Face-off between the CAESAR Lightweight Finalists: ACORN vs. Ascon

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

[1] Virginia Tech, Blacksburg VA 24061, USA
[2] George Mason University, Fairfax VA 22030, USA

Abstract. Authenticated ciphers potentially provide resource savings and security improvements over the joint use of secret-key ciphers and message authentication codes. The CAESAR competition has aimed to choose the most suitable authenticated ciphers for several categories of applications, including a lightweight use case, for which the primary criteria are performance in resource-constrained devices, and ease of protection against side channel attacks (SCA). In March 2018, two of the candidates from this category, ACORN and Ascon, were selected as CAESAR contest finalists. In this research, we compare two SCA-resistant FPGA implementations of ACORN and Ascon, where one set of implementations has area consumption nearly equivalent to the defacto standard AES-GCM, and the other set has throughput (TP) close to that of AES-GCM. The results show that protected implementations of ACORN and Ascon, with area consumption less than but close to AES-GCM, have 23.3 and 2.5 times, respectively, the TP of AES-GCM. Likewise, implementations of ACORN and Ascon with TP greater than but close to AES-GCM, consume 18% and 74% of the area, respectively, of AES-GCM.

Keywords: Side-channel · DPA · CAESAR · authenticated cipher · countermeasure · FPGA · FOBOS.

1 Introduction

This article is an extended version of [15][3,4].

Authenticated ciphers offer the promise of better efficiency and higher security for modern cryptographic applications, particularly those constrained by

[3] ©2018 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.

[4] This paper is partially based upon work supported by the National Science Foundation under Grant No. 1718434, the Semiconductor Research Corporation under Grant No. 2017-TS-2772, and Air Force Research Laboratory/DARPA under Grant no. FA8650-18-1-7819.

resources, power, and energy, such as lightweight devices in the Internet of Things (IoT). Specifically, authenticated ciphers combine the cryptographic services of confidentiality, integrity, and authentication into one algorithmic construct, which is often less-resource intensive than separately-implemented encryption, e.g., block or stream ciphers, and message authentication mechanisms, such as keyed-hash functions.

Current and projected cryptographic competitions and standardization processes are evaluating authenticated ciphers based on several criteria. The Competition for Authenticated Encryption: Security, Applicability, and Robustness (CAESAR), was started in 2013 to develop authenticated ciphers which provide improvements over the defacto standard, AES-GCM [2]. During the evaluation of 15 candidates in CAESAR Round 3, the CAESAR committee specified use cases for which candidates for final round and final portfolio would be chosen. One of these use cases is for lightweight applications, i.e., candidate ciphers that exhibit good performance in resource-challenged devices (such as small FPGAs or ASICs), and have natural side-channel resistance [5]. In March 2018, the CAESAR committee announced the final round candidates, where ACORN and Ascon were chosen as the contenders in the lightweight category.

In August 2018, the U.S. National Institute of Standards and Technology (NIST) announced the start of a lightweight cryptography (LWC) standardization process. This process will evaluate authenticated ciphers for their performance in resource-constrained environments. Candidate submissions should additionally lend themselves to countermeasures against side-channel attacks [32].

While cryptographic algorithms are generally secure against brute-force attacks that recover sensitive information, physical implementations of cryptography are vulnerable to attacks which analyze leaking information, called side-channel attacks (SCA). One powerful side-channel attack is Differential Power Analysis (DPA), where the adversary is able to measure minute changes in power output of the device resulting from the manipulation of one or more inputs, hypothesize the contents of a secret key fragment, and conduct several electronic measurements to statistically derive the correct secret key [27, 28]. Cryptographic implementations deployed as part of lightweight devices in the IoT, often located in remote locations with little physical protection, are especially vulnerable to these types of attacks. Therefore, the fielding of authenticated ciphers with efficient and robust side-channel resistance is paramount.

LWC authenticated cipher candidates can be evaluated in many respects, including performance in hardware and cost of implementation of countermeasures against DPA. Whereas simultaneous hardware evaluations of a large number of candidates is a daunting task, the paucity of candidates in the CAESAR final round permits a more in-depth look at them. In this research, we contribute to the CAESAR final round and projected NIST LWC standardization process by comparing side-channel resistant FPGA implementations of ACORN and Ascon. We define two optimization targets for our side-channel resistant algorithms as follows:

1. *Area-equivalent*: Candidate implementations with approximately the same area as AES-GCM
2. *Throughput-equivalent*: Implementations with approximately the same throughput (TP) as AES-GCM

We also establish the following controls for our comparison:

1. All protected (i.e., side-channel resistant) implementations use the same type and order of countermeasures for DPA protection: 3-share threshold implementations (TI) [33], resistant against 1st order DPA.
2. All implementations are compliant with the CAESAR Hardware Applications Programming Interface for Authenticated Ciphers (HW API) [24].
3. All implementations are realized using the Development Package for CAESAR HW Implementations [11].
4. Countermeasures for all protected implementations are verified with Test Vector Leakage Assessment (TVLA) methodology [12, 19], using the Flexible Open-source workBench fOr Side-channel analysis (FOBOS) in the same target device: the Spartan-6 FPGA on the Digilent Nexys-3 board [10].

Our contributions are as follows:

1. We provide a direct estimation of the area and performance differences between these CAESAR finalists.
2. We provide an early evaluation of potential contenders in the NIST LWC standardization process, which explicitly prioritizes third-party analysis of submissions.

Our implementations, tested in actual hardware, using a realistic input-output environment and publicly-available Development Package and test vectors, provide a high-confidence approximation of suitable real-world applications.

2 Background and Previous Work

2.1 Authenticated Ciphers

Authenticated ciphers combine the functionality of a block or stream cipher with a keyed-hash function to deliver confidentiality, integrity, and authentication in one algorithm. Inputs consist of *Message*, Associated Data (*AD*), which is not encrypted but used for *Tag* generation, a public message number (*Npub*), and a secret *Key*. The outputs of authenticated encryptions include *Ciphertext* and *Tag*, which is a function of all input values. In authenticated decryption, the user must supply *Ciphertext*, as well as the original *Npub*, *AD*, *Key*, and *Tag*. The cipher compares the received *Tag* with a *Tag'* computed based on input values during tag verification. If $Tag = Tag'$, then the decrypted *Plaintext* is released.

2.2 Specifications of Ciphers in this Research

ACORN. ACORN (v3) is a bit-based sequential authenticated cipher, based on a stream cipher [38]. It processes one bit per step, but is very fast in both hardware and software as up to 32 steps can be processed in parallel. Length information for *AD* and *Message* are not required a priori. Additionally, the length of padding is fixed in ACORN, meaning that data is not required to be grouped into multiples of padded blocks.

ACORN has a 293-bit state using a concatenation of six Linear Feedback Shift Registers (LFSRs), shown in Figure 1. In this figure, the non-linear feedback bit f_i is a function of selected bits of the internal state. An authenticated encryption or decryption consists of the following four phases:

1. Initialization: The state is initialized with the concatenation of *Key* and *Npub* (i.e., an initialization vector), and run for 1792 steps.
2. Process Associated Data: *AD* is used to update the state, and runs for a minimum of 256 steps, even if there is no *AD*.
3. Encryption/Decryption: One *Plaintext* bit p_i, or one *Ciphertext* bit c_i updates one bit of the state S_i, using one keystream bit ks_i. The output bit is produced as $c_i = p_i \oplus ks_i$, or $p_i = c_i \oplus ks_i$, and ks_i is calculated as a function of selected bits of the internal state.
4. Finalization: a *Tag* *T* is generated using the last $|T|$ keystream state bits. During decryption, *T'* is compared to *T*, and *Plaintext* is released if *T' = T*.

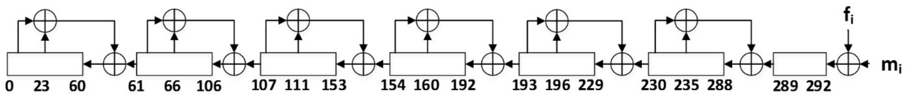


Fig. 1: ACORN top-level datapath. All buses are 1-bit unless noted.

Ascon. Ascon (v1.2) is a sponge-based cipher which shares some construction similarities with the SHA-3 contest winner Keccak [17].

Sponge constructions act on a variable length input and produce an arbitrary fixed-length output as a function of $[f, pad, r]$, where f is a permutation function, pad is a padding rule applied to all or part of the last block, and r is the bitrate. The sponge operates with a state size of bitwidth b, where $b = r+c$. The capacity c provides an additional measure of security to form the state size. The sponge operates in two phases – absorbing and squeezing.

In the absorbing phase, input bits (such as *AD* or *Plaintext*) are XORed into the state r bits at a time until the entire header or message is included in the state. With each squeeze, r bits of state are returned to form output blocks [6].

Ascon has a sponge state of 320 bits and two permutations p^a and p^b. The authors' primary recommendation for the CAESAR lightweight applications use case is Ascon-128, which includes a 128-bit *Key*, a 128-bit *Npub*, a 128-bit *Tag*, a 64-bit data block, and 12 and 6 rounds of p^a and p^b, respectively.

Ascon authenticated encryption or decryption consists of four phases – initialization, Associated Data (AD), *Plaintext* or *Ciphertext*, and finalization – as shown in Figure 2. Permutation p^a applies to initialization and finalization, and p^b applies to AD and *Plaintext* or *Ciphertext*.

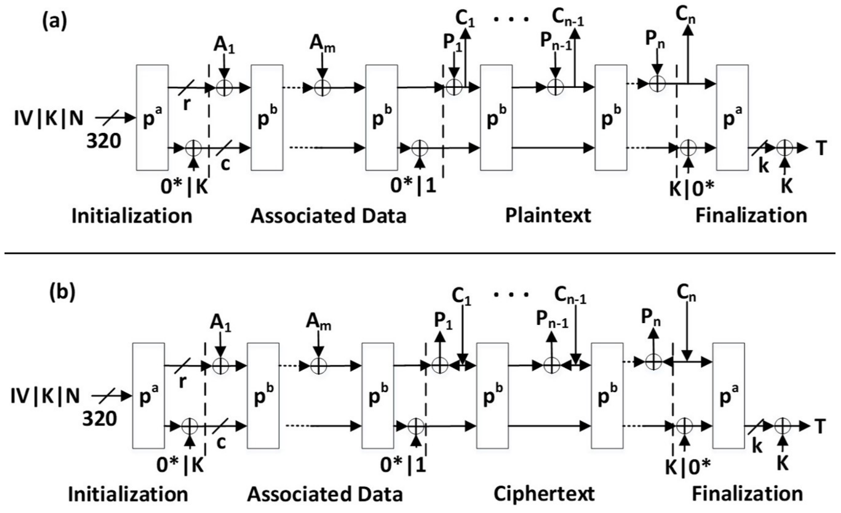


Fig. 2: Authenticated Encryption(a) and Decryption(b) in Ascon.

Substitution-permutation network (SPN)-based permutations p^a and p^b consist of three transformations p_c, p_l, and p_s such that $p = p_l \circ p_s \circ p_c$. The transformations are defined as follows:

1. Round-constant addition p_c: A one-byte round constant is added to the least significant byte of state register x_2, e.g., a 64-bit register, during each round i prescribed for the particular permutation, as $x_2 = x_2 \oplus c_i$.
2. Substitution layer p_s: 64 five-bit S-Boxes are applied across the entire 320-bit state.
3. Linear diffusion layer p_l: A diffusion layer is applied across each 64-bit state register using the transformations described in [17].

AES-GCM. AES-GCM, or Galois Counter Mode, is a current NIST standard for authenticated encryption, specified in [31]. Confidentiality is achieved using a variation of the counter mode (GCTR), where an incrementing counter is encrypted, and XORed to each new block of *Message*. The initial value of the counter is the initialization vector (IV), which is formed using a 96-bit public message number *Npub* concatenated with 31 bits of zero and the constant 1

($Npub|0^{31}|1$). Increments are additions modulo 2^{32}. The final value of *Ciphertext* is truncated to be the same length as the final block of *Plaintext*. Meanwhile, authentication is assured by the GHASH, or keyed hash performed on all blocks of authenticated data AD, *Ciphertext*, and $len(AD)|len(C)$. The result of the GHASH is a *Tag T*. In our research, T has the length of 128 bits.

The cryptographic primitive for the GCTR is AES with a 128-bit key. Only the AES encryption is required, which allows some possibilities for optimization of the AES implementation. The GHASH is a multiplication in GF(2^{128}) by the hash subkey *H*, which is initialized by an encryption of the zero vector 0^{128}, followed by an XOR with the next block of *Ciphertext*. The AES-GCM authenticated encryption flow is shown in Figure 3.

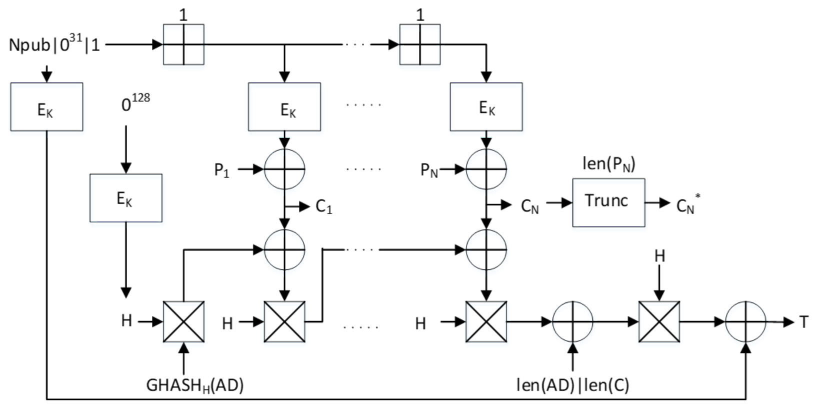


Fig. 3: Authenticated encryption flow in AES-GCM. $\times$ denotes multiplication in GF^{128}.

2.3 Previous Hardware Implementations of Ciphers in this Research

ACORN. Previous hardware implementations of ACORN include a high-speed version with 32 parallel output bits (ACORN-32) and a medium-speed version with 8 parallel output bits (ACORN-8) [25]. Both versions are quite efficient in hardware, at 1244 LUTs and 500 LUTs in the Virtex-6 FPGA, for ACORN-32 and ACORN-8, respectively [8].

Ascon. There have been several ASIC and FPGA implementations of Ascon, both unprotected and protected against DPA, using various architectural approaches. The authors of [23] detail several implementations of Ascon in 90 nm ASIC technology. These include Ascon-*fast*, which executes a permutation round in one clock cycle or faster (using various combinations of unrolled rounds); Ascon *64-bit*, which uses an ALU-like approach to perform permutations in a 64-bit datapath using microcodes in lieu of a dedicated controller, and executes a round in 59 clock cycles; and Ascon *x-low-area*, which is a serial implementation and requires 512 clock cycles per round. The authors discuss 3-share TI-protected versions of several of the above implementations, including Ascon-*fast* and Ascon *x-low-area*. The relative growth ratios in area for protected versions compared to unprotected versions for Ascon-*fast* and Ascon *x-low-area* are 3.8 and 2.5, respectively.

Later, the authors in [21] discuss versions of Ascon resistant to Higher Order DPA of order d up to $d = 15$, which are designed to reduce refresh random masking requirements using a combination of Domain Oriented Masking (DOM) techniques [22] and time-domain separation of non-linear masking operations used in software [3], in an approach called Unified Masking Approach (UMA). The authors provide results only for protected implementations; not for corresponding unprotected versions. However, the reported area for their 90 nm ASIC implementation with only one S-Box instantiated in their UMA approach, is 10.8 kGE (Kilo Gate Equivalent), and this implementation requires 402 clock cycles per round. This is nearly equivalent to the results reported in [23] for the same technology for Ascon *x-low-area*, which requires 512 clock cycles per round and occupies 9.19 kGE. Although the VHDL source code discussed in [21] is available at [20], it is not easily comparable to other protected CAESAR candidates, since it does not conform to the CAESAR HW API.

There are other unprotected hardware implementations that do not include corresponding protected versions. These include a high-speed Ascon implementation, which uses basic iterative architecture and requires one clock cycle per round. This implementation in several FPGAs (including the Spartan-6), available at [1], is similar to the Ascon-*fast* (unprotected) implementation reported in [23], except that it is fully compliant with the CAESAR HW API, and uses the high-speed Development Package at [11].

Additionally, an unprotected lightweight version of Ascon is reported in [39], which uses an ALU-like approach with accompanying microcode instructions on a 64-bit datapath, similar to that reported in [23]. This version, implemented in the Spartan-6 FPGA, completes a round in 38 clock cycles. Additionally, it is implemented using an updated lightweight version of the CAESAR HW Development Package, available at [11], which reduces the overhead of the previous Development Package designed for high-speed applications, and allows the selection of reduced external data bus widths.

AES-GCM. Since AES-GCM has existed as a defacto standard for authenticated ciphers for more than a decade, hardware and software implementations are ubiquitous in literature. For example, the authors of [37] document a side-channel protected implementation with TP of 15.24 Gbps, using 38,241 slices on the Virtex-7 FPGA. Although the authors claim resistance to 1st order DPA, no countermeasure verification results are provided. The authors also remark that such a design would require 175.24 Gbps for random refresh masking, which is infeasible in current technology.

2.4 Side Channel Attacks and Verification of Countermeasures

ACORN. The only reported verification of DPA countermeasures in ACORN is recorded in [14], and is discussed subsequently.

Ascon. In [21], the authors use TVLA to perform a leakage assessment of their protected Ascon implementations. Using a fixed-versus-random methodology, the authors demonstrate resistance to leakage up to 3rd order DPA ($d = 3$). However, the t-tests are not evaluated on actual hardware; the authors use recorded signal traces from post-synthesis simulations of the netlists.

In [34], the authors perform a DPA attack on an unprotected version of Ascon-128 in the Spartan-6 FPGA, and on a 3-share TI-protected version of a miniature version of Ascon, using only 4-bit registers (for an equivalent state size of 20 bits). Using the SAKURA-G evaluation board, the authors recover one-half of the secret key by an attack on the 64-bit $x_0^{'}$ state register using 50,000 traces. An attack on the TI-protected miniature version takes 150,000 traces to recover the 4-bit $x_0^{'}$ register, and 900,000 traces to recover the 4-bit $x_1^{'}$ register.

AES-GCM. AES-GCM differs from other ciphers in this study in that it has a significant non-linear operation outside of the cryptographic primitive, namely, multiplication in $GF(2^{128})$ modulo the polynomial $x^{128} + x^7 + x^2 + x + 1$. 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 [31], the secret *Key* itself is never used in the multiplier. Rather, combinations of *Message*, *AD*, and block length information are processed by the multiplier.

However, 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 [18]. Another example is a 1st order DPA attack on AES in the counter mode [26]. Additionally, Belaïd 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 [4]. The AES-GCM implementation documented in [37] notes these potential vulnerabilities, and provides a TI-protected multiplier.

2.5 Previous Studies Comparing Costs of DPA Countermeasures for Multiple Ciphers

A comparative study of costs of protecting 10 CAESAR Round 3 candidates was conducted in [14]. The authors' analysis showed that costs of protecting ACORN-8 and a version of Ascon-128 with a 64-bit datapath include area growth by factors of 5.0 and 3.1, respectively. However, protected implementations in this study used an earlier CAESAR HW Development Package not optimized for lightweight applications, and included the costs of an embedded PRNG, which likely skew results. Our study improves upon [14] by:

1. Performing a two-dimensional comparison of subject ciphers in terms of area- and throughput-equivalency to a known standard, AES-GCM.
2. Using a newer, more efficient HW Development Package [11].
3. Removing the artificial costs of including a PRNG in the reported area of protected implementations.

3 Methodology

3.1 Threshold Implementations

In order to establish a baseline for "side-channel resistant" implementations, we apply one type of countermeasure to all protected implementations. We choose threshold implementations (TI), which have wide acceptance as a provably-secure DPA countermeasure, and are documented at [33]. In TI, sensitive data is separated into shares, on which computations are performed independently. TI provide DPA security in the presence of glitches, provided that they satisfy three properties:

1. *Non-completeness*: Each share must lack at least one piece of sensitive data.
2. *Correctness*: The final recombination of the result must be correct.
3. *Uniformity*: An output distribution should match the input distribution.

TI which are both non-complete and uniform are often challenging. We apply a typical method for ensuring uniformity, which is to refresh TI shares after non-linear transformations using additional randomness.

As a constant in all of our protected implementations, we use a hybrid 2-share / 3-share approach, where all of the linear transformations in each cipher are protected using two shares, and expand to three shares only for non-linear transformations. We compress our shares to two shares following each non-linear operation. This increases our randomness requirements, as we are required to provide for both "resharing" and mask "refreshing," but simplifies our design tasks.

3.2 Verification of Improved Resistance of Countermeasures

After deploying our countermeasures against DPA, we seek to verify that countermeasures actually improve resistance against DPA. One method to accomplish this task is attack-based testing, i.e., measuring the delta in the number of "traces" required to recover a n-bit key fragment, before and after the inclusion of countermeasures (e.g., [34]).

However, attack-based testing requires development of a power model and attack strategy, which are highly dependent on changing architecture, and are timely endeavors. Therefore, we leverage the TVLA methodology, introduced in [12, 19] and clarified in [35]. Using the Welch's t-test, this leakage detection methodology rapidly determines whether or not we can distinguish between two populations, e.g., Q_0 and Q_1. In our case, we leverage a "fixed versus random" non-specific t-test, where we randomly interleave either a fixed test vector, or randomly-generated test vectors possessing the same length and protocol. Using means and variances of our fixed and random populations, we compute a figure of merit t. If $|t| > 4.5$, we reason that we can distinguish between the two populations, and that our design is "leaking information."

A t-test is a quick, rough-estimate of leakage, that does not tell us if a DPA attack will succeed or fail, and cannot recover sensitive information. As discussed in [36], it is dangerous to rely solely on a t-test for security, as there can be false-positives and false-negatives. However, it is an appropriate method for our comparison of ACORN and Ascon, in that we are able to plausibly establish a baseline of identically-protected implementations in order for fair benchmarking.

3.3 CAESAR HW API and Lightweight Development Package

Our FPGA cipher implementations are fully compliant with the CAESAR HW API, and use the newly-released Development Package for lightweight applications. This approach is advantageous because:

1. The use of a single API, with conforming interface and protocol for all candidates, is an important step in ensuring fair evaluations.
2. Our use of validated development tools and ready-made test vectors, publicly-available at [11], provides an optimal environment for third-party verification and improvement of our efforts.

A summary of an authenticated cipher constructed with the Development Package is shown in Figure 4. In this framework, the user develops an application in CipherCore, and interfaces with the provided PreProcessor and PostProcessor for input and output, respectively. The three modules are enclosed in the AEAD module, which is the nucleus for authenticated cipher benchmarking and comparison.

3.4 Leakage Detection using FOBOS

To conduct TVLA on unprotected and protected implementations, we use the Flexible Open-source workBench fOr side-channel analysis (FOBOS), available at [10]. In our instance of FOBOS, we instantiate victim algorithms on the Spartan-6 FPGA in the Digilent Nexys-3 board. Our procedure for conducting t-tests on authenticated ciphers is summarized as follows:

1. Test vectors are generated using development tools at [11]. They are wrapped in a layer of FOBOS protocol and stored in an attached PC. In order to generate random test vectors, instances of *Npub*, *AD*, *Message*, and *Tag* are replaced with random values of equal length.

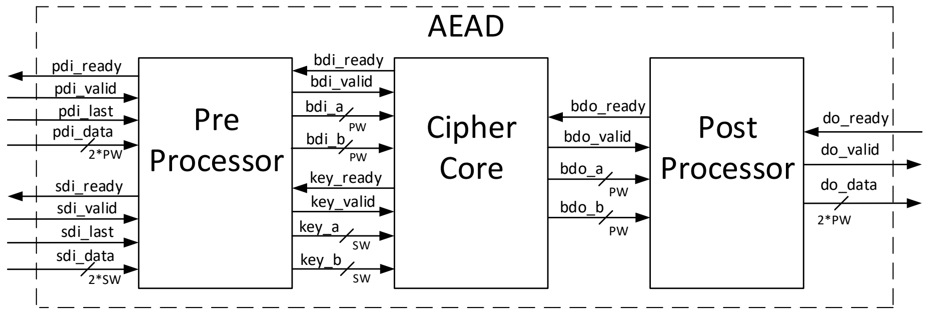


Fig. 4: Implementation of an Authenticated cipher developed using the CAESAR HW Development Package.

2. Randomness required for initial share separation is generated in software, and applied to test vectors prior to start of trace acquisition.
3. Upon acquisition start, test vectors (500 - 2000 bytes long) are processed by the victim cipher. A trigger synchronizes sample acquisition (15000 - 20000 samples per trace) with an Oscilloscope; trace results are stored off-line in the PC.
4. Upon completion of trace acquisitions, t-tests are processed off-line using analysis software available at [10]. T-test results can be processed either graphically or analytically using the FOBOS toolset – we present graphical results in this research.

In our research, any required randomness for protected implementations is provided from low-grade pseudo-randomness, consisting of a variable number of concatenated 16-bit LFSRs. This PRNG is located in the FOBOS wrapper; it is physically instantiated in the victim board but outside of AEAD (see Figure 4). Therefore, any costs of the PRNG are not included in cipher benchmarking, but are included as part of power and energy calculations. PRNGs are initiated using a random seed, generated in software, prior to the start of every trace. The FOBOS framework as used in this research is depicted in Figure 5.

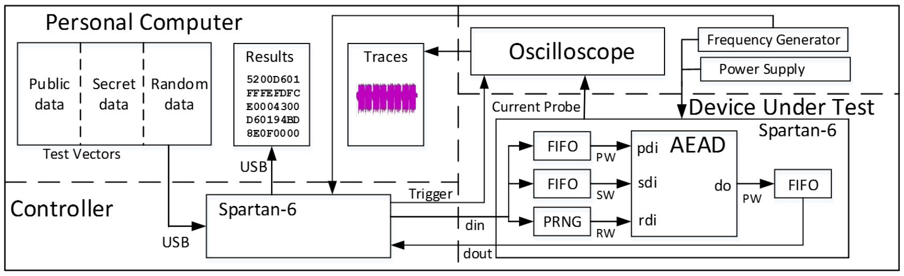


Fig. 5: FOBOS framework [16].

4 Construction of Protected Implementations

4.1 ACORN

Given the flexibility of ACORN, we are free to implement any version from ACORN-1 (serial) up to ACORN-32. In order to select design targets for protected ACORN that are either area-equivalent or TP-equivalent to AES-GCM, we estimate final results based on [14]. In this case, the authors report an area of 2732 LUTs in the Spartan-6 FPGA for a protected version of ACORN-8. However, the implementations in [14] included large PRNGs, whereas we do not include PRNG cost. Since the area of protected AES-GCM in [14] is relatively large (4828 LUTs), we target the largest version of ACORN: ACORN-32. As a TP target, we note that TP of ACORN-8 reported in [14] is 570.6 Mbps. Dividing 570.6 Mbps by 8 results in 71.4 Mbps, which is close to the AES-GCM TP of 68.8 Mbps reported in [14]. Therefore, we aim for ACORN-1 as our TP-equivalent case.

We build on the TI-protection scheme described in [14]. We choose to execute the state update in two clock cycles instead of one, in order to distribute the non-linearity across two clock cycles. For our ACORN-32, we instantiate ten 32-bit hybrid 2- / 3- share TI-protected `and` modules, each of which consumes 64 random reshare, and 32 random refresh bits, to maintain the TI uniformity during each call. When distributed over two clock cycles, this results in an average of 480 random bits per clock cycle. In ACORN-1, there are ten 1-bit TI-protected `and` modules, which consume a total of only 20 random reshare, and 10 random refresh bits per state update. In a two-cycle architecture, only 15 random bits are required per clock cycle. The ACORN protected state update is shown in Figure 6.

It is worth noting that the CAESAR HW API [24] is not currently defined for external bus widths less than 8 bits. Therefore, although internally "serial," ACORN-1 communicates with external clients via an 8-bit bus.

4.2 Ascon

We must likewise choose AES-GCM area-equivalent and TP-equivalent targets for a protected Ascon. Ascon implementations with full-width datapaths, i.e., 64-bit block size in the case of Ascon-128, and basic-iterative architectures (i.e., one clock cycle per round), are not ideal for protection against DPA. As discussed in [21], it is imperative to analyze the parts of the algorithm that are susceptible to glitches and separate calculations into smaller independent hardware modules. In order to minimize resources required for a 3-share TI-protected `and` module, reduce required randomness, and reduce vulnerability due to long logic paths vulnerable to glitching, we implement a hybrid 2- / 3- share TI-protected Ascon permutation, which can execute one round in five clock cycles, assuming a 64-bit permutation datapath. We use the bitslice S-Box discussed in [17], and instantiate only one hybrid 2- / 3- share TI-protected `and` module.

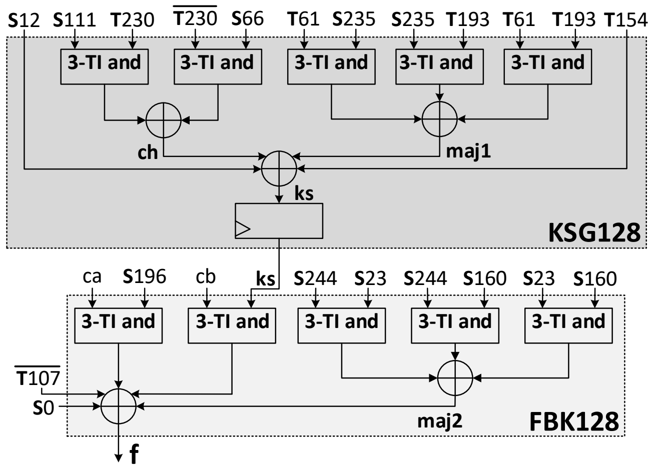


Fig. 6: ACORN Protected State Update. All buses are 1-bit unless noted.

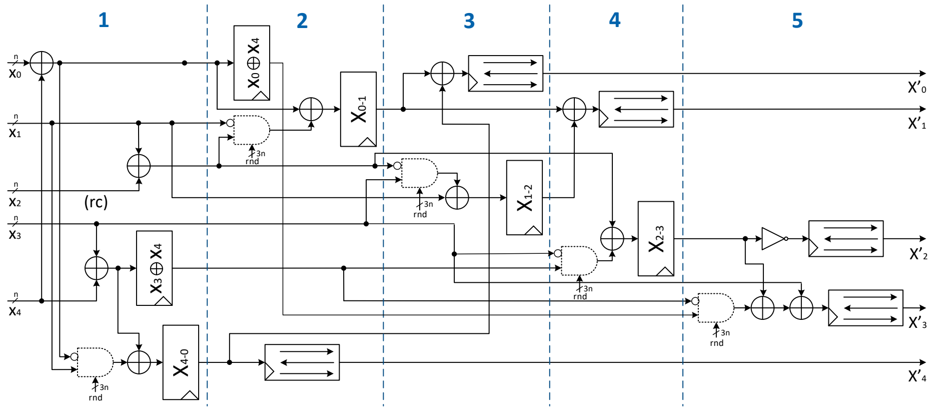


Fig. 7: Ascon-large 5-cycle Permutation. Bus width n is 64 bits.

Such an approach is advantageous when we have multiple design targets, given that we can "modulate" the size of the permutation by dividing the datapath in two (i.e., 64 bits to 32 bits) and doubling the number of clock cycles (i.e., 5 cycles to 10 cycles per round).

The largest Ascon we can make using this strategy is the 64-bit/5-cycle version, called Ascon-large. Given the reported area in [14] of 6364 LUTs using a similar architecture, which includes a costly PRNG, we assume Ascon-large as our AES-GCM area-equivalent target. In Ascon-large, the internal datapath of the permutation is 64 bits, and one 64-bit, multiplexed TI-protected and operation takes place in every clock cycle. Permutations (shown by alternating arrows in Figure 7) are performed in the stage in which the corresponding row register result is written, i.e., Stage 2 (x_4), Stage 3 (x_0), Stage 4(x_1), and Stage 5 (x_2 and x_3). Ascon-large requires 192 random bits per clock cycle – 128 bits for resharing (from two to three shares), and 64 bits to satisfy the TI uniformity property.

The TP of protected Ascon reported in [14] is 134.6 Mbps. Dividing 134.6 Mbps by 2 results in 67.3 Mbps, which is close to the TP of AES-GCM in the same study. Therefore, we choose to divide the Ascon permutation once, and assume that a 32-bit permutation datapath will achieve an AES-GCM TP-equivalent target. We call this Ascon-small. In Ascon-small, the permutation takes 12 clock cycles per round, and the internal datapath is 32 bits. One 32-bit TI-protected `and` operation takes place in cycles 1 through 10. Cycles 11 and 12 are reserved for 32-bit permutations, in order to shorten critical paths and reduce sensitive information available in a given clock cycle. Ascon-small requires 96 random bits per clock cycle, including 64 for resharing and 32 for refresh masking.

4.3 AES-GCM

The AES cryptographic primitive is documented in [30]. We leverage AES designs discussed in [29, 7, 14], which use the Tower Fields method to more efficiently perform calculations in subfields of GF(2^8). Since even one TI-protected S-Box is costly, we instantiate only one 3-share TI-protected 8-bit inverter. Outside the inverter, we leverage the methods described in [7] to employ a hybrid 2-/3-share TI approach, where linear calculations, such as round key addition, column multiplications, basis conversions, affine transformations, etc., are conducted on only two shares to save resources.

The resulting protected design has a 5-stage pipeline, where a 128-bit block encryption executes in 205 clock cycles. This construction requires 16 bits of fresh randomness for resharing from two to three shares, and 24 fresh remasking bits, for a total of 40 random bits per clock cycle.

As discussed above, DPA vulnerabilities identified in AES-GCM necessitate the protection of the non-linear $GF(2^{128})$ multiplication. Therefore, we use a 3-share TI-protected multiplier, which executes a two-operand multiplication in 128 clock cycles. 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 operations occurring inside the AES core.

5 Results

5.1 Results of TVLA

Using the above methodology, t-tests are conducted on unprotected cipher implementations. Results are shown for AES-GCM (Figure 8a), ACORN-1 (Figure 9a), ACORN-32 (Figure 9c), Ascon-small (Figure 10a), and Ascon-large (Figure 10c). The t-tests, using 2,000 FOBOS-generated traces, generally show that unprotected implementations leak information, although unprotected ACORN-1 comes close to passing. In t-test figures, time domain samples are shown on the x-axis, and t-values are shown on the y-axis. We note that the existence of leakage does not prove the existence of DPA vulnerability; it merely provides a suspicion that sensitive data requires countermeasure protection.

After applying the hybrid 2- /3-share TI protection, as described above, protected implementations are retested. Results of passing t-tests, using 2,000 traces, are shown for AES-GCM (Figure 8b), ACORN-1 (Figure 9b), ACORN-32 (Figure 9d), Ascon-small (Figure 10b), and Ascon-large (Figure 10d). While it is possible that protected implementations could exhibit leakage at a higher number of traces, our methodology establishes a sufficient plausible baseline for cipher implementations protected against 1st order DPA, in order to benchmark results and compare differences in size and performance.

5.2 Benchmarking of Unprotected and Protected Implementations

Cipher implementations are designed by the authors of this research using register-transfer level (RTL) methodology in VHDL. Our implementation results are reported as post place-and-route (PAR) results using Xilinx 14.7 ISE for the Spartan-6 (xc6slx16 csg324-3) FPGA. We further optimize results using the ATHENa optimization tool with a strategy optimized for TP/A ratio [9]. The ATHENa tool suppresses generation of Block RAMs to ensure a more fair comparison. All ciphers include a serial wrapper, since the external interfaces of some implementations have bus widths for incoming random data that exceed the Spartan-6 capabilities.

Table 1: Unprotected and Protected Cipher Implementations in Spartan-6 FPGA

Algorithm	Area [LUT]	Area Ratio vs AES-GCM	Freq [MHz]	TP [Mbps]	TP Ratio vs AES-GCM	TP/A [Mbps/LUT]	TP/A Ratio vs AES-GCM
Unprotected							
ACORN-1	446	0.22	141.9	70.9	0.93	0.159	3.33
ACORN-32	1,396	0.69	147.7	2,363.7	24.31	1.693	35.50
Ascon-small	1,640	0.80	146.1	114.0	1.17	0.070	1.46
Ascon-large	1,725	0.85	148.4	237.4	2.44	0.138	2.89
AES-GCM	2,039	1.00	157.3	97.2	1.00	0.048	1.00
Protected							
ACORN-1	784	**0.18**	156.6	78.3	**1.02**	0.100	5.77
ACORN-32	4,072	**0.92**	111.5	1,784.0	**23.30**	0.438	25.31
Ascon-small	3,278	**0.74**	117.0	91.4	**1.19**	0.028	0.87
Ascon-large	3,673	**0.83**	119.8	191.7	**2.50**	0.052	3.01
AES-GCM	4,429	1.00	124.0	76.7	1.00	0.017	1.00

Case 1: Implementations area-equivalent to AES-GCM. The protected implementation of ACORN-32 consumes 4072 LUTs, which is close to (92%) the area of our protected AES-GCM, with 4429 LUTs. However, the TP of ACORN-32 is 1,784 Mbps, which is 23.3 times that of AES-GCM at 76.7 Mbps. The protected implementation of Ascon-large (with a 64-bit permutation datapath) consumes 3673 LUTs, which is 83% of the area of AES-GCM, and has a TP of 191.1 Mbps, which is 2.5 times that of AES-GCM.

Case 2: Implementations TP-equivalent to AES-GCM. The protected implementation of ACORN-1, i.e., a true stream cipher, has a TP of 78.3 Mbps, which is slightly greater than that of AES-GCM, but has an area of 784 LUTs, which is only 18% that of AES-GCM. In the case of Ascon, Ascon-small (with a 32-bit permutation datapath), has a TP of 91.4 Mbps, which is 1.2 times the TP of AES-GCM, and has an area of 3278 LUTs, which is 74% that of AES-GCM.

5.3 Measurement of Power and Energy

Although we do not design protected implementations with AES-GCM-equivalent power and energy targets, we are nevertheless able to make this comparison. We measure power using an extension of FOBOS, by measuring amplified voltage across a 1 Ohm shunt resistor, during device operation with multiple test vectors, at 10 MHz on the Spartan-6 FPGA. Measured power results are shown in Table 2.

The two AES-GCM area-equivalent implementations of ACORN-32 and Ascon-large both draw more power than AES-GCM, despite the fact that they are physically smaller. The order of increasing power consumption, AES-GCM, Ascon-large, and ACORN-32, is the same as the order of increasing critical path (i.e., the inverse of maximum frequency), which suggests that increased intra-cycle toggle rates due to glitching is possibly a factor. However, the large power consumption of these ciphers is also possibly correlated to the larger PRNGs required for these implementations, or to differences in internal datapath widths.

Conversely, ACORN-1 and Ascon-small both use less power than AES-GCM. In fact, the protected version of ACORN-1 uses only 13% more power than its unprotected counterpart.

We compute energy per bit (E/bit) (nJ/bit) as

$$E/bit(nJ/bit) = \frac{Power(mJ/s)}{TP_{10MHz}(Mbps)} \tag{1}$$

By this metric, all protected implementations are more energy efficient that AES-GCM, particularly ACORN-32, with only 6% of the energy usage per bit of AES-GCM.

5.4 Analysis

Protected implementations of ACORN exhibit very high performance in hardware. The low degree of non-linearity in the state update facilitates an easy pro-

Table 2: Power and Energy of Cipher Implementations at 10MHz in Spartan-6 FPGA

Algorithm	Power [mW]	Power Ratio vs AES-GCM	Power Growth Ratio vs Unprotected	E/bit [nJ/bit]	E/bit Ratio vs AES-GCM	E/bit Growth Ratio vs Unprotected
Unprotected						
ACORN-1	7.6	0.78	-	1.520	0.97	-
ACORN-32	11.7	1.21	-	0.073	0.05	-
Ascon-small	9.2	0.95	-	0.575	0.37	-
Ascon-large	10.6	1.09	-	0.663	0.42	-
AES-GCM	9.7	1.00	-	1.569	1.00	-
Protected						
ACORN-1	8.6	0.46	1.13	1.720	0.57	1.13
ACORN-32	27.4	1.47	2.34	0.171	0.06	2.34
Ascon-small	15.9	0.85	1.73	2.037	0.67	1.84
Ascon-large	22.8	1.22	2.15	1.425	0.47	2.15
AES-GCM	18.7	1.00	1.93	3.024	1.00	1.93

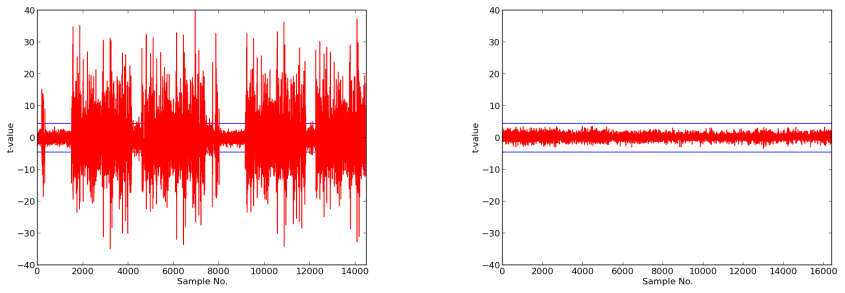
tection scheme using a 3-share threshold implementation. As such, a protected ACORN-1, with very low area, relatively high TP, low power, good energy efficiency, and low external randomness requirement, is an excellent fit for security applications requiring authenticated transactions in the IoT. Our ACORN-32 implementation, while realizing very high TP with relatively low area, is less optimal for lightweight applications than ACORN-1, in that it it requires 480 random bits per clock cycle, and is relatively high-power.

Ascon, likewise, is more efficiently protected against DPA than AES-GCM, thanks to the low algebraic degree of its S-Box. However, Ascon inherently has a larger protection cost than some other CAESAR-candidate ciphers, including ACORN, because of its sponge construction with large internal state of 320 bits, meaning 64 5-bit S-Boxes must be conducted in every round, regardless of whichever architecture one chooses.

Unfortunately, our reduction in permutation datapath width from 64 bits (Ascon-large) to 32 bits (Ascon-small) results in area savings of only 11%. This comes at a cost of doubling of the number of clock cycles per p^b permutation, from 40 to 82. Although the size of the 3-share TI-protected `and` module is reduced in Ascon-small, the increase in number of required multiplexers and number of states in the permutation FSM minimizes most gains. Additionally, the overhead in the authenticated cipher layer datapath and controller, and FSM logic necessary to meet the requirements of the CAESAR HW API, remains relatively constant, even while the permutation shrinks.

Further, our conservative application of both reshare and refresh masking drives the external randomness requirements to numbers higher than desirable for lightweight applications, namely 96 or 192 random bits per clock cycle for Ascon-small or Ascon-large, respectively. In the case of both ACORN and Ascon, it might be possible to reduce randomness requirements, using strategies such as Domain Oriented Masking (DOM), Uniform Masking Approach (UMA), or "Changing of the Guards," as discussed in [22, 21, 13]. Randomness for refresh masking to maintain the TI Uniformity property can also be ensured by calculating the minimum number of bits necessary for output distribution to preserve the input distribution, but this is computationally complex, and is not often attempted in practice [7].

We do not claim that our implementations are the smallest or most efficient possible, even at a protection level of 1st order DPA ($d = 1$). Our ACORN implementations are either significantly smaller (ACORN-1) or faster (ACORN-32) than protected ACORN-8 implemented in [14], which is our only basis for comparison. In the case of Ascon, however, we can compare ACORN to the side-channel protected Ascon at [20] and discussed in [21]. In order to compare the smallest possible version, we obtain place and route result for [20] in the lightweight Artix-7 FPGA, with 1st order protection ($d = 1$) and a single S-Box. The results are 1754 LUTs with a TP of 11.1 Mbps, which is 2.2 times larger, with only 14.2% of the TP of ACORN-1. In summary, it is difficult to improve upon the size and performance of the protected ACORN-1.

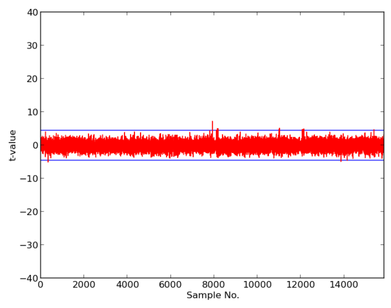


(a) AES-GCM UnPr (b) AES-GCM Pr

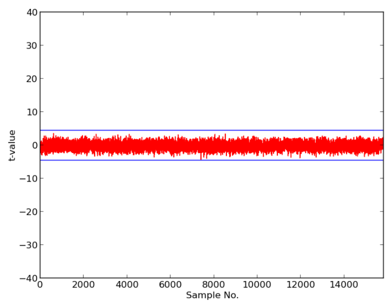
Fig. 8: t-test results for unprotected (UnPr) and protected (Pr) AES-GCM implementations, where "S" denotes "small" and "L" denotes "large." Samples are shown on the x-axis, and t-values are shown on the y-axis.

6 Conclusions

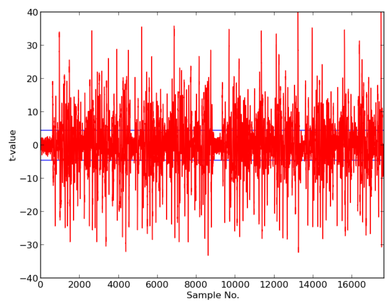
In this research, we compared side-channel resistant FPGA implementations of the CAESAR lightweight-finalists ACORN and Ascon, using two optimization targets. First, we compared versions of ACORN and Ascon that are roughly "area-equivalent" to AES-GCM, and noted gains in throughput (TP). Next, we compared versions of ACORN and Ascon that were close to "TP-equivalent" to



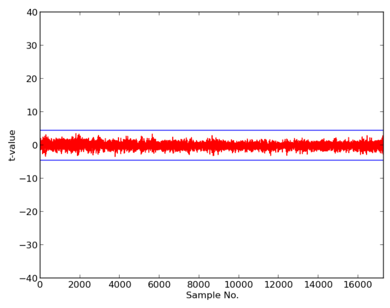
(a) ACORN1 UnPr



(b) ACORN1 Pr



(c) ACORN32 UnPr

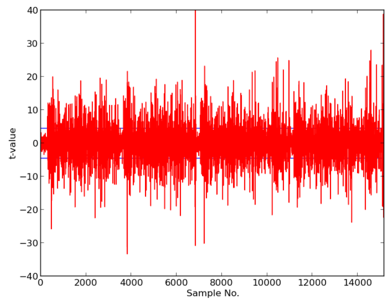


(d) ACORN32 Pr

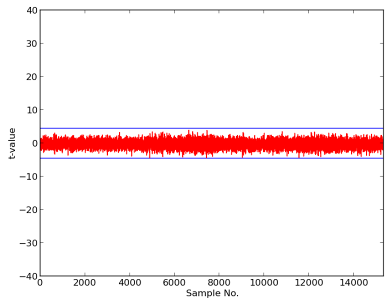
Fig. 9: t-test results for unprotected (UnPr) and protected (Pr) ACORN implementations, where "S" denotes "small" and "L" denotes "large." Samples are shown on the x-axis, and t-values are shown on the y-axis.

AES-GCM, and noted reductions in area (LUTs). We observed that AES-GCM area-equivalent implementations of the CAESAR finalists, protected against 1st order DPA with resistance affirmed using Test Vector Leakage Methodology (TVLA), have significantly higher TP, namely 23.3 and 2.5 times for ACORN-32 and Ascon-large, respectively. Additionally, we observed that AES-GCM TP-equivalent protected versions of ACORN-1 and Ascon-small have 18% and 74%, respectively, the area of AES-GCM.

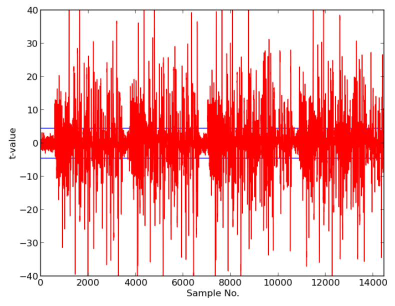
We declare ACORN as the "winner of the face-off" – it is clear that ACORN, particularly ACORN-1, is the most efficient side-channel resistant CAESAR lightweight finalist, in terms of low area, power, and external randomness, which are most relevant for lightweight applications in IoT devices. If insertion of countermeasures is required, 1st order DPA protection additions create an area overhead of only 76%, even with the inclusion of a robust I/O capability.



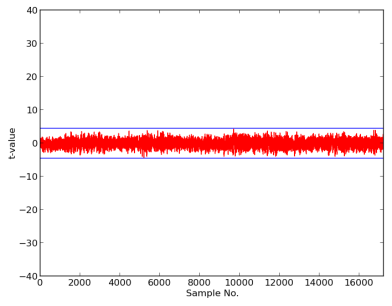
(a) Ascon-S UnPr



(b) Ascon-S Pr



(c) Ascon-L UnPr



(d) Ascon-L Pr

Fig. 10: t-test results for unprotected (UnPr) and protected (Pr) Ascon implementations, where "S" denotes "small" and "L" denotes "large." Samples are shown on the x-axis, and t-values are shown on the y-axis.

7 Areas for Future Work

A more thorough investigation of these ciphers in the future could include alternative techniques for protection, improved techniques for random number generation including reduction of randomness requirements, measurements using improved TVLA methods, and characterization of improved DPA resistance through attack-based testing, such as Correlation Power Analysis (CPA).

References

1. GMU Source Code of Round 3 & Round 2 CAESAR Candidates, AES-GCM, AES, AES-HLS, and Keccak Permutation F, https://cryptography.gmu.edu/athena/index.php?id=CAESAR_source_codes
2. CAESAR Competition for Authenticated Encryption: Security, Applicability, and Robustness (2012), http://competitions.cr.yp.to/caesar.html

3. Barthe, G., Dupressoir, F., Faust, S., Grégoire, B., Standaert, F.X., Strub, P.Y.: Parallel Implementations of Masking Schemes and the Bounded Moment Leakage Model. In: Advances in Cryptology – EUROCRYPT 2017. pp. 535–566 (2017)
4. Belaïd, S., Fouque, P.A., Gérard, B.: Side-Channel Analysis of Multiplications in GF(2^{128}). In: Advances in Cryptology – ASIACRYPT 2014. pp. 306–325 (2014)
5. Bernstein, D.J.: Cryptographic Competitions (Jul 2016), https://groups.google.com/forum/#!forum/crypto-competitions
6. Bertoni, G., Daemen, J., Peeters, M., Van Assche, G.: Duplexing the Sponge: Single-Pass Authenticated Encryption and Other Applications. In: Selected Areas in Cryptography. pp. 320–337 (2012)
7. Bilgin, B., Gierlichs, B., Nikova, S., Nikov, V., Rijmen, V.: A More Efficient AES Threshold Implementation. In: Progress in Cryptology – AFRICACRYPT 2014. pp. 267–284 (2014)
8. CERG: ATHENa Database of Results: Authenticated Encryption FPGA Ranking, https://cryptography.gmu.edu/athena
9. CERG: Automated Tool for Hardware Evaluation (ATHENa), https://cryptography.gmu.edu/athena/index.php?id=ATHENa
10. CERG: Flexible Open-source workBench fOr Side-channel analysis (FOBOS) (Oct 2016), https://cryptography.gmu.edu/fobos/
11. CERG: Development Package for Hardware Implementations Compliant with the CAESAR Hardware API, v2.0 (Dec 2017), https://cryptography.gmu.edu/athena/index.php?id=CAESAR
12. Cooper, J., DeMulder, E., Goodwill, G., Jaffe, J., Kenworthy, G., Rohatgi, P.: Test Vector Leakage Assessment (TVLA) Methodology in Practice. International Cryptographic Module Conference (2013)
13. Daemen, J.: Changing of the Guards: A Simple and Efficient Method for Achieving Uniformity in Threshold Sharing. In: Cryptographic Hardware and Embedded Systems – CHES 2017. pp. 137–153 (2017)
14. 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: 2018 IEEE International Symposium on Hardware Oriented Security and Trust (HOST). pp. 147–152 (April 2018)
15. Diehl, W., Farahmand, F., Abdulgadir, A., Kaps, J.P., Gaj, K.: Face-off between the CAESAR Lightweight Finalists: ACORN vs. Ascon. In: 2018 International Conference on Field Programmable Technology (ICFPT), Naha, Japan. (Dec 2018)
16. Diehl, W., Farahmand, F., Abdulgadir, A., Kaps, J.P., Gaj, K.: Fixing the CLOC with Fine-grain Leakage Analysis. In: 2018 Workshop on Attacks and Solutions in Hardware Security (ASHES '18), Toronto, ON. pp. 75–80 (Oct 2018)
17. Dobraunig, C., Eichlseder, M., Mendel, F., Schläffer, M.: Ascon v1.2 (Sep 2016), https://competitions.cr.yp.to/round3/asconv12.pdf, accessed February 16, 2018
18. Ferguson, N.: Authentication Weaknesses in AES-GCM (May 2005), https://csrc.nist.gov/csrc/media/projects/block-cipher-techniques/documents/bcm/comments/cwc-gcm/ferguson2.pdf, accessed April 12, 2018
19. Goodwill, G., Jun, B., Jaffe, J., Rohatgi, P.: A Testing Methodology for Side Channel Resistance Validation. NIST Non-invasive Attack Testing Workshop (2011)
20. Groß, H.: DOM and UMA Masked Hardware Implementations of Ascon (2017), https://github.com/hgrosz/ascon_dom, 2017, accessed April 7, 2018
21. Groß, H., Mangard, S.: Reconciling $d+1$ Masking in Hardware and Software. International Conference on Cryptographic Hardware and Embedded Systems (CHES 2017) pp. 115–136 (2017)

22. Gross, H., Mangard, S., Korak, T.: Domain-Oriented Masking: Compact Masked Hardware Implementations with Arbitrary Protection Order. In: Proceedings of the 2016 ACM Workshop on Theory of Implementation Security. TIS '16
23. Groß, H., Wenger, E., Dobraunig, C., Ehrenhöfer, C.: Suit up! - Made-to-Measure Hardware Implementations of ASCON. In: 2015 Euromicro Conference on Digital System Design, DSD 2015, Madeira, Portugal, August 26-28, 2015. pp. 645–652
24. Homsirikamol, E., Diehl, W., Ferozpuri, A., Farahmand, F., Yalla, P., Kaps, J.P., Gaj, K.: CAESAR Hardware API. Cryptology ePrint Archive, Report 2016/626 (2016), http://eprint.iacr.org/2016/626.pdf
25. Huang, T.: Round 3 Hardware Submission: ACORN (Jul 2017), https://groups.google.com/forum/#!forum/crypto-competitions
26. Jaffe, J.: A First-Order DPA Attack Against AES in Counter Mode with Unknown Initial Counter. In: Cryptographic Hardware and Embedded Systems - CHES 2007. pp. 1–13 (2007)
27. Kocher, P., Jaffe, J., Jun, B.: Differential Power Analysis. In: Advances in Cryptology — CRYPTO' 99. pp. 388–397 (1999)
28. Kocher, P., Jaffe, J., Jun, B., Rohatgi, P.: Introduction to Differential Power Analysis. Journal of Cryptographic Engineering **1**(1), 5–27 (Apr 2011)
29. Moradi, A., Poschmann, A., Ling, S., Paar, C., Wang, H.: Pushing the limits: A very compact and a threshold implementation of AES. In: Advances in Cryptology – EUROCRYPT 2011. pp. 69–88 (2011)
30. National Institute of Standards and Technology: Report on the Development of the Advanced Encryption Standard (AES) (Oct 2000), http://csrc.nist.gov/archive/aes/round2/r2report.pdf
31. National Institute of Standards and Technology: Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC. NIST SP800-38D (Nov 2007), http://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-38d.pdf
32. National Institute of Standards and Technology: Submission Requirements and Evaluation Criteria for the Lightweight Cryptography Standardization Process (Aug 2018), https://csrc.nist.gov/Projects/Lightweight-Cryptography
33. Nikova, S., Rechberger, C., Rijmen, V.: Threshold Implementations Against Side-Channel Attacks and Glitches. In: Information and Communications Security. pp. 529–545 (2006)
34. Samwel, N., Daemen, J.: DPA on Hardware Implementations of Ascon and Keyak. In: Proceedings of the Computing Frontiers Conference, CF'17, Siena, Italy, May 15-17, 2017. pp. 415–424. ACM (2017)
35. Schneider, T., Moradi, A.: Leakage Assessment Methodology. Journal of Cryptographic Engineering **6**(2), 85–89 (Jun 2016)
36. Standaert, F.X.: How (not) to Use Welchs T-test in Side-Channel Security Evaluations. Cryptology ePrint Archive, Report 2017/138 (2017), https://eprint.iacr.org/2017/138.pdf
37. Vliegen, J., Reparaz, O., Mentens, N.: Maximizing the throughput of threshold-protected AES-GCM implementations on FPGA. In: 2017 IEEE 2nd International Verification and Security Workshop (IVSW). pp. 140–145 (July 2017)
38. Wu, H.: ACORN: A Lightweight Authenticated Cipher (Sep 2016), https://competitions.cr.yp.to/round3/acornv3.pdf, accessed March 23, 2018
39. Yalla, P., Kaps, J.P.: Evaluation of the CAESAR Hardware API for Lightweight Implementations. In: International Conference on Reconfigurable Hardware (ReConFig 2017). pp. 1–6 (Dec 2017)