Overview of Round One Power Attacks on SHA-3 Based MAC

Kevin Millar, Chun-Yi Chu, and Marcin Łukowiak

***Abstract*—Digital signatures and message authentication codes are the two most common applications of cryptographic hash functions. Others range from ensuring data integrity to randomization functions and key derivation. Because of the recent break of the SHA-1 hash function, it is expected that in the nearest future there will be an increasing interest in the new SHA-3 algorithm. SHA-3 implements a subset of the Keccak family and was released as the NIST standard in 2015. SHA-3 based MAC is a keyed-hash message authentication function, which can be used to verify both the data integrity of a message and its source. Though Keccak is cryptographically secure, implementations of the algorithm may be susceptible to power analysis attacks if not sufficiently protected. This work implements and analyzes two correlation power analysis (CPA) attacks targeting the round one operations of a SHA-3 based MAC implementation on an FPGA.**

***Index Terms*—SHA-3, power attacks, FPGA.**

I. Introduction

IN February 2017, it was announced that the commonly used SHA-1 hash function had been broken through discovery of the first collision [1]. This was somewhat expected as a theoretical attack on SHA-1 with 2^{63} operations (versus the simple birthday paradox implied bound of 2^{80} operations) was known since 2005. It was this theoretical attack that led NIST to call for the creation of a new SHA-3 hash function which would replace (or complement) the existing family of secure hash standards (SHS). There were 64 candidate algorithms submitted to the NIST competition, and after five years of rigorous scrutiny, Keccak was announced as the winner on October 2, 2012. On August 5 2015, SHA-3 was officially standardized and published as FIPS PUB 202 [2].

Power analysis attacks (PAA) are some of the best known and most powerful side channel attacks (SCA). They exploit the correlation between a computational model and the measured power consumption of the implementation to reveal secret data, e.g. the key used in the computations.

The focus of this research was to demonstrate power analysis attacks using the correlation power analysis (CPA) technique to extract the secret key of a SHA-3 based MAC implementation on an FPGA.

This work implements and analyzes two different CPA attacks targeting the first round output. The first was presented by Luo et al. in [3]. In this attack, inverse round operations are performed to determine the relationship between the round one output (R_1) and the first round θ output (θ_{out}). The correlation between the Hamming weight of R_1 ($HW(R_1)$) and the measured power traces is then calculated and the key bits are extracted from the derived key relations. Our two-step CPA attack targeting SHA-3 based MAC was first introduced in [4]. In this attack, the correlation between $HW(\theta_{out})$ and the measured power traces are calculated. Round operations are then performed with the top key guesses to obtain R_1. The process is then repeated to determine the maximum correlation with R_1, and the key bits are selected from the top results.

The organization of this paper is as follows: Section II discusses the background of the SHA-3 hash function and presents the basic concepts of SHA-3 based MAC and HMAC. Early works on power attacks against SHA-3 based MAC are presented in Section III. The methodologies of two round one power attacks on SHA-3 based MAC are discussed in Section IV. The attack hardware and software setup utilized in this work is discussed in Section V. The results are presented and analyzed in Section VI. Finally, Section VII presents the conclusions.

II. Background

A. Sponge Construction

SHA-3 utilizes the sponge construction consisting of two phases: absorbing and squeezing. In the absorbing phase, the padded message is absorbed into the state. In the squeezing phase, the hash value is read from a subset of the state. A diagram of the sponge construction is shown in Fig. 1.

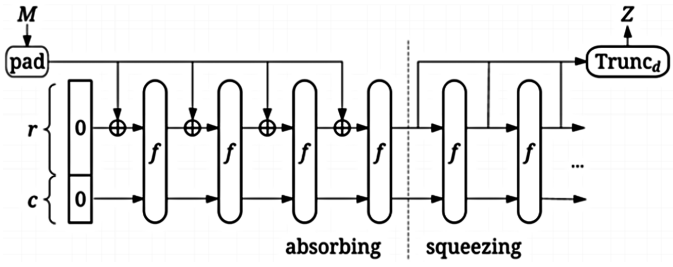

Fig. 1. Sponge Construction of SHA-3 [5].

The state of SHA-3 consists of 1600 bits and is divided into two parts: 1024-bit rate (r) and 576-bit capacity (c). The rate is the number of bits that the sponge absorbs in each stage and the capacity is the number of bits that get appended to every r bits to form the state. The state data is organized as a 3D array from left to right starting at the bottom plane and proceeding into the upper planes [2]. The state consists of five planes, each containing five lanes of 64 bits as shown in Fig. 2. The f function performs a random permutation of the

K. Millar, C.-Y. Chu and M. Łukowiak are with the Rochester Institute of Technology, Rochester, NY USA (e-mails: {kdm8162, cc4234, mxleec}@rit.edu)

state and is the core of the algorithm. The sponge construction is beneficial in that it can accept messages of arbitrary length and produce a hash value of fixed size without the risk of length extension attacks.

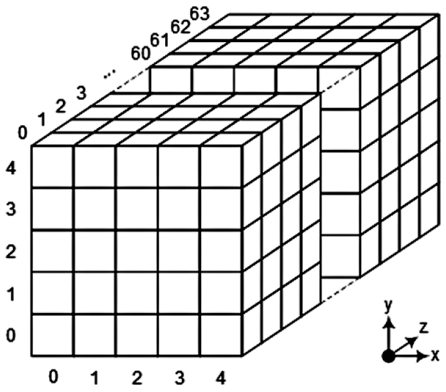

Fig. 2. The state of SHA-3.

B. SHA-3 Round Function

The f round function performs a random permutation of the state, and is the core of the SHA-3 algorithm. There are 24 rounds, each consisting of five transformations: Theta (θ), Rho (ρ), Pi (π), Chi (χ), and Iota (ι). Hence, a single round can be defined as

$$R = \iota \circ \chi \circ \pi \circ \rho \circ \theta(Input).$$

Let $a[x][y][z]$ denote a bit in the 3D array state, $a[x][y]$ a single lane and $a[x]$ a single sheet.

1) Theta Transformation, θ: The θ transformation provides diffusion across the lanes of the state. It is defined as an XOR operation between a single bit of the state and the bits of two columns. This operation is performed on every bit within the state.

$$\theta[x][y][z] \leftarrow a[x][y][z] \oplus (\oplus_{y'=0}^{4}\ a[x-1][y'][z]) \oplus\ (\oplus_{y'=0}^{4}\ a[x+1][y'][z-1])$$

Implementation of the θ operation is often performed in two steps. The first step generates the θ_{plane}, and is obtained using a 5-bit XOR operation across each column of the state.

$$\theta_{plane}[x][z] \leftarrow \oplus_{y'=0}^{4}(a[x][y'][z])$$

The second step calculates the θ_{out} value using a 3-bit XOR between the bit to be modified and the two bits of θ_{plane} corresponding with the columns required by the θ calculation.

$$\theta_{out}[x][y][z] \leftarrow a[x][y][z] \oplus (\theta_{plane}[x-1][z]) \oplus (\theta_{plane}[x+1][z-1])$$

The separation of the θ transformation into two steps is demonstrated in Fig. 3. In this figure, θ_1 represents the generation of the θ_{plane} and θ_2 represents the calculation of θ_{out}.

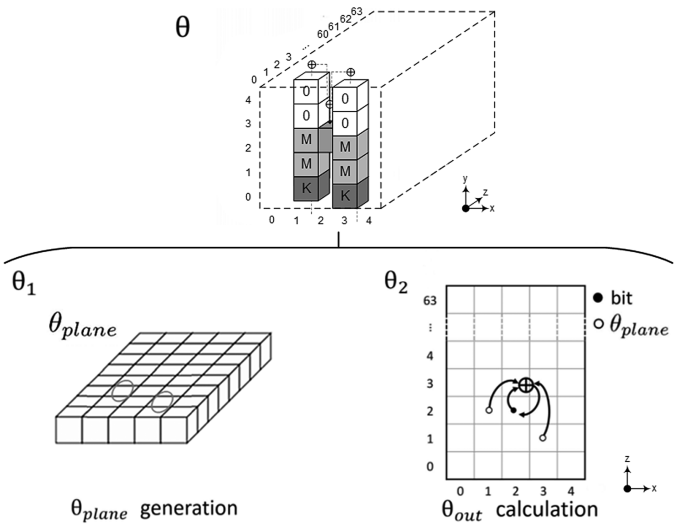

Fig. 3. The separation of the θ transformation into θ_1 and θ_2 [6].

2) Rho Transformation, ρ: The ρ transformation provides inter-slice diffusion of the state. It performs a binary rotation over each lane in the state with a different offset for each lane.

$$\rho[x][y][z] \leftarrow a[x][y][z-(t+1)(t+2)/2],$$
with t satisfying $0 \le t < 24$ and
$$\begin{pmatrix}0 & 1\\ 2 & 3\end{pmatrix}^{t}\begin{pmatrix}1\\0\end{pmatrix} = \begin{pmatrix}x\\y\end{pmatrix} \text{ in GF}(5)^{2\times 2},$$
or $t=-1$ if $x=y=0$

3) Pi Transformation, π: The goal of the π transformation is to disturb the horizontal and vertical alignment of each slice. This is achieved by shuffling each row of lanes to a corresponding column.

$$\pi[x][y] \leftarrow a[x'][y'], \text{ with } \begin{pmatrix}x\\y\end{pmatrix} = \begin{pmatrix}0 & 1\\ 2 & 3\end{pmatrix}\begin{pmatrix}x'\\y'\end{pmatrix}$$

4) Chi Transformation, χ: The χ transformation is the only non-linear step in the f function and utilizes XOR, AND, and NOT operations. This transformation operates on each row of the state and performs binary operations between the five bits within the row.

$$\chi[x][y][z] \leftarrow a[x][y][z] \oplus \left(\overline{a[x+1][y][z]} \cdot a[x+2][y][z]\right)$$

5) Iota Transformation, ι: The ι transformation is the last operation in the round function. It XORs a round-dependent constant i_r with the center lane of the state to disrupt the symmetry between rounds.

$$\iota[0][0] \leftarrow a[0][0] \oplus \text{RoundConstant}[i_r]$$

C. SHA-3 Based MAC and HMAC

A well known message authentication code (MAC) construction is the keyed-hash message authentication code (HMAC) that uses a cryptographic hash function in combination with a secret key [6]. The HMAC is used to verify the integrity of a message and its origin and is widely popular due to its provable security given the underlying hash function is secure [6]. The HMAC processes an arbitrary length

message M and a secret key K to produce a fixed-length output.

$$\text{HMAC}(K, M) = H((K \oplus opad) \parallel H((K \oplus ipad) \parallel M))$$

H is a cryptographic hash function, K is a secret key, M is the message to be authenticated, $ipad$ is the inner padding and $opad$ is the outer padding [6]. The hash function in the HMAC algorithm is used twice. The inner hash takes $(K \oplus ipad)$, appends the message and hashes it. The outer hash function of the HMAC prepends the digest of the inner hash function with another key variant $(K \oplus opad)$ and produces the HMAC digest. With the FIPS PUB 198 standard usage of HMACs, there have been no successful attacks on HMAC SHA-1 as the outer application of the hash function masks the intermediate result of the internal hash. Unlike the Merkle-Damgård construction of SHA-1, the output of the sponge construction of SHA-3 reveals only a part of the final state. Therefore, it is believed that SHA-3 can be used in a direct MAC construction in which the secret key is directly prepended to the message before hashing [5].

$$\text{MAC}(K, M) = H(K \parallel M)$$

The secret key K used in SHA-3 based MAC is the target of all attacks described in this paper.

III. Early Work

Taha and Schaumont performed differential power analysis (DPA) on MAC-Keccak for variable key lengths in [6]. This analysis targeted software implementations of MAC-Keccak. Case studies were performed by applying power analysis attacks (PAAs) for varying key lengths targeting the θ operation during the first round. DPA iterations were executed on measured power traces of SHA-3 based MAC with known input messages to obtain the unknown variables necessary to extract the secret key. For key-lengths of 768 and 896 bits, DPA was used to first extract the unknown variables necessary to calculate the θ_{plane}. Once this information was extracted, DPA was performed on θ_{out} to extract the key-bit values from the XOR between the key-bits and the θ_{plane}. For a key-length of 1024 bits, DPA iterations were executed to recover the state after the θ operation. The ρ and π permutations were then performed, and further DPA iterations were executed to extract the unknown variables of the χ operation. From this information, the secret key could be extracted. The reference software implementation of Keccak was programmed onto a 32-bit Microblaze processor on a 90 nm Xilinx Spartan-3E FPGA. Each attack was performed yielding success rates of about 90%, 80%, and 40%, respectively, when applied to 50,000 traces. This work showed that unprotected software implementations of SHA-3 based MAC are vulnerable to DPA attacks and that these attacks become increasingly difficult for larger key lengths.

In [7], Luo et al. performed further side channel vulnerability analysis of SHA-3 based MAC targeting a hardware implementation on an FPGA. Because the intermediate variables used to attack software implementations may not correlate directly to power traces generated by hardware implementations, a new attack target and leakage model were needed. An attack was developed taking advantage of the hardware implementation of the θ transformation in which it was divided into two steps. The θ_{plane} calculation for each column occurs between 4-bits of the known message and a single bit of the key allowing for the key bits to be directly extracted. To increase the signal-to-noise ratio (SNR) of the attack target, 8-bits of the state were attacked concurrently. This method was used only to find the key bits of a single lane as the correlation between the power traces and $HW(\theta_{plane})$ was not very strong resulting in a low overall success rate. Once a full key lane was known, the θ_{out} calculation was used as a second attack target to extract the remainder of the key. Power traces were obtained from a reference implementation of MAC-Keccak programmed onto a Sasebo-GII board containing a 65 nm Xilinx Virtex-5 FPGA. This two-step θ attack was able to successfully extract a single byte of the key achieving a 90% success rate with 500,000 traces.

A methodology by which to simulate PAAs on cryptographic implementations through Synopsys design tools was introduced by Smith and Łukowiak in [8] targeting AES. This work was later extended by Tran et al. in [9] targeting SHA-3 based MAC with keys of varying bit length. CPA attacks targeting the first round θ output, specifically the initial calculation of θ_{plane}, were performed on simulated power consumption waveforms generated by Synopsys EDA tools with a 130 nm CMOS standard cell library.

IV. Round 1 Power Attacks

Two round one attacks targeting SHA-3 based MAC were analyzed. The input data is composed of the secret key, message, and padding. The authenticated message is sent in plaintext and is known to the attacker. A 320-bit (40 byte) key is chosen for simplicity, but both presented attacks can be modified for keys of variable length. The key completely fills the bottom plane of the state as shown in Fig. 4. Each succeeding plane is filled with the message, and the capacity c of the initial state is filled with 0s.

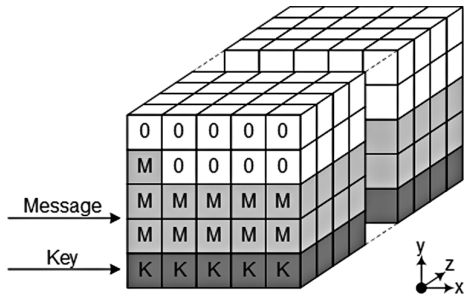

Fig. 4. Initial content of the state register containing the 320-bit key.

A. Round 1 Output Attack

A CPA attack targeting the first round output of a hardware implementation of MAC-Keccak was introduced by Luo et al. in [3]. The correlation between measured power traces and $HW(R_1)$ were found to be stronger than the correlation between measured power traces and $HW(\theta_{out})$. Because of this, higher key extraction success rates were achieved with fewer traces when compared to previous θ-based attacks.

The attack methodology and selected attack target are illustrated in Fig. 5. It is assumed that both the initial value of the state register and input message are known to the attacker.

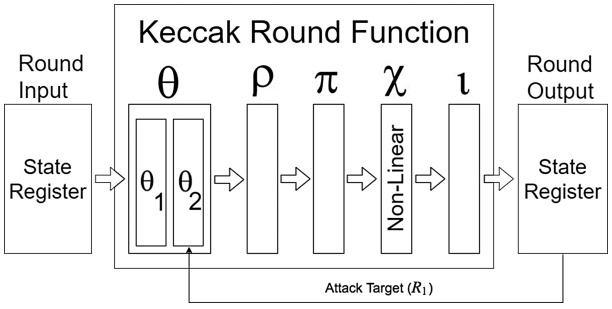

Fig. 5. Target of the round one output attack on SHA-3 based MAC.

The interaction between the key bits and each of the known bits of the state in the calculation of R_1 are analyzed to produce a mapping from R_1 to the initial state. Because the ι operation performs only a binary XOR with a known round constant, the attack first focuses on the χ operation. Each bit of $\chi[x][y][z]$ is dependent upon three bits of $\pi[x][y][z]$.

$$\chi[x][y][z] \leftarrow \pi[x][y][z] \oplus \left(\overline{\pi[x+1][y][z]} \cdot \pi[x+2][y][z]\right)$$

The π and ρ operations change only the positions of the bits within the state and retain their values allowing for each bit of $\pi[x][y][z]$ to be directly mapped to a bit of $\theta[x][y][z]$ by performing the inverse π and ρ operations π^{-1} and ρ^{-1}, respectively.

$$\begin{cases} \pi[x][y][z] \longleftrightarrow \theta[x_0][y_0][z_0] \\ \pi[x+1][y][z] \longleftrightarrow \theta[x_1][y_1][z_1] \\ \pi[x+2][y][z] \longleftrightarrow \theta[x_2][y_2][z_2] \end{cases}$$

Each bit of $\chi[x][y][z]$ can then be represented as a binary operation between three bits of $\theta[x][y][z]$.

$$\chi[x][y][z] = \theta[x_0][y_0][z_0] \oplus \left(\overline{\theta[x_1][y_1][z_1]} \cdot \theta[x_2][y_2][z_2]\right)$$

The bits of $\theta[x][y][z]$ are the result of operations containing either two or three key bits, and the relation between these key bits can be extracted as an XOR as shown in Fig. 6. Though DPA could be performed on each bit individually, this attack method would require a large number of traces due to low SNR. This attack would also be computationally expensive as it would require separate analysis of each bit within the 1600-bit state. The presented attack instead targets each 5-bit row of the state requiring separate analysis of each of the 320 rows.

At the completion of the χ operation, each row contains 1-bit originally from the bottom plane and 4-bits from the other planes. Therefore, each row of R_1 contains 4-bits with two XOR key relations and 1-bit with three XOR key relations. The π^{-1} and ρ^{-1} operations are performed on each row to obtain the mapping relationship between $\pi[x][y][x]$ and $\theta[x][y][z]$ of each bit in the row. This mapping allows for extraction of the key relations involved in the θ calculation of each bit in the row. It is then possible to perform correlation calculations for each of the 32 possible values of each enumerated key relation.

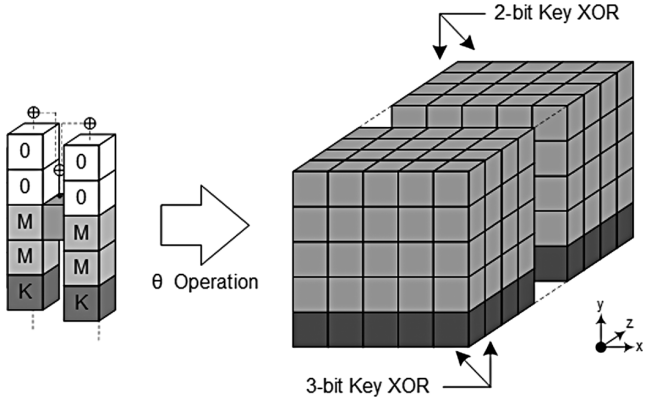

Fig. 6. The θ transformation producing 1280 2-bit key XOR relations and 320 3-bit key XOR relations.

This is achieved by calculating the round one output of each row for each key relation guess, calculating the Hamming weight of the row output, and measuring the correlation between the Hamming weight of the output and the measured power traces. The key relation producing the largest correlation is saved, and the process is repeated across each row in the state. Analysis of the full R_1 results in 320 unique 3-bit XORs and 1280 2-bit XORs between key bits of which 320 are unique. This allows for perfect extraction of the full 320-bit key by pairing each 3-bit relation with its corresponding 2-bit relation and performing a binary XOR to obtain the individual key bit.

B. Two-Step Round 1 Attack

This work proposes the round one attack first introduced in [4] consisting of two steps to improve the success rate of secret key extraction. The first attack target is the θ_{out} calculation. This target provides a higher SNR over the θ_{plane} and allows for successful retrieval of the relationship between two key bits. Efficiency of the first attack stage was improved by adopting the iterative methodology described in [10].

The second attack target exploits the key relationship as the intermediate result necessary to attack R_1. This allows for full retrieval of the individual key bits. The selected attack targets and their relation is illustrated in Fig. 7.

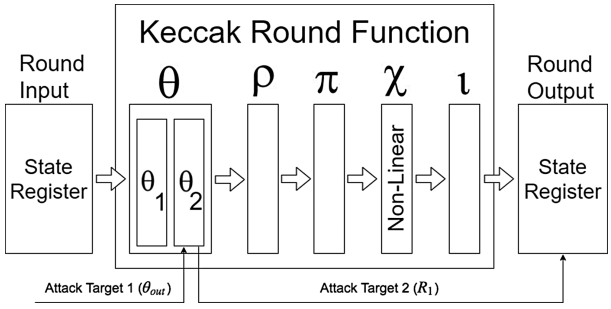

Fig. 7. Targets of the two-step round one attack on SHA-3 based MAC.

1) Attack Target 1 – θ_{out} Calculation: It is assumed that the message content of the four upper planes is known to the attacker. Therefore, if the result of $\theta_{plane}[x-1][z] \oplus \theta_{plane}[x+1][z-1]$ is found, it can be used to obtain the following key relationship: $K[x-1][z] \oplus K[x+1][z-1]$. To increase the SNR for CPA, 8-bits are attacked simultaneously.

2) Attack Target 2 – θ_{out} to Round Output Calculation: After successfully attacking θ_{out}, the intermediate results can be utilized to model the execution of the other four transformations (ρ, π, χ, ι) to completely obtain R_1. The main benefit of targeting R_1 is the non-linear property of the χ transformation. This property provides large differences in the output for small differences in the input key guesses producing a larger correlation for the attack. The main challenge of this approach is in tracing the location of the key-related output bits in R_1. The following briefly explains the location of the target key affected bits in R_1 after each transformation.

The first round operation is the θ transformation. The key plane is XORd with θ_{plane} and is the known result of the first attack target. As a result, the transformed key affected bits remain in the bottom plane of the state. The second is the ρ transformation in which a circular shift is applied to each lane along the z-axis with all bits remaining in the same plane. The third is the π transformation as illustrated in Fig. 8 with the transformed key affected bits colored in gray. As a result of the π operation, the transformed key bits of the bottom plane align with the y-axis.

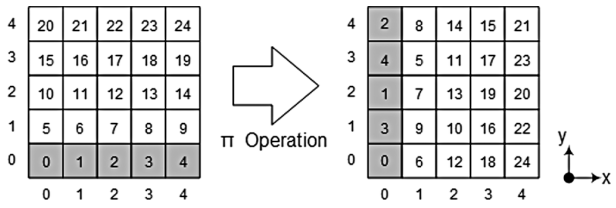

Fig. 8. The π permutation with transformed key bits colored in gray.

The fourth is the χ transformation, the non-linear operation in which a single transformed key bit will affect three output bits. The left side of Fig. 9 shows the χ operation applied to a single row of the state with the transformed key bit colored in light gray and the target key affected bits colored in dark gray. As a result of χ, three sheets of the state will be affected by the transformed key plane.

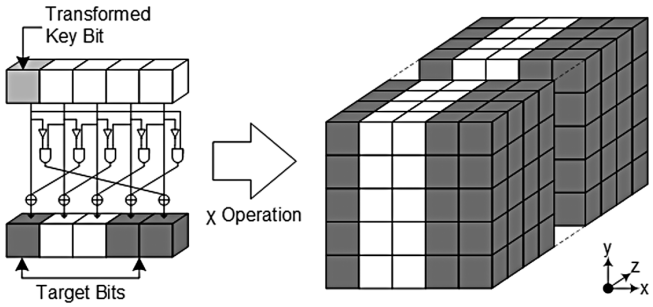

Fig. 9. The χ transformation with transformed key bits colored in light gray and attack target bits colored in dark gray.

The last is the ι transformation, a simple XOR operation between a 64-bit round constant and the center lane of the state. The first round constant is a value of 0x0000000000000001 and affects only a single bit of the state.

After careful analysis of the round transformations, the result of R_1 can be modeled completely. As only three sheets of R_1 are affected by the bits of the θ_{out} key plane, the other two sheets can be ignored. Like the first attack target, 8-bits of the key are attacked simultaneously to obtain a stronger correlation.

V. Hardware and Software Setup

The three main components of the attack setup are as follows: SAKURA-X board, oscilloscope, and PC. Together, these components are responsible for execution of the SHA-3 hardware implementation, capture of the power consumption waveforms, and recording and processing of data. Developed software in this work includes control software designed in C# and attacking scripts written in MATLAB. The control software manages operation of the SAKURA-X board, oscilloscope automation, and data collection. Each attack script calculates the Hamming weight data model, loads the necessary power traces, and performs the respective CPA attack as described. Fig. 10 shows a diagram of the hardware setup and system connections for the attack.

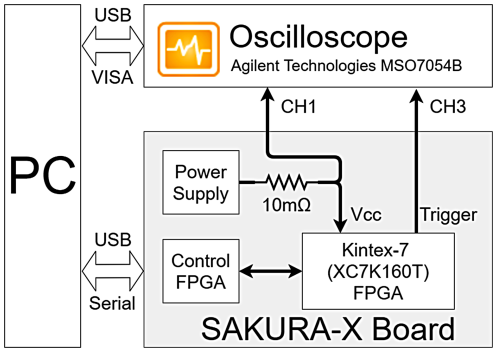

Fig. 10. Diagram of the attack hardware setup.

The PC connects to the SAKURA-X board and oscilloscope through a USB interface. Channel 1 of the oscilloscope connects to the 1.0V power input of the Xilinx Kintex-7 FPGA on the SAKURA-X board using an SMA to BNC cable. Channel 3 of the oscilloscope connects to the CN8 Header Pin1 as the trigger input using a basic 1:10 probe. A photo of the hardware setup is shown in Fig. 11.

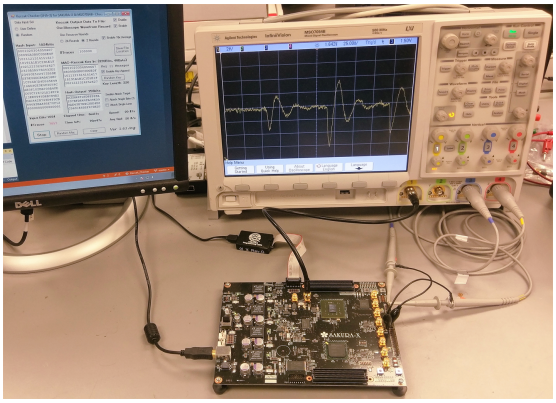

Fig. 11. Photo of the hardware setup.

The final results were obtained from each attack script running on a Windows 10 laptop equipped with an Intel Core i7-5700HQ processor, 12GB of RAM, and a 1TB hard drive. An Agilent Technologies MSO7054B Mixed Signal

Oscilloscope was used in this work with the 16 times average acquisition mode enabled to reduce noise. For programming of the FPGA, all implementations were generated using Xilinx Vivado 2016.4 with default settings for synthesis, mapping, and routing.

VI. Results and Analysis

Both round one power attacks were implemented and tested on the same set of traces to allow for direct comparison between them. The Keccak design team provides HDL implementations of Keccak that can be downloaded from [11]. The high-speed core was utilized with Keccak-1600 and programmed onto the SAKURA-X board. Random message data was generated and SHA-3 based MAC was implemented with a random 40-byte key. The generated power traces were captured by a controlled oscilloscope for each set of random input data. The complete data set contained 500,000 traces focused on the first two rounds of the Keccak hashing algorithm obtained with 16x averaging and a resolution of 1000 points per trace.

To verify the captured power traces, the correlation between $HW(R_1)$ and the measured power traces was calculated as shown in Fig. 12. This graph shows that a significant correlation exists at the time the R_1 data is written to the state register.

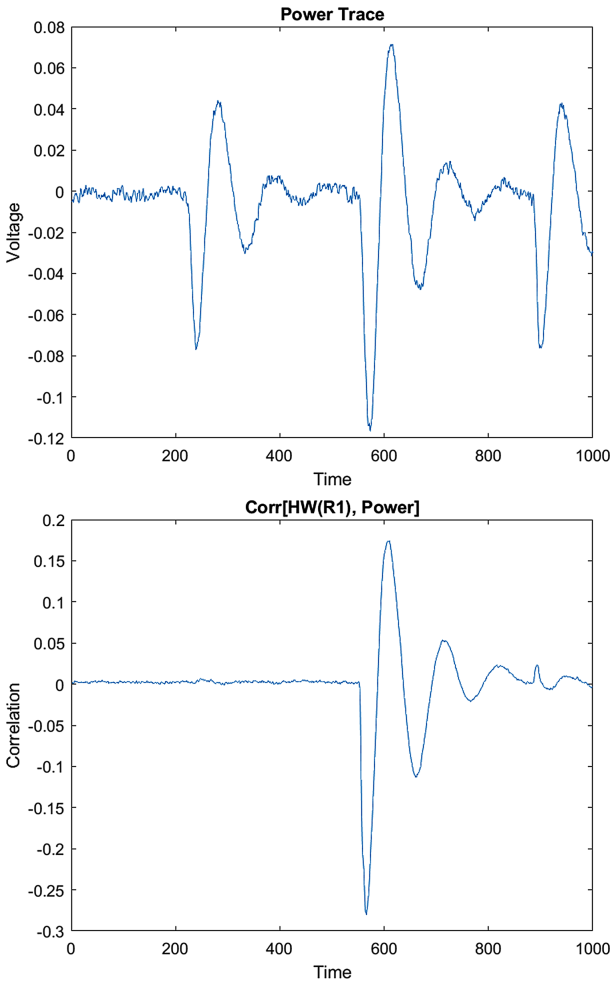

Fig. 12. A single power trace (top) and the correlation between $HW(R_1)$ and 500,000 power traces (bottom).

A. Round 1 Output Attack

Because the R_1 based attack targets each row of the MAC-Keccak state, the correlation between the Hamming weight of a single row of R_1, $R_1[0:4][4][0]$, and the power traces were calculated with both 50,000 and 200,000 traces as shown in Fig. 13. These plots show that the correlation

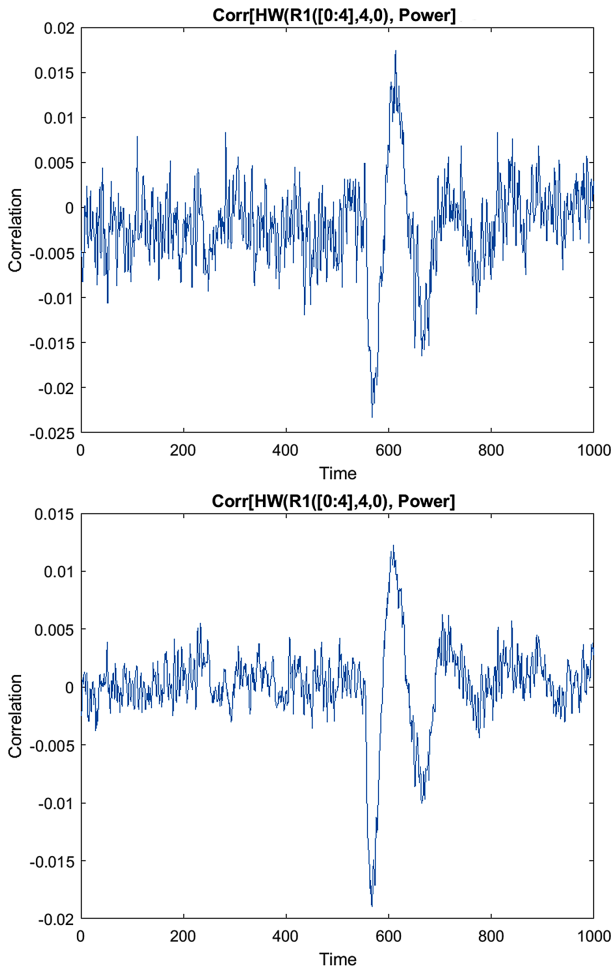

Fig. 13. The correlation between the Hamming weight of $R_1[0:4][4][0]$ with 50,000 traces (top) and 200,000 traces (bottom).

between $HW(R_1)$ and the power traces does exist; however, the SNR is quite low for small trace amounts and a large quantity of traces may be required to extract the full key successfully.

The initial stages of the attack were implemented targeting the row $R_1[0:4][4][0]$. The enumeration of the key relations for this row results in a decimal value of 23 and was successfully extracted for two subsets of the data set containing 50,000 and 200,000 traces as shown in Fig. 14.

Though the attack was able to successfully extract the correct value of the key relation with 50,000 traces, a correlation of 0.02 for the enumerated value of 6 came very close to the correlation of 0.023 achieved by the correct enumerated key relation value of 23. With 200,000 traces, this false positive was diminished almost entirely further suggesting that noise will need to be overcome through analysis of a large number of traces.

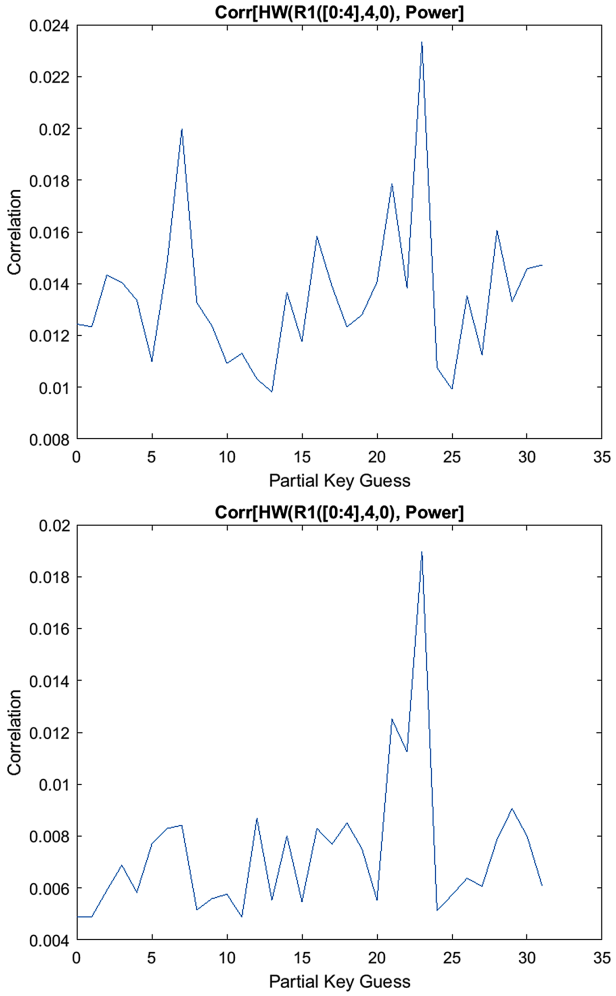

Fig. 14. The maximum correlation between $HW(R_1[0:4][4][0])$ and the enumerated key relations for 50,000 traces (top) and 200,000 traces (bottom).

The attack was then extended to determine the key relations for each row of R_1 and to extract the full 320-bit key. Subsets of the data set, incremented by 10,000 traces, were analyzed to determine the success rate of extracting the full key for each quantity of traces producing the results shown in Fig. 15.

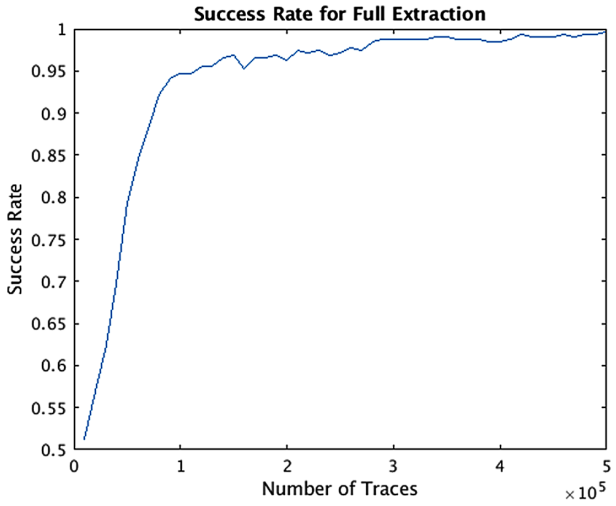

Fig. 15. The success rate for full key extraction of the round one output attack with varying numbers of traces.

The attack successfully extracted 319 of the 320 key bits with 500,000 traces reaching a maximum success rate of 99.69%.

B. Two-Step Round 1 Attack

The legend of "Top N" indicates that the target key byte is in the Top N of ranked correlation results, and the success rate represents the number of extracted key bytes that are verified to be correct. For example, because the key is 40 bytes long, "Top 1" with a success rate of 0.8 indicates that 32 of the 40 key bytes have been found.

1) Attack Target 1: The first attack targets the θ_{out} calculation. Correlation results of Top 16 key guess were saved for each byte of the state. By accepting a large number of possible key guesses, the chances of final key recovery were increased. The attack then establishes the power consumption matrix for all partial key guesses in the Top 16 result and determines which key guesses produce a larger correlation with the measured power traces. An iterative process as described in [10] is then initialized to remove the low correlation key guesses and reduce the number of saved key guess from 16 to 8. This process is then repeated, reducing the number of saved guess by half until Top 1 values are found.

Fig. 16 shows a plot of the success rate when attacking the 40-byte key by targeting the θ_{out} calculation. In the case of 500,000 traces, the success rate for Top 1 is 1.0. This indicates that the key relationship $(K[x-1][z] \oplus K[x+1][z-1])$ can be retrieved for the entire key plane.

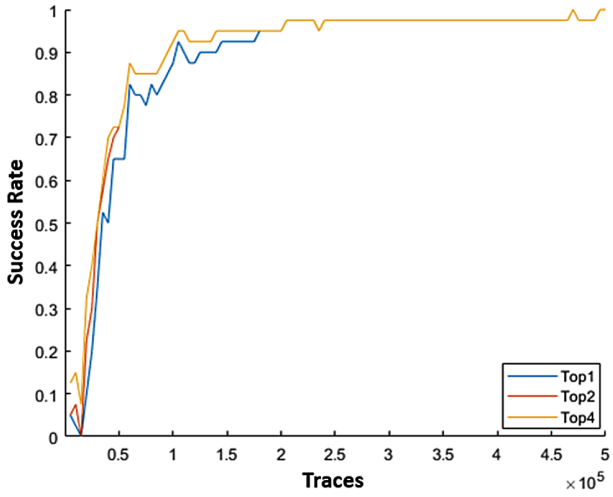

Fig. 16. The success rate of attack target one.

2) Attack Target 2: The second attack target utilizes the key relationship results obtained by the first attack target and models the execution of the other four transformations to obtain R_1. Through this attack, the entire key plane can be retrieved. The results here show the success rates for the second attack target only. Because this target requires the intermediate values obtained from the first attack, to isolate the effects of the success rates from the first attack, the success rates for the second attack are calculated using the correct key values as input.

Because this attack targets the non-linear round output, the success rates are very high. As presented in Fig. 17, the Top 8 correlation has a success rate of 1.0, and the Top 1 correlation has a success rate of 0.975. This success rate corresponds with successful extraction of 39 of the 40 key bytes.

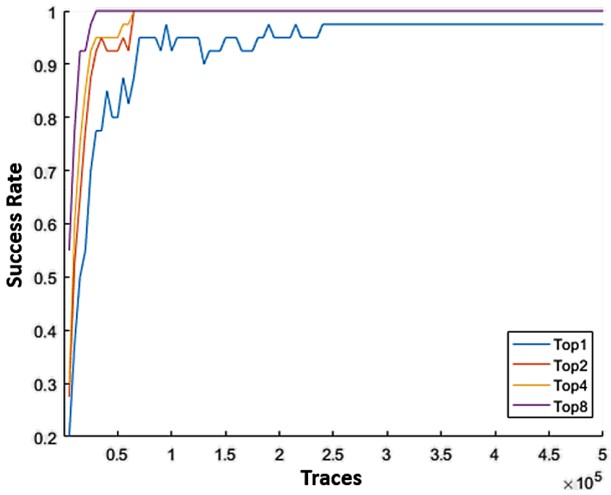

Fig. 17. The success rate of attack target two.

3) Combined Attack Result: After successful verification of both attack targets, the attack was combined to perform a complete two-step attack. Results of the first attack target were obtained and used to recover the key through implementation of the second attack. The success rate of the combined attack is shown in Fig. 18.

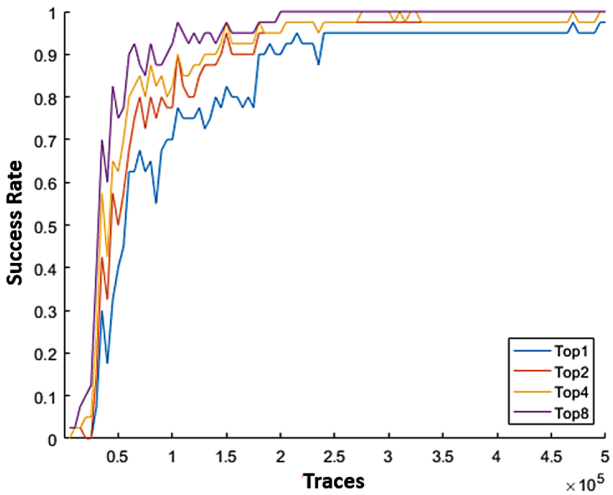

Fig. 18. The success rate of the combined two-step round one attack.

The Top 1 success rate becomes stable after nearly 250,000 traces with a value of 0.95. This corresponds with successful extraction of 38 of the 40 key bytes. For the Top 8 correlation, the success rate is 1.0 (100%) after about 200,000 traces. The highest success rate is obtained for 500,000 traces with a Top 1 correlation value of 0.975. This means that 39 of the 40 key bytes were successfully retrieved. The only missing key byte is contained within the Top 2 set. By combining different attack targets, the success rate for recovering the entire key plane is 0.95 for 250,000 traces and 0.975 for 500,000 traces.

C. Comparison of Attacks

The top result of each attack performed on subsets of the data set incremented by 10,000 traces is shown in Fig. 19.

Both attacks successfully extracted the 320-bit authentication key with a 90% success rate for 200,000 traces. The round one output attack successfully extracted 319 of the 320 key bits with 500,000 traces reaching a maximum success rate of 99.69%. The two-step attack successfully extracted 39 out of 40 key bytes achieving a maximum success rate of 97.5% key extraction with 500,000 traces.

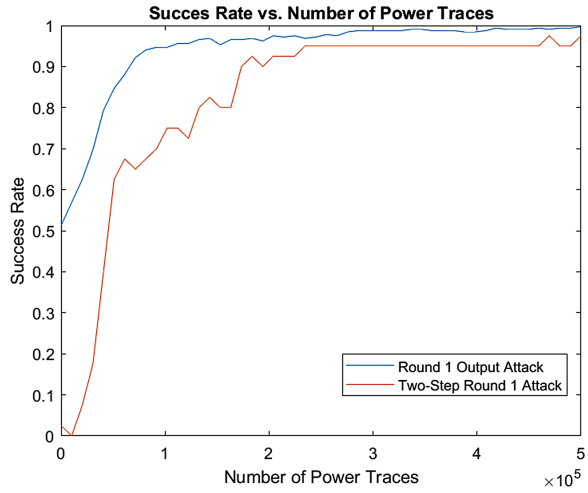

Fig. 19. The success rate for full key extraction of both the round one output and two-step round one attacks with varying numbers of traces.

VII. Conclusion

This work analyzed and compared two round one power analysis attacks that use CPA to extract the secret key of SHA-3 based MAC. The round one output attack introduced by Luo et al. in [3] was implemented and compared to the proposed two-step round one attack. Both attacks are viable and were able to successfully extract the private authentication key from SHA-3 based MAC showing that unsecured implementations are vulnerable to PAA. These attacks are not restricted to a 320-bit key and may be modified to target keys of variable length. Counter measures are necessary to ensure that future implementations of SHA-3 based MAC are secure against SCA.

References

[1] M. Stevens, E. Bursztein, P. Karpman, A. Albertini, Y. Markov, A. P. Bianco, and C. Baisse, "Announcing the first sha1 collision," *Google Security Blog*, Feb. 2017. [Online]. Available: https://security.googleblog.com/2017/02/announcing-first-sha1-collision.html

[2] *SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions*, NIST Std., Aug. 2015. [Online]. Available: http://dx.doi.org/10.6028/NIST.FIPS.202

[3] P. Luo, Y. Fei, X. Fang, A. A. Ding, D. R. Kaeli, and M. Leeser, "Side-channel analysis of mac-keccak hardware implementations," in *Proceedings of the Fourth Workshop on Hardware and Architectural Support for Security and Privacy*, ser. HASP '15. New York, NY, USA: ACM, 2015, pp. 1:1–1:8. [Online]. Available: http://doi.acm.org/10.1145/2768566.2768567

[4] C. Chu and M. Lukowiak, "Two step power attack on sha-3 based mac," in *2018 25th International Conference "Mixed Design of Integrated Circuits and System" (MIXDES)*, June 2018, pp. 209–214.
[5] G. Bertoni, J. Daemen, M. Peeters, and G. V. Assche, "Cryptographic sponge functions," 2011. [Online]. Available: http://sponge.noekeon.org/CSF-0.1.pdf,2011
[6] M. Taha and P. Schaumont, "Differential power analysis of mac-keccak at any key-length," in *Advances in Information and Computer Security*, K. Sakiyama and M. Terada, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 68–82.
[7] P. Luo, Y. Fei, X. Fang, A. A. Ding, M. Leeser, and D. R. Kaeli, "Power analysis attack on hardware implementation of mac-keccak on fpgas," in *2014 International Conference on ReConFigurable Computing and FPGAs (ReConFig14)*, Dec 2014, pp. 1–7.
[8] K. Smith and M. Łukowiak, "Methodology for simulated power analysis attacks on aes," in *2010 - MILCOM 2010 MILITARY COMMUNICATIONS CONFERENCE*, Oct 2010, pp. 1292–1297.
[9] X. D. Tran, M. Łukowiak, and S. P. Radziszowski, "Effectiveness of variable bit-length power analysis attacks on sha-3 based mac," in *MILCOM 2016 - 2016 IEEE Military Communications Conference*, Nov 2016, pp. 794–799.
[10] C. Clavier, J.-L. Danger, G. Duc, M. A. Elaabid, B. Gérard, S. Guilley, A. Heuser, M. Kasper, Y. Li, V. Lomné, D. Nakatsu, K. Ohta, K. Sakiyama, L. Sauvage, W. Schindler, M. Stöttinger, N. Veyrat-Charvillon, M. Walle, and A. Wurcker, "Practical improvements of side-channel attacks on aes: feedback from the 2nd dpa contest," *Journal of Cryptographic Engineering*, vol. 4, no. 4, pp. 259–274, Nov 2014. [Online]. Available: https://doi.org/10.1007/s13389-014-0075-9
[11] "Keccak hardware implementation in vhdl version 3.1." [Online]. Available: http://keccak.noekeon.org/KeccakVHDL-3.1.zip

Kevin Millar is a student at the Rochester Institute of Technology, New York working towards a BS/MS degree in computer engineering. He is a member of the Honors Program, and was named an Outstanding Undergraduate Scholar in Spring 2017. He is a member of the Applied Cryptography and Information Security laboratory under the advisement of Dr. Marcin Lukowiak and is working towards the completion of his Master's degree. His primary research interests include cryptography and reconfigurable computing.

Chun-Yi Chu obtained his MS degree in the Department of Computer Engineering at Rochester Institute of Technology and BS in the Department of Electronic and Computer Engineering at National Taiwan University of Science and Technology. His research interests are focused on applied cryptography, hardware- software cryptosystems and side channel attack countermeasures.

Marcin Lukowiak obtained a BS/MS dual degree in the Department of Control and Systems Engineering, and a Ph.D. degree in the Faculty of Electrical Engineering, both at Poznan University of Technology, Poland. He has over 20 years of academic experience and currently is an Associate Professor in the Department of Computer Engineering at Rochester Institute of Technology. His professional interests are concentrated in cross-disciplinary areas involving reconfigurable computing, cryptographic engineering, hardware and hardware-software systems, and high performance heterogeneous computing.