iSTELLAR: intermittent Signature aTtenuation Embedded CRYPTO with Low-Level metAl Routing

Jeremy Blackstone*, Debayan Das†, Alric Althoff‡, Shreyas Sen†, Ryan Kastner*
*University of California San Diego
{jblackst, kastner}@ucsd.edu
†Purdue University {das60, shreyas}@purdue.edu
‡Tortuga Logic {alric}@tortugalogic.com

***Abstract*—An adversary can exploit side-channel information such as power consumption, electromagnetic (EM) emanations, acoustic vibrations or the timing of encryption operations to derive the secret key from an electronic device. Signature aTtenuation Embedded CRYPTO with Low-Level metAl Routing (STELLAR) is a technique to mitigate power and EM-based attacks, however, it incurs 50% power overhead. This work presents `iSTELLAR`, which reduces the power overhead by operating STELLAR intermittently utilizing an intelligent scheduling algorithm. The proposed scheduling algorithm for `iSTELLAR` determines the optimal locations during the crypto operation to turn STELLAR ON, and thereby reduces the power overhead by >30% compared to the normal STELLAR operation, while eliminating the information leakage.**

I. Introduction

The development of devices containing greater amounts of sensitive information has sparked the creation of many strong, mathematically secure cryptographic algorithms. Unfortunately, side channel analysis (SCA) attacks bypass these algorithms by monitoring the effects of the algorithm on a physical platform through power consumption, electromagnetic emanations (EM), timing of operations, or acoustic vibrations. By measuring these aspects of the physical computations, an attacker is able to discover sensitive information, e.g., extracting the secret key from a cryptographic algorithm.

Signature aTtenuation Embedded CRYPTO with Low-Level metAl Routing (STELLAR) mitigates power and EM SCA by routing the cryptographic core through the lower metal layers, and then embedding the crypto core locally within a signature attenuation hardware, which suppresses the critical correlated crypto signature significantly before passing it through to the higher metal layers to connect to the external pins [1]. STELLAR, proposed in 2019 [1], was shown to be secure against EM and power SCA even after 1M encryptions for an AES-128 [2]. However, STELLAR incurs a 50% power overhead to actively suppress the crypto power signature (when compared to the unprotected crypto core). To address this issue, we propose `iSTELLAR` – a programmable technique to implement STELLAR intermittently based on the side-channel leakage, thereby providing security while reducing the power overhead.

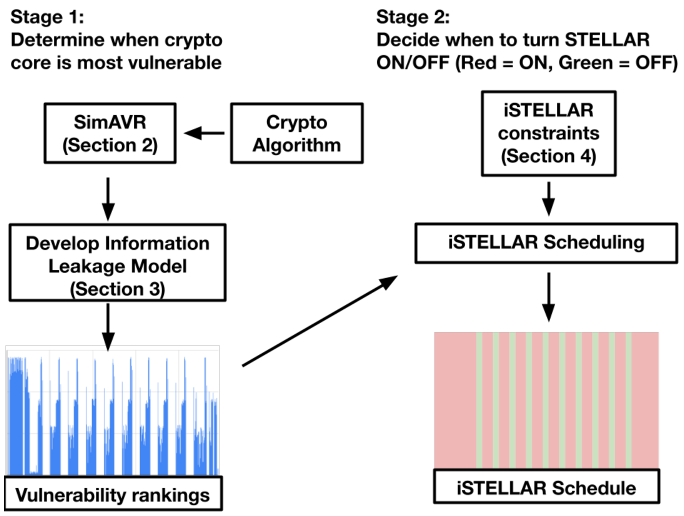

Stage 1:
Determine when crypto core is most vulnerable
SimAVR (Section 2)
Crypto Algorithm
Develop Information Leakage Model (Section 3)
Vulnerability rankings
Stage 2:
Decide when to turn STELLAR ON/OFF (Red = ON, Green = OFF)
iSTELLAR constraints (Section 4)
iSTELLAR Scheduling
iSTELLAR Schedule

Fig. 1: STELLAR provides high resilience to side channel attacks but requires 50% power overhead. `iSTELLAR` performs STELLAR intermittently to reduce overhead in two stages. In the first stage we generate joint mutual information (JMI) rankings to determine which parts of the algorithm's execution reveal information about the key. In the second stage, we generate an iSTELLAR schedule to determine when STELLAR should be turned ON (shown in red) or OFF (shown in green).

The goal of `iSTELLAR` is to minimize the power consumption while maximizing the security of the crypto implementations against SCA attacks. Unfortunately, `iSTELLAR` cannot instantly turn ON and OFF, as it requires a turn ON delay of a few cycles. In this paper, we study those start-up constraints and propose scheduling algorithms to only turn ON `iSTELLAR` at critical time points to minimize both power overhead and information leakage (i.e., maximize SCA security).

In order to maximize the SCA security, we determine which periods of time to turn STELLAR ON based on information leakage as shown in figure 1. We develop an information leakage model by selecting an algorithm and performing an implementation of the selected algorithm in SimAVR [3]. Using our information leakage model, we rank the information leakage of the implementation cycle by cycle based on its usefulness to an attacker in a side channel attack. Utilizing these information leakage rankings and iSTELLAR's constraints, the iSTELLAR scheduling algorithm decides the cycles in which to turn STELLAR ON or turn it OFF.

The major contributions of this paper are:

- Analysis of how to operate STELLAR intermittently.
- A careful accounting of a specific trade-off space to maximize security and minimize power consumption.
- Analysis of algorithms to minimize information leakage under specific power thresholds.

The remainder of this paper is organized as follows. Section II defines the threat model, describes STELLAR, and motivates the use of iSTELLAR. Section III outlines the security evaluation. Section IV presents iSTELLAR scheduling algorithms, which aim to minimize power overhead and information leakage. Section V describes the results of performing iSTELLAR algorithms on three implementations of cryptographic algorithms. Section VI discusses related work and Section VII provides concluding remarks.

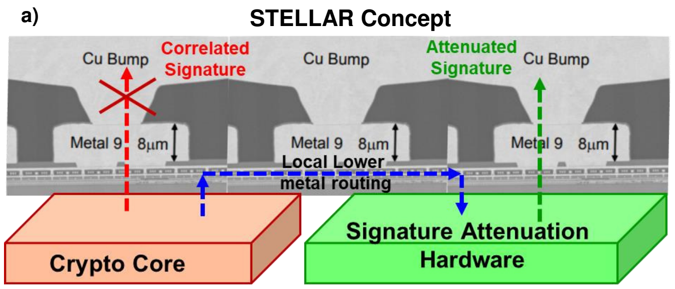
II. Background

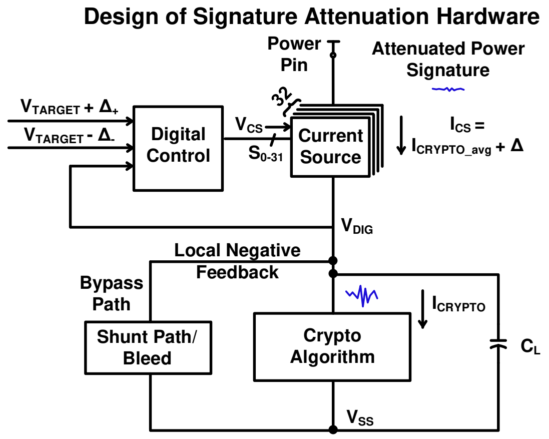
A. Threat Model

We assume the attacker has physical possession of the target device, is able to run security critical programs with arbitrary inputs, and is capable of collecting detailed power or EM traces. Our experimental results target the attack of cryptographic algorithms running on a microcontroller. Our analysis can be extended to other scenarios given appropriate attacks and power/EM models. We assume the attacker is aware of when the computation of the cryptographic algorithm begins with microsecond level precision (e.g., by using simple power analysis) and is able to collect multiple traces. We make no assumptions about the equipment the attacker uses to collect traces.

B. STELLAR

The concept of STELLAR is shown in figure. 2(a). The STELLAR technique proposes routing the crypto core within the lower metal layers and then embedding it within a *signature attenuation hardware (SAH)* which suppresses the correlated critical crypto signature significantly before it reaches to the higher metal layers. figure 2(b) shows the design of the SAH, which ensures that the current from the top *current source (CS)* remains almost equal to the average crypto current independent of the crypto core switching. STELLAR prevents against both EM as well as power SCA attacks. For EM SCA protection, STELLAR is placed locally within the lower-level metal layers embedding the crypto core, which is also routed in the lower metals. It has been shown that the higher metal layers leak more EM radiation compared to the lower metals due to its larger dimensions. Hence, STELLAR uses the SAH circuit to attenuate the critical signature within the lower metal layers before it goes through the higher metals to prevent against EM SCA. The SAH reduces the correlated signal to noise ratio (SNR) significantly and hence provides high power SCA immunity as well [1], [4].





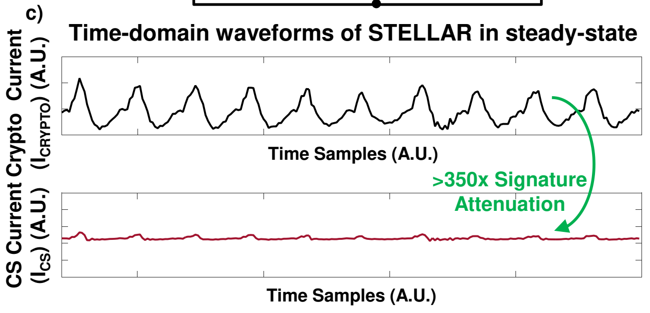


Fig. 2: (a) STELLAR technique; and (b) design of the Signature Attenuation Hardware (SAH). (c) Time-domain measurements of the unprotected and protected AES with the SAH (STELLAR) shows $> 350\times$ current-domain signature attenuation.

As shown in figure 2(b), the SAH is designed with a CS on top which provides a high output impedance ensuring that the voltage fluctuations across the entire execution of AES are significantly suppressed before the critical information reaches the supply pin. Now, as the top CS current cannot be exactly equal to the average crypto current, it is set to the closest quantization level, and the quantization error (Δ) is bypassed through a shunt bleed path. Also, to compensate for process, voltage and temperature (PVT), a switched mode digital control loop with a guard band is used which turns ON or OFF the required number of CS slices to maintain an average crypto current from the top. Recently, in 2021 [5], a fully-synthesizable implementation of the STELLAR has also been proposed. iSTELLAR aims to intermittently turn ON the STELLAR countermeasure – only protecting the most sensitive computations. Hence, it is important to analyze the start-up constraints involved with the SAH design.

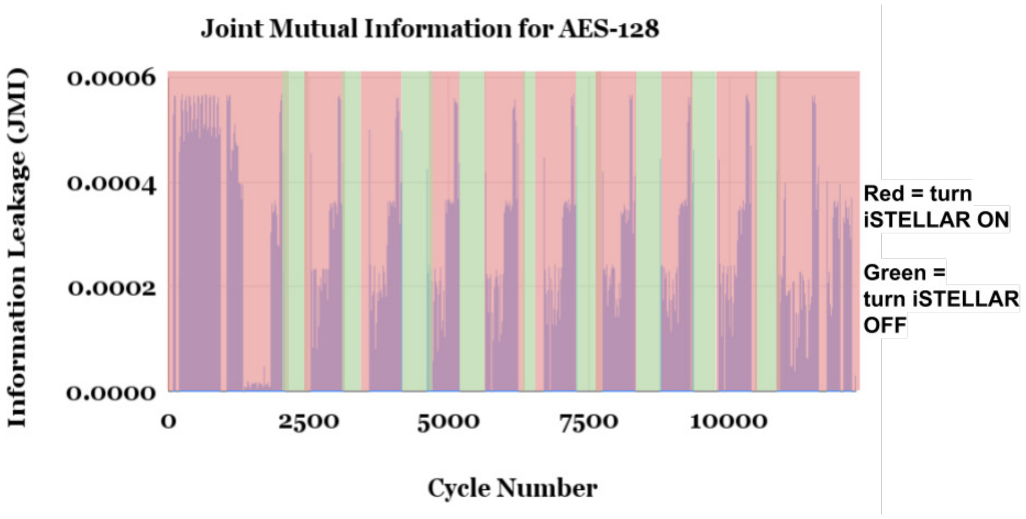


Fig. 3: The joint mutual information (JMI) of a subset of a power trace collected from an AES-128 implementation from DPA Contest v4.2 [6]. The x-axis shows the time (cycle number) and the y-axis the JMI, which corresponds to the amount of information leakage. Cycles that have larger JMI reveal more information about the key.

C. Motivation

Figure 3 provides an example of why STELLAR incurs larger than necessary overhead by protecting all portions of the computation. It shows information leakage of a software implementation of AES-128 (obtained from the DPA Contest v4.2 [6]) running on an AVR microcontroller. The leakage is computed using joint mutual information (JMI)(JMI explained in detail in section III). Samples of the power trace that have larger JMI values reveal more information about the secret key. While some cycles have very high information leakage (noted by the high peaks in JMI), other cycles have low leakage or zero leakage. Given this, we can turn STELLAR ON for the cycles with high leakage (shown in red), and turn STELLAR OFF for the cycles with low or no leakage to minimize the power consumption (shown in green).

III. Security Evaluation

`iSTELLAR` requires security evaluation in order to determine when it is most beneficial to turn STELLAR ON and how effective STELLAR will be when turned ON. We choose to use joint mutual information(JMI) as a quantitative metric to determine the most vulnerable points in the algorithm's execution and use minimum traces to disclosure (MTD) [7] as a qualitative metric to determine how effective STELLAR will in mitigating SCA.

A. Joint Mutual Information (JMI)

1) Usage: We use JMI to rank time periods within the traces and these rankings are used to determine when we turn STELLAR ON/OFF [8], [9]. We calculate JMI using equation 1.

$$JMI_i = \sum_{j \in B} I(f(t_i, \hat{m}, \hat{s}) \frown f(t_j, \hat{m}, \hat{s}); \hat{s}). \tag{1}$$

$a \frown b$ calculates the concatenation between a and b, and f is a function used to represent a trace. The trace takes an independently and uniformly random message and the secret key from vectors $\hat{m}$ and $\hat{s}$. I(C; D) determines the mutual information for a and b which determines how much we can learn about C based on how it is related to D. It is calculated using equation 2 where H(C) is the entropy of C and $H(C|D)$ is the conditional entropy of C given D. B is the set of indices (i) that we have chosen to turn STELLAR ON.

$$I(C; D) = H(C) - H(C|D). \tag{2}$$

As indices are added to B, they are given a ranking based on their JMI. The index with the greatest mutual information with the key will be ranked highest and would be selected first. The algorithm repeats this process with successive time indices ranked according to how much easier they would make it for an attacker to recover the key. Once the time indices have been ranked, the JMI scores are normalized so that the sum of all the JMI rankings is 1.

We determine JMI during design time to ensure we perform `iSTELLAR` in a manner that is consistent across any and all executions of the cryptographic algorithm. The `iSTELLAR` schedule is determined before the execution starts, and is the same regardless of the data that is being computed. Thus, this does not introduce a timing channel.

2) Reasoning: One reason we choose joint mutual information (JMI) as a metric is because it considers how each point in an algorithm's execution is related to all the other points [10], [11]. Consider a scenario where $x1 \oplus x2 = y$ under the assumption that x1 and x2 are statistically independent Boolean variables [12]. In this scenario, the mutual information between x1 and y =0 and the mutual information between x2 and y is 0. However, if we concatenate x1 and x2 ,the mutual information between the concatenation of x1, x2 and y is greater than 0 because the information from these Boolean

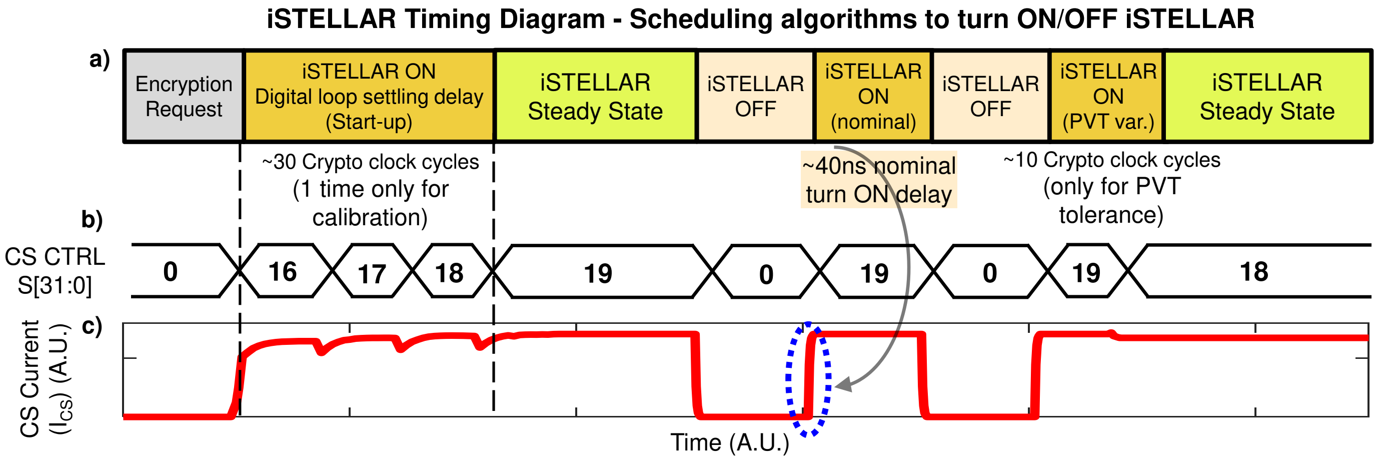

iSTELLAR Timing Diagram - Scheduling algorithms to turn ON/OFF iSTELLAR
a)
Encryption Request
iSTELLAR ON Digital loop settling delay (Start-up)
iSTELLAR Steady State
iSTELLAR OFF
iSTELLAR ON (nominal)
iSTELLAR OFF
iSTELLAR ON (PVT var.)
iSTELLAR Steady State
~30 Crypto clock cycles (1 time only for calibration)
~40ns nominal turn ON delay
~10 Crypto clock cycles (only for PVT tolerance)
b)
CS CTRL S[31:0]
0
16
17
18
19
0
19
0
19
18
c)
CS Current (I_CS) (A.U.)
Time (A.U.)

Fig. 4: (a) Timing diagram of the `iSTELLAR` operation: At start-up, for the first time (only for one-time calibration), the delay is contributed by the digital control loop settling delay, which takes $\sim$ 3 cycles to settle to the number of CS slices to ensure average crypto current from the top CS and the digital loop runs at a $10\times$ lower frequency than the crypto core. In steady state, the main delay is caused due to the settling time of the analog biases to set the top CS stage in saturation to ensure a high output impedance and maximize the signature attenuation of the current source. (b) Output of the digital control loop determines the number of CS slices to be turned ON. (c) CS current with the `iSTELLAR` operation, showing $\sim 40ns$ turn ON delays for all different conditions. The steady state delay constraint (circled in blue) is also illustrated in figure 5 which needs to be accounted for in the `iSTELLAR` scheduling algorithm.

variables completely determines y. Similarly, a SCA security evaluation metric such as the Test Vector Leakage Assessment (TVLA) may determine the time index of a security-sensitive algorithm has low vulnerability by considering just that index. However, combining functions or multivariate histograms with a few time indices, it may still be attractive to an adversary if considered alongside additional time indices. While metrics that only consider one point in an algorithm's execution at a time are very effective for determining vulnerability to a number of powerful attacks [13]–[16], an implementation may still be vulnerable to attacks which consider multiple points in an algorithm's execution at a time [17]–[19].

Another reason we choose joint mutual information as a metric is because it provides a numeric score [10]. Having a numeric score allows us to precisely compare each cycle in an algorithm's execution to other cycles and precisely compare one potential `iSTELLAR` schedule to any other potential `iSTELLAR` schedules. We assume that each cycle in an algorithm's execution that handles sensitive information has the potential to reveal it if left unprotected and this is indicated by a non-zero JMI value. However, different cycles may reveal differing amounts of information to an adversary. Time periods with high JMI values indicate high vulnerability to SCA attacks and similarly time periods with low JMI values indicate low vulnerability to SCA attacks.

While we performed our analysis using JMI, our methodology can easily use other leakage assessment metrics. Any metric capable of providing a numeric score to compare one cycle in an algorithm's execution to another could be used in place of JMI to rank and determine the time points that are most vulnerable to SCA.

B. Minimum Traces to Disclosure (MTD)

1) Usage: MTD is defined as the number of measurements necessary to distinguish the correct secret key guess from other incorrect secret key guesses in a side channel attack [7]. Figure 2(c) shows that the signature in case of the protected implementation is significantly suppressed by $> 350\times$ compared to the unprotected implementation, thereby promising a $> 350^2\times$ improvement in the MTD. Recently, STELLAR has been fabricated in TSMC 65nm process using AES256 as the crypto core [20]. Embracing signature attenuation and local lower metal routing, the countermeasure achieved a MTD of $> 1B$, which to the best of our knowledge is the best reported result to date. Due to the efficiency of STELLAR, we assume that if STELLAR is turned ON, then the cycles it protects will not leak sufficient information for an adversary to launch a SCA attack and these cycles will assume a JMI value of 0.

2) Reasoning: We chose minimum traces to disclosure (MTD) as a metric because it models the time and effort it would take for an adversary to successfully launch an SCA attack in a practical scenario.**We did not use MTD to compare each cycle in algorithm's execution to the others because it is computationally expensive when considering how each point in an algorithm's execution is related to all other points**. If we need to consider how each point is related to all other points we must calculate the MTD for (n:1), (n:2), (n:3)...(n:m) where m is the number of time points an adversary uses for their attack and $m < n$. This increases the time complexity to O(n:m).

IV. `iSTELLAR`

STELLAR can remain ON for any amount of time. However, if we turn STELLAR OFF to conserve power during cycles with low or no leakage, we must wait a certain amount of time for STELLAR to complete its start-up process. As a

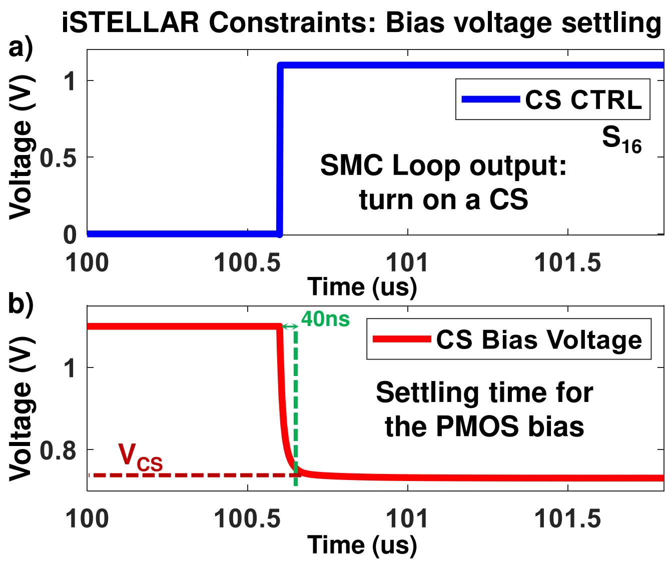


Fig. 5: Transient analysis of the CS bias voltage settling time. (a) As the digital controller turns on a CS stage, (b) the PMOS voltage requires a settling time of $\sim 40ns$ to bias the CS into saturation to provide a high output impedance for STELLAR.

result, we must determine when to turn STELLAR ON/OFF while allowing it the necessary time to start back up after being turned OFF.

A. Constraints

The timing diagram for the `iSTELLAR` operation is shown in figure 4. At start-up, the signature attenuation hardware (SAH) requires a few cycles for the digital switched mode control (SMC) loop to set the CS current at the average crypto current. The SMC, which runs at $10\times$ lower frequency compared to the crypto core, requires $\sim 2-3$ cycles to reach steady state, which is $\sim 20-30$ crypto clock cycles. In steady state, the scheduling algorithms for `iSTELLAR` intermittently turn the SAH ON and OFF depending on the information leakage content in the traces. Once the digital loop is stable and the STELLAR operates in steady-state, the `iSTELLAR` algorithm may want to turn STELLAR OFF after a certain time point. With STELLAR turned OFF, once `iSTELLAR` turns it ON again, the CS biases need to settle to the correct bias voltages to ensure the transistors remain in saturation for a high output impedance thereby leading to a high signature attenuation.

Figure 5(a) shows that one of the 32 bits of the SMC output (S_{16}) switches from zero to one, which means that the corresponding CS slice should be turned ON. Now, to turn ON a CS slice, the bias voltage at the gate of the PMOS needs to transition from the supply voltage (1.1V) to the CS bias voltage (V_{CS}). With the AES operating at 50MHz, the settling time for the PMOS bias is $\sim 40ns$. This implies that, we need to wait for $\sim 40ns$ once we turn OFF STELLAR to turn it back ON. For the purpose of our calculations in sections IV-C and V-B we conservatively estimate this analog settling time to have a turn ON delay of 3 cycles in our digital loop to account for a broad range of clock speeds. This analog settling time as shown in figure 5 is $\sim 40ns$, and this is the typical steady state delay the `iSTELLAR` needs to handle each time the STELLAR SAH circuit is turned ON and OFF. Note that this delay is due to the settling of the analog bias voltages to turn ON the PMOS CS slice, and is equal to the RC time constant for the node, where R is the on resistance and C is the gate capacitance of the PMOS. Now, if the frequency of the crypto core is increased, the size of the PMOS CS slice can be increased accordingly so that the RC time constant remains the same (R reduces, C increases due to larger size). Hence, `iSTELLAR` only needs to deal with the analog bias settling delay of 40ns as it turns OFF and ON the PMOS CS (figure 5). While this is typically the only delay, `iSTELLAR` may require up to 10 crypto clock cycles (equivalent to 1 cycle of the SMC loop) to account for the PVT(process/voltage/temperature) variations and settle to the optimal number of CS slices, as illustrated in figure 4.

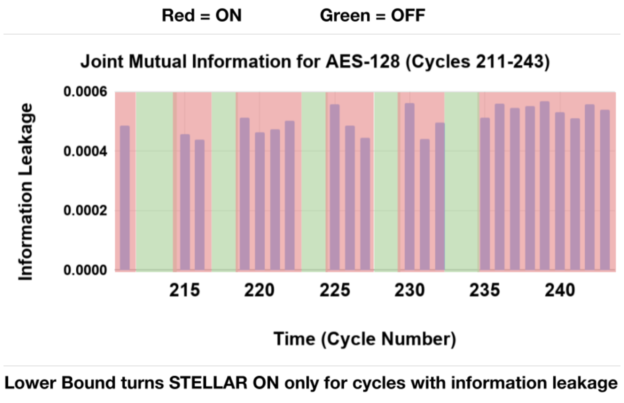
1) Overheads & Performance of iSTELLAR: As discussed, in steady state operation of the STELLAR, the only constraint for turning it ON and OFF is the turn ON delay for the analog bias to bring the top current source stage into saturation region. Our scheduling algorithm for `iSTELLAR` ensures that at the end of its OFF time, it turns ON the STELLAR circuit for the bias settling. Hence, there are no performance overheads associated with the turn ON or OFF of the STELLAR hardware.

Overall, the area overhead for STELLAR is $\sim 40\%$ and the power overhead is $\sim 50\%$ [1], which is drastically reduced using the proposed `iSTELLAR` technique without incurring any performance penalty.

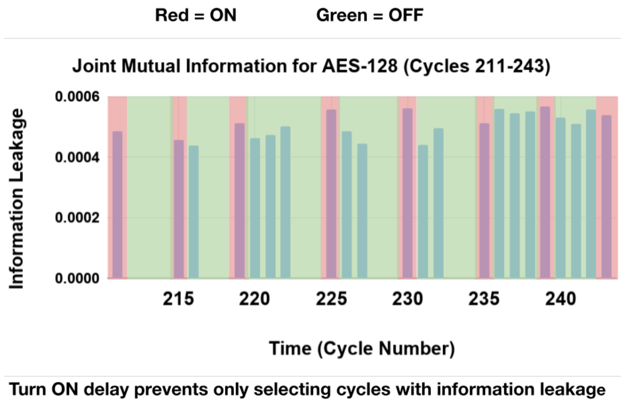
B. `iSTELLAR` Lower Bound

To determine the best times to turn STELLAR ON, we began by developing a best-case scenario algorithm to minimize power consumption and maximize security to establish `iSTELLAR`'s lower bound for power consumption. In this algorithm, we assume we can turn STELLAR back ON immediately after turning it OFF.

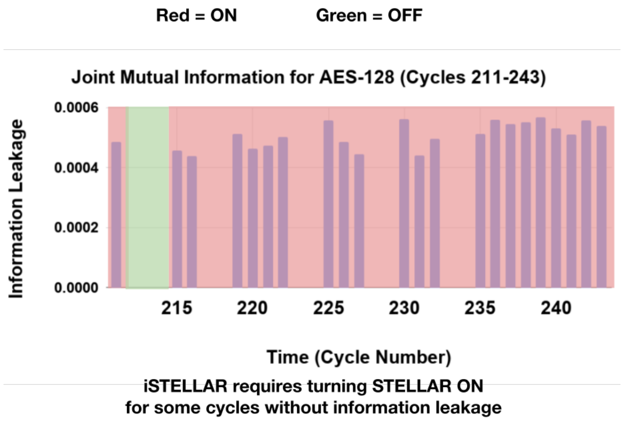
First, we determine the total information leakage for a trace according to each cycle's JMI. Next, we find the highest leakage cycle and mark it as a cycle where we would like to turn STELLAR ON. After this, we add it to a list of cycles we plan to turn STELLAR ON for and determine the new total information leakage and power overhead for turning STELLAR ON. We continue this process for every cycle that has a JMI greater than zero or until we reach a power overhead threshold selected by the designer. For example, in figure 6a, we would mark cycles in the following order: 239, 230, 225, 242, 238, 237, 243, 240, 235, 219, 241,222, 232, 226, 211, 221, 220, 215, 227, 231, 216. After marking these cycles we would choose to turn STELLAR ON for the cycles we selected (shown in red) and OFF for the cycles we did not select (shown in green).



(a) `iSTELLAR` Lower Bound



(b) Challenge



(c) `iSTELLAR`

Fig. 6: (a) By turning ON STELLAR (shown in red), we can hide parts of the trace, which effectively eliminates the information leakage, that is, JMI = 0. (b) However, if we turn it OFF for cycles where there is no information leakage, we must wait for a set amount of cycles (shown in green) for it to start back up . (c) If we keep STELLAR ON for some cycles that have no information leakage, then we are able to hide nearby cycles with high information leakage.

C. `iSTELLAR` Scheduling

Using the lower bound algorithm under realistic constraints presents an interesting challenge due to the turn ON delay required after turning STELLAR OFF. In most circumstances, this will be the 40 ns turn ON delay which we conservatively estimate to be 3 crypto clock cycles to account for a broad range of clock speeds. This means that after we turn STELLAR OFF, we must wait 3 clock cycles before turning STELLAR back ON. As a result, if any cycle we plan to select is within 3 cycles of a cycle we have already selected, we will have to ignore it because it would violate the 3 cycle turn ON delay. Similarly, if we want to account for the 10 cycle turn ON delay due to PVT variations, we ignore cycles that would violate the 10 cycle turn ON delay

In figure 6b, the lower bound algorithm would mark cycles 230 and 239 because they have the highest JMI. Next, it would mark cycle 225 rather than cycle 236 because it is within 3 cycles of another cycle we have already marked. For the same reason, cycles 216-218, 220-224, 226-229, 231-234,236-238, and 240-242 would be ignored and cycles 215, 219, 235 and 243 would be marked.

To further decrease information leakage, we chose to check for and address cycles that occur during the turn ON delay between STELLAR being turned OFF and being turned back ON. If a new cycle we plan to mark would violate the turn ON delay of a cycle we have already marked, we will keep STELLAR ON between the cycle we plan to mark and the cycle we have already marked.

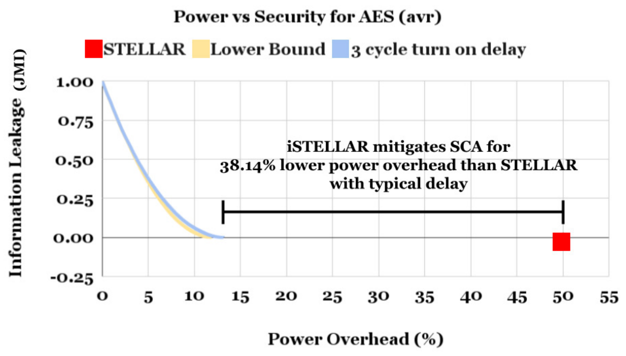
Figure 6c shows how using this approach addresses the challenge from our first approach. Using this approach, the lower bound algorithm would still mark cycles 230 and 239 because they have the highest JMI. Next, it would mark cycle 236 and realize that it would violate the turn ON delay if we turned STELLAR OFF then tried to turn it ON again for cycle 239. As a result, we would also mark cycles 237 and 238 even though they do not have the next highest JMI. This process continues until we have marked all of the cycles that have a JMI value greater than 0. It will also mark cycles 216-217, 223-224, 228-229 and 233-234 even though they have a JMI value of 0 to ensure STELLAR stays ON between cycles that would violate each others' turn ON delay. However, it will not mark cycles 212-214 because no surrounding cycles violate each others' turn ON delay time. Even though we use power to turn STELLAR ON for cycles where STELLAR is unnecessary, we are able to achieve higher security at a lower power overhead.

V. Results

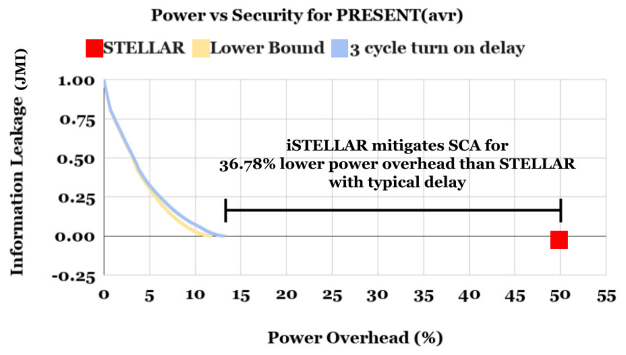
Using `iSTELLAR`'s constraints, the JMI rankings for each cycle, `iSTELLAR`'s power overhead lower bound and an `iSTELLAR` scheduling algorithm, we are able to develop an iSTELLAR schedule to minimize information leakage and power overhead. We can choose to keep STELLAR OFF for as many cycles as we choose, to make tradeoffs between power and security. This section evaluates these tradeoffs.

A. Experimental Setup

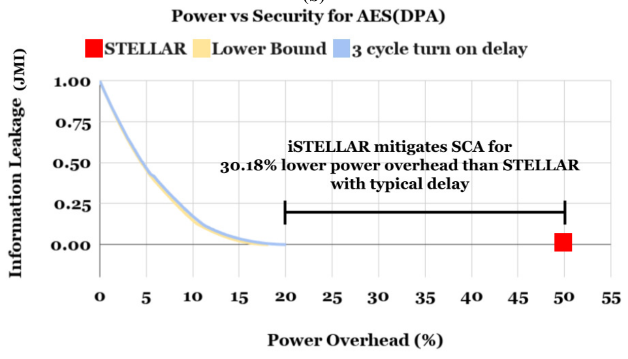
We develop power traces using SimAVR to simulate an Atmel ATmega328 chip [3]. SimAVR is capable of executing binaries compiled by the avr-gcc toolchain as they would be run on an AVR microcontroller and we use it to collect power traces using a Hamming distance leakage model [13].



(a)

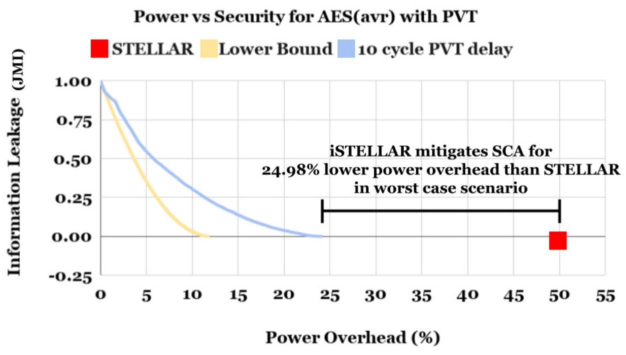


(b)

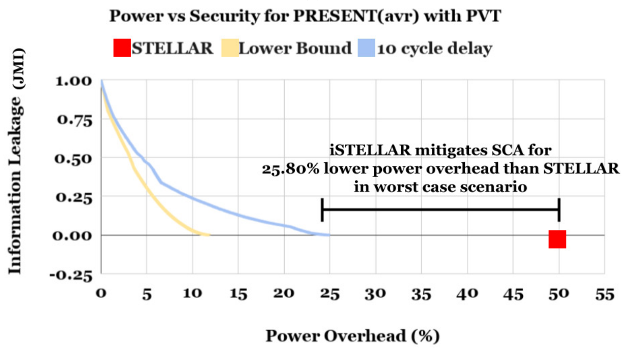


(c)

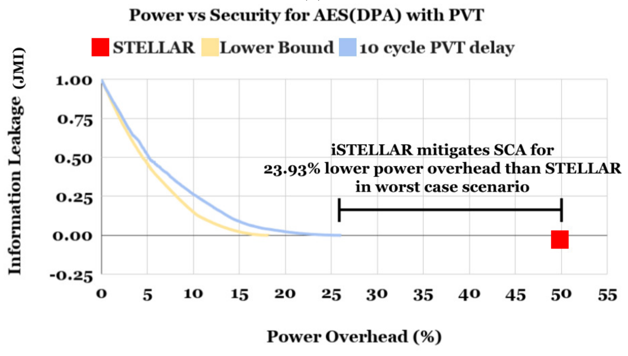
Fig. 7: (a) Considering the 3 cycle turn ON delay, iSTELLAR achieves 38.14% less power overhead on AES(avr), (b) 36.78% less power overhead on PRESENT(avr), and (c) 30.18% less power overhead than STELLAR on AES(DPA)



(a)



(b)



(c)

Fig. 8: (a) Considering the process/voltage/temperature (PVT) variations, iSTELLAR achieves 24.98% less power overhead on AES (avr), (b) 25.80% less power overhead on PRESENT (avr), (c) and 23.93% less power overhead than STELLAR on AES (DPA).

Although we evaluate power traces, a similar experimental setup could be performed to use EM traces.

To perform our evaluation, we collect power traces for 2^{14} experimental plaintext and secret key vectors on an implementation of AES-128 from DPA Contest v4.2 [6], an implementation of PRESENT [21] from the avr library and an implementation of AES from the avr library. Under this model, each time point in a trace consists of the difference in Hamming distance between an opcode and its predecessor for different experimental plaintext and secret key vectors [12]. For our information leakage model we assume that toggling a bit consumes one bit of power and leaving a bit unchanged consumes no power. Our information leakage evaluation is independent of the instruction type or the type of the data.

B. *Power vs Security*

Since STELLAR is responsible for eliminating information leakage, we consider any decision to keep STELLAR ON/OFF as a tradeoff between power and security. We can choose to maximize security by ensuring STELLAR is ON for every cycle that leaks information or we can minimize power overhead by having the designer select a specific power threshold and only turning STELLAR ON for cycles the iSTELLAR scheduling algorithm has marked up to that point. Once we establish an iSTELLAR schedule, the schedule will be constant for all executions of the algorithm regardless of the input data to avoid introducing a timing channel.

Figure 7a shows the power and security for using iSTELLAR on an implementation of AES-128 from the avr library. The red square labeled STELLAR is representative

TABLE I: Power Overhead for iSTELLAR

Algorithm	AES(avr)	PRESENT(avr)	AES(DPA)
STELLAR	50%	50%	50%
Lower Bound	11.87%	12.67%	19.82%
3 cycle turn ON delay	13.14%	13.22%	20.05%
10 cycle PVT delay	24.20%	25.02%	26.07%

of the baseline STELLAR technique. This assumes that no `iSTELLAR` scheduling algorithm is used and STELLAR remains ON for the entire algorithm's execution. Under these conditions, STELLAR incurs a 50% power overhead, but is able to eliminate power and EM leakage to reduce the sum of JMI rankings to 0. This means it is not possible to launch an SCA attack because an adversary will not be able to differentiate between different secret key hypotheses by measuring the power consumption or electromagnetic emanations.

The yellow line in figure 7a labeled Lower Bound assumes the scenario outlined in section IV-B which ignores `iSTELLAR`'s turn ON delay constraint to establish a best-case scenario for minimizing power consumption and maximizing security. Under these conditions, the sum of JMI rankings starts at 1 which means power and EM leakage is completely unmitigated and vulnerable to attack. If we choose to turn STELLAR on, only for cycles that have a JMI value great than 0, we are able to reduce the sum of JMI rankings to 0 for 11.87% power overhead. Additionally, if 11.87% power overhead is still too high, the designer can establish a power threshold, and the iSTELLAR scheduling algorithm is able to maximize security for that threshold.

The blue line in figure 7a labeled 3 cycle turn on delay assumes the scenario outlined in section IV-C which requires a 3 cycle turn ON delay to turn STELLAR back ON after it has been turned OFF. Under these conditions, we are able to reduce the sum of JMI rankings to 0 for 13.14% power overhead, only 1.27% higher than the Lower Bound scenario. Furthermore, we are also able to maximize security within a power threshold for around the same power overhead as the Lower Bound scenario.

Figure 7b shows the power and security for using `iSTELLAR` on an implementation of PRESENT from the avr library and 7c shows the power and security for using an implementation of AES-128 from DPA Contest v4.2. Figure 8 repeats the scenarios for 7 with the a 10 cycle PVT turn ON delay rather than the typical 3 cycle turn ON delay to account for worst case conditions. These results are summarized in table I.

C. Discussion

We believe that the efficiency of iSTELLAR may be dependent on the leakage distribution in the implementations of the different algorithms. The implementation of AES-128 from DPA Contest v4.2 may have the power overhead closest to the Lower Bound scenario because its leakage is distributed over fewer cycles. While 28.71% of the cycles in this implementation of AES-128 have a JMI value of 0, only 14.06% of the cycles in the AES-128 implementation from the avr library have a JMI value of 0, and only 13.57% of the cycles in the PRESENT from the avr library have a JMI value of 0. We further note that the implementations of PRESENT and AES-128 from the avr library have similar leakage distributions and similar power overheads necessary to turn STELLAR ON for.

VI. Related Work

Power side-channel countermeasures attempt to modify the power trace signal to hide any information related to the key. Some techniques add active equalization circuitry to diminish power variations during execution and keep the power supply constant [22], [23]. Other techniques use signal attenuation hardware to reduce the power cost of noise injection [24] or use a suppression circuit to reduce low frequency power variations and a low-pass filter to reduce high frequency power variations [25]. There are some ideas to use internal power sources which an adversary cannot modify e.g., a charge-pump circuit using on-chip capacitors [26] and a switched capacitor circuit to isolate an AES core from the power supply line [27]. The disadvantage to these works is that they were not applied selectively in order to allow designers to make trade-offs between performance, area, and security. While computational blinking [12] also identifies non-uniformity in information leakage, it implements a switched capacitor circuit rather than using signal attenuation hardware. As a result, applying the different techniques intermittently has different accommodations and requires different constraints. STELLAR is one of the most recent proposals as a signature attenuation-based countermeasure and has been demonstrated to achieve the highest security (MTD of 1B traces) with only $\sim 50\%$ power and $\sim 40\%$ area overheads [4], [5]. Moreover, it is a generic countermeasure (agnostic to any crypto algorithm) without any performance degradation.

VII. Conclusion

Although STELLAR provides protection from power and EM SCA, it incurs larger than necessary power overhead because many of the cycles it protects do not have any information leakage. By turning STELLAR ON and OFF, we are able to to eliminate all information leakage with minimal power overhead. However, turning STELLAR OFF, we must give it a set amount of cycles to turn back ON. Utilizing our proposed scheduling algorithm for `iSTELLAR`, we address this issue by turning STELLAR ON for all cycles with high information leakage as well as some of prior cycles with low information leakage to avoid violating startup constraints. We were able to eliminate information leakage with 38.14% lower power overhead.

Acknowledgements

This work was supported primarily by NSF grant No. 1563767. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect those of the National Science Foundation.

References

[1] D. Das et al., "Stellar: A generic em side-channel attack protection through ground-up root-cause analysis.," in *HOST*, pp. 11–20, 2019.

[2] P. FIPS, "197: the official aes standard," *Figure2: Working scheme with four LFSRs and their IV generation LFSR1 LFSR*, vol. 2, 2001.

[3] M. Pollet, "Simavr," 2017 https://github.com/buserror/simavr.

[4] D. Das et al., "EM and Power SCA-Resilient AES-256 Through >350x Current-Domain Signature Attenuation and Local Lower Metal Routing," *IEEE Journal of Solid-State Circuits*, pp. 1–1, 2020. Conference Name: IEEE Journal of Solid-State Circuits.

[5] A. Ghosh et al., "36.2 An EM/Power SCA-Resilient AES-256 with Synthesizable Signature Attenuation Using Digital-Friendly Current Source and RO-Bleed-Based Integrated Local Feedback and Global Switched-Mode Control," in *IEEE ISSCC*, vol. 64, pp. 499–501, Feb. 2021. ISSN: 2376-8606.

[6] C. Clavier et al., "Practical improvements of side- channel attacks on aes: feedback from the 2nd dpa contest," *Journal of Cryptographic Engineering*, pp. 259–274, 2014.

[7] K. Tiri, D. Hwang, A. Hodjat, B.-C. Lai, S. Yang, P. Schaumont, and I. Verbauwhede, "Prototype ic with wddl and differential routing–dpa resistance assessment," in *International Workshop on Cryptographic Hardware and Embedded Systems*, pp. 354–365, Springer, 2005.

[8] P. Meyer et al., "Information-theoretic feature selection in microarray data using variable complementarity," *IEEE Journal of Selected Topics in Signal Processing*, pp. 261–274, 2008.

[9] H. Yang et al., "Data visualization and feature selection: New algorithms for nongaussian data," *Advances in Neural Information Processing Systems*, pp. 687–693, 2000.

[10] A. Althoff, J. Blackstone, and R. Kastner, "Holistic power side-channel leakage assessment: Towards a robust multidimensional metric," in *2019 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pp. 1–8, IEEE, 2019.

[11] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *Journal of machine learning research*, vol. 3, no. Mar, pp. 1157–1182, 2003.

[12] A. Althoff et al., "Hiding intermittent information leakage with architectural support for blinking," *International Symposium on Computer Architecture*, pp. 638–649, 2018.

[13] E. Brier et al., "Correlation power analysis with a leakage model," *International Workshop on Crypto- graphic Hardware and Embedded Systems*, pp. 16–29, 2004.

[14] P. Kocher et al., "Differential power analysis," in *Annual International Cryptology Conference*, pp. 388–397, Springer, 1999.

[15] E.-P. Duan, Y.-J. Yan, and P.-Z. Li, "Correlation electromagnetic analysis against aes cryptographic on implementations of fpga," *Computer Engineering and Design*, vol. 33, no. 8, pp. 2926–2930, 2012.

[16] G.-l. Ding, Z.-x. Li, X.-l. Chang, and Q. Zhao, "Differential electromagnetic analysis on aes cryptographic system," in *2009 Second Pacific-Asia Conference on Web Mining and Web-based Application*, pp. 120–123, IEEE, 2009.

[17] S. Chari, J. R. Rao, and P. Rohatgi, "Template attacks," in *International Workshop on Cryptographic Hardware and Embedded Systems*, pp. 13–28, Springer, 2002.

[18] B. Gierlichs, L. Batina, B. Preneel, and I. Verbauwhede, "Revisiting higher-order dpa attacks," in *Cryptographers' Track at the RSA Conference*, pp. 221–234, Springer, 2010.

[19] L. Mather, E. Oswald, and C. Whitnall, "Multi-target dpa attacks: Pushing dpa beyond the limits of a desktop computer," in *International Conference on the Theory and Application of Cryptology and Information Security*, pp. 243–261, Springer, 2014.

[20] D. Das et al., "27.3 em and power sca-resilient aes-256 in 65nm cmos through 350× current-domain signature attenuation," in *IEEE ISSCC*, pp. 424–426, 2020.

[21] A. Bogdanov, L. R. Knudsen, G. Leander, C. Paar, A. Poschmann, M. J. Robshaw, Y. Seurin, and C. Vikkelsoe, "Present: An ultra-lightweight block cipher," in *International workshop on cryptographic hardware and embedded systems*, pp. 450–466, Springer, 2007.

[22] R. Muresan et al., "Protection circuit against differ- ential power analysis attacks for smart cards," *IEEE Transactions on Computers*, pp. 1540–1549, 2008.

[23] R. Muresan et al., "On-chip current flattening circuit with dynamic voltage scaling," *IEEE International Symposium on Circuits and Systems*, pp. 4–pp, 2006.

[24] D. Das et al., "Asni: Attenuated signature noise injection for low-overhead power side-channel attack immunity," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 65, no. 10, pp. 3300–3311, 2018.

[25] R. Williams et al., "An on-chip signal suppression countermeasure to power analysis attacks," *IEEE Transactions on Dependable and Secure Computing*, pp. 179–189, 2004.

[26] S. Perri et al., "An integrated countermeasure against differential power analysis for secure smart- cards," *IEEE International Symposium on Circuits and Systems*, pp. 4–pp, 2006.

[27] C. Tokunaga et al., "Secure aes engine with a local switched-capacitor current equalizer," in *2009 IEEE International Solid-State Circuits Conference-Digest of Technical Papers*, pp. 64–65, IEEE, 2009.