Updates on ACE

Mark Aagaard[1], Riham AlTawy[2], Guang Gong[1], Kalikinkar Mandal[3], and Raghvendra Rohit[4]

[1] Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, Canada
[2] Department of Electrical and Computer Engineering, University of Victoria, Victoria, Canada
[3] Faculty of Computer Science, University of New Brunswick, Fredericton, Canada
[4] University of Rennes, CNRS, IRISA, France

Abstract. ACE is a permutation based authenticated encryption (AE) and hash algorithm which provides 128-bit security for AE and hash with a single hardware circuit. In this note, we report updates on ACE since its selection as a round 2 candidate of the NIST lightweight cryptography standardization competition [3]. This report includes: new third party cryptanalysis and implementation results, applications of ACE in IEEE 802.11X and CoAP handshake protocols for Internet of Things, and comparisons with current NIST standards. We do not plan any future tweaks for ACE.

1 Security Analysis

Our round 2 submission document [1] already includes an in-depth security analysis of ACE permutation and its AE and hash modes. In a nutshell, the AE algorithm ACE-$\mathcal{AE}$-128 with 16 steps of ACE permutation offers 128-bit security in a nonce-respecting scenario while the hash algorithm ACE-$\mathcal{H}$-256 with 256-bit message digest also provides 128-bit security [1, Table 3.1]. At the time of writing this report, we are not aware of any third party attacks on ACE. However, there are some recent results where the round-reduced ACE permutation and side channel security of ACE-$\mathcal{AE}$-128 have been investigated. We briefly describe them below.

Impossible differential distinguishers on round-reduced ACE. Liu et al. [7] have analyzed the security of ACE permutation against impossible differential distinguishers using the characteristic matrix method [11]. They showed that there are no impossible differential distinguishers for more than 9 steps of ACE. Further, they were able to find only 8 steps impossible differentials. The two 8-step impossible differentials are $(0,0,0,\alpha,0) \not\rightarrow (\beta,0,0,0,0)$ and $(0,\alpha,0,0,0) \not\rightarrow (\beta,0,0,0,0)$ where α and β are 64-bit words. Note that the total number of steps of ACE permutation is 16. Thus, the security margin, i.e., $1 - \frac{\#\text{ attacked rounds}}{\#\text{ total rounds}}$ of ACE against impossible differentials is at least 50%.

Leakage resistance of ACE-$\mathcal{AE}$-128. In a recent work, Bellizia et al. [5] have analyzed the leakage resistance of NIST LWC round 2 candidates by classifying them into modes and primitives. ACE has been shown to be CIML2 and CCAmL1 secure. Adopting the [5, Def. 1 & 2], CIML2 refers to the *"ciphertext integrity with misuse-resistance (i.e., no constraint on nonces) and leakage in encryption and decryption"*, while CCAmL1 means *"chosen ciphertext security with nonce misuse-resilience (i.e., fresh challenge nonce) and leakage in encryption only"*. For the proof details, the reader is referred to [5].

In summary, the existing results on ACE do not affect our security claims.

2 New Implementation Results

2.1 Software Performance

Masked implementation by Belaïd et al. [4]. Belaïd et al. have introduced a new framework called Tornado which automatically generates the masked bitsliced implementation of a primitive which is

provably-secure in the register probing model. The security of ACE and its performance has been analyzed along with other round 2 candidates. Below, we list few observations from [4].

- To be secure in the register probing model, ACE round function requires a refresh gadget.
- Although ACE is designed to keep hardware efficiency in mind, it ranks eighth in performance for masked bitsliced implementations [4, Table 6].

Software implementation and benchmarking by Weatherley [12]. ACE is included in Rhy's Weatherley software benchmarking framework where the performance of round 2 candidates is evaluated on 8-bit and 32-bit microcontrollers. For the details on the performance, the reader is referred to [12].

2.2 Hardware Performance

A comprehensive analysis of parallel implementations of ACE was presented in [2], and we included the implementation results for four ASIC technologies in [1]. For example, unparallelized ACE has the area of 4250 GE and reaches a throughput of 360 Mbps using ST Micro 65 nm library. Throughput of 984 Mbps was achieved with the 8× parallel implementation, with area 7240 GE. In [2], we also presented energy per bit, measured as the average value while performing cryptographic operations over 8192 bits of data at 10 MHz, which for unparallelized ACE ST Micro 65 nm implementation yields 27.9 nJ. Further, in [1], we have included implementation results for Xilinx Spartan-3 and Spartan-6, and for Intel/Altera Stratix IV. An LWC API compliant [8] implementation for ACE will be available shortly.

3 Features

3.1 Applications and Use-cases

ACE is primarily a hardware-oriented lightweight authenticated encryption and hash algorithm which is designed to achieve low *area, power and energy*, and as such, it mainly targets RFID and sensor network applications. For instance, the combined unparallelized hardware circuit (including both AE and hash functionality) of ACE has the area of 4250 GE and reaches a throughput of 360 Mbps in ST Micro 65 nm library.

As a concrete application, we have investigated ACE to implement the key derivation function (KDF) and generate the message integrity check (MIC) in IEEE 802.11X [6] and CoAP [14] handshake mutual authentication and key establishment protocols for IoT applications [13]. Given that the majority of IoT devices are quipped with microcontrollers, we provide performance evaluation of ACE for `KDF` and `MIC` functionalities and handshaking and data protection protocols on microcontrollers. Our experimental results show that ACE take about 3,089 and 2,966 ms to complete the IEEE 802.11X authentication protocol using MSP430F2370, and Cortex-M3, respectively. For the data protection protocol, ACE achieves a throughput of 10 and 55 Kbits/s on MSP430F2370 and Cortex-M3, respectively to encrypt and authenticate a plaintext of 1024 bits and an associated data of 128-bits. For generating a hash of 1024 bits message, the corresponding throughputs are 16 and 63 Kbits/s on MSP430F2370 and Cortex-M3, respectively. More details can be found in [13].

3.2 Comparison with current NIST standards

Comparison with AES-GCM. A fair comparison of ACE-$\mathcal{AE}$-128 with AES-GCM is hard unless both ciphers are implemented in the same technology. However, in ASICs, we believe that ACE-$\mathcal{AE}$-128 outperforms AES-GCM in *area*. This is because AES-GCM requires a 128-bit finite field multiplier which is generally expensive in ASICs.

Comparison with SHA2 and SHA3 hash functions. From a hardware perspective (specifically area), the hash functions SHA-2 [9] and SHA-3 [10] do not fit into the lightweight applications. The former uses modular addition in its round function which are too expensive in hardware while for the later the state size (1600 bits) is too large. Thus, it is unfair to compare SHA-2 and SHA-3 with ACE-$\mathcal{H}$-256.

4 Tweaks

We propose no tweaks in the design of ACE. Our rationale is to keep the same number of rounds for both authenticated encryption and hashing functionalities to have a compact hardware implementation.

References

[1] Mark Aagaard, Riham AlTawy, Guang Gong, Kalikinkar Mandal, and Raghvendra Rohit. Ace: An authenticated encryption and hash algorithm. *Submission to NIST-LWC (Round 2)*, 2019. https://csrc.nist.gov/CSRC/media/Projects/lightweight-cryptography/documents/round-2/spec-doc-rnd2/ace-spec-round2.pdf,.

[2] Mark D. Aagaard, Marat Sattarov, and Nuša Zidarič. Hardware design and analysis of the ACE and WAGE ciphers. NIST LWC Workshop 2019. Also available at https://arxiv.org/abs/1909.12338.

[3] Lawrence Bassham, Cagdas Calik, Donghoon Chang, Jinkeon Kang, Kerry McKay, and Meltem Sonmez Turan. Lightweight cryptography: Round 2 candidates, 2019. https://csrc.nist.gov/Projects/lightweight-cryptography/round-2-candidates.

[4] Sonia Belaïd, Pierre-Évariste Dagand, Darius Mercadier, Matthieu Rivain, and Raphaël Wintersdorff. Tornado: Automatic generation of probing-secure masked bitsliced implementations. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 311–341. Springer, 2020.

[5] Davide Bellizia, Olivier Bronchain, Gaëtan Cassiers, Vincent Grosso, Chun Guo, Charles Momin, Olivier Pereira, Thomas Peters, and François-Xavier Standaert. Mode-level vs. implementation-level physical security in symmetric cryptography a practical guide through the leakage-resistance jungle. In *Crypto*, 2020.

[6] IEEE 802.11 Working Group et al. 802.11ax - ieee draft standard for information technology – telecommunications and information exchange between systems local and metropolitan area networks. *IEEE Std*, 2019.

[7] Jingyi Liu, Guoqiang Liu, and Longjiang Qu. A new automatic tool searching for impossible differential of nist candidate ace. *Mathematics*, 8(9):1576, 2020.

[8] Ekawat Homsirikamol William Diehl Jens-Peter Kaps Michael Tempelmeier, Farnoud Farahmand and Kris Gaj. Implementer's guide to hardware implementations compliant with the hardware api for lightweight cryptography, v1.0.1 (nov 2019). https://cryptography.gmu.edu/athena/LWC/LWC_HW_Implementers_Guide.pdf.

[9] NIST. Secure hash standard. Federal Information Processing Standards Publication 180-4, 2015. https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.180-4.pdf.

[10] NIST. Sha-3 standard: Permutation-based hash and extendable-output functions. Federal Information Processing Standards Publication 202, 2015. https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.202.pdf.

[11] Bing Sun, Meicheng Liu, Jian Guo, Vincent Rijmen, and Ruilin Li. Provable security evaluation of structures against impossible differential and zero correlation linear cryptanalysis. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 196–213. Springer, 2016.

[12] Rhys Weatherley. Lightweight cryptography primitives. https://rweather.github.io/lightweight-crypto/.

[13] Yunjie Yi, Guang Gong, and Kalikinkar Mandal. Implementation of lightweight ciphers and their integration into entity authentication with IEEE 802.11 physical layer transmission, 2020. In submission at IEEE Internet of Things Journal.

[14] K. Hartke C. Bormann Z. Shelby, K. Hartke. The constrained application protocol (coap). RFC 7252, 2014. https://tools.ietf.org/pdf/rfc7252.pdf.