thresholding Method on the EMFI Dataset. While there is minimal difference in accuracy for the NBC over different parts of the dataset, the accuracy of threshold method is significantly affected.

$$\begin{aligned} &\forall i \text{ in } i = \{0, 1, ..., 12\} \text{ and,} \\ &X = \{V_i\} \text{ and } X' = \{V_i'\}, \text{ where,} \\ &V_i' = V_i \text{ if } i = 0, \text{ else } V_i' = (V_i + V_{i-1})/2 \end{aligned} \tag{1}$$

Secondly, a linear search algorithm, performed over 80% of the dataset (similar to training in ML methods), finds the bounds of both the classes (0: Non-injection, 1:Injection). The bounds are Lower/Upper for class Zero/One. The lower bound for class 0 is sufficient as a threshold boundary for classification, and is, thus, chosen as the threshold for each sensor. Thus, a test function simply places the incoming sensor statuses within the bounds and predict the classes. This implies that a value higher/lower (as per calibration) than the threshold would send an alarm signal stating "injection detected" scenario.

B. Accuracy Evaluation and Comparison with best ML method

Post optimizing the threshold, a test analysis with the remaining 20% of the dataset is performed. The results are then compared with the results from ML analysis as shown in the Figure 8. The graph clearly depicts that the results of NBC are not affected by the position of the EM probe on the chip, and offers a near equal prediction in all cases. On the other hand, the accuracy of the threshold method has dependency over the localization of EM injection i.e., it only works well in cases where the sensing is better.

A detailed comparison of both the methods with False Positives and Negatives is shown in Table I. It is evident that false positives is nearly accurate in both cases i.e. both methods are able to pass non-injection scenarios. However, the real challenge is to detect an injection activity and the rate of detection is much higher (**97.00**%) in case of ML method as compared to the threshold method (**65.32**%).

	(VALUES IN %)				
	False Positive	**False Negative**	**Acc: w/o injection**	**Acc: with injection**	**Overall**
	Naive Bayes Classifier				
PART 1	0.00	2.50	100.00	95.00	**97.50**
PART 2	0.00	2.02	100.00	95.96	**97.98**
PART 3	0.00	0.28	100.00	99.45	**99.72**
PART 4	0.00	0.00	100.00	100.00	**100.00**
Combined	0.00	1.49	100.00	**97.00**	**98.51**
	Thresholding Method				
PART 1	0.08	39.00	99.85	22.00	60.92
PART 2	0.00	18.83	100.00	62.35	81.17
PART 3	0.01	8.76	99.99	82.43	91.23
PART 4	0.00	0.00	100.00	100.00	100.00
Combined	0.00	17.18	100.00	**65.32**	82.82

Table I: In-depth comparison of Threshold and the best performing ML Model (NBC).

VI. Some more results

Machine learning methods are an exquisite solution to manage the uncertainty and aggregating multivariate (noise) information. To that end, we perform supervised ML on all DS outputs, in diverse situation to output a ML-based model, that acts as "a collective threshold" for binary classification. Figure 9 presents the detection accuracy on a selected test dataset, to compare the efficiency of the Smart Monitor with single DS signals. Given the extremely tiny amplitude

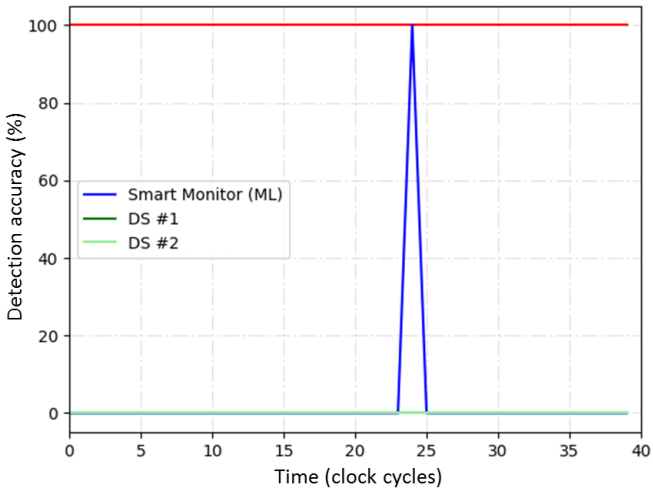

Figure 9: Comparison of the injection detection rate of Smart-Monitor with 2 individual DS signals arbitrary chosen. The EM injection is performed at clock cycle 24 and its duration is exactly one cycle. The detection accuracy, on a selected test dataset with an injection at fixed duration, achieves 100% while keeping a null false positive rate at nominal cycles.

of the EM variations, the single sensor fails to detect the perturbation. The teaming strategy (Smart Monitor) provides an intelligent aggregation of single DS signal and eliminates false negatives at the time of injection while keeping the false positive rate at 0 during unperturbed computation.

We quantitatively evaluate the effect of sensor teaming using a second dataset, generated from a chip embedding 32 DSs. Figure 10 presents statistics of EMFI detection rate (true positive performance value) when the statuses from 1 to 32 sensors are used as inputs of the ML model during the learning phase. The graph shows the diversity of detection performance on different training and validation datasets, and more importantly with different selections of sensors. It illustrates the sensor teaming approach as a winning strategy and moreover proves that the extracted information highly depends on the selected sensors. Flyer points show that using a small number of sensors, as little as 1, can lead to excellent accuracy, up to 100%, but also to a very poor performance, showing the importance of the considered DS.

This gives insight on which sensors brings the most useful information (discriminating between nominal and non-nominal situations) and thus reveals the precise localization of the attack as the position of a few DS that could be matched on Figure 4. Choosing a high number of sensors (over 15) guarantees an accuracy around 90% and can achieve up to 100%. The 10% margin is due to material uncertainty over the EM injection timing, leading to misalignments on the supervised dataset, and occasionally inducing misclassified items.

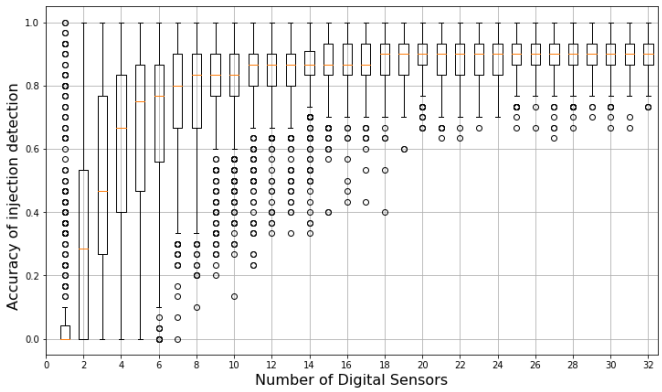

Figure 10: Boxplots of EMFI detection performances (true positive detection) for various sensor aggregation size and various test datasets. The statistics are gathered over 500 experiments by randomly splitting the dataset into train and test and randomly selecting the DS to consider during the training phase. The boxes are drawn from quartile 1 (Q1) to quartile 3 (Q3) while the whiskers are sets to Q1 - 1.5IQR and Q3 + 1.5IQR with IQR being the interquartile range. The flyers points represents samples beyond the whiskers limits. The number of sensors included in the collective machine learning model is studied to show the benefits of the approach for EMFI detection compared to individual sensor threshold-based model. The average detection score considering 20 DS is increased by 3 times comparing to using only 2 DS.

VII. Conclusion

If EMFI on chip focuses on local effect, it often shows a very characteristic diffusion phenomena of the signal that can be of useful information. An imaginable evolution of the Smart Monitor would be to consider DS signals as temporal sequences of cycles, in order to detect more efficiently the injection timing and locality by learning the inherent noise induced by the perturbation.

In our implementation effort to contrast inferencing of the DS data to detect a physical EM injection on the chips, we show how machine learning methods are more robust in terms of injection localization of the EM probes and also near accurate to classify between a non-injected and injected state. We present in our case simple linear ML algorithms and for comparative analysis we use MLP. Complex non-linear neural-networks based AI techniques are not implemented as to maintain a low-power light-weight core on the hardware fabric with very-high throughput. Owing to the fact that the Digital Sensors produce continuous status signals, the data distribution is Gaussian, and therefore we use Gaussian NBC as the final choice, since it works best for this problem.

As a final perspective, we attract the reader's attention on the fact that the Smart Monitor itself can be the target of attacks. The adversary's goal is to manipulate the chip sensors such that it can be corrupted while being misled to be under

a nominal or benignly abnormal operating conditions. Hence, widespread use of AI methods should be carefully tailored for security purpose, requiring high-value security expertise and methodology for trustworthy AI implementation.

ACKNOWLEDGMENTS

The French FUI program CSAFE+ funded part of this work. Besides, the side-channel analysis part of this work has benefited from a funding via TeamPlay (https://teamplay-h2020.eu/), a project from European Union's Horizon2020 research and innovation programme, under grant agreement N° 779882.

REFERENCES

[1] S. Guilley, J.-L. Danger, R. Nguyen, and P. Nguyen, "System-Level Methods to Prevent Reverse-Engineering, Cloning, and Trojan Insertion," in *ICISTM (PPREW workshop)*, ser. Communications in Computer and Information Science, S. Dua, A. Gangopadhyay, P. Thulasiraman, U. Straccia, M. A. Shepherd, and B. Stein, Eds., vol. 285. Springer, 2012, pp. 433–438.

[2] N. Miura, Z. Najm, W. He, S. Bhasin, X. T. Ngo, M. Nagata, and J. Danger, "PLL to the rescue: a novel EM fault countermeasure," in *Proceedings of the 53rd Annual Design Automation Conference, DAC 2016, Austin, TX, USA, June 5-9, 2016*, 2016, pp. 90:1–90:6. [Online]. Available: http://doi.acm.org/10.1145/2897937.2898065

[3] W. He, J. Breier, S. Bhasin, N. Miura, and M. Nagata, "An FPGA-compatible PLL-based sensor against fault injection attack," in *22nd Asia and South Pacific Design Automation Conference, ASP-DAC 2017, Chiba, Japan, January 16-19, 2017*, 2017, pp. 39–40. [Online]. Available: https://doi.org/10.1109/ASPDAC.2017.7858291

[4] M. Bahrepour, N. Meratnia, and P. J. M. Havinga, "Use of AI techniques for residential fire detection in wireless sensor networks," in *Proceedings of the Workshops of the 5th IFIP Conference on Artificial Intelligence Applications & Innovations (AIAI-2009), Thessaloniki, Greece, April 23-25, 2009*, 2009, pp. 311–321. [Online]. Available: http://ceur-ws.org/Vol-475/AIAEP/33-pp-311-321-409.pdf

[5] M. A. Alsheikh, S. Lin, D. Niyato, and H. P. Tan, "Machine learning in wireless sensor networks: Algorithms, strategies, and applications," *IEEE Communications Surveys and Tutorials*, vol. 16, no. 4, pp. 1996–2018, 2014. [Online]. Available: https://doi.org/10.1109/COMST.2014.2320099

[6] C. Deshpande, B. Yuce, N. F. Ghalaty, D. Ganta, P. Schaumont, and L. Nazhandali, "A configurable and lightweight timing monitor for fault attack detection," in *IEEE Computer Society Annual Symposium on VLSI, ISVLSI 2016, Pittsburgh, PA, USA, July 11-13, 2016*, 2016, pp. 461–466. [Online]. Available: https://doi.org/10.1109/ISVLSI.2016.123

[7] A. Facon, S. Guilley, X. T. Ngo, and T. Perianin, "Hardware-enabled AI for Embedded Security: A New Paradigm," in *2019 3rd International Conference on Recent Advances in Signal Processing, Telecommunications Computing (SigTelCom)*, March 2019, pp. 80–84, Hanoi, Vietnam.

[8] N. Selmane, S. Bhasin, S. Guilley, and J.-L. Danger, "Security evaluation of application-specific integrated circuits and field programmable gate arrays against setup time violation attacks," *IET Information Security*, vol. 5, no. 4, pp. 181–190, December 2011, DOI: 10.1049/iet-ifs.2010.0238. [Online]. Available: http://www.scopus.com/inward/record.url?eid=2-s2.0-82055176045{&}partnerID=40{&}md5=06115dddc18329ee1ff40f6af025c9ce