Active and Passive Side-Channel Key Recovery Attacks on Ascon

Keyvan Ramezanpour[1], Abubakr Abdulgadir[2], William Diehl[1], Jens-Peter Kaps[2], and Paul Ampadu[1]

[1] Virginia Tech, Blacksburg, VA 24061, USA{rkeyvan8,wdiehl,ampadu}@vt.edu
[2] George Mason University, Fairfax, VA 22033, USA {aabdulga,jkaps}@gmu.edu

Abstract. Physical exposure of devices to adversaries in unprotected environments in the era of the Internet of Things (IoT) necessitates evaluation of cryptographic hardware implementations against side-channel analysis (SCA). The Ascon authenticated cipher has been accepted to Round 2 of the U.S. National Institute of Standards and Technology (NIST) Lightweight Cryptography (LWC) Standardization Process, and was selected as the first choice of the CAESAR committee for the lightweight use case. In this paper, we evaluate the vulnerability of Ascon to both passive and active SCA attacks. Using a lightweight implementation of Ascon on an Artix-7 FPGA, we demonstrate a successful statistical ineffective fault analysis (SIFA) attack using voltage glitches on the supply pin of the FPGA chip. Using only 280 correct values of the output authentication tags under fault injection into a pair of S-boxes, one subset of the secret key (equivalent to two bits) is recovered. We also demonstrate that a power analysis attack using a deep learning technique is able to find the secret key using 24K power traces during S-box computations at the beginning of the Initialization stage of Ascon. Conversely, classical DPA and CPA attacks fail to find the correct key with more than 40K traces.

Keywords: Ascon · Fault injection analysis · SIFA · Side Channel Analysis · Deep Learning · FPGA · Voltage glitch · FOBOS.

1 Introduction

Increased connectivity of devices and ubiquitous data transfer in the era of the Internet of Things (IoT) increase demand for security provisions for a wide range of applications from high performance to lightweight computing platforms. The growth of resource-constrained IoT devices has necessitated the development of lightweight cryptography (LWC) for low-overhead implementation of various security services in hardware. Authenticated encryption with associated data (AEAD) integrates confidentiality, integrity and authentication in a single algorithm, possibly facilitating lightweight implementations. Authenticated ciphers, which are based on symmetric cryptography, generate a unique fixed-length output, called the authentication tag, for every encrypted message. The tag can be

used by the receiver to verify the integrity of transmitted data and authenticity of the sender.

Standardized cryptographic algorithms are usually secure against cryptanalysis, in which an attacker attempts to recover the secret key by observing a set of inputs and the corresponding outputs of an algorithm. However, physical exposure of IoT and mobile devices in unprotected environments to potential adversaries has enabled a class of physical attacks on the hardware implementation of the algorithms, called side-channel analysis (SCA). The U.S. National Institute of Standards and Technology (NIST) has launched a multi-year standardization process for lightweight cryptography (LWC), in which authenticated ciphers are evaluated for efficient implementation on resource-constrained platforms, and ease of inclusion of countermeasures against side-channel attacks.

The Ascon authenticated cipher is one of the candidates selected for the Round 2 of NIST LWC competition [8]. It was also announced by the committee of the Competition for Authenticated Encryption: Security, Applicability and Robustness (CAESAR) as the first choice for the lightweight use case in February 2019 [5]. Ascon claims a balanced design suitable for lightweight implementation, on both hardware and software, and efficient implementation of countermeasures against SCA. In this paper, we analyze Ascon against two types of active and passive SCA attacks.

The runtime signatures of a hardware platform executing a cryptographic algorithm leak information about the processed data, exploited in an SCA attack to recover the secret key. The most common signatures of a device used in different SCA attacks include electromagnetic (EM) emanations from the interconnects of circuits [10], [18], and power consumption of devices [13], [20], [9], [3], [19]. In these SCA techniques, the attacker does not interfere with the execution of the device and infers secret information by *passively* observing the device behavior. In contrast, fault injection analysis (FIA) is an *active* SCA, in which the attacker observes the response of the device to a fault injection during the execution of the algorithm [22], [35], [31], [6].

Power analysis (PA) is the most common passive SCA attack in which the power consumption of a device executing the algorithm is analyzed to find the secret data. Classical PA techniques, such as differential power analysis (DPA) and correlation power analysis (CPA) [33], [3], require a leakage model for a successful attack. The leakage model is a mathematical function that describes the relationship between the measurements, i.e., power traces in PA, and the secret data processed by the device.

The leakage model is the characteristic of the hardware implementation and the particular cipher operation under attack. Classical PA techniques, such as differential power analysis (DPA) and correlation power analysis (CPA) [33], [3], require prior knowledge of the leakage model for a successful attack. To address the uncertainty in leakage model, profiling SCA techniques estimate the proper model from a set of training data, collected from a reference hardware platform identical to the device under attack [4], [1], [11]. In this paper, we demonstrate an SCA attack based on deep learning, called SCA with reinforcement learning

(SCARL), that requires no prior knowledge of the leakage model and a much smaller amount of power traces than DPA and CPA attacks, to find the secret key of Ascon.

Fault injection analysis is an active SCA technique, in which an attacker induces a fault in an intermediate variable, i.e., the result of an internal computation, of a cipher by applying an external stimulation on the hardware during runtime, such as a voltage/clock glitch [35], [32], or electromagnetic radiation [21]. In a well-designed cryptographic algorithm, the intermediate variables assume maximum entropy. However, as a result of fault injection, specific features appear in the distribution of sensitive variables under attack that reduce the entropy. The reduced entropy of a variable under fault injection is equivalent to the leakage of secret data in a PA attack.

In this paper, we analyze the vulnerability of the Ascon authenticated cipher to passive and active SCA attacks at two abstraction levels: algorithmic and implementation. We use a lightweight implementation of Ascon on an Artix-7 FPGA and deploy a PA attack and a voltage glitch attack to recover the entire 128-bit secret key. We demonstrate that, at the algorithmic level, the initialization of the cipher state with the secret key enables an attacker to find the entire key in a PA attack when the nonce values are known. Further, the addition of the secret key at the end of the Finalization stage of the cipher, to generate the authentication tag, makes it vulnerable to biased FIA attacks. At implementation level, we demonstrate that the power consumption of S-box computations have nonzero mutual information with the processed secret data. Additionally, a simple voltage glitch attack induces a large bias at the output of the S-box computations that can be exploited in a biased fault attack.

The rest of the paper is organized as follows: We describe background of the Ascon cipher, side-channel attacks, and related work in Section 2. In Section 3 we describe the attack setup for both attacks. In Section 4 we discuss the key recovery using the biased fault attack, and discuss the key recovery attack using power analysis in Section 5. Results are presented in Section 6, and the paper concludes in Section 7.

2 Background and Related Work

2.1 Description of Ascon

The structure of the authenticated cipher Ascon is based on Sponge construction, shown in Fig.1. The state of the cipher, denoted by S, consists of five 64-bit words x_0, x_1, x_2, x_3 and x_4. The plaintext, as well as the associated data, is divided into blocks of r bits. Every block of data is absorbed into first r bits of the state. These r bits of the state are called the rate bits. The remaining private $c = 320 - r$ bits of the state are capacity bits. The number of capacity bits directly affects the privacy and authentication security bounds of a sponge-based cipher [28]. The corresponding blocks of ciphertext are squeezed out of the sponge construction.

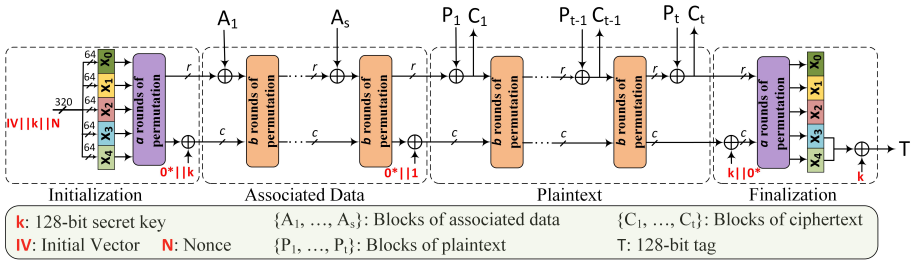

Fig. 1. Sponge-based structure of authenticated encryption in the Ascon cipher [26].

In the "Initialization" stage, the state of Ascon is initialized by the concatenation of an Initial Vector (IV), the secret key (k) and a nonce (N), i.e., $S = IV||k||N$. The nonce adds some level of protection to the cipher against differential cryptanalysis techniques. Nonce-based ciphers provide different outputs for the same message encrypted multiple times. However, differential techniques, such as DFA, require the same key and initial state for a single message encrypted more than once. The power consumption of the hardware during processing the state initialized with the secret key is the target of our PA attack.

After initializing the state of Ascon, "a" rounds of a permutation function are performed on the state. At the end of the Initialization stage, the 128-bit secret key is XORed with the last 128 capacity bits of the state. In the "Associated Data" stage, the r-bit blocks of associated data are absorbed into the sponge by XORing them with the rate bits. After absorption of each block of data, "b" rounds of permutation are performed. In the "Plaintext" stage, the blocks of plaintext are absorbed and the ciphertext blocks are squeezed out.

By processing all plaintext blocks, Ascon proceeds to the "Finalization" stage to generate the tag for authentication. At the beginning of the Finalization stage, the 128-bit secret key is XORed with the first 128 capacity bits of the state. Next, "a" rounds of permutation are performed. Finally, the last 128 capacity bits are XORed with the secret key to generate the tag. In our proposed fault attack, the last permutation round of the Finalization stage is the target of fault injection, and the tag values are analyzed to retrieve the secret key.

The permutation function of Ascon consists of Addition of Constants, Substitution Layer (S-box) and Diffusion Layer. The S-box is a 5-bit nonlinear function. The inputs to the S-box are the bits of five state words x_0 to x_4, one bit from each word. Conceptually, there are 64 S-boxes corresponding to the 64 bits of the state words. The updated state at the output of the S-box operations is processed by the linear diffusion layer. The linear diffusion function operates on each word $x_i, i = 0, \cdots, 4$ separately.

There are five different diffusion functions for every word of the state, which mixes bits within the words. By representing each word x_i as a vector of 64 bits, the diffusion functions can be represented in the matrix form as

$$\boldsymbol{\Sigma}_i(\boldsymbol{x}_i) = (\mathrm{L}_i \boldsymbol{x}_i) \bmod 2, \quad i = 0, 1, ..., 4 \tag{1}$$

In this relation, $\boldsymbol{x}_i$ is the vector representation of a state word, and L_i is the corresponding linear mapping matrix of dimension 64×64. The matrix multiplication is computed modulo-2. The first row of matrices L_i has nonzero elements at locations which represent the rotational shift values of the corresponding diffusion function. As an example for the state word $\boldsymbol{x}_0$, the first row of matrix L_0 is a vector with elements 0, 19 and 28 equal to 1, while the rest are 0. The next rows of the matrix are obtained by rotating the previous row to the right by one element. In our fault attack, we will use the matrix representation of the diffusion layer.

2.2 Power Analysis

In a typical power side-channel analysis, the power consumption corresponding to a key-dependent operation is used to recover the secret key. Assume the cipher operation $F_k() : \mathbb{F}_2^n \to \mathbb{F}_2^m$ maps known input data $Z \in \mathbb{F}_2^n$ to an unknown intermediate variable $X \in \mathbb{F}_2^m$, called the sensitive or secret data, in which k is a subset of the secret key. The function $F_k()$ usually involves a non-linear, or *confusion*, operation in a cryptographic algorithm for a successful key recovery. In most block ciphers, including Ascon, the confusion operation is the so-called substitution layer, or S-box. One class of PA techniques uses the power consumption during the computation of $F_k()$ to find the secret data X. Other techniques inspect the power consumption when the variable X is loaded into a memory element in the hardware, as exploited in [30] to find the secret key of Ascon in a DPA attack.

The binary representation of the secret data is not convenient for power analysis, since, the power consumption is not necessarily correlated with the individual bits of the data. The numerical normal form [34] is a generic representation of a Boolean variable that captures the bit dependencies of the data. In this form, the m-bit Boolean variable $\bar{\mathbf{x}} = (x_i)_{i=0,1,\cdots,m}$ is represented by the monomials $X^U = \prod_{i=1}^{m} x_i^{u_i}$ of degree $d = HW(U)$, in which, $U \in \mathbb{F}_2^m \backslash \{\mathbf{0}\}$ and $HW(U)$ is the Hamming weight of U. To define the mutual information between the power traces and the secret data, the leakage function $L() : \mathbb{F}_2^m \to \mathbb{R}$ is introduced to map the Boolean space of the data to the real-valued space of power measurements. A generic leakage model thus can be defined as

$$L(X) = \alpha_0 + \sum_{U \in \mathbb{F}_2^m \backslash \{0\}} \alpha_U X^U + \epsilon \tag{2}$$

in which $\alpha_U \in \mathbb{R}, U \in \mathbb{F}_2^m$ are real-valued coefficients of the model, and ϵ is a noise term.

In a PA attack, the goal is to find the mutual information between the power measurements and the leakage of the sensitive variable X^*, calculated with a key candidate k^*. The correct key exhibits the highest information. The mutual information is a mathematical definition for the amount of information content of measurements about the secret data. However, calculating mutual information empirically might result in poor accuracy. As a result, existing SCA techniques

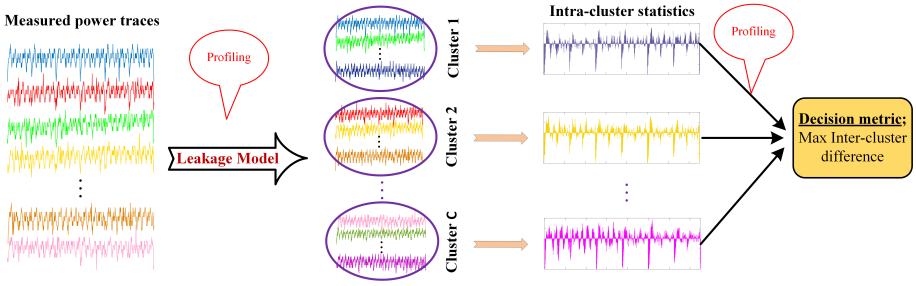

Fig. 2. Conceptual description of model-based power analysis techniques using a leakage model.

employ alternative statistics to rank the key candidates based on the information content of the measurements.

A wide range of SCA techniques use clustering to indirectly evaluate the mutual information between measurements and secret data; the power measurements are divided into two or more clusters according to the leakage model of 2. The inter-cluster separation provides a statistical metric for information content of the measurements. A conceptual representation of the model-based PA is shown in Fig. 2. A classical, yet powerful, clustering model-based technique is differential power analysis (DPA) in which the difference between the mean of power traces in two clusters is used as the statistical relation to rank the key candidates [13], [14]. The correct key exhibits the largest inter-cluster difference.

2.3 Fault Injection Analysis

The observations of a fault attack are the result of computations after a fault is injected into a sensitive variable. Assume the key-dependent cipher operation $Z = F_k(X;Y)$ maps an unknown sensitive variable $X \in \mathbb{F}_2^m$ to a known output $Z \in \mathbb{F}_2^n$. The variable Y is a subset of the secret state disjoint from the sensitive variable X and k is a subset of the secret key. In most FIA techniques, the operation $F_k()$ involves a diffusion function that mixes the bits of X and Y with a linear operation. When k is known the value of X can be calculated uniquely from the output. An exception is the fault sensitivity analysis (FSA) which exploits the correlation between the secret data and fault sensitivity defined as the intensity at which errors start to appear at the output [16].

Differential fault analysis (DFA) is a class of FIA techniques in which fault-free and faulty outputs of the cipher for the same plaintext and initial state are used to calculate the error in an intermediate variable. Certain properties of the error are exploited to identify the correct key among all key candidates [15], [17]. While DFA recovers the secret key with a small number of fault injections, as for example in [23], it assumes a strong fault model; fault manifestation must be precise and both faulty and fault-free outputs are required. Statistical DFA techniques, such as differential fault intensity analysis (DFIA) relax the adver-

sary model as they can tolerate imprecise fault injections, however, multiple encryptions with the same input and initial conditions are still required.

Biased fault analysis techniques relax the assumptions of fault model, however, more fault injections are required to recover the secret key. In biased techniques, the deviation of the distribution of an intermediate variable under fault injection from the maximum entropy distribution (which is the uniform distribution for Boolean variables) is the leakage of secret data. The deviation is often referred to as fault bias. In biased techniques, running multiple encryptions with the same input and initial conditions, necessary to calculate the error, might not be required. Further, noisy fault inductions, in which timing or location of the fault are not precise, can be tolerated.

Statistical ineffective fault analysis (SIFA) is a biased technique that exploits the bias of correct values of an intermediate variable under fault injection [7]. Hence, SIFA only requires the correct output of the cipher which is a great advantage in the presence of countermeasures detecting errors. Further, the authors in [25] extend SIFA by introducing fault intensity map analysis (FIMA), which combines observables of fault bias, and the variation in fault distribution with fault intensity, to recover the secret key of a cipher with improved efficiency. Deep learning has also been augmented with FIMA in [24] resulting in a highly efficient attack called FIMA with neural network key distinguisher (FIMA-NN).

A subset fault analysis (SSFA) attack on Ascon is introduced in [12] that reduces the search space of the secret key to 64 bits. The major vulnerability of the Ascon S-box exploited in this attack is the fact that by setting the third bit of the S-box input to 0, the XOR of the last two output bits is biased. A SIFA attack on Ascon is also demonstrated in [26]. The advantage of this attack is the fact that SIFA is successful even in the presence of countermeasures that detect errors in intermediate variables. While the foregoing works focus on algorithmic level vulnerability of Ascon, in this paper, we demonstrate a practical voltage glitch attack on an FPGA implementation of Ascon, based on the SIFA attack of [26].

3 Attack Setup

The setup for implementing the power analysis and fault attack with voltage glitch is shown in Fig. 3 which uses the Flexible Open-source workBench fOr Side-channel analysis (FOBOS) [2]. Our FOBOS instance includes a NewAE CW305 Artix-7 FPGA board that executes the Ascon cipher. The target of the attack is the FPGA chip on this board. A Digilent Nexys A7 is also used as a control board that synchronizes the target board under attack and a host PC. The control board receives the required data, including the input of encryption/decryption algorithms and configuration parameters for the target device, from the host PC. The control board provides the target board with the input data, control and clock signals. It also receives the results of computations and sends them back to the host PC.

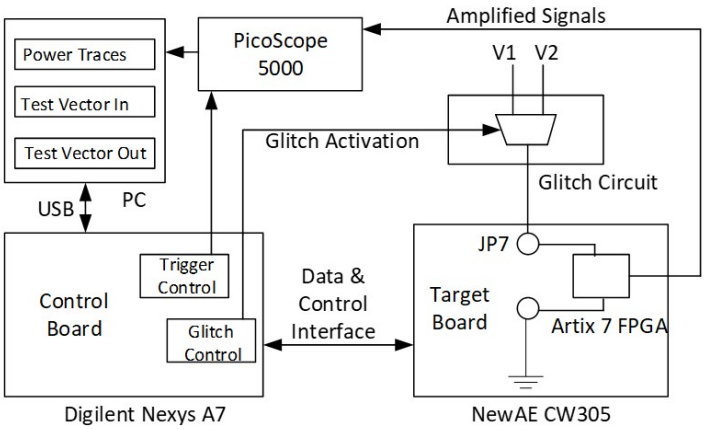

Fig. 3. Fault-enabled Flexible Open-source workBench fOr Side-channel analysis (FOBOS) setup for deploying power analysis and voltage glitch fault attacks on Ascon.

Our FOBOS setup uses a PicoScope 5000 with 20 dB amplification for measuring the power consumption of the FPGA chip on the board under attack during execution of Ascon. The scope is also synchronized with the clock signal provided by the control board, and collects 125 samples of power consumption per clock cycle. We measure the power traces during the S-box computations of Ascon at the beginning of Initialization stage.

To deploy a voltage glitch attack on Ascon during execution, we use a single-pole double-throw (SPDT) analog switch to control the supply of the operating device. The common pin of the switch is connected to the supply voltage of the Artix-7 FPGA on the target board. The two input pins of the switch are connected to an operating voltage source (V1) and a glitch source (V2). When the select pin of the switch is logic-0, the common pin is normally connected to the V1 input. With a logic-1 at the select pin, the V2 input is connected to the common pin, thus, the supply voltage of the target FPGA.

The control signal at the select pin of the analog switch is provided by the control board. The configuration parameters of the glitch generator include the starting time and the duration of the voltage glitch, both in terms of clock cycles. These parameters are set by the user on the host PC. Accordingly, the control board generates the proper pulse for the select pin. Ideally, we desire to apply the voltage glitch over one clock cycle during which an S-box operation is executed. However, the voltage regulators on the target FPGA board filter out short voltage glitches. Hence, to apply an effective fault injection, we are required to start the glitch at earlier time and maintain the underfed voltage over multiple cycles.

In our experiments, we set the operating conditions of the target FPGA, executing Ascon, at a supply voltage of 0.8V (V1) and a clock frequency of 10MHz. We observed that a glitch voltage of 0.51V (V2) results in an effective biased fault injection. In order to attack S-box i, executed at clock cycle C_i, referenced

to the beginning of the cipher execution, we need to switch the supply voltage to the the glitch value of 0.51V at 21 cycles earlier ($C_i - 21$), and switch back the supply to the operating value of 0.8V at cycle C_i. Under these conditions, the S-box i experiences the fault injection with a biased distribution at the output.

We implemented the Ascon authenticated cipher on the Artix-7 FPGA on the target board; the implementation is available for inspection at [29]. The implementation is lightweight as it includes only one instantiation of the S-box hardware shared by all 64 S-box operations in Ascon. The S-box hardware is a bit-sliced implementation. After initializing the 320-bit state, the S-box operations in a round of permutation are conducted during the next 64 clock cycles, with one operation at every cycle. This implementation is consistent with the flow of operations shown in Fig. 4.

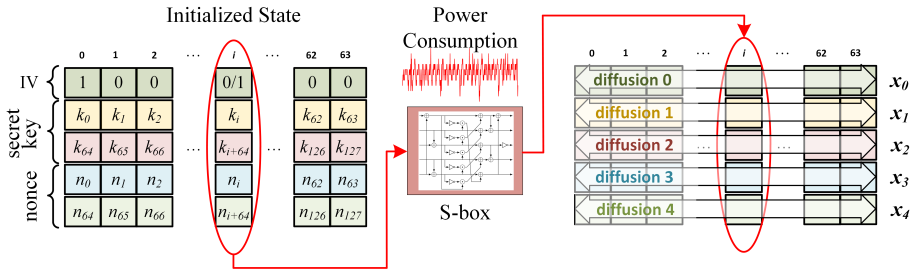

Fig. 4. S-box operation and diffusion over the 320-bit state initialized with nonce, secret key and IV, at the beginning of Initialization stage of Ascon. The power consumption during S-box computations is used in the SCA attack.

4 Key Recovery with Fault Attack

In the proposed attack methodology, faults are injected into the operation of two selected S-boxes, during every encryption. The permutation operation of the last round of the Finalization stage in Ascon is shown schematically in Fig. 5. After constant addition, S-boxes operate on the state, and diffusion functions mix the bits within every 64-bit word x_i. The correct values of the tags, under ineffective faults, are collected for analysis. Correct tags can be identified by two methods: 1) An authenticated decryption in which no plaintext is released due to a failed authentication denotes an incorrect tag, or 2) We can compare tag values with and without fault injections.

The data at the output of the S-box pairs under attack are calculated from tag values using the inverse operation of the diffusion functions under each key hypothesis. Using the matrix representation of (1), the inverse diffusion mappings are simply the inverse matrices L_i^{-1}. Considering $\boldsymbol{x}_i$ and $\boldsymbol{\Sigma}_i$ as the input and output words of the diffusion layer, the inverse diffusion can be represented as

$$\boldsymbol{x}_i = (\mathrm{L}_i^{-1}\boldsymbol{\Sigma}_i) \bmod 2, \quad i = 0, 1, ..., 4 \tag{3}$$

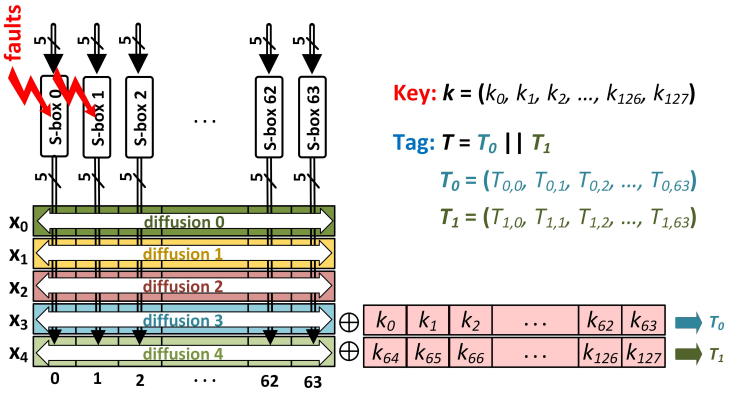

Fig. 5. The location of fault injection into a pair of S-box computations at the last round of Finalization stage in Ascon [26].

We represent an inverse diffusion matrix in terms of its rows as

$$\mathrm{L}_i^{-1} = \left[\boldsymbol{l}_0^{(i)T}, \boldsymbol{l}_1^{(i)T}, \cdots, \boldsymbol{l}_{63}^{(i)T}\right]^T, \quad i = 0, 1, \cdots, 4 \tag{4}$$

In this relation, $\boldsymbol{l}_j^{(i)T}$ is the j-th row of the inverse diffusion matrix corresponding to the state word $\boldsymbol{x}_i$. The superscript T represents matrix transpose.

In our proposed method, the 128-bit key is divided into words of length w bits, where we assume that w is a power of 2. Bits 3 and 4 at the output of S-box j can be calculated with a linear combination of bits within the key words. The coefficients of the combination are determined by the j-th row of the inverse diffusion matrices L_3^{-1} and L_4^{-1}. This can be shown by rewriting the equations for bits 3 and 4 of the j-th S-box as

$$\begin{aligned} s_3^{(j)} &= \sum_{s=0}^{64/w-1} \left(\sum_{r=s\cdot w}^{(s+1)\cdot w-1} T_{0,r} \quad l_{j,r}^{(3)} \right) \oplus K_s^{(j)} \\ s_4^{(j)} &= \sum_{s=0}^{64/w-1} \left(\sum_{r=s\cdot w}^{(s+1)\cdot w-1} T_{1,r} \quad l_{j,r}^{(4)} \right) \oplus K_{s+64/w}^{(j)} \end{aligned} \tag{5}$$

In these equations, the key bit combinations for $s = 0, 1, \cdots, 128/w - 1$ are defined as

$$K_s^{(j)} = \sum_{r=s\cdot w}^{(s+1)\cdot w-1} k_r \quad l_{j,r} \tag{6}$$

We used the concatenation $\boldsymbol{l}_j = (\boldsymbol{l}_j^{(3)}, \boldsymbol{l}_j^{(4)})$ to simplify the notation. In order to calculate bits 3 and 4 at the output of S-box j, we only need the key bit combinations instead of the individual 68 bits of the key used in the calculations

of (5). Using vector representation, a key hypothesis for S-box j will be

$$\mathbf{K}^{(j)} = \left(K_0^{(j)}, K_1^{(j)}, \cdots, K_{128/w-1}^{(j)}\right) \tag{7}$$

Consider an example in which double fault injection targets a pair of S-boxes 0 and 1. Hence, the key hypothesis is the vector $(\mathbf{K}^{(0)}, \mathbf{K}^{(1)})$. Using the fault analysis, the values of $K_s^{(0)}$ and $K_s^{(1)}$ for $s = 0, 1, \cdots, 128/w - 1$ are estimated. Every $K_s^{(j)}$ is a linear combination of w bits of the key within the word s. When the fault injection targets different pairs of S-boxes, different combinations of the w bits are obtained resulting in a set of w binary equations for the w key bits in a word. Then the w bits of each key word s are calculated by solving a set of linear equations corresponding to that word. The details of key recovery algorithm are explained in [26].

5 Key Recovery with Power Analysis

In the context of the attack model described in Section 2.2, the 5-bit S-box operations of Ascon, at the beginning of the Initialization stage, are the targets of our PA attack. At the beginning of Initialization, the most significant bit (MSB) at the input of the i-th S-box is the i-th bit of the IV which is known. The next two bits of the input are bits k_i and k_{i+64} of the secret key, respectively. The two least significant bits (LSB) of the input are bits n_i and n_{i+64} of the *public message number*, or *nonce*. We measure the power consumption of the S-box computation for multiple values of nonce. The known input data to the S-box i is a 2-bit variable $Z_i \in \mathbb{F}_2^2 = (n_i, n_{i+64})$ corresponding to two bits of the nonce. The secret data $X_i \in \mathbb{F}_2^5$ is the 5-bit variable at the output of the S-box. The subset of the key that is estimated using the power consumption of S-box i is $k = (k_i, k_{i+64})$.

5.1 Classical DPA and CPA Attacks

We conduct the classical DPA and CPA attacks with two commonly used leakage models, i.e., Hamming weight (Hw) and most significant bit (MSB). In the Hw model, the leakage of secret data in (**??**) is $L^{Hw}(X) = \sum_i x_i$, in which $X = (x_0, x_1, \cdots, x_4)$ are the 5-bit values at the output of the S-boxes under attack. Using the MSB model, the leakage is simply $L^{MSB} = x_0$. In a DPA attack, the power traces are grouped into two clusters based on the leakage.

We collected power measurements of the FPGA for 40K encryptions with random nonce values. Let $\mathbf{T}_j$ denote the power measurements during encryption with the corresponding nonce value $n_j = (n_{j,0}, n_{j,1}, n_{j,2}, \cdots, n_{j,127})$ for $j = 1, 2, \cdots, 40K$. For the DPA attack, we separated the power traces of S-boxes 0 and 1. Hence, a total of 80K power traces $\hat{\mathbf{T}}_j, j = 1, 2, \cdots, 80K$ are available, in which the first 40K traces correspond to S-box 0 and the rest to S-box 1. The corresponding input data are denoted by $Z_j = (n_{j,0}, n_{j,64}), j = 1, 2, \cdots, 40K$ and $Z_j = (n_{j,1}, n_{j,65}), j = 40K + 1, 40K + 2, \cdots, 80K$. The subset of the secret

key (key candidate) that is estimated using this attack is $k = (k_0, k_{64}, k_1, k_{64})$, with 16 possible values.

For every value of the key candidate $k = k^*$, we calculate the output of the S-boxes. The output values $X_j^*, j = 1, 2, \cdots, 80K$ are calculated with k^* and the corresponding Z_j. The values of data leakage are then calculated as $L^{\mathcal{G}}(X_j^*)$, in which $\mathcal{G}$ is either Hw or MSB. Let μ_L^* denote the mean of the calculated leakage values. Next, we assign power traces with $L^{\mathcal{G}}(X_j^*) < \mu_L^*$ to cluster C_0^* and traces with $L^{\mathcal{G}}(X_j^*) > \mu_L^*$ to cluster C_1^*. Finally, we calculate the difference between the mean traces in the two clusters. The sample with the highest difference is the rank of key candidate k^*.

The CPA attack is similar to DPA but with different rank statistics. Given the power traces $\hat{\mathbf{T}}_j, j = 1, 2, \cdots, 80K$ and the calculated leakage values $L^{\mathcal{G}}(X_j^*) > \mu_L^*$, as described above, the Pearson's correlation coefficient between the samples of the power traces and the leakage values is used to rank the key candidate k^*. The sample with the highest absolute value of the correlation coefficient determines the rank of the key.

5.2 Deep Learning Power Analysis

We use SCA with reinforcement learning (SCARL) to find the secret key of the Ascon authenticated cipher using the power consumption during S-box computations at the beginning of Initialization stage. Using SCARL, an attacker does not require prior knowledge of the leakage model. SCARL estimates the proper model from the measurements. In the first step, the raw power measurements are mapped into an intermediate representation using an LSTM autoencoder. Next, a leakage model is estimated using the autoencoder representation. Finally, the secret key can be found using the estimated model and a clustering technique, similar to a DPA attack. The details of the SCARL algorithm are further explained in [27].

Assume $\{\tilde{\mathbf{f}}_j\}_{j=1}^{80K}$ are a set of features that the autoencoder extracts from the corresponding raw power traces $\mathbf{T}_j$. In SCARL, a reinforcement learning algorithm assigns the features into two clusters C_0 and C_1 under two conditions: 1) the features are assigned evenly to the clusters such that the cardinality of the clusters are as close as possible, and 2) the difference between the average of the features in two clusters, i.e., inter-cluster difference, is maximum.

We label the clusters by $l_j = \{0, 1\}$. After clustering by the reinforcement learning algorithm, The coefficients of the generic leakage model (2) are estimated for the key candidate k^* using MSE; i.e.,

$$\alpha_U^* = \min_{\alpha_U} \mathrm{E}_j \left[|L(X_j^*) - l_j|^2 \right],\ U \in \mathbb{F}_2^m \tag{8}$$

For an m-bit secret data X, the highest order of the leakage model is m. For the output of the Ascon S-box $m = 5$. In SCARL, it is assumed that the components of the leakage model with order $d = HW(U) \leq m_0$ with $m_0 < m$ have the largest contribution to the leakage. Hence, we set $\alpha_U^* = 0, HW(U) > m_0$

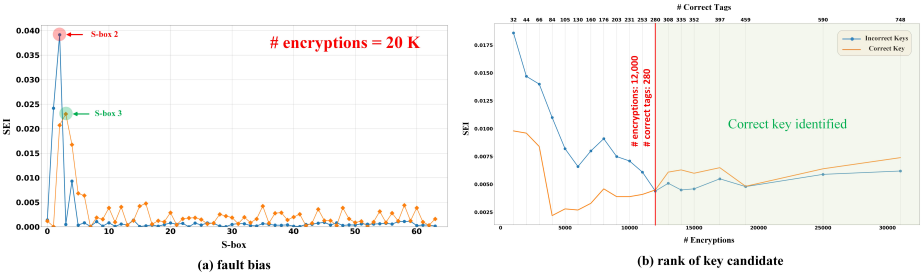

Fig. 6. Results of voltage glitch fault attack on the S-box operations of Ascon; (a) bias of correct values under fault injection; (b) rank of key candidates versus number of fault injections.

and find the low order leakage as

$$l_j^* = \alpha_0^* + \sum_{U \in \mathbb{F}_2^m \setminus \{0\}, HW(U) \leq m_0} \alpha_U^* (X_j^*)^U \tag{9}$$

Now, we recluster the power features based on this low order model. We calculate the mean of the leakage as $\mu_l^* = \mathrm{E}_j[l_j^*]$, and define the clusters $C_0^* = \{\mathbf{c}_j | l_j^* > \mu_l^*\}$ and $C_0^* = \{\mathbf{c}_j | l_j^* < \mu_l^*\}$. The rank of key candidate k^* is thus

$$\mathcal{R}(k^*) = \max \ \mathrm{E}_{\mathbf{c}_j \in C_0^*}[\mathbf{c}_j] - \mathrm{E}_{\mathbf{c}_j \in C_1^*}[\mathbf{c}_j] \tag{10}$$

6 Experimental Results

6.1 Fault Attack

The results of the voltage fault injection at two consecutive S-box operations are shown in Fig. 6. In part (a) of the figure, the bias of two least significant bits (LSB) of all 64 S-boxes, using only correct values, is plotted. In this figure the bias of data at the output of S-boxes is shown for two fault injection experiments. In one experiment, S-box 2 is the target of fault injection, while in the second one, the target of the attack is S-box 3.

The results of Fig. 6 reveals a significant vulnerability of the Ascon S-box (with bit-sliced implementation) to biased fault attacks. As explained in Section 4, addition of the secret key at the end of Finalization stage for tag generation, is the vulnerability of the Ascon cipher to biased fault attacks. However, we note that an attacker can observe only the two LSB bits of S-box outputs given the tag values. Hence, for a successful attack, the distribution of the two LSB bits of the S-box must be biased. As observed in part (a) of Fig. 6, the bias of last two LSB at the output of the S-boxes under attack is significantly larger than other S-boxes. This observation demonstrates the effectiveness of voltage glitch on the FPGA implementation, and the vulnerability of Ascon S-box to biased fault attacks.

The rank of key candidates versus the number ineffective fault injections is shown in Fig. 6 (b). The key candidates correspond to S-boxes 2 and 3. It is observed that using only 280 ineffective fault injections, the correct key is identified. However, to collect the 280 correct tags, 12K faulted encryptions are required. This demonstrates that the rate of ineffective faults is only 2.3%. This low rate of ineffective faults results in a large bias of correct values shown in Fig. 6 (a).

6.2 Power Analysis Attack

The result of the DPA attack using Hw and MSB leakage models is shown in Fig. 7 (a). It is observed that with the Hw leakage model there are 7 incorrect key candidates with a higher rank than the correct key. However, with the MSB model, the rank of the correct key is among the top 4 key candidates. We note that rank of the correct key and the key candidate 8 are equal in this case. These results imply that the power consumption of S-boxes in Ascon are more correlated with the MSB of the output values.

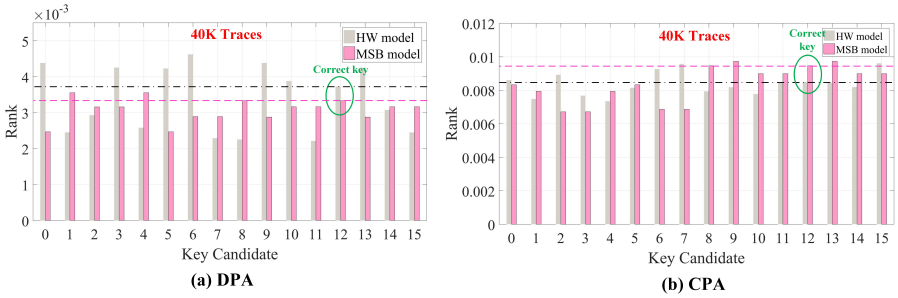

Fig. 7. Rank of key candidates at the input of S-boxes 0 and 1 in a power analysis attack with Hw and MSB leakage models; (a) DPA attack, (b) CPA attack.

The CPA attack also shows similar performance to the DPA attack. The rank of key candidates with 40K power traces in the CPA attack are shown in part (b) of Fig. 7. Similar to DPA, the MSB leakage model shows improved performance. However, there are still two incorrect key candidates with higher rank than the correct key with 40K power measurements. Again, the incorrect key 8 has the same rank as the correct key.

The rank of key candidates versus the number of power measurements (encryptions) in a SCARL attack is shown in Fig. 8 (b). It is observed that if at least 24K measurements are collected, SCARL is able to identify the correct key. In this figure, the rank of key candidate corresponding to 4 bits of the secret key at the input of S-boxes 0 and 1 is plotted. This is in contrast to the results of DPA and CPA attacks, in Fig. 7, which fail to identify the correct key with 40K measurements.

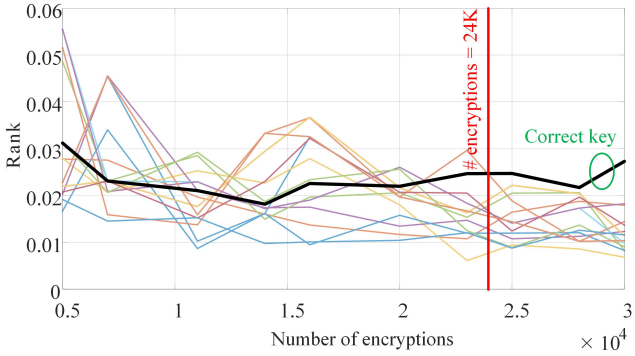

Fig. 8. Rank of key candidates at the input of S-boxes 0 and 1 in SCARL attack versus the number of collected power traces (during multiple encryptions).

7 Conclusions

We demonstrated the vulnerability of the Ascon authenticated cipher, as a candidate in the second round of the NIST LWC competition, to power analysis and fault attacks. The addition of the secret key at the end of the Finalization stage of the cipher, for generating the authentication tag, enables an attacker to find the secret key with a biased fault attack. Using a voltage glitch on an FPGA implementation of Ascon, we successfully recovered one subset of the secret key (equivalent to two bits) with 280 output correct values of tag under fault injection into a pair of S-box computations. Further, we demonstrated that the initialization of the cipher state with the secret key is a vulnerability to power analysis (PA) attacks. While classical PA attacks, such as DPA and CPA, failed to find the correct key with more than 40K power traces, we employed a deep learning technique to recover the secret key with only 24K traces.

References

1. Bartkewitz, T., Lemke-Rust, K.: Efficient template attacks based on probabilistic multi-class support vector machines. In: International Conference on Smart Card Research and Advanced Applications. pp. 263–276. Springer (2012)
2. CERG: Flexible Open-source workBench fOr Side-channel analysis (FOBOS) (October 2016), https://cryptography.gmu.edu/fobos/, https://cryptography.gmu.edu/fobos/
3. Chakraborty, A., Mondal, A., Srivastava, A.: Correlation Power Analysis Attack against STT-MRAM Based Cyptosystems. IACR Cryptology ePrint Archive **2017**, 413 (2017)
4. Chari, S., Rao, J.R., Rohatgi, P.: Template attacks. In: International Workshop on Cryptographic Hardware and Embedded Systems. pp. 13–28. Springer (2002)
5. D. J. Bernstein: Cryptographic competitions (2016), https://competitions.cr.yp.to/caesar.html

6. Dobraunig, C., Eichlseder, M., Gross, H., Mangard, S., Mendel, F., Primas, R.: Statistical ineffective fault attacks on masked aes with fault countermeasures. In: International Conference on the Theory and Application of Cryptology and Information Security. pp. 315–342. Springer (2018)
7. Dobraunig, C., Eichlseder, M., Korak, T., Mangard, S., Mendel, F., Primas, R.: SIFA: Exploiting Ineffective Fault Inductions on Symmetric Cryptography. IACR Transactions on Cryptographic Hardware and Embedded Systems pp. 547–572 (2018)
8. Dobraunig, C., Eichlseder, M., Mendel, F., Schläffer, M.: Ascon v1.2. Submission to the NIST LWC Competition (2019)
9. Fabšič, T., Gallo, O., Hromada, V.: Simple power analysis attack on the QC-LDPC McEliece cryptosystem. Tatra Mountains Mathematical Publications **67**(1), 85–92 (2016)
10. Gandolfi, K., Mourtel, C., Olivier, F.: Electromagnetic analysis: Concrete results. In: International workshop on cryptographic hardware and embedded systems. pp. 251–261. Springer (2001)
11. Jap, D., Stöttinger, M., Bhasin, S.: Support vector regression: exploiting machine learning techniques for leakage modeling. In: Proceedings of the Fourth Workshop on Hardware and Architectural Support for Security and Privacy. p. 2. ACM (2015)
12. Joshi, P., Mazumdar, B.: A Sub-Set Fault Analysis Attack on ASCON. Cryptology ePrint Archive, Report 2019/1370 (2018), https://eprint.iacr.org/2019/1370
13. Kocher, P., Jaffe, J., Jun, B.: Differential power analysis. In: Annual International Cryptology Conference. pp. 388–397. Springer (1999)
14. Le, T.H., Clédière, J., Canovas, C., Robisson, B., Servière, C., Lacoume, J.L.: A proposition for correlation power analysis enhancement. In: International Workshop on Cryptographic Hardware and Embedded Systems. pp. 174–186. Springer (2006)
15. Li, W., Zhang, W., Gu, D., Cao, Y., Tao, Z., Zhou, Z., Liu, Y., Liu, Z.: Impossible differential fault analysis on the LED lightweight cryptosystem in the vehicular ad-hoc networks. IEEE Transactions on Dependable and Secure Computing **13**(1), 84–92 (2016)
16. Li, Y., Sakiyama, K., Gomisawa, S., Fukunaga, T., Takahashi, J., Ohta, K.: Fault Sensitivity Analysis. In: International Workshop on Cryptographic Hardware and Embedded Systems. pp. 320–334. Springer (2010)
17. Liu, Y., Cui, X., Cao, J., Zhang, X.: A hybrid fault model for differential fault attack on AES. In: 2017 IEEE 12th International Conference on ASIC (ASICON). pp. 784–787. IEEE (2017)
18. Longo, J., De Mulder, E., Page, D., Tunstall, M.: SoC it to EM: electromagnetic side-channel attacks on a complex system-on-chip. In: International Workshop on Cryptographic Hardware and Embedded Systems. pp. 620–640. Springer (2015)
19. Luo, C., Fei, Y., Zhang, L., Ding, A.A., Luo, P., Mukherjee, S., Kaeli, D.: Power analysis attack of an AES GPU implementation. Journal of Hardware and Systems Security **2**(1), 69–82 (2018)
20. Mahanta, H.J., Azad, A.K., Khan, A.K.: Power analysis attack: A vulnerability to smart card security. In: 2015 International Conference on Signal Processing and Communication Engineering Systems. pp. 506–510. IEEE (2015)
21. Ordas, S., Guillaume-Sage, L., Maurine, P.: Electromagnetic fault injection: the curse of flip-flops. Journal of Cryptographic Engineering **7**(3), 183–197 (2017)
22. Patranabis, S., Chakraborty, A., Nguyen, P.H., Mukhopadhyay, D.: A biased fault attack on the time redundancy countermeasure for AES. In: International workshop

on constructive side-channel analysis and secure design. pp. 189–203. Springer (2015)
23. Piret, G., Quisquater, J.J.: A differential fault attack technique against spn structures, with application to the aes and khazad. In: International workshop on cryptographic hardware and embedded systems. pp. 77–88. Springer (2003)
24. Ramezanpour, K., Ampadu, P., Diehl, W.: Fault Intensity Map Analysis with Neural Network Key Distinguisher. In: Proceedings of the 3rd ACM Workshop on Attacks and Solutions in Hardware Security Workshop. pp. 33–42 (2019)
25. Ramezanpour, K., Ampadu, P., Diehl, W.: FIMA: Fault Intensity Map Analysis. In: Constructive Side-Channel Analysis and Secure Design. pp. 63–79. Springer (2019)
26. Ramezanpour, K., Ampadu, P., Diehl, W.: A statistical fault analysis methodology for the ascon authenticated cipher. In: 2019 IEEE International Symposium on Hardware Oriented Security and Trust (HOST). pp. 41–50. IEEE (2019)
27. Ramezanpour, K., Ampadu, P., Diehl, W.: SCARL: Side-Channel Analysis with Reinforcement Learning on the Ascon Authenticated Cipher. arXiv preprint arXiv:2006.03995 (2020)
28. Saarinen, M.J.O.: Beyond modes: Building a secure record protocol from a cryptographic sponge permutation. In: Cryptographers' Track at the RSA Conference. pp. 270–285. Springer (2014)
29. SAL: Lightweight Implementation of Ascon Authenticated Cipher (October 2020), https://github.com/vtsal/ascon_lwc_aead/
30. Samwel, N., Daemen, J.: Dpa on hardware implementations of ascon and keyak. In: Proceedings of the Computing Frontiers Conference. pp. 415–424 (2017)
31. Schellenberg, F., Finkeldey, M., Gerhardt, N., Hofmann, M., Moradi, A., Paar, C.: Large laser spots and fault sensitivity analysis. In: 2016 IEEE International Symposium on Hardware Oriented Security and Trust (HOST). pp. 203–208. IEEE (2016)
32. Singh, A., Kar, M., Chawla, N., Mukhopadhyay, S.: Mitigating power supply glitch based fault attacks with fast all-digital clock modulation circuit. In: 2019 Design, Automation & Test in Europe Conference & Exhibition (DATE). pp. 19–24. IEEE (2019)
33. Sugawara, T., Homma, N., Aoki, T., Satoh, A.: Differential power analysis of aes asic implementations with various s-box circuits. In: 2009 European Conference on Circuit Theory and Design. pp. 395–398. IEEE (2009)
34. Whitnall, C., Oswald, E., Standaert, F.X.: The myth of generic DPA... and the magic of learning. In: Cryptographers' Track at the RSA Conference. pp. 183–205. Springer (2014)
35. Yuce, B., Ghalaty, N.F., Santapuri, H., Deshpande, C., Patrick, C., Schaumont, P.: Software fault resistance is futile: Effective single-glitch attacks. In: 2016 Workshop on Fault Diagnosis and Tolerance in Cryptography (FDTC). pp. 47–58. IEEE (2016)