cryptography

MDPI

Article

Comparison of Cost of Protection against Differential Power Analysis of Selected Authenticated Ciphers †

William Diehl [1,*], Abubakr Abdulgadir [2], Farnoud Farahmand [2], Jens-Peter Kaps [2] and Kris Gaj [2]

[1] The Bradley Department of Electrical and Computer Engineering, Virginia Polytechnic Institute and State University, Blacksburg, VA 24061, USA

[2] Department of Electrical and Computer Engineering, George Mason University, Fairfax, VA 22030, USA; aabdulga@gmu.edu (A.A.); ffarahma@gmu.edu (F.F.); jkaps@gmu.edu (J.-P.K.); kgaj@gmu.edu (K.G.)

* Correspondence: wdiehl@vt.edu; Tel.: +1-540-231-3771

† This paper is an extended version of our paper published in the International Symposium on Hardware Oriented Security and Trust, Washington DC, USA, 1–7 May 2018.

Received: 31 July 2018; Accepted: 10 September 2018; Published: 19 September 2018

Abstract: Authenticated ciphers, which combine the cryptographic services of confidentiality, integrity, and authentication into one algorithmic construct, can potentially provide improved security and efficiencies in the processing of sensitive data. However, they are vulnerable to side-channel attacks such as differential power analysis (DPA). Although the Test Vector Leakage Assessment (TVLA) methodology has been used to confirm improved resistance of block ciphers to DPA after application of countermeasures, extension of TVLA to authenticated ciphers is non-trivial, since authenticated ciphers have expanded input and output requirements, complex interfaces, and long test vectors which include protocol necessary to describe authenticated cipher operations. In this research, we upgrade the FOBOS test architecture with capability to perform TVLA on authenticated ciphers. We show that FPGA implementations of the CAESAR Round 3 candidates ACORN, Ascon, CLOC (with AES and TWINE primitives), SILC (with AES, PRESENT, and LED primitives), JAMBU (with AES and SIMON primitives), and Ketje Jr.; as well as AES-GCM, are vulnerable to 1st order DPA. We then use threshold implementations to protect the above cipher implementations against 1st order DPA, and verify the effectiveness of countermeasures using the TVLA methodology. Finally, we compare the unprotected and protected cipher implementations in terms of area, performance (maximum frequency and throughput), throughput-to-area (TP/A) ratio, power, and energy per bit (E/bit). Our results show that ACORN consumes the lowest number of resources, has the highest TP/A ratio, and is the most energy-efficient of all DPA-resistant implementations. However, Ketje Jr. has the highest throughput.

Keywords: cryptography; authenticated cipher; field programmable gate array; power analysis; side channel attack; countermeasure; lightweight; TVLA; *t*-test

1. Introduction

Today's environment of large and high-speed centralized cloud-based computing is expanding into tomorrow's smaller and lightweight edge-based computing, which will consist of billions of devices in the "Internet of Things" (IoT). IoT devices are both resource-constrained, especially in terms of power and energy, and particularly vulnerable to exploitation and compromise, since they are more likely to be physically accessible by an adversary.

Authenticated ciphers, such as AES-GCM, are well-suited for lightweight edge devices in the IoT, since they combine the functionality of confidentiality, integrity, and authentication services, and can

potentially provide more efficient integration of disparate cryptographic components, such as block ciphers and keyed hashes [1].

Input and output fields, and computational processes of authenticated ciphers, are introduced in [2], and are summarized below:

1. *Message*—An input field to authenticated encryption consisting of *Plaintext* to be encrypted to *Ciphertext*, which is an output from authenticated encryption.
2. *Ciphertext*—An output from authenticated encryption, and input to authenticated decryption, which consists of data to be decrypted to *Plaintext*.
3. *AD* (Associated Data)—Data accompanying *Message* that will not be encrypted, but contains ancillary information such as header or protocol information.
4. *Npub*—A public message number; usually a nonce (number used once).
5. *Key*—A secret key, used for encryption and decryption to ensure confidentiality, and used in keyed hash functions to ensure integrity and authenticity.
6. *Tag*—A function of all blocks of *AD*, *Message*, *Npub*, and *Key*, which is produced at the conclusion of message encryption, and provides a value which is used by the recipient to verify integrity and authenticity.

During authenticated decryption, *Tag*, forwarded from the module performing authenticated encryption, is compared to *Tag*′, computed by the module performing authenticated decryption, as a function of *Ciphertext*, *Npub*, *AD*, and *Key*. If *Tag* = *Tag*′, then the authentication is considered valid, and *Ciphertext* is released. A notional authenticated cipher is shown in Figure 1.

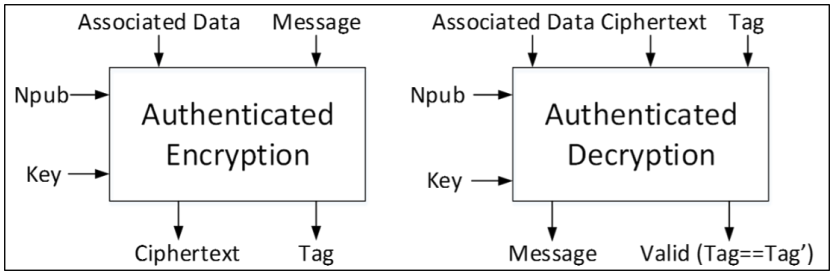


Figure 1. Notional authenticated cipher.

Cryptographic algorithms, which are either well-researched or endorsed by government standardization organizations (e.g., NIST), are generally secure against cryptanalysis or brute-force attacks, given currently available computing power. However, cryptography is implemented in physical devices, which are subject to information leakage, and which can be exploited through so-called side-channel attacks, such as Differential Power Analysis (DPA), through which an attacker can often recover sensitive information.

Several competitions and standardization efforts are evaluating authenticated cipher candidates worldwide to determine the most optimal algorithms for various applications, including lightweight cryptography suitable for the IoT. One of these efforts is the Competition for Authenticated Encryption: Security, Applicability, and Robustness (CAESAR), which has evaluated a large number of candidate authenticated ciphers over the last four years, in order to ultimately select a final portfolio of ciphers that offer advantages over AES-GCM, and are suitable for widespread adoption [3]. Beginning in Round 3, the CAESAR committee identified use-cases for which candidates would be evaluated. One of these use cases is for lightweight applications, for which desired characteristics include "natural ability to protect against side-channel attacks" [4].

Another example is the NIST Lightweight Cryptography (LWC) Standardization Process, which is a multi-year effort to examine categories of lightweight cryptographic algorithms (including authenticated ciphers) in order to choose eventual U.S. federal standards. The NIST LWC

standardization process takes into account candidate ciphers' abilities to be protected against side-channel attacks. Additionally, the NIST LWC standardiztion process emphasizes the desire for third-party analysis, i.e., analysis by researchers other than the authors of cipher submissions [5].

Given the widespread interest in developing and standardizing authenticated ciphers, it is useful to examine implementations of authenticated ciphers with specified lightweight use-cases to compare resistance of unprotected and protected implementations to DPA, as well as their costs of protection. However, to date, there has been no study of the side-channel resistance of a large group of authenticated ciphers, implemented using the same methodology and same test equipment, and no study of their comparative costs of protection against DPA.

In this work, we demonstrate a methodology for determining vulnerabilities of authenticated ciphers to DPA, and evaluating the effectiveness of DPA countermeasures. We use an existing Test Vector Leakage Assessment (TVLA) methodology (i.e., *t*-test, further discussed in "Materials and Methods") [6,7], and upgrade the Flexible Open-source workBench fOr Side-channel analysis (FOBOS) [8], to perform TVLA on authenticated ciphers. The FOBOS interface with the victim cipher implementation is standardized by leveraging the CAESAR Hardware Applications Programming Interface (API) for Authenticated Ciphers, which was adopted by the CAESAR committee in May 2016 [9,10]. Additionally, our use of the CAESAR Hardware API Development Package, available at [11] enables a repeatable and exportable test methodology for all CAESAR candidates.

Using the augmented FOBOS, we demonstrate *t*-tests on unprotected implementations of the CAESAR Round 3 variants of the ACORN, Ascon, CLOC-SILC, JAMBU, and Ketje families of authenticated ciphers, described in [12–16], respectively. We choose these ciphers since their authors have specified an intended lightweight use case for their respective ciphers. We additionally analyze an existing defacto authenticated cipher standard AES-GCM, described in [17], for purposes of comparison. We use register transfer level (RTL) VHDL implementations of AES-GCM, Ascon, CLOC-AES, JAMBU-AES, and SILC-AES available at [18], ACORN at [19], CLOC-TWINE, SILC-PRESENT, and SILC-LED at [20], JAMBU-SIMON available at [21], and Ketje Jr. available at [22]. The authenticated ciphers investigated in this research, including relevant characteristics, are summarized in Table 1.

Table 1. Authenticated ciphers examined in this research.

Authenticated Cipher	Spec	Implementation	Key Size [bits]	Block Size [bits]	Tag Size [bits]
AES-GCM	[17]	CERG GMU [18]	128	128	128
ACORN	[12]	CCRG NTU [19]	128	1	128
Ascon	[13]	CERG GMU [18]	128	64	128
CLOC-AES	[14]	CERG GMU [18]	128	128	64
CLOC-TWINE	[14]	CLOC-SILC Team [20]	80	64	32
SILC-AES	[14]	CERG GMU [18]	128	128	64
SILC-PRESENT	[14]	CLOC-SILC Team [20]	80	64	32
SILC-LED	[14]	CLOC-SILC Team [20]	80	64	32
JAMBU-AES	[15]	CERG GMU [18]	128	64	64
JAMBU-SIMON	[15]	CCRG NTU [21]	96	48	48
Ketje Jr.	[16]	Ketje-Keyak Team [22]	96	32	64

AES is Advanced Encryption Standard; GCM is Galois Counter Mode; CERG is Cryptographic Engineering Research Group; GMU is George Mason University; CCRG is Coding and Cryptography Research Group; NTU is National Technical University; CLOC is Compact Low-overhead Counter Feedback Mode; SILC is Simple Lightweight Counter Feedback Mode. Spec is specification.

After demonstrating vulnerabilities of the unprotected cipher implementations to DPA, we seek to employ countermeasures to mitigate vulnerabilities. Although there are several types of DPA countermeasures, including algorithmic and non-algorithmic countermeasures, we limit our research to one particular algorithmic countermeasure called threshold implementations (TI).

TI, introduced in [23], involve the separation of sensitive data into "shares". Computations are subsequently performed on individual shares, in order to prevent an adversary from being able to simultaneously have access to all sensitive data. TI are based on the concepts of secret sharing and

multi-party communications, where the communications of a single party do not provide sufficient information to reveal the contents of an entire message [23–25].

TI are designed to provide security in CMOS technologies which are subject to glitches, where multiple level transitions per clock cycle can be observed by an attacker and correlated to recover sensitive data [26]. In order to be provably secure against power analysis in the presence of glitches, algorithmic countermeasures constructed using TI should adhere to the following three properties, as discussed in [23]:

1. Non-completeness. Every function is independent of at least one share of each of the input variables. Defined formally, if $z = F(x, y)$, and x and y are divided into n shares, then

$$z_1 = f_1(x_2, x_3, \ldots, x_n, y_2, y_3, \ldots, y_n),$$

$$z_2 = f_2(x_1, x_3, \ldots, x_n, y_1, y_3, \ldots, y_n),$$

$$\vdots$$

$$z_n = f_n(x_1, x_2, \ldots, x_{n-1}, y_1, y_2, \ldots, y_{n-1}).$$

   Since z_i does not depend on x_i and y_i, it cannot leak information about x_i or y_i.
2. Correctness. The sum of the output shares gives the desired output. Formally,

$$z = \oplus_{i=1}^{n},$$

   where

$$z_i = N(x, y).$$

3. Uniformity. A realization of sharing $z = F(x, y)$ is uniform if for all distributions of the inputs x and y, the output distribution preserves the input distribution.

A nonlinear second-degree algebraic function, such as $z = xy$ (e.g., a 2-input `and` gate), can be shared using three TI shares, since $d + 1$ shares are required to share a function of degree d. However, as discussed in [27,28], achieving TI sharings which are simultaneously non-complete and uniform is not trivial. The uniformity property can be achieved by supplying fresh random bits (e.g., "resharing" or "remasking" randomness); however, this requires augmenting an implementation with a source of randomness, which must either be imported into the device, or generated internally at run-time. Thus, the engineer must make a decision whether to use fewer TI shares, which increases the amount of required additional randomness, or more TI shares, which results in more required logic.

Although TVLA has been used to show vulnerabilities in block ciphers, and to confirm the effectiveness of countermeasures to DPA (e.g., [29,30]), it has not previously been used to demonstrate improved resistance for a large group of authenticated ciphers. In this research, we enhance the methodology of [30] to provide the first documented methodology suitable for analyzing side-channel resistance, and evaluating the effectiveness of countermeasures against side-channel attacks (SCA), for a large number of authenticated ciphers (i.e., 11 cipher variants in this research). Our methodology uses a free and open-source SCA test bench (FOBOS), published specification for the CAESAR Hardware API for Authenticated Ciphers, associated Development Package, and publicly-available source codes for the unprotected cipher implementations in this research. As such, it should be possible for other researchers to either duplicate, or improve upon these results.

Having established a baseline of identically protected implementations of authenticated ciphers, we compare unprotected and protected implementations in terms of FPGA resources in the Spartan-6 FPGA (LUTs and slices), maximum frequency (MHz), throughput (Mbps), throughput-to-area (TP/A) ratio (Mbps/LUT), power (mW), and energy per bit (E/bit) (nJ/bit), in order to determine absolute and relative costs of protection.

The key contributions of this work are as follows:

1. While it is well-known that the implementation of countermeasures against DPA is costly in terms of resources and performance, comparison between multiple ciphers often occurs using ambiguous metrics, performed by diverse research groups, and operating on different hardware and test architectures. This work presents a methodology for the comparison of the costs of protection against 1st order DPA which are suitable for adaptation across all authenticated ciphers, and could assist evaluation and standardization committees in selection of the best candidates.
2. This work performs a large-scale analysis of 10 CAESAR candidate authenticated ciphers, with comparison to a defacto standard of AES-GCM, which provides implementation data to support evaluation of CAESAR Final Round candidates, and provides early support to the NIST Lightweight Cryptography Standardization Project.
3. In addition to providing a large-scale comparison of protected implementations of authenticated ciphers, this research provides analysis and insights of the structures of individual ciphers which could spur further research into improved DPA protection techniques.

2. Results

2.1. Protection of Authenticated Ciphers against DPA

2.1.1. ACORN

ACORN is unique among authenticated ciphers in this research in that it is the only stream cipher. ACORN can be implemented serially, or in n-bits of output generated in parallel. For this research, we evaluate an 8-bit version of ACORN (i.e., ACORN-8) available at [19]. In order to shorten the critical path and reduce the possibility of glitches occurring along long nonlinear computational paths, we execute the state update in two clock cycles instead of one. Our resulting modification includes ten 8-bit hybrid 2-/3-share TI-protected `and` functions, each of which consumes 16 random bits, including eight for reshare, and eight for refresh masking, in each iteration of the nonlinear state update, to meet requirements of the TI uniformity property. Although this requires a total of 240 random bits per iteration, only 120 random bits per clock cycle are required, since the nonlinear state update executes in two clock cycles. Additional random bits are provided by a pseudo random number generator (PRNG) which augments the protected ACORN implementation. An abbreviated representation of the ACORN state update is shown in Figure 2.

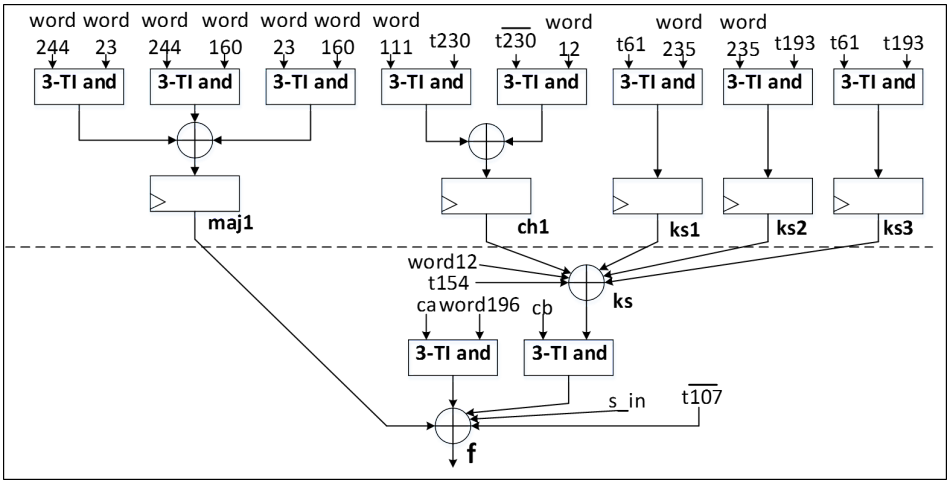


Figure 2. Simplified ACORN two-cycle state update. The dashed line indicates the boundary between clock cycles. All signals and buses represent two TI-shares, where the bus width of each single share is 8 bits. Signal names are derived from [12].

2.1.2. Ascon

Ascon uses a custom substitution-permutation (SPN) transformation based on duplex sponge modes like Monkey Duplex [13]. Since our goal is to evaluate implementations of authenticated cipher candidates prepared for the CAESAR competition, we evaluate the implementation at [18]. However, this implementation is designed using a basic-iterative architecture with large internal datapath (i.e., 64-bit blocks), and is optimized for throughput-to-area (TP/A) ratio. This results in a longer critical path of unregistered intermediate nonlinear computations, which is not ideal for protection against DPA. As an example, three-share TI protection of a nonlinear state update, occurring in a single clock cycle across the 320 bits of an Ascon state, would consume an enormous amount of logic and required randomness, since the required amount of area grows quadratically in the order of protection d.

Therefore, we implement a hybrid 2-/3- share TI-protection scheme which executes one round in seven clock cycles. We use the bitslice S-Box discussed in [13], and instantiate only one hybrid 2-/3-share 64-bit TI-protected `and` module, which uses 192 random bits per clock cycle—128 bits for the resharing from two to three shares, and 64 bits to meet the TI uniformity property. Round constant addition occurs in Stage 0, S-Box operations occur in Stages 0 through 6, and the linear permutation layer occurs in Stage 6.

The additional required randomness is provided by a 192-bit PRNG, which is integrated into the authenticated cipher, and additionally performs pre-whitening during state initialization to begin round computations with an average Hamming Weight (HW) of 0.5-per-bit. In order to facilitate a fair comparison with an analogous unprotected implementation using the same architecture, we modify the unprotected version of Ascon at [18] to use the same seven-cycle-per-round architecture. The modified S-Box is shown in Figure 3.

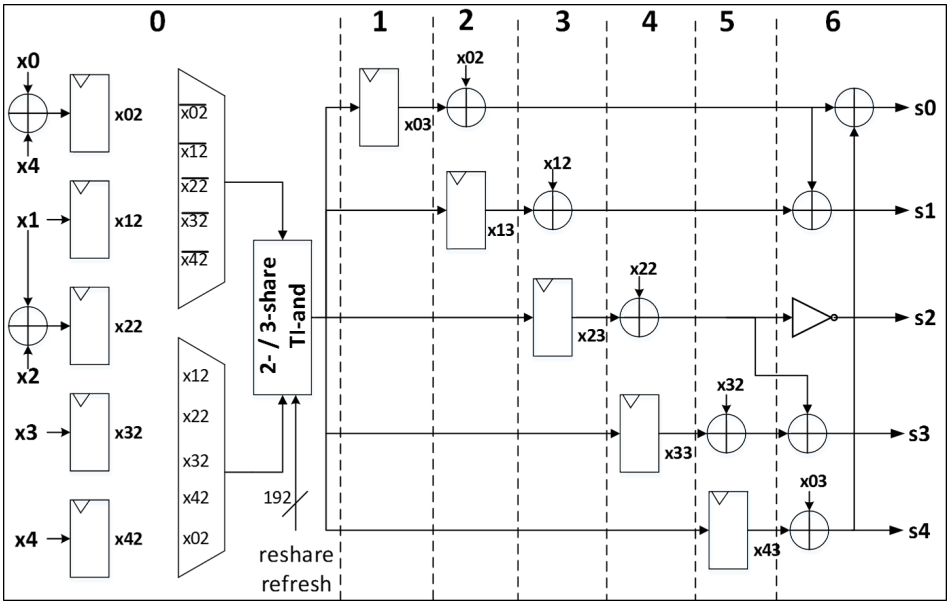


Figure 3. 7-cycle S-Box. Stages are separated by dashed lines, and numbered across the top. All bus widths are 64-bits, unless indicated.

2.1.3. TI Protection of AES in AES-GCM, CLOC, SILC, and JAMBU

TI-protected versions of AES are documented in [27–31]. We improve upon the hybrid 2-/3-share 5-stage pipelined version in [30] by upgrading the pipeline with TI-protection for round keys generated on the fly. Our TI-protected AES uses an S-Box implemented entirely with combinational logic vice look-up tables, as described in [32,33]. We use the method of Tower Fields, in which inversions in $GF(2^8)$ are represented as operations in $GF(2^4)$, which are in turn represented in $GF(2^2)$, and finally

in $GF(2^1)$, which individually have low computational costs, and are easier to protect against DPA using TI countermeasures.

Since the $GF(2^8)$ inverter portion of a TI-protected S-box is costly, we instantiate only one 3-share TI-protected 8-bit inverter. Apart from the 8-bit field inversion, all of the remaining S-Box transformations, including the affine transformations and basis conversions inside the S-Box, and round key additions, column multiplications, and row permutations outside the S-Box, are linear. We leverage a method discussed in [27] to employ a hybrid 2-/3-share TI approach, where linear calculations are performed on only two shares to save resources, and expansion to three shares occurs only when necessary to conduct nonlinear operations.

Our resulting protected design has a 5-stage pipeline, where one S-Box operation commences every clock cycle. A 128-bit round completes every 20 cycles, and a 128-bit block encryption executes in 205 clock cycles (including five cycles to prime the pipeline). The design uses 16 bits of fresh randomness for resharing from two to three shares, and two fresh remasking bits per $GF(2^2)$ multiplier and multiplier-scalar instance, resulting in a total of 40 random bits required for each S-Box (i.e., per clock cycle). The required refresh randomness is supplied by a PRNG integrated into the AES core. The S-Box computation, distributed over five pipeline stages, is shown in Figure 4.

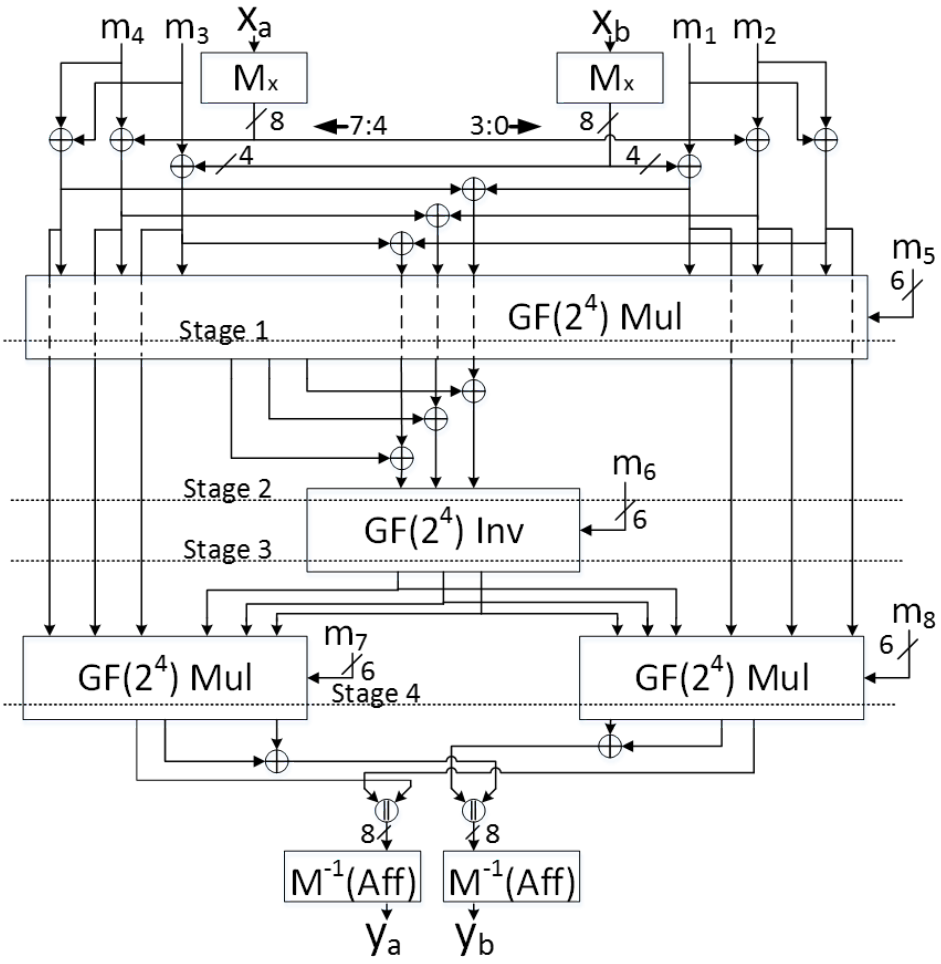


Figure 4. 8-bit 5-stage pipelined TI-protected S-Box. Dotted lines denote stage boundaries; dashed lines indicate pass-thru, i.e., no interaction with component. M_x represent change to normal basis, and M^{-1} are combined affine transformation and change to standard basis. m_i indicate random refresh bits, with bus widths as indicated, adapted from [30].

The authenticated ciphers at [18] using AES as a cryptographic primitive (i.e., AES-GCM, CLOC, SILC, and JAMBU) are optimized for high-speed operations, and use a full-width AES core which executes a 128-bit block encryption in 10 clock cycles. However, it is not feasible to build a full-width TI-protected AES with basic iterative architecture, due to:

1. Quadratic increase in resources for TI-protection,
2. Large number of random refresh bits required,
3. Probability of increased vulnerability to SCA due to long paths of combinational logic along which glitches can occur.

Therefore, in order to facilitate a relevant benchmarking comparison between unprotected and protected ciphers, we replace the full-width AES with an unprotected version of our 8-bit, 5-stage pipelined AES in the unprotected implementations of AES-GCM, CLOC, SILC, and JAMBU.

2.1.4. TI Protection of the $GF(2^{128})$ Multiplier in AES-GCM

AES-GCM is different than all of the other authenticated ciphers in this study, in that it has a significant nonlinear operation outside of the cryptographic primitive, namely, multiplication in $GF(2^{128})$ modulo the polynomial $x^{128} + x^7 + x^2 + x + 1 (P(x))$. Since the TI-protection of this multiplier is bound to be costly, an interesting question is whether or not this operation should be protected to prevent vulnerability to DPA. According to [17], the secret key itself is never used in the multiplier. Rather, combinations of $Plaintext$, $Ciphertext$, AD, and block length information are processed by the multiplier.

There are known weaknesses associated with AES-GCM. One example is the Ferguson Observation, where it is possible to create a tag linearly dependent on the hash key K_H, since multiplications by 2^n are linear [34]. Another example is a 1st order DPA attack on AES in the counter mode [35]. Additionally, Belaid et al. concluded that attacking a multiplier in AES-GCM could provide knowledge of the AES secret key K_S, and determined that a designer should mask the multiplier to protect against Hamming Weight (HW) leakage in registered values [31]. The AES-GCM implementation documented in [36] discusses these potential vulnerabilities, and includes a TI-protected multiplier.

Even if we did not protect the multiplier, we would be required to combine the two shares of sensitive data leaving the AES core, perform the multiplication, and reshare the data into two shares for subsequent operations, which would require additional logic and randomness. Additionally, an unprotected multiplier would not satisfy our verification methodology, since it would fail our t-test, regardless of vulnerabilities to subsequent key recovery attacks.

Based on the above logic, we develop a low-cost 3-share TI-protected multiplier, which computes $a \cdot b \, mod P(x)$, where a and b are 128-bit operands. The multiplier executes a two-operand multiplication in 128 clock cycles. This does not affect the overall throughput for large messages, since it is still less than the 205 clock cycles required for an AES block encryption. The multiplier is low-cost in area, since multiplication in each clock cycle is only a 128-by-1 bit multiplication, followed by parallel `xor` gates to provide reduction modulo $P(x)$. The randomness for resharing from two to three shares in the multiplier is recycled from a shift register containing randomness at the input stages of the AES core provided by the PRNG. The use of recycled randomness presents a potential vulnerability for higher orders of DPA, but is practically uncorrelated to our multiplicands after sensitive data is permutated by an entire AES block encryption. The 3-share TI-protected multiplier is shown in Figure 5. A modified version of the resulting protected implementation of AES-GCM is available to the public at [18].

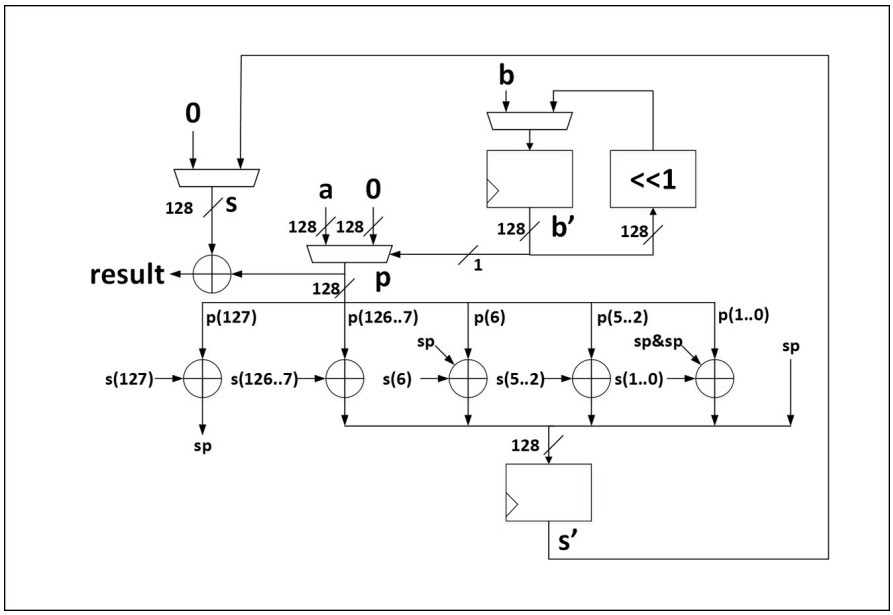


Figure 5. Three-share TI-protected GF(2^{128}) multiplier used in AES-GCM. All signals are triplicated inside the multiplier for 3-share computation.

2.1.5. JAMBU-SIMON

The authors of JAMBU-SIMON provided an implementation with unrolled architecture, i.e., four rounds per clock cycle, as their CAESAR Round 3 hardware submission [21]. We choose to directly apply 3-share TI protection to the unrolled JAMBU-SIMON implementation, although we risk information leakage along long critical paths due to the possibility of multiple hardware glitches.

As demonstrated in TI-protected implementations of SIMON discussed in [30,37], achieving efficient 3-share TI protection is relatively straightforward. This is because SIMON uses only a single 2-input 48-bit `and` gate to achieve nonlinearity, meaning that a 3-share TI of this second-degree function is achieved without the need for composite functions or cascading nonlinear gates.

Since each share lacks at least one of the component shares in its calculation, the TI non-completeness property is satisfied. The uniformity property is satisfied by considering the key shares, which are mixed into each TI-share calculation, as a source of entropy which is uncorrelated to share data [37]. Since refreshing randomness is not required in 3-share TI-protected SIMON, the DPA-resistant implementation of SIMON is relatively efficient. A sample SIMON round is shown in Figure 6.

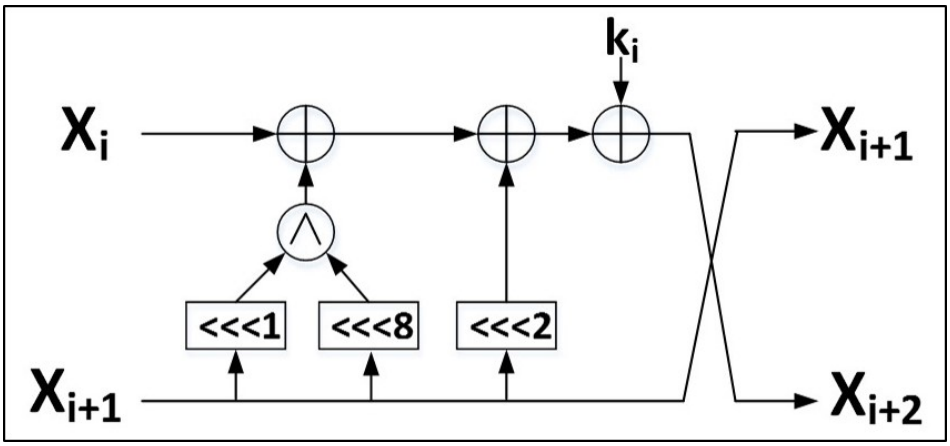


Figure 6. One round of SIMON 96/96 primitive used in JAMBU-SIMON. All bus widths are 48 bits.

2.1.6. PRESENT and LED in SILC

PRESENT and LED use the same 4-bit S-Box. The S-Box is a cubic function, but can be decomposed into two quadratic functions, which can be computed using 3-TI shares [38,39]. As discussed in [38,39], the output of the composite functions is a permutation on the input bits, meaning that the TI uniformity property is satisfied without additional random refresh bits. We choose to protect all 16 S-Boxes in parallel (plus associated S-Boxes for round key updates) in the full-width datapath (i.e., 64 bits), basic iterative architecture, as implemented in [20]. We implement strict 3-share TI-protection, and thus do not require any random reshare bits. The composite 3-share TI-protected S-Box used in PRESENT and LED is shown in Figure 7.

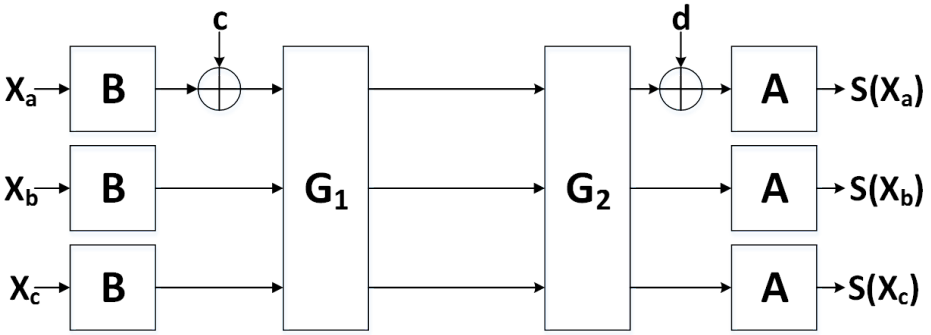


Figure 7. 3-share TI-protected S-Box used in authenticated ciphers with PRESENT and LED, adapted from [30,38,39], where A and B are matrix multiplications, c and d are constants, and G is a composite function. All bus widths are 4 bits.

2.1.7. CLOC-TWINE

Like PRESENT and LED, TWINE also uses a 4-bit S-Box of cubic degree which can be decomposed into two quadratic functions. Using Fermat's Little Theorem, as discussed in [30,40], we can compute $x^{14} \equiv x^{-1}$ in $GF(2^4)$, which decomposes into two nonlinear multipliers, and several low-cost linear computations, as shown in Figure 8.

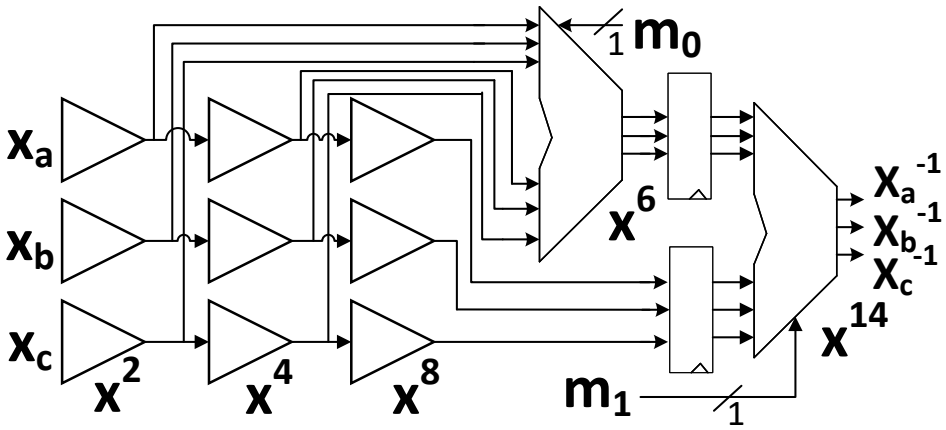


Figure 8. 3-share TI GF(2^4) inverter used in TWINE. x^2, x^4, x^8 are produced by squares; $x^6 = x^2 \cdot x^4$ is an intermediate product; $x^{14} = x^6 \cdot x^8$ is the final product; and all bus widths are 4 bits, except for single random bits m_0 and m_1, adapted from [30].

Since the outputs of the multipliers are not permutations on the input, they do not satisfy the TI uniformity property, and require the injection of additional refresh randomness if we desire to use only three TI shares. We therefore add two bits of refresh randomness per S-Box per clock cycle, for a total of 20 random bits per clock cycle, including four bits for the S-Boxes for round key updates.

The sufficiency of only two random refresh bits is demonstrated through Monte Carlo simulations, which confirm an output probability of at least 0.498 for all output bits, given an equally likely 0 or 1 at all input bits.

Like PRESENT and LED, we seek to implement a strict 3-share TI-protected implementation on the full-width (64-bit) datapath, using the basic iterative architecture, as implemented in [20].

2.1.8. Ketje Jr.

The authors of the Ketje Jr. implementation at [22] use a basic-iterative architecture with full-width datapath, presumably to maximize TP/A ratio. This means that a large number of nonlinear operations will occur in each clock cycle, and increase the chances of providing inadequate protection against DPA due to propagation of glitches. Additionally, the protection cost is potentially large. However, Ketje Jr. is the smallest of available CAESAR Round 3 Ketje specifications, and has a state of only 200 bits, which increases our chances of success.

TI protection of Ketje Jr. is relatively straightforward, since Ketje Jr. uses the $Keccak - p^*$ transformation (adapted from the $Keccak - p$ in SHA-3). Only one transformation χ is nonlinear, and protection is provided by a 3-share TI-protected `and` module. We implement a hybrid 2-/3-share TI-protection, using two shares outside the χ transformation, resharing to three shares in χ, and recombining to two shares for the remainder of the round. We use 200 bits of resharing randomness per clock cycle, which is provided by an integrated PRNG. The protected $Keccak - p^*$ is shown in Figure 9.

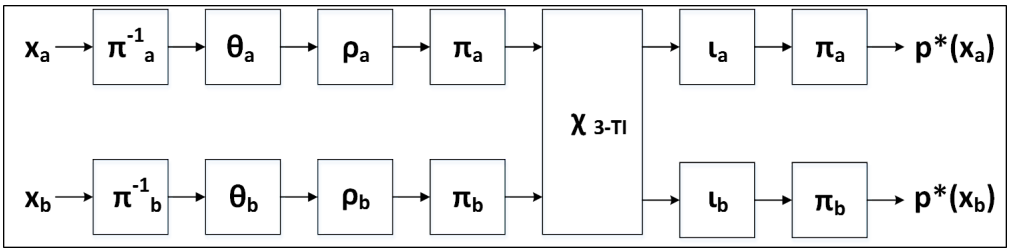


Figure 9. Hybrid 2-/3-share TI-protected $Keccak - p^*$ transformation, used in Ketje Jr.

2.1.9. Summary of Authenticated Ciphers

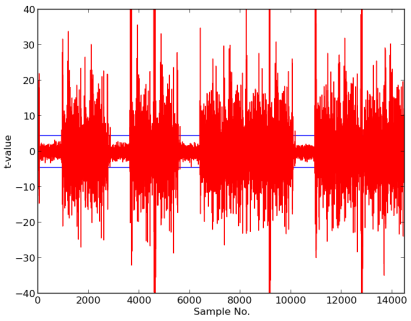
The characteristics of the authenticated ciphers investigated in this research, including the architectural choices used to achieve protection against 1st order DPA, are summarized in Table 2.

Table 2. Characteristics of authenticated ciphers examined in this research. "Formula for Throughput" is for a long message consisting of *Message* (i.e., *Plaintext* or *Ciphertext*), where $fclk$ is clock frequency. "Pipelined" abbreviated "Pipl".

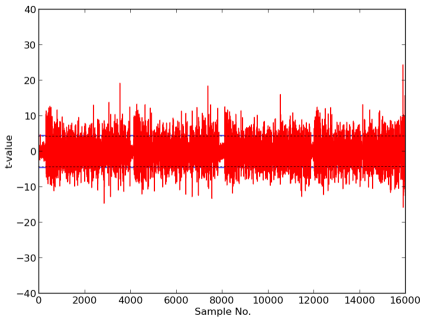
Cipher	Architecture	Rounds	Cycles Per Block	Formula for Throughput	Random Bits Per Clock Cycle
AES-GCM	8-bit 5-stage Pipl. AES, 128-cycle GF multiplier	10	218	(128/218)*fclk	40
ACORN	8-bit 2-cycle folded	-	21	4*fclk	120
Ascon	64-bit 7-cycle folded	7	49	(64/49)*fclk	192
CLOC-AES	8-bit 5-stage Pipl. AES (two cores)	10	206	(128/206)*fclk	40
CLOC-TWINE	64-bit basic-iterative	36	70	(64/70)*fclk	20
SILC-AES	8-bit 5-stage Pipl. AES (two cores)	10	205	(128/205)*fclk	40
SILC-PRESENT	64-bit basic-iterative	31	64	(64/64)*fclk	0
SILC-LED	64-bit basic-iterative	48	98	(64/98)*fclk	0
JAMBU-AES	8-bit 5-stage Pipl. AES	10	205	(64/205)*fclk	40
JAMBU-SIMON	48-bit Unrolled x4	52	13	(48/13)*fclk	0
Ketje Jr.	32-bit basic-iterative	2	2	(32/2)*fclk	200

2.2. Power Analysis of Unprotected Cipher Implementations

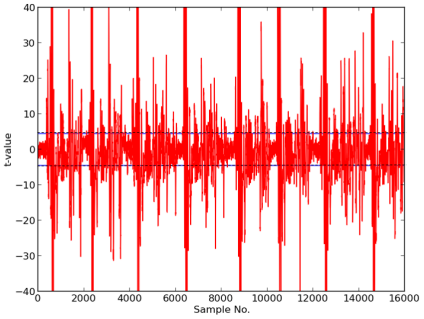
We use TVLA and FOBOS test architecture to detect information leakage in AES-GCM, ACORN, Ascon, CLOC (AES, TWINE), SILC (AES, PRESENT, LED), JAMBU (AES, SIMON), and Ketje Jr.; from which we infer vulnerability to DPA. We perform 2000 "fixed-versus-random" high fidelity traces (i.e., between 16,000 and 20,000 samples per trace), using test vectors created by `aeadtvgen.py` [11], consisting of between four and eight combinations of authenticated encryption and decryption. The *t*-tests are performed on the Digilent Nexys-3 victim board, and instantiated in the Spartan 6 FPGA (xc6slx16csg324-3). For *t*-tests, the ciphers are clocked at 781 KHz, in order to minimize capacitive and inductive effects and present a cleaner DPA target. The results, shown in Figure 10a–k, indicate significant leakage in all cipher implementations. The results are as expected for unprotected implementations.



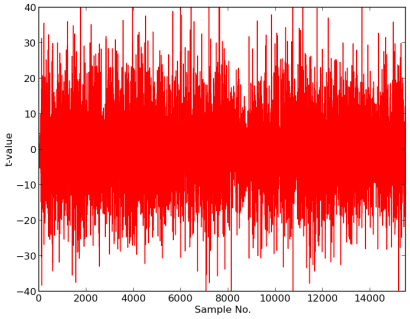
(**a**) Unprotected AES-GCM



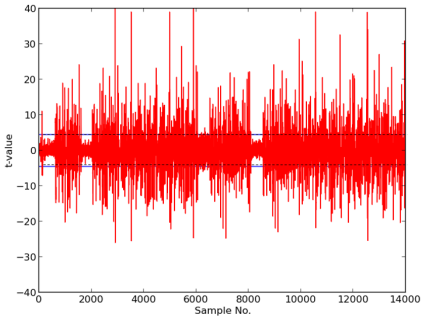
(**b**) Unprotected ACORN



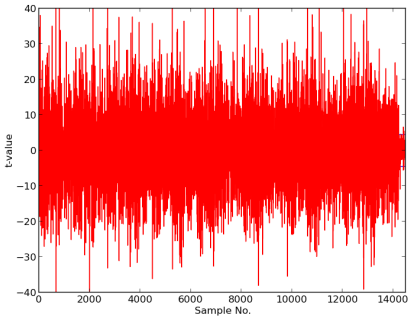
(**c**) Unprotected Ascon



(**d**) Unprotected CLOC-AES

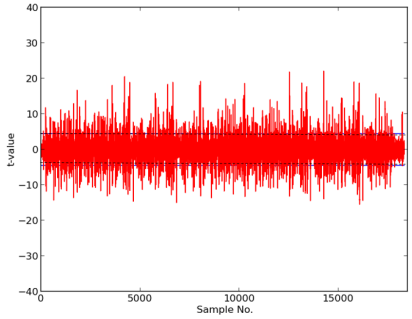


(**e**) Unprotected CLOC-TWINE

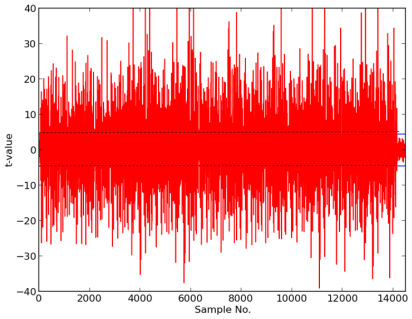


(**f**) Unprotected SILC-AES

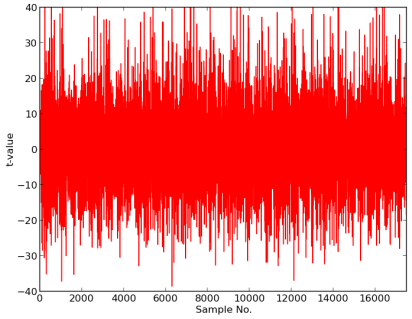
Figure 10. *Cont.*



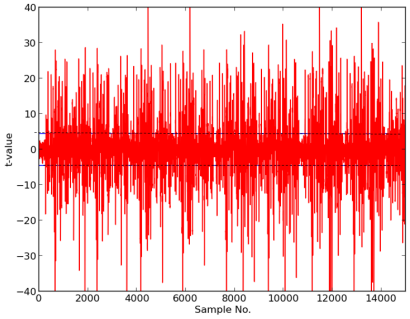
(**g**) Unprotected SILC-PRESENT



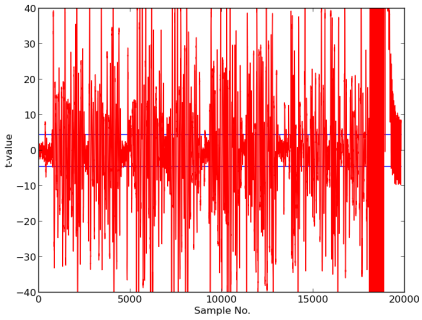
(**h**) Unprotected SILC-LED



(**i**) Unprotected JAMBU-AES



(**j**) Unprotected JAMBU-SIMON



(**k**) Unprotected Ketje Jr.

Figure 10. Results of *t*-tests on unprotected cipher implementations. Time domain (samples) are on the *x*-axis, *t*-values are on the *y*-axis. Horizontal lines denote $t = \pm 4.5$. Sample values where $|t| > 4.5$ indicates information leakage and potential vulnerability to DPA.

2.3. First Attempt at Protection of AES-JAMBU

We attempt to produce an implementation of AES-JAMBU, resistant to 1st order DPA. The JAMBU layer above the AES core consists of linear operations, and is separated into two shares, using random data supplied through the `rdi` interface. We note that padding, a required operation for JAMBU (and most authenticated ciphers), is not a linear operation. However, padding is performed in PreProcessor (and not in CipherCore, shown in Figure 11) in this implementation, and will be subsequently addressed.

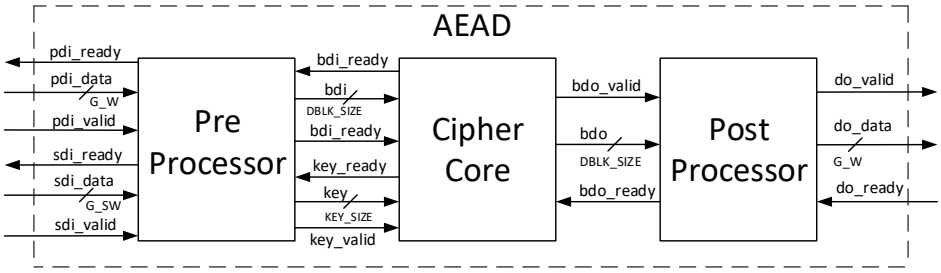
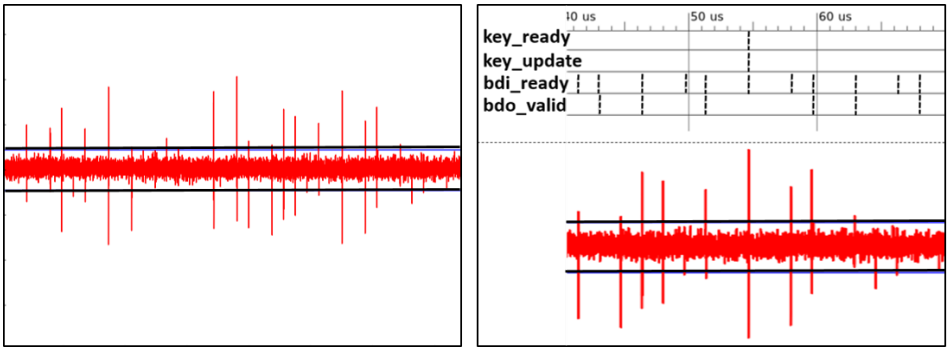


Figure 11. External interface of the authenticated cipher module (AEAD), compliant with the CAESAR Hardware API [9,10], and the internal top-level block diagram of AEAD supported by the Development Package [11].

When share separation is performed in CipherCore, the amount of initial randomness required is equal to $\#bits_{rnd} = \#bits_{bdi} + \#bits_{key}$. Note that this does not include refresh masking required during cipher operation, which is provided by an integrated PRNG. The arrival timing of random data is also important; it must be received prior to the separation of incoming data in `bdi` and `key` (defined in [41]). In our first iteration, we achieve synchronization through modification of the AES-JAMBU controller. However, this is a suboptimal approach that is not standardized for multiple authenticated ciphers, and defeats our goal of protection and analysis of a large number of ciphers (a remedy will be discussed subsequently).

We perform a *t*-test on AES-JAMBU as modified above. The results (shown at left in Figure 12) indicate that leakage is significantly reduced, yet still present at numerous discrete points throughout the time domain. We align the time domain of the *t*-test plot with significant simulation events (in Xilinx iSim), and note that the location and frequency of *t*-test spikes where $|t| > 4.5$ appears to match important I/O events (shown at right in Figure 12). We hypothesize that leakage occurs because PreProcessor and PostProcessor register unmasked data (both entering and leaving the cipher), which allows the *t*-test, or a potential attacker, to correlate sensitive data. In other words, the Development Package I/O processors themselves (as implemented in [11]) leak information.



(**a**) (**b**)

Figure 12. (**a**) *t*-test result for protected version of AES-JAMBU CipherCore; (**b**) time-domain alignment of leakage spikes with key I/O signal events in Xilinx iSim simulation of the same cipher.

2.4. Second Attempt at Protection of AES-JAMBU

To validate our hypothesis, we produce a 2-/3-share TI-protected version of Pre- and PostProcessor, by modifying the designs at [11]. This ensures that unmasked sensitive data is never registered during cipher operation. As discussed, padding is a nonlinear operation, and is performed using a 3-share TI-protected `or` module in PreProcessor.

Since randomness is required in the PreProcessor, we streamline the ingestion of randomness by designing a separate `rdi` PreProcessor, an approach which is generic and applicable to all protected authenticated ciphers using the Development Package at [11]. In this approach, the amount of

randomness required in `rdi` is $\#bits_{rnd} = (\#bits_{publicdata} + \#bits_{keydata}) \times (d-1)$, where d is the number of TI shares.

The successful t-test of JAMBU-AES using the modified AEAD, including modified Pre- and PostProcessor, and `rdi` interface, is shown in Figure 13. The updated and augmented CAESAR HW interface, with modified I/O processors, is shown in Figure 14.

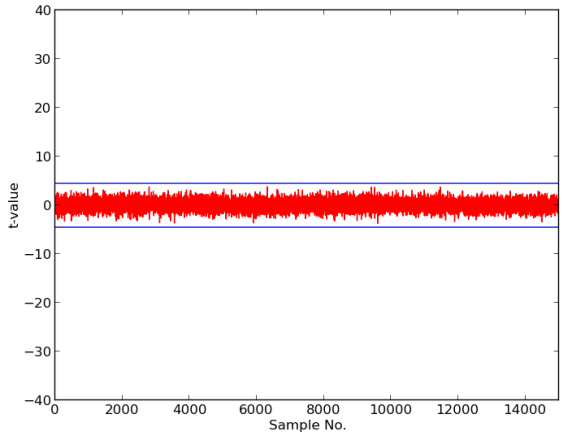


Figure 13. Successful t-test of JAMBU-AES after modification of AEAD, including Pre- and PostProcessors.

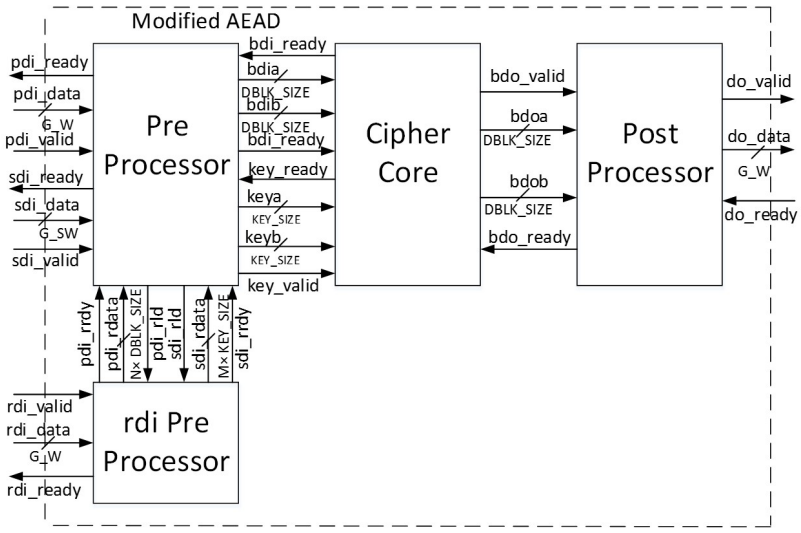
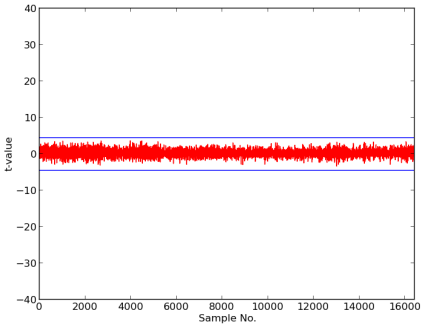


Figure 14. External interface of the modified authenticated cipher module (Modified AEAD), and the internal top-level block diagram of Modified AEAD, augmented with a `rdi` PreProcessor.

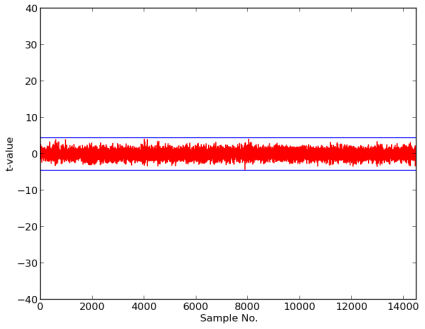
2.5. Protection of Remaining Authenticated Ciphers

All remaining authenticated cipher implementations are updated using their respective cryptographic primitives, and instantiated in the Modified AEAD module, as described above. The authenticated cipher layers above the primitive in CipherCore are protected using either 2-share TI (e.g., AES-GCM, ACORN, Ascon, CLOC-AES, SILC-AES, JAMBU-AES and Ketje Jr.) or 3-share TI (e.g., CLOC-TWINE, SILC-PRESENT, SILC-LED, JAMBU-SIMON). TI protection of cipher functionality within CipherCore is generally trivial, except that one must take care to account for occasional nonlinear operations, such as padding, and ensure that derived control functions do not leak information.

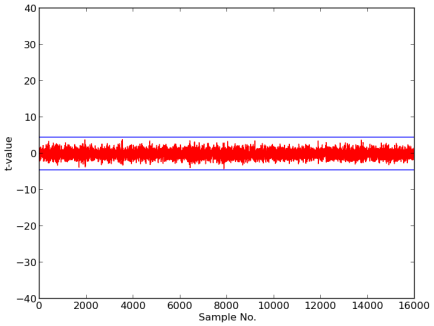
The t-test leakage detection methodology shows that the AES-GCM, ACORN, Ascon, SILC, JAMBU, and Ketje Jr. cipher implementations pass the t-test, and are resistant to 1st order DPA. The results of t-test protected implementations are shown in Figure 15a–h.



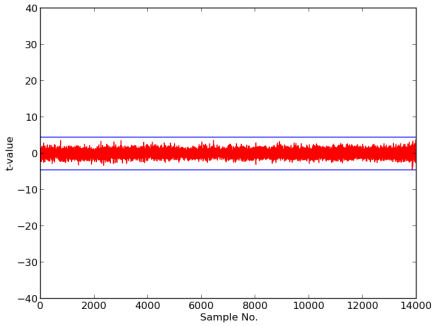
(**a**) Protected AES-GCM



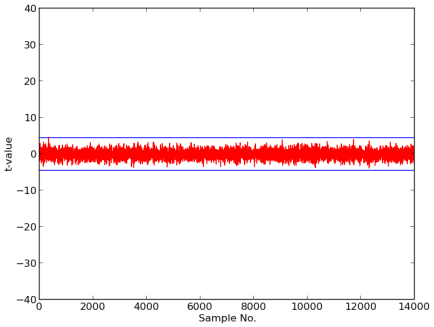
(**b**) Protected ACORN



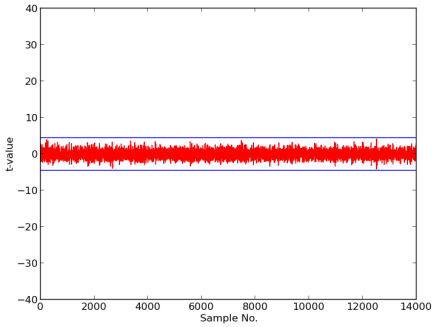
(**c**) Protected Ascon



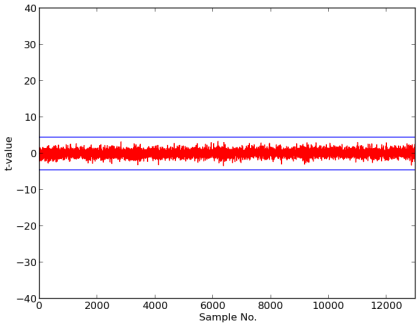
(**d**) Protected SILC-AES



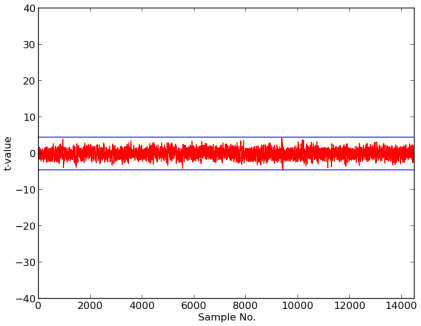
(**e**) Protected SILC-PRESENT



(**f**) Protected SILC-LED



(**g**) Protected JAMBU-SIMON



(**h**) Protected Ketje Jr.

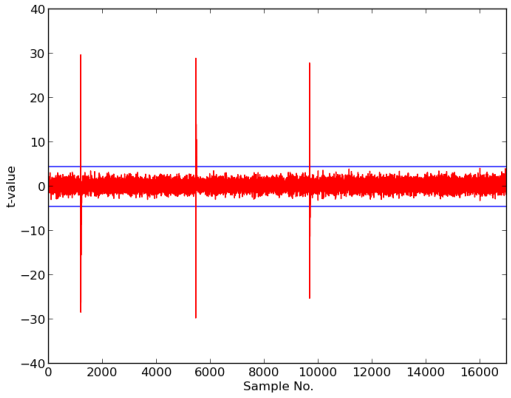
Figure 15. Results of t-tests on AES-GCM, ACORN, Ascon, SILC, JAMBU (SIMON), and Ketje Jr. authenticated cipher implementations, protected against 1st order DPA. The dashed lines denote $t = \pm 4.5$.

2.6. Conditional Protection of CLOC Implementations

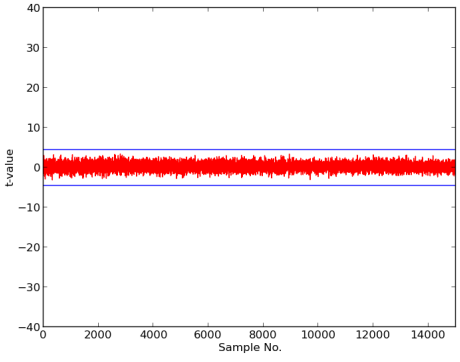
The CLOC specification contains a data-dependent decision condition. According to [14], a tweak is performed based on the most significant bit (msb) of the first word of associated data $A[1]$. Whether the algorithm is implemented in hardware or software, some decision mechanism must choose whether or not to use the tweaked result.

We hypothesize that the *t*-test will detect this data-dependent decision condition, regardless of our TI-protected implementation. To test this hypothesis, we first conduct a *t*-test of our protected implementation of CLOC-AES, with an unconstrained fixed-versus random test vector, i.e., where $msb(A[1])$ is randomly assigned a value of 0 or 1. The results, shown in Figure 16a, show that the *t*-test generally passes, but unambiguously fails at discrete points.

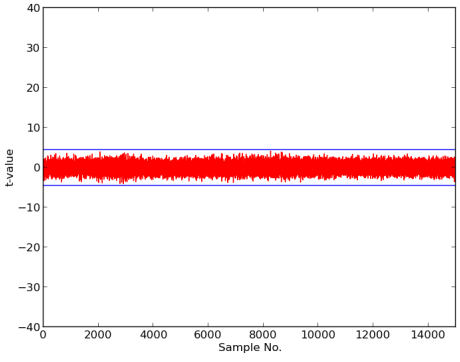
Next, we generate two sets of alternative test vectors, using `aeadtvgen.py` and the FOBOS fixed-versus-random test vector generator, where $msb(A[1])$ is constrained to be either 0 or 1, and re-run the *t*-tests. The results in Figure 16b,c show fully passing *t*-tests for both test vectors, which supports our hypothesis that the *t*-test is sensitive enough to detect this data-dependent decision.



(**a**) msb(A[1]) unconstrained



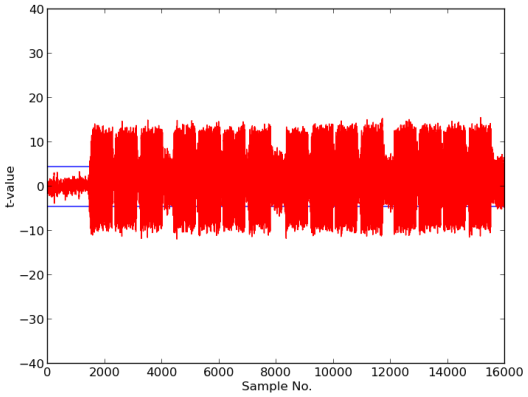
(**b**) msb(A[1])=0



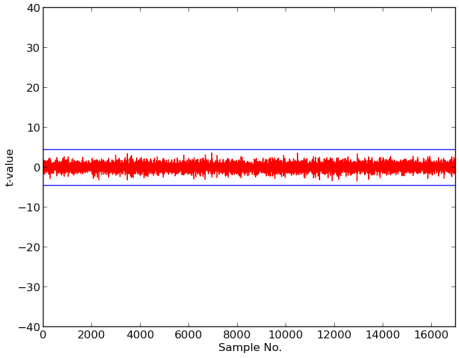
(**c**) msb(A[1])=1

Figure 16. Results of *t*-tests on protected implementation of CLOC-AES. Time domain (samples) are on the *x*-axis, *t*-values are on the *y*-axis. The horizontal lines denote $t = \pm 4.5$.

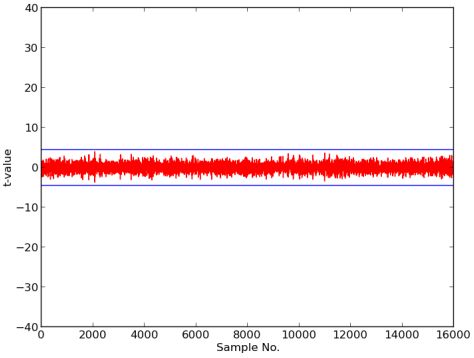
Our manipulation of *t*-test vectors to ameliorate *t*-test failures based on $msb(A[1])$ shows that we have achieved a "conditionally protected" version of CLOC-AES resistant to 1st order DPA. An analogous series of *t*-tests, with similar results, is illustrated for the CLOC-TWINE cipher in Figure 17a–c.



(**a**) msb(A[1]) unconstrained



(**b**) msb(A[1])=0



(**c**) msb(A[1])=1

Figure 17. Results of t-tests on protected implementation of CLOC-TWINE. Time domain (samples) are on the x-axis, t-values are on the y-axis. The horizontal lines denote $t = \pm 4.5$.

2.7. Benchmarking of Unprotected and Protected Cipher Implementations

We implement the unprotected and protected versions of the 11 ciphers using Xilinx ISE on the Spartan 6 FPGA. The results are further optimized for throughput-to-area ratio using the ATHENa optimization tool [42]. During optimization, we prohibit Block RAM generation, in order to ensure that area (LUTs and slices) are directly comparable. The results, in terms of LUTs, slices, frequency, throughput, and throughput-to-area (TP/A) ratio, are shown in Table 3 and displayed in Figure 18. Note that protected implementations include any required PRNG in place-and-route results.

Using the FOBOS architecture, we estimate mean power (P_{mean}) and maximum power (P_{max}) (mW) for all ciphers at 10 MHz, where the victim board is supplied by an external frequency generator. E/bit (nJ/bit) is computed as $P_{mean}(mJ/s)/TP(Mbps)$. Results are shown in Table 4, and displayed in Figures 19 and 20.

Table 3. Optimized implementation results of authenticated ciphers in Spartan-6 FPGA.

Cipher	Area		Ratio	Freq	TP	Ratio	TP/A	Ratio
	LUT	Slices	Pr/UnPr [LUT]	MHz	Mbps	UnPr/Pr (Mbps)	Mbps/LUT	UnPr/Pr [Mbps/LUT]
Unprotected (UnPr)								
AES-GCM	1947	688	-	176	103.4	-	0.0531	-
ACORN	549	269	-	226.6	906.2	-	1.6507	-
Ascon	2048	755	-	195.5	255.4	-	0.1247	-
CLOC-AES	2496	1108	-	150	93.2	-	0.0373	-
CLOC-TWINE	1536	485	-	171.2	156.5	-	0.1019	-
SILC-AES	1975	755	-	163	101.7	-	0.0515	-
SILC-PRESENT	2057	610	-	238.8	238.8	-	0.1161	-
SILC-LED	1990	699	-	203.4	132.8	-	0.0667	-
JAMBU-AES	1073	527	-	163.1	50.9	-	0.0475	-
JAMBU-SIMON	1105	311	-	137.9	509.3	-	0.4609	-
Ketje Jr.	1242	363	-	96.9	1550.4	-	1.2483	-
Protected (Pr)								
AES-GCM	4828	1870	2.48	116.8	68.57	1.51	0.0142	3.74
ACORN	2732	1032	4.98	142.7	570.6	1.59	0.2089	7.9
Ascon	6364	2062	3.11	103.1	134.6	1.9	0.0212	5.89
CLOC-AES	5900	2157	2.36	104.2	64.7	1.44	0.011	3.4
CLOC-TWINE	6467	2073	4.21	70.7	64.7	2.42	0.01	10.19
SILC-AES	4865	1899	2.46	102.8	64.2	1.59	0.0132	3.91
SILC_PRESENT	4624	1638	2.25	116.6	116.6	2.05	0.0252	4.6
SILC-LED	4780	1550	2.4	92	60.1	2.21	0.0126	5.31
JAMBU-AES	2869	1105	2.67	122.4	38.2	1.33	0.0133	3.56
JAMBU-SIMON	3140	1243	2.84	58.7	216.7	2.35	0.069	6.67
Ketje Jr.	4800	1879	3.86	59.6	954	1.63	0.1987	6.28

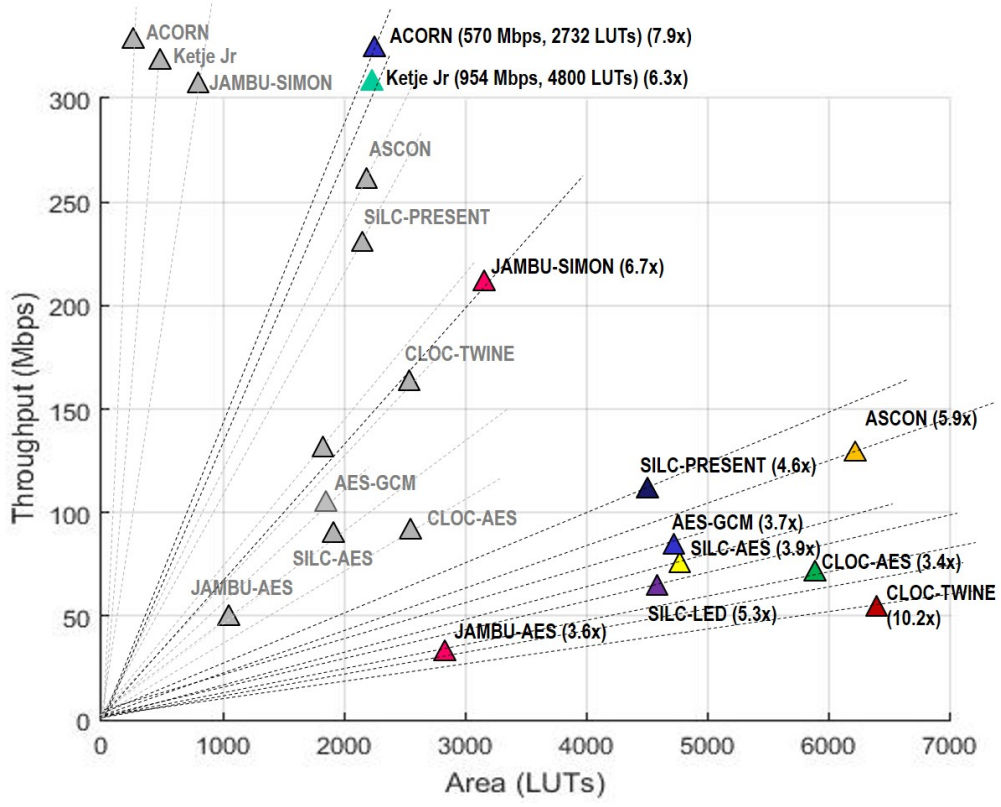


Figure 18. Throughput, area (in LUTs), and throughput-to-area (TP/A) ratios of unprotected and protected authenticated cipher implementations. Unprotected versions are shown in gray triangles, while protected versions are depicted with darker triangles. The relative increase in TP/A ratio is shown next to protected versions.

Table 4. Power and E/bit measured on Spartan-6 FPGA at 10 MHz.

Cipher	Power (mW)		Ratio	Pmax-Pmean	Energy	Ratio
	Pmean	Pmax	Pr/UnPr [mW]	% Diff	[nJ/bit]	Pr/UnPr [nJ/bit]
Unprotected (UnPr)						
AES-GCM	10.3	11.5	-	11.7	1.754	-
ACORN	7.8	8.6	-	9.9	0.195	-
Ascon	10.5	11.5	-	8.8	0.805	-
CLOC-AES	12.4	14	-	12.9	1.996	-
CLOC-TWINE	10.3	11.6	-	12.5	1.129	-
SILC-AES	10.6	13.1	-	23.6	1.698	-
SILC-PRESENT	9.7	10.7	-	9.8	0.972	-
SILC-LED	10.9	12	-	10.1	1.666	-
JAMBU-AES	9.4	10	-	6.7	3.001	-
JAMBU-SIMON	19.7	21	-	6.6	0.534	-
Ketje Jr.	22	26.5	-	20.5	0.138	-
Protected (Pr)						
AES-GCM	23.9	28.1	2.32	17.6	4.07	2.32
ACORN	16.8	18.3	2.15	8.9	0.419	2.15
Ascon	34.8	37.5	3.31	7.7	2.664	3.31
CLOC-AES	33.1	36.4	2.67	10	5.327	2.67
CLOC-TWINE	71.6	86.2	6.95	20.1	7.848	6.95
SILC-AES	23.7	30	2.24	26.6	3.796	2.24
SILC-PRESENT	25.3	28.5	2.6	13	2.526	2.6
SILC-LED	40.2	44.5	3.7	10.6	6.162	3.7
JAMBU-AES	17.8	19.2	1.9	7.9	5.702	1.9
JAMBU-SIMON	96.5	111.2	4.9	15.2	2.614	4.9
Ketje Jr.	105.3	128.7	4.86	22.2	0.658	4.77

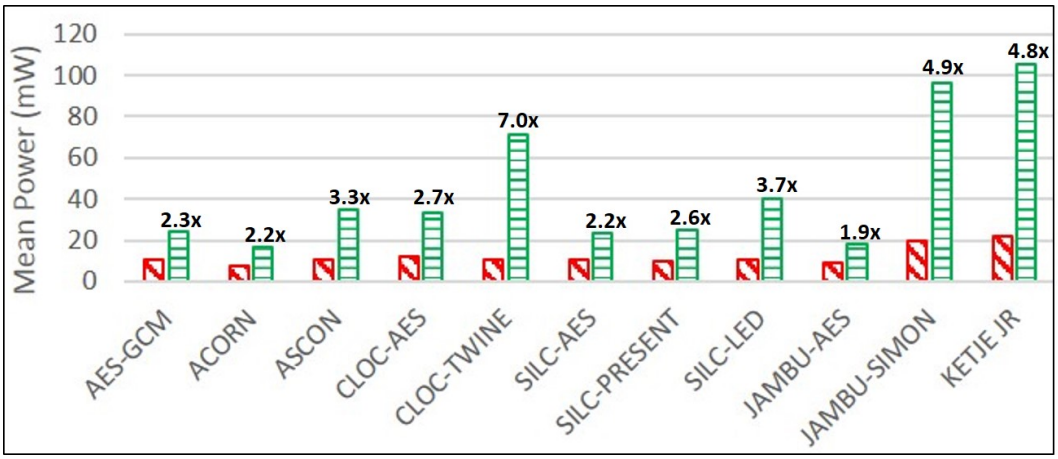


Figure 19. Mean power of unprotected (using diagonal lines) and protected (using horizontal lines) cipher implementations. The relative increase is shown above protected versions.

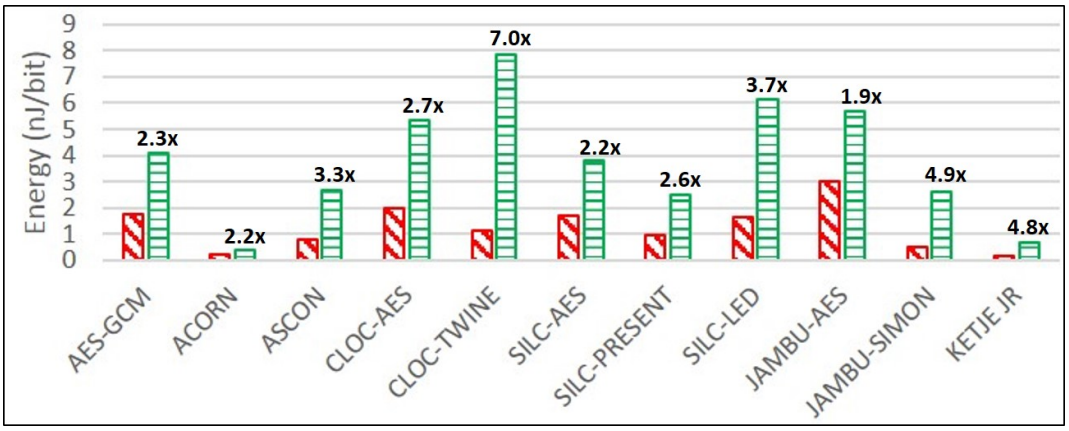


Figure 20. E/bit of unprotected (using diagonal lines) and protected (using horizontal lines) cipher implementations. The relative increase is shown above protected versions.

3. Discussion

For both the unprotected and protected implementations, ACORN is the smallest in terms of LUTs, followed by JAMBU-AES and JAMBU-SIMON. CLOC-AES and SILC-AES are larger than JAMBU-AES, since the CLOC and SILC implementations at [18] instantiate two AES cores, whereas JAMBU-AES uses only one. AES-GCM (with one AES core) is nearly the size of SILC-AES (with two AES cores), since the AES-GCM $GF(2^{128})$ multiplier compares in size to the 8-bit pipelined AES core. Ketje Jr. and Ascon are relatively large due to their full-width basic-iterative architectures.

In terms of throughput, Ketje Jr. is highest among both unprotected and protected implementations, followed by ACORN and JAMBU-SIMON. However, ACORN has the highest TP/A ratio, followed by Ketje Jr. and SIMON-JAMBU, for both unprotected and protected implementations.

ACORN, followed by JAMBU-AES and SILC-PRESENT, have the lowest mean power consumption, as measured on the Spartan-6 FPGA at 10 MHz. For protected versions, ACORN uses the lowest mean power, followed by JAMBU-AES and SILC-AES.

Protected implementations resistant to DPA are generally not "constant-power" implementations. However, a minimal difference between P_{mean} and P_{max} is desirable, from both an engineering standpoint, and for reducing potential vulnerability to power analysis attacks. Ascon, JAMBU-AES, and ACORN have the lowest difference between P_{mean} and P_{max}, while SILC-AES, Ketje Jr.; and CLOC-TWINE have the greatest difference.

Ketje Jr. is the most energy-efficient of the unprotected cipher implementations, followed by ACORN and JAMBU-SIMON. For protected ciphers, ACORN is the most efficient, followed by Ketje Jr. and SILC-PRESENT.

The average number of LUTs increases by a factor of 3.1, and the throughput decreases by a factor of 1.8, when comparing unprotected to protected implementations. The reduction in throughput results from a 1.8 factor decrease in average maximum frequency, which is due to increase in critical path and routing congestion in the protected versions. The average TP/A ratio of the protected implementations decreases by a factor of 5.6 compared to the unprotected versions. The average power and E/bit of protected implementations increase by a factor of 3.4 compared to unprotected implementations. However, the growth factor for area, and reduction factors for TP and TP/A ratios (respectively) for individual protected cipher versions vary widely. In terms of area (LUTs), protected versions of SILC-PRESENT, CLOC-AES, and SILC-LED have the lowest growth factors over unprotected versions, while ACORN, CLOC-TWINE, and Ketje Jr. have the highest growth factors. The reason for high area growth factors is a combination of architecture required for protection against DPA, and additional required randomness. For example, the high growth factor in ACORN is due to the addition of a

PRNG capable of sourcing 120 random bits per clock cycle, the size of which is comparable to the area of the protected ACORN not including the PRNG.

In terms of throughput, the lowest reduction ratios for protected cipher implementations are for JAMBU-AES, CLOC-AES, and AES-GCM, while the highest reduction ratios are for CLOC-TWINE, JAMBU-SIMON, and SILC-LED. Since architectures for protected and unprotected versions are analogous, this means that DPA protection most negatively affects the combination of critical path and routing congestion for CLOC-TWINE, JAMBU-SIMON and SILC-LED, and least affects JAMBU-AES, CLOC-AES, and AES-GCM. If we expand lowest reduction cost to fourth place, we note that SILC-AES and ACORN have nearly equivalent costs. This shows that the 8-bit pipelined AES core itself has a relatively low cost of protection against DPA.

In terms of throughput-to-area (TP/A) ratio, the lowest reduction ratios for protected cipher implementations are CLOC-AES, JAMBU-AES and AES-GCM, and the highest reduction ratios are CLOC-TWINE, ACORN, and JAMBU-SIMON. If we expand highest reduction ratios to four places, Ketje. Jr. has the next highest reduction cost. This shows that the best three overall performing protected ciphers (Ketje Jr.; ACORN, and JAMBU-SIMON) also have the highest relative protection costs. CLOC-TWINE has the highest ratio, indicating that either our protection of the TWINE primitive, or implementation of the protected 3-share CLOC-TWINE, is sub-optimal and could be improved.

JAMBU-AES, ACORN, and SILC-AES have the lowest growth ratios in power and energy consumption comparing protected to unprotected implementations, while CLOC-TWINE, JAMBU-SIMON, and Ketje Jr. have the highest growth ratios. This is a positive result for ACORN, since the highest performing protected cipher implementation (in terms of TP/A ratio) also has a very low growth in power consumption, at least at 10 MHz. While we have already noted the possibly sub-optimal DPA protection used in CLOC-TWINE, the high power and energy growths of JAMBU-SIMON and Ketje Jr. are explained by the use of architectures optimized for TP/A ratio (i.e., full-width datapath with basic iterative architectures), since the additional overhead of TI-protected modules results in the use of more than five times the additional computations in the same clock cycle compared to unprotected versions.

Table 5 ranks all authenticated ciphers in this study, in terms of absolute and relative costs of protections, as described above.

Table 5. Rankings of ciphers in terms of absolute and relative costs of protection.

Cipher	LUT	Area Fctr.	TP	TP Fctr.	TP/A	TP/A Fctr.	Pwr	Pwr Fctr.	E/bit	E/bit Fctr.
Unprotected										
AES-GCM	6	-	8	-	8	-	4	-	9	-
ACORN	1	-	2	-	1	-	1	-	2	-
Ascon	9	-	4	-	4	-	6	-	4	-
CLOC-AES	11	-	10	-	11	-	9	-	10	-
CLOC-TWINE	5	-	6	-	6	-	5	-	6	-
SILC-AES	7	-	9	-	9	-	7	-	8	-
SILC-PRESENT	10	-	5	-	5	-	3	-	5	-
SILC-LED	8	-	7	-	7	-	8	-	7	-
JAMBU-AES	2	-	11	-	10	-	2	-	11	-
JAMBU-SIMON	3	-	3	-	3	-	10	-	3	-
Ketje Jr.	4	-	1	-	2	-	11	-	1	-

Table 5. *Cont.*

Cipher	LUT	Area Fctr.	TP	TP Fctr.	TP/A	TP/A Fctr.	Pwr	Pwr Fctr.	E/bit	E/bit Fctr.
Protected										
AES-GCM	7	5	6	3	6	3	4	4	7	4
ACORN	1	11	2	4	1	10	1	2	1	2
Ascon	10	8	4	7	5	7	7	7	5	7
CLOC-AES	9	2	7	2	10	1	6	6	8	6
CLOC-TWINE	11	10	8	11	11	11	9	11	11	11
SILC-AES	8	4	9	5	8	4	3	3	6	3
SILC-PRESENT	4	1	5	8	4	5	5	5	3	5
SILC-LED	5	3	10	9	9	6	8	8	10	8
JAMBU-AES	2	6	11	1	7	2	2	1	9	1
JAMBU-SIMON	3	7	3	10	3	9	10	10	4	10
Ketje Jr.	6	9	1	6	2	8	11	9	2	9

Fctr is Factor; TP is Throughput; TP/A is Throughput-to-area ratio; Pwr is Power consumption; E/bit is energy per bit.

In general, comparison with results from previous research is difficult because there are very few reports on comparative costs of DPA protection of authenticated ciphers. An exception is [43], where the authors construct several unprotected and DPA-protected versions of the Ascon authenticated cipher, and synthesize results in ASIC technologies. Although implementation areas (i.e., gate equivalents) are not directly comparable to FPGA results, we can examine the relative increases in area for the protected implementations. Additionally, the authors of [43] use the same protection methodologies used in our research—a 3-share threshold implementation, which facilitates fair comparison.

In [43], the authors produce one version of Ascon called Ascon-*fast*. This version has a 64-bit datapath, and computes one round in 59 clock cycles. As such, it is similar to our protected version which has a 64-bit datapath, and computes one round in 49 clock cycles. The authors observe a 3.83 factor increase in area in the protected implementation, which compares to our observation of a 3.11 factor growth in the protected implementation. One notable observation is that the authors of two different Ascon implementations (i.e., [43] and this work) have both observed a relatively high cost of protection for analogous Ascon architectures. In contrast, the authors of [43] observe only a 2.45 factor cost of protection in the serialized version Ascon-*x-low-area*, which completes one round in 512 clock cycles.

Although not directly comparable, one can examine costs of our 3-share TI-protected authenticated cipher implementations versus costs for 3-share TI-protected block cipher implementations. Studies of the protection costs of several block ciphers are published in [29,30], in which results for the AES, SIMON, and PRESENT block ciphers can be loosely compared to authenticated ciphers in our research using the same primitives, such as the CLOC-SILC and JAMBU families, as well as AES-GCM.

The resulting matrix of comparisons of area growth factors for protected implementations is shown in Table 6. Of note, the average area growth factor of all protected AES implementations, including block ciphers and authenticated ciphers, is 2.53, which is less than the average observed cost of 3.1 for all authenticated ciphers in this research. Since protected AES implementations in [29,30], and all of this research, use a similar TI-protection strategy leveraging 8-bit datapaths and field inversions in Tower Fields, one can infer that this TI-protection technique is relatively efficient.

Table 6. Comparison of area growth factors of protected implementations with previous work.

Block Cipher	Authenticated Cipher	[30]	[29]	[43]	This Work
	Ascon-*fast*	-	-	3.83	3.11
	Ascon-*x-low-area*	-	-	2.45	-
AES		2.57	2.62	-	-
	AES-GCM	-	-	-	2.48
	CLOC-AES	-	-	-	2.36
	SILC-AES	-	-	-	2.46
	JAMBU-AES	-	-	-	2.67
SIMON		3.49	4.61	-	-
	JAMBU-SIMON	-	-	-	2.84
PRESENT		3.46	3.23	-	-
	SILC-PRESENT	-	-	-	2.25

In this research, we examine implementations of CAESAR Round 3 candidate authenticated ciphers which are fully (or nearly-fully) compliant with the CAESAR HW API for Authenticated Ciphers [9,10]. The version of ACORN at [19] enables a close-to-optimal protection against DPA using threshold implementations, since it uses a small datapath width (i.e., 8 bits), and has a maximum of two cascaded `and` gates in its nonlinear state update computation path. In general, however, the implementations available at [18,20–22] are not optimized for TI protection. Specifically, they have either large datapath widths (e.g., 128 bits for AES-based ciphers, 64 bits for Ascon, SILC-PRESENT, SILC-LED, CLOC-TWINE, etc.), basic iterative architectures with multiple nonlinear operations performed in parallel (e.g., Ascon, CLOC-SILC cipher variants, Ketje Jr.), or even unrolled architectures with multiple rounds completed in a single cycle (e.g., JAMBU-SIMON).

While the above choices of architecture provide optimal throughput-to-area (TP/A) ratios, they are suboptimal when attempting TI-protection. Some reasons include:

1. Wide datapaths with multiple TI-protected gates in the same clock cycle lead to a large growth of resources (which increase quadratically in order of protection), and large power consumption, which is not optimal for IoT devices.
2. Multiple cascaded nonlinear computations, occurring in the same clock cycle, increase the probability of enabling power correlations based on glitch transitions in CMOS logic, which have the potential to leak sensitive information [26].
3. The amount of randomness (measured in random bits per clock cycle) required for resharing from two to three TI shares, or required to meet the TI uniformity property, increases with wide datapaths and with basic iterative or unrolled architectures. This increases the required output of either an internal randomness source (such as a PRNG), or external randomness provided through an interface.

Therefore, authenticated ciphers, optimized for TI protection, should be constructed with small internal datapaths (e.g., 8 or 16 bits), and with a maximum of one logic level of nonlinear functions (e.g., `and`) conducted in a single clock cycle, which could result in pipelined or folded (e.g., multi-cycle) architectures. This approach has been fully adopted for modification of AES-based ciphers (i.e., reduced datapath and pipelined architecture), and partially adopted for ACORN and Ascon (e.g., multi-cycle architectures). However, these techniques should be investigated for all authenticated cipher candidates, and is left to future research.

Future research could include investigation of additional pairs of authenticated ciphers, investigation of cipher versions which are optimized for protection against DPA, and measurement of power and energy at higher frequencies, i.e., closer to actual maximum operating frequencies. The use of attack-based testing methods (such as Correlation Power Analysis) to quantify improved resistance of protected versions to DPA (including higher orders of DPA) could provide additional insight into

the relative costs of protection of the subject ciphers. Additionally, the techniques in this research could be adapted to investigate costs of protection for future cryptographic competitions, such as for post-quantum resistant public key cryptography.

4. Materials and Methods

Our research leverages methods and methodologies applied for comparing costs of DPA resistance for block ciphers, described in [30], but expanded to apply to authenticated ciphers.

Differential Power Analysis (DPA) is used to analyze differences between observed power measurements, and power based on hypothetical contents of a sensitive intermediate variable, according to a power model. However, determining the correct power model is time consuming, can require extensive trial and error, and can be completely invalidated by changes in the associated architecture [30,44,45].

One method of analyzing cryptographic implementations for information leakage is introduced in [6,7] and further described in [46], and is called the Test Vector Leakage Assessment (TVLA). This methodology uses common statistical methods such as the Welch's t-test to determine whether two distributions are different from one another. Advantages of this leakage assessment methodology are that it locates information leakage without having to conduct a more time-consuming DPA attack, is a "black-box" testing approach, in that it does not require extensive internal knowledge of the implementation, and can quickly assess the effectiveness of countermeasures. However, it cannot be used to recover a secret key, or provide immediate information about the difficulty of a prospective DPA attack.

In TVLA, a confidence factor t is calculated as

$$t = \frac{\mu_0 - \mu_1}{\sqrt{\frac{{s_0}^2}{n_0} + \frac{{s_1}^2}{n_1}}},$$

where μ_0 and μ_1 are means of distributions Q_0 and Q_1 (to be subsequently defined), s_0 and s_1 are standard deviations, and n_0 and n_1 are the cardinality of the distributions, or the number of samples.

Assuming a normally distributed probability density function (pdf) $f(t)$, a probability of accepting a null hypothesis p is calculated as $p = 2\int_{|t|}^{\infty} f(t)dt$. We start with distributions Q_0 and Q_1, and assume the null hypothesis, i.e., that we are unable to distinguish between Q_0 and Q_1. We designate a "threshold", e.g., $|t| > 4.5$, beyond which we reject the null hypothesis. If we exceed this threshold during a t-test involving Q_0 and Q_1, we reject the null hypothesis, and conclude that the device is leaking information.

In our research, we use the so-called "fixed versus random" t-test, where we preselect some "fixed" input data D (which for authenticated ciphers consists of a test vector including *Message*, *AD*, and *Npub*), and randomly interleave the feeding of D, or random data, to the algorithm [7,46].

In order to conduct a fixed versus random t-test, we instantiate the cipher on a physical device (e.g., FPGA or microcontroller), isolate external noise sources, monitor changes in voltage or current that occur in response to varying input, capture data from thousands of repetitive traces, and perform offline statistical analysis to diagnose vulnerabilities.

In order to avoid noise and corrupted analysis, we wish to prevent external I/O during trace collection. Additionally, we require test vectors, such as *Message* and *Key*, which reflect a fixed-versus-random methodology, are suitable for thousands of repetitions, and are available at the cipher module at the start of every trigger event. These conditions are easily met for the typical block cipher, where there are only a few (e.g., 16) bytes each of *Message* and *Key* for every trace event. These few bytes of data are stored in the cipher module itself prior to trigger, or in a thin-veneer of buffers on the test board. Additionally, the cipher-test architecture interface is typically trivial, consisting of (for example), m-bit *Message*, n-bit *Key*, p-bit *Ciphertext* ports, clock, and control signals. Likewise, the only protocol events for block cipher operations are typically "start" and "done". As a result, it is usually easy for cipher developers to send their designs to a "power analysis

test shop", and assume that the tester will be able to adapt the block cipher to their test architecture. The above assumptions, however, do not hold for authenticated ciphers. In order to detect all possible leakage in an authenticated cipher, one should test a variety of sequences of operations, including key initialization, *Npub* and *AD* processing, authenticated encryption and decryption, and *Tag* generation and verification. This requires a test vector potentially thousands of bytes long, interlaced with protocol that describe the entire range of permitted authenticated cipher operations. A sample authenticated cipher test vector is shown in Figure 21.

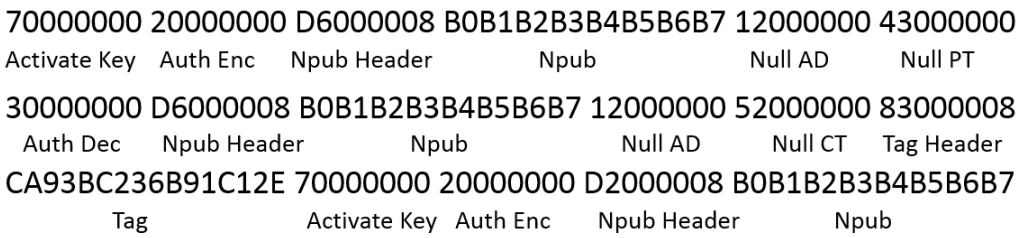


Figure 21. Sample authenticated cipher test vector.

This long test vector must be provided to the victim board (but remain outside the cipher) prior to the trigger event. In contrast to the block cipher, long test vectors will arrive and depart the cipher unit during the trace event, but should not enter or exit the victim board during the event. Additionally, an authenticated cipher must have a more-complex external interface to encompass the range of possible operations. It is not reasonable to expect that a laboratory engineer could adapt each individual custom-designed authenticated cipher interface to a power analysis test bench, and, if so, the expense in time and resources would preclude performing a large-scale analysis of DPA-resistance of multiple authenticated ciphers.

The adoption by the CAESAR Committee of the CAESAR HW API for Authenticated Ciphers enables the definition of an interface which is applicable to all CAESAR-candidate authenticated ciphers, and is readily compatible with our DPA test bench and benchmarking processes. This API, available at [9,10], includes a protocol for all required authenticated cipher operations, as summarized above. The API also specifies an AXI-compatible external interface, shown in Figure 11, and further described in [41]. Additionally, the Development Package for the CAESAR HW API contains a test vector generator, `aeadtvgen.py`, which facilitates construction of deterministic and comprehensive test vectors required for our power analysis [11].

We adapt the Flexible Open-source Workbench for Side-channel analysis (FOBOS) to perform TVLA for side-channel leakage detection on authenticated ciphers. FOBOS is designed to measure resistance to power analysis side-channel attack (SCA) and evaluate the effectiveness of countermeasures [8]. It is trigger-activated, captures power analysis data in a specified window using an oscilloscope, and stores data offline for post-run analysis in a personal computer (PC). FOBOS uses separate control and victim boards, where the control board interfaces with host personal computer and external peripheral devices, and the victim board instantiates the device under test (DUT).

In our instance of FOBOS, the oscilloscope used is the Agilent Technologies DSO6054A (Santa Clara, CA, USA), and the control and victim boards are the Digilent Nexys-3 (Seattle, WA, USA) with Xilinx Spartan 6 FPGA . However, the components of FOBOS are built in a modular fashion so that the entire experimental setup can easily be adapted for different control and victim FPGA boards, oscilloscopes, and attack techniques.

For authenticated ciphers, the FOBOS DUT victim wrapper is configured with separate FIFOs (First-in, First-out) corresponding to the data ports prescribed in [9], including public data interface `pdi`, secret data interface `sdi`, and data output `do`. A fourth FIFO is aligned to the random data interface `rdi`, which augments [9] to provide random data necessary for initial masking of public and secret data in protected ciphers. The FOBOS architecture, updated for authenticated ciphers, is shown

in Figure 22. The baseline FOBOS software suite, including acquisition and offline side-channel analysis packages, is coded in Python and is available for download at [8].

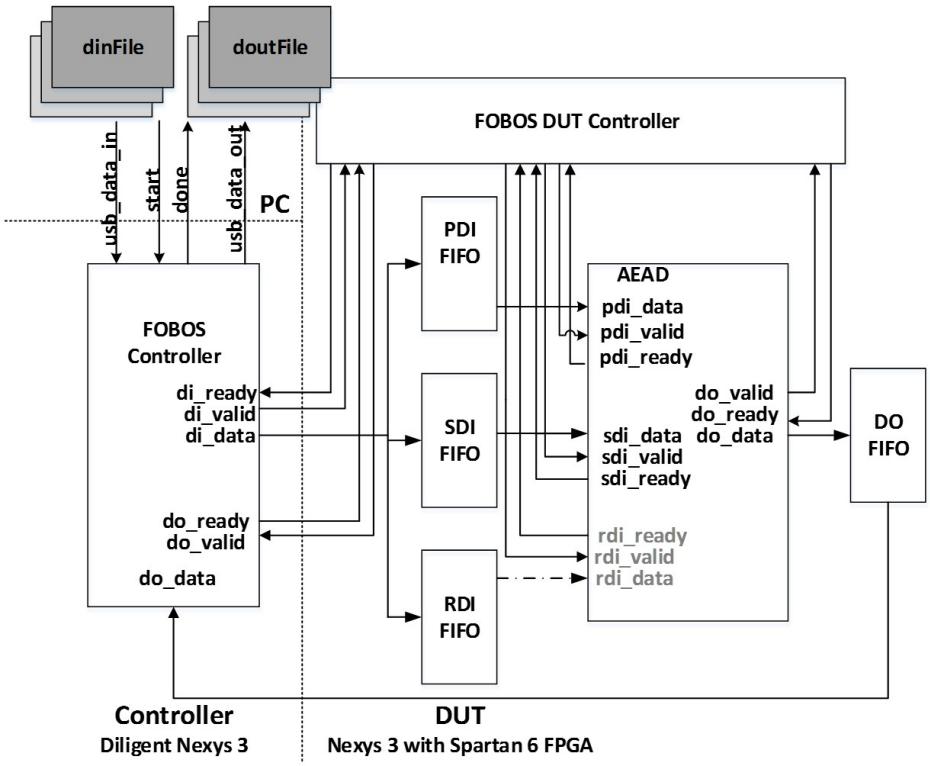


Figure 22. FOBOS architecture modified for *t*-test leakage detection on authenticated ciphers.

The procedure for performing *t*-tests on authenticated ciphers using FOBOS is summarized below:

1. The test vector `dinFile.txt`, created by `aeadtvgen.py`, is pre-formatted using a FOBOS parsing utility. It contains thousands of consecutive vectors of randomly-interleaved fixed or "random" data, where random data is substituted for all instances of $Npub$, AD, $Message$, $Ciphertext$, and Tag. The test vectors are wrapped in a layer of FOBOS-specific protocol, which determines their FIFO address on the victim board.
2. Two separate bitstreams, FOBOS Controller (control board), and FOBOS DUT (which contains FOBOS DUT wrapper and victim cipher) are instantiated in hardware.
3. The acquisition process `dataAcquisiton.py` is run from the PC. Each vector is loaded by the FOBOS Controller into FOBOS DUT. FOBOS Controller provides an oscilloscope trigger upon completion of test vector loading. Power measurements, sensed by a current probe and measured in the oscilloscope, are sent to the PC for offline analysis. Data output (e.g., $Ciphertext$) from each trace is accumulated in `doutFile.txt`. Output data, although not used in the non-specific *t*-test, is valuable for ensuring proper cipher operation.
4. At the completion of all traces, the tester performs offline analysis on traces, stored in .npy format [47]. A utility routine "splits" the collected power traces into two distributions Q_0 and Q_1, according to a "fixed-versus-random" metafile created during test vector generation. The tester then runs the *t*-test utility on distributions Q_0 and Q_1, which generates a two-dimensional display of samples (corresponding to the time domain on the x-axis), and t-values, where sustained and repeatable results of $|t| > 4.5$ are considered a sign of vulnerability to DPA leakage.

We adapt the FOBOS architecture to measure power consumed by the Spartan-6 1.2V bus, e.g., VCC_{INT}, by measuring current through a 1 Ω shunt resistor. Measured current is amplified

by a TI INA225 amplifier (Dallas, TX, USA), collected by the attached oscilloscope, and transferred to a host computer for post-acquisition power computation.

Power measurements are recorded at discrete time intervals corresponding to sample rate. Between 10 and 100 traces (using various test vectors of up to 2000 bytes each) are used to generate power traces. The power measurements contain a combination of static and dynamic power at each sample, where dynamic power sourced by VCC_{INT} accounts for about 95% of total dynamic power, according to Xilinx Power Analyzer (XPA) simulations. The victim board itself, including hardware outside the DUT but instantiated in the DUT wrapper, accounts for some static and dynamic power usage, which results in some error in power measurement. However, this error is expected to be nearly constant across all evaluated authenticated cipher candidates, so that the relative difference between observed power is accurate.

During post-analysis, mean power (P_{mean}) is computed by averaging instantaneous power measurements over the entire time domain, while maximum power (P_{max}) is estimated by sampling the highest peaks during each trace. E/bit (nJ/bit) is then estimated as $P_{mean}(mJ/s)/TP(Mbps)$, where TP (throughput) is the throughput of an authenticated encryption of a long message.

5. Conclusions

In this research, we expanded the Test Vector Leakage Assessment (TVLA) methodology to enable comprehensive large-scale analysis of authenticated ciphers, in order to determine resistance to DPA side-channel attack, and to verify effectiveness of countermeasures against DPA. Our methodology, which leverages the Flexible Open-source workBench fOr Side-channel analysis (FOBOS) test bench, CAESAR Hardware API for Authenticated Ciphers, and related Development Package, confirms that unprotected implementations of AES-GCM, ACORN, Ascon, CLOC (AES and TWINE), SILC (AES, PRESENT, and LED), JAMBU (AES and SIMON), and Ketje Jr., in the Spartan-6 FPGA, have significant information leakage and are likely vulnerable to DPA.

We then constructed protected implementations of all above ciphers, and verified their improved resistance to 1st order DPA using TVLA as implemented on FOBOS. In the case of the CLOC authenticated cipher (i.e., CLOC-AES and CLOC-TWINE), we demonstrated leakage due to a data-dependent conditional decision in the CLOC specification. Although CLOC-AES and CLOC-TWINE protected implementations did not pass a *t*-test with generic test vectors, we use modified test vectors to demonstrate that conditionally-protected CLOC authenticated cipher implementations achieved improved resistance to 1st order DPA.

Our results showed that ACORN had the lowest area of the protected cipher implementations, followed by JAMBU-AES and JAMBU-SIMON. Likewise, ACORN had the highest throughput-to-area (TP/A) ratio, followed by Ketje Jr. and JAMBU-SIMON. ACORN was also the most energy efficient of the protected implementations (i.e., used the lowest energy per bit), followed by Ketje Jr. and SILC-PRESENT, according to our evaluations on the Spartan-6 FPGA at a fixed frequency of 10 MHz.

Given our large-scale analysis of multiple protected implementations of authenticated ciphers, we are able to generally characterize costs of protection against 1st order DPA. The area of protected implementations increased by an average factor of 3.1, the throughput decreased by a factor of 1.8, and the TP/A ratio decreased by a factor of 5.6, when comparing protected to unprotected implementations. The energy per bit of protected implementations increased by an average factor of 3.4 compared to unprotected implementations.

SILC-PRESENT had the lowest relative growth in area, while JAMBU-AES had the lowest reduction in throughput, and CLOC-AES had the lowest reduction in TP/A ratio, when comparing protected to unprotected cipher versions. JAMBU-AES had the lowest growth in power and energy per bit.

Our results and repeatable methodologies demonstrated in this research can be used to experimentally develop improved algorithmic side-channel protection techniques for existing and future cipher specifications.

Author Contributions: Conceptualization, W.D.; Methodology, W.D.; A.A.; and J.-P.K.; Software, W.D.; A.A.; and F.F.; Validation, W.D. and A.A.; Formal Analysis, W.D. and K.G.; Investigation, W.D. and K.G.; Resources, J.-P.K. and K.G.; Data Curation, J.-P.K.; Writing—Original Draft Preparation, W.D. and F.F.; Writing—Review & Editing, W.D.; F.F.; and K.G.; Visualization, W.D. and F.F.; Supervision, W.D.; J.-P.K.; and K.G.; Project Administration, K.G.; Funding Acquisition, K.G. and J.-P.K.

Funding: This research was funded by the National Science Foundation under Grant No. 1314540.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

AD	Associated Data
AEAD	Authenticated Encryption with Associated Data
AES	Advanced Encryption Standard
API	Applications Programming Interface
AXI	AMBA-Extensible Interface
BRAM	Block Random Access Memory
CAESAR	Competiton for Authenticated Encryption: Security, Applicability, and Robustness
CCRG	Coding and Cryptography Research Group
CERG	Cryptographic Engineering Research Group
CLOC	Compact Low-overhead Counter Feedback Mode
CMOS	Complementary Metal-Oxide Semiconductor
DPA	Differential Power Analysis
DUT	Device under test
E/bit	Energy per bit
FIFO	First-in First-out
FOBOS	Flexible Open-source Workbench for Side-channel analysis
FPGA	Field Programmable Gate Array
FSM	Finite State Machine
GCM	Galois Counter Mode
GF	Galois Field
GMU	George Mason University
HW	Hamming Weight/Hardware
I/O	Input/Output
IoT	Internet of Things
KHz	Kilohertz
LUT	Look Up Table
LWC	Lightweight Cryptography
Mbps	Megabits per second
MHz	Megahertz
msb	most significant bit
mW	milliwatt
NIST	National Institute of Standards and Technology
nJ	nanojoule
Npub	Public Message Number
NTU	National Technical University
pdi	public data input
Pipl	Pipelined
PRNG	Pseudo Random Number Generator
RAM	Random Access Memory
rdi	random data input
RTL	Register transfer level
SCA	Side-channel attack/Side-channel analysis
sdi	secret data input

SHA	Secure Hash Algorithm
SILC	Simple lightweight Counter Feedback Mode
SPN	Substitution Permutation Network
SWaP	Size, Weight, and Power
TI	Threshold Implementation
TP	Throughput
TP/A	Throughput-to-area
TVLA	Test Vector Leakage Assessment
U.S.	United States
VHDL	Very High-Speed Hardware Design Language

References

1. Diehl, W.; Abdulgadir, A.; Farahmand, F.; Kaps, J.P.; Gaj, K. Comparison of Cost of Protection Against Differential Power Analysis of Selected Authenticated Ciphers. In Proceedings of the 2018 IEEE International Symposium on Hardware Oriented Security and Trust (HOST), Washington, DC, USA, 30 April–4 May 2018; pp. 147–152.
2. Rogaway, P. Authenticated-Encryption with Associated-Data. In Proceedings of the ACM Conference on Computer and Communications Security (CCS'02), Washington, DC, USA, 18–22 November 2002.
3. CAESAR Competition for Authenticated Encryption: Security, Applicability, and Robustness. 2012. Available online: http://competitions.cr.yp.to/caesar.html (accessed on 12 September 2018).
4. Bernstein, D. Cryptographic Competitions. 2016. Available online: https://groups.google.com/forum/#!forum/crypto-competitions (accessed on 16 September 2018).
5. Submission Requirements and Evaluation Criteria for the Lightweight Cryptography Standardization Process, NIST. 2018. Available online: https://csrc.nist.gov/CSRC/media/Projects/Lightweight-Cryptography/documents/final-lwc-submission-requirements-august2018.pdf (accessed on 16 September 2018).
6. Cooper, J.; DeMulder, E.; Goodwill, G.; Jaffe, J.; Kenworthy, G.; Rohatgi, P. Test Vector Leakage Assessment (TVLA) Methodology in Practice. In Proceedings of the International Cryptographic Module Conference, Gaithersburg Area, MD, USA, 24–26 September 2013.
7. Goodwill, G.; Jun, B.; Jaffe, J.; Rohatgi. P. A Testing Methodology for Side Channel Resistance Validation. In Proceedings of the NIST Non-Invasive Attack Testing Workshop, Todai-ji Cultural Center Nara, Japan, 25 September–27 September 2011.
8. CERG. Flexible Open-Source workBench for Side-Channel Analysis (FOBOS). 2016. Available online: https://cryptography.gmu.edu/fobos/ (accessed on 12 September 2018).
9. Homsirikamol, E.; Diehl, W.; Ferozpuri, A.; Farahmand, F.; Yalla, P.; Kaps, J.; Gaj, K. CAESAR Hardware API. Cryptology ePrint Archive, Report 2016/626. 2016. Available online: https://eprint.iacr.org/2016/626.pdf (accessed on 19 September 2018).
10. Homsirikamol, E.; Diehl, W.; Ferozpuri, A.; Farahmand, F.; Yalla, P.; Kaps, J.; Gaj, K. Addendum to the CAESAR Hardware API v1.0. 2016. Available online: https://cryptography.gmu.edu/athena/CAESAR_HW_API/CAESAR_HW_API_v1.0_Addendum.pdf (accessed on 16 September 2018).
11. CERG. Development Package for Hardware Implementations Compliant with the CAESAR Hardware API, v2.0. 2017. Available online: https://cryptography.gmu.edu/athena/index.php?id=CAESAR (accessed on 16 September 2018).
12. Wu, H. ACORN, A Lightweight Authenticated Cipher (v3). 2016. Available online: https://competitions.cr.yp.to/round3/acornv3.pdf (accessed on 16 September 2018).
13. Dobraunig, C.; Eichlseder, M.; Mendel, F.; Schläffer, M. ASCON v1.2. 2016. Available online: https://competitions.cr.yp.to/round3/asconv12.pdf (accessed on 16 September 2018).
14. Iwata, T.; Minematsu, K.; Guo, J.; Morioka, S.; Kobayashi, E. CLOC and SILC v3. 2016. Available online: https://competitions.cr.yp.to/round3/clocsilcv3.pdf (accessed on 12 September 2018).
15. Wu, H.; Huang, T. The JAMBU Lightweight Authenticated Encryption Mode. 2016. Available online: http://www3.ntu.edu.sg/home/wuhj/research/caesar/caesar.html (accessed on 16 September 2018).
16. Bertoni, G.; Daemen, J.; Peeters, M.; Van Assche, G.; Van Keer, R. CAESAR Submission: Ketje V2. 2016. Available online: https://competitions.cr.yp.to/round3/ketjev2.pdf (accessed on 16 September 2018).

17. Dworkin, M. Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC. Available online: https://www.nist.gov/publications/recommendation-block-cipher-modes-operation-galoiscounter-mode-gcm-and-gmac?pub_id=51288 (accessed on 12 September 2018).
18. CERG. GMU Source Code of CAESAR Round 3 Candidates. 2017. Available online: https://cryptography.gmu.edu/athena/index.php?id=CAESAR_source_codes (accessed on 16 September 2018).
19. Huang, T. Round 3 Hardware Submission: ACORN, 2017. Available online: https://groups.google.com/forum/#!forum/crypto-competitions (accessed on 16 September 2018).
20. Iwata, T. HW for CLOC and SILC 64-bit BC. 2017. Available online: https://groups.google.com/forum/#!forum/crypto-competitions (accessed on 16 September 2018).
21. Huang, T. SIMON-JAMBU. 2017. Available online: https://groups.google.com/forum/#!forum/crypto-competitions (accessed on 16 September 2018).
22. Bertoni G. Ketje-Keyak Team. 2017. Available online: https://github.com/guidobertoni/caesar_gmu_vhdl (accessed on 12 September 2018).
23. Nikova, S.; Rechberger, C.; Rijmen, V. Threshold Implementations Against Side-Channel Attacks and Glitches. In Proceedings of the International Conference on Information and Communications Security, Raleigh, NC, USA, 4–7 December 2006; pp. 529–545. [CrossRef]
24. Shamir A. How to Share a Secret. *Commun. ACM* **1979**, *22*, 612–613. [CrossRef]
25. Yao, A. Protocols for Secure Computation. In Proceedings of the 23rd Annual Symposium on Foundations of Computer Science, Chicago, IL, USA, 3–5 November 1982; pp. 160–164. [CrossRef]
26. Mangard, S.; Pramstaller, N.; Oswald, E. Successfully attacking masked AES hardware implementations. In Proceedings of the 7th International Workshop on Cryptographic Hardware and Embedded Systems, Edinburgh, UK, 29 August–1 September 2005; pp. 157–171. [CrossRef]
27. Bilgin, B.; Gierlichs, B.; Nikova, S.; Nikov, V.; Rijmen, V. A More Efficient AES Threshold Implementation. In *Lecture Notes in Computer Science, Proceedings of the 7th International Conference on Cryptology in Africa, Marrakesh, Morocco, 28–30 May 2014*; Springer: Cham, Switzerland, 2014; pp. 267–284. [CrossRef]
28. Moradi, A.; Poschmann, A.; Ling, S.; Paar, C.; Wang, H. Pushing the Limits: A Very Compact and a Threshold Implementation of AES. In *Lecture Notes in Computer Science, Proceedings of the 30th Annual International Conference on the Theory and Applications of Cryptographic Techniques, 15–19 May 2011*; Springer: Berlin/Heidelberg, Germany, 2011; pp. 69–88. [CrossRef]
29. Sadhukhan, R.; Patranabis, S.; Ghoshal, A.; Mukhopadhyay, D.; Saraswat, V.; Ghosh, S. An Evaluation of Lightweight Block Ciphers for Resource-Constrained Applications: Area, Performance, and Security. *J. Hardw. Syst. Secur.* **2017**, *1*, 203–218. [CrossRef]
30. Diehl, W.; Abdulgadir, A.; Kaps, J.; Gaj, K. Comparing the Cost of Protecting Selected Lightweight Block Ciphers Against Differential Power Analysis in Low-Cost FPGAs. *Computers* **2018**, *7*, 28. [CrossRef]
31. Vliegen, J.; Reparaz, O.; Mentens, N. Maximizing the throughput of threshold-protected AES-GCM implementations on FPGA. In Proceedings of the 2nd International Verification and Security Workshop (IVSW), Thessaloniki, Greece, 3–5 July 2017; pp. 40–145.
32. Canright, D.; Batina, L. A Very Compact 'Perfectly Masked' S-Box for AES. *Appl. Cryptogr. Netw. Secur.* **2008**, *5037*, 446–459. [CrossRef]
33. Gaj, K.; Chodowiec, P. FPGA and ASIC Implementations of AES. In *Cryptographic Engineering*; Springer: Boston, MA, USA, 2009; pp. 235–294.
34. Ferguson, N. Authentication Weaknesses in AES-GCM, Microsoft Corporation. 2005. Available online: https://csrc.nist.gov/csrc/media/projects/block-cipher-techniques/documents/bcm/comments/cwc-gcm/ferguson2.pdf (accessed on 16 September 2018).
35. Jaffe, J. A First-Order DPA Attack Against AES in Counter Mode with Unknown Initial Counter. In *Cryptographic Hardware and Embedded Systems—CHES 2007*; Paillier, P., Verbauwhede, I., Eds.; Springer: Berlin, Germany, 2007; Volume 4727, pp. 1–13.
36. Belaid, S.; Fouque, P.; Gerard, B. Side Channel Analysis of Multiplications in $GF(2^{128})$. In *Advances in Cryptology—ASIACRYPT 2014*; Sarkar, P., Iwata, T., Eds.; Springer: Berlin, Germany, 2014; Volume 8874, pp. 306–325. [CrossRef]
37. Shahverdi, A.; Taha, M.; Eisenbarth, T. Lightweight Side Channel Resistance: Threshold Implementations of Simon. *IEEE Trans. Comput.* **2017**, *66*, 661–671. [CrossRef]

38. Poschmann, A.; Moradi, A.; Khoo, K.; Lim, C.; Wang, H.; Ling, S. Side-Channel Resistant Crypto for Less than 2300 GE. *J. Cryptol.* **2011**, *24*, 322–345. [CrossRef]
39. Kutzner, S.; Nguyen, P.; Poschmann, A.; Wang, H. On 3-Share Threshold Implementations for 4-Bit S-Boxes. Available online: https://eprint.iacr.org/2012/509.pdf (accessed on 12 September 2018).
40. Rivain, M.; Prouff, E. Provably Secure Higher-Order Masking of AES. In Proceedings of the International Workshop on Cryptographic Hardware and Embedded Systems, Santa Barbara, CA, USA, 17–20 August 2010; pp. 413–427. [CrossRef]
41. Homsirikamol, E.; Yalla, P.; Farahmand, F.; Diehl, W.; Ferozpuri, A.; Kaps, J.; Gaj, K. Implementer's Guide to the CAESAR Hardware API v2.0. 2017. Available online: https://cryptography.gmu.edu/athena/CAESAR_HW_API/CAESAR_HW_Implementers_Guide_v2.0.pdf (accessed on 16 September 2018).
42. CERG. Automated Tool for Hardware Evaluation (ATHENa). 2017. Available online: https://cryptography.gmu.edu/athena/ (accessed on 16 September 2018).
43. Groß, H.; Wenger, E.; Dobraunig, C.; Ehrenhöfer, C. Ascon hardware implementations and side-channel evaluation. *Microprocess. Microsyst. Embed. Hardw. Des.* **2017**, *52*, 470–479. [CrossRef]
44. Kocher, P.; Jaffe, J.; Jun, B. Differential Power Analysis. In Proceedings of the 19th International Conference on Cryptology (CRYPTO 99), Santa Barbara, CA, USA, 15–19 August 1999; ISBN 3-540-66347-9.
45. Kocher, P.; Jaffe, J.; Jun, B.; Rohatgi, P. Introduction to Differential Power Analysis. *J. Cryptogr. Eng.* **2011**, *1*, 5–27. [CrossRef]
46. Schneider, T.; Moradi, A. Leakage Assessment Methodology. *J. Cryptogr. Eng.* **2016**, *6*, 85–89. [CrossRef]
47. Kern, R. A Simple File Format for NumPy Arrays. 2007. Available online: https://docs.scipy.org/doc/numpy-1.14.0/neps/npy-format.html (accessed on 16 September 2018).