An Update on the LWC Finalist Sparkle

Christof Beierle[1], Alex Biryukov[2], Luan Cardoso dos Santos[2], Johann Großschädl[2], Amir Moradi[1], Léo Perrin[3], Aein Rezaei Shahmirzadi[1], Aleksei Udovenko[2,4], Vesselin Velichkov[5], and Qingju Wang[2]

[1]Ruhr University Bochum, Germany
[2]DCS and SnT, University of Luxembourg, Luxembourg
[3]Inria, Paris, France
[4]CryptoExperts, Paris, France
[5]University of Edinburgh, U.K.

September 30, 2022

sparkle-lwc.github.io
sparklegrupp@googlegroups.com

Introduction

Sparkle is a family of lightweight cryptographic permutations used to build the Schwaemm Authenticated Encryptions with Associated Data (AEAD), and the Esch hash functions. These algorithms are optimized for efficiency on micro-controllers, which allows them to be top performers in terms of speed and memory consumption at the same time. Furthermore, unlike for instance for the AES, it is easy to write constant-time implementations.

In this short note, we summarise recent implementation improvements along with respective benchmarks, and survey recent works providing additional security analysis for the Sparkle algorithms.

Implementations and Benchmarks

The Sparkle group has slightly improved the Assembler implementations of the permutations for 8-bit AVR ATmega and 32-bit ARM Cortex-M3/M4 microcontrollers compared to the second round of evaluation. The source code of these improved implementations can be found in the GitHub repository at `https://github.com/cryptolu/sparkle`. However, the surrounding C code for the mode and the full AEAD and hash algorithm did not change since the second round. In addition, the Sparkle group developed new size-optimized and speed-optimized Assembler implementations of the Sparkle384 permutation for the 32-bit RISC-V platform. These implementations can optionally make use of the 32-bit rotation instructions provided by the bit-manipulation extension if it is available.

In the past two years, various software implementations of Sparkle, Schwaemm, and Esch have been contributed by third-party developers. For example, Cheng et al. present in [CGM+22] highly-optimized Assembler implementations of Schwaemm256-128 and the other nine final-round AEAD candidates for a 32-bit RISC-V processor that supports the bit-manipulation extension, which means single-cycle rotate instructions are available. Furthermore, they also designed Instruction Set Extensions (ISEs) for each candidate, whereby the custom instructions adhere to the standard three-register format (i.e. two source registers and one destination register). Experimental results obtained with an FPGA implementation of the 32-bit Rocket core show that Schwaemm256-128 is the most efficient final-round candidate on RISC-V, both with and without

ISEs. For example, an ISE-supported implementation of SCHWAEMM256-128 outperforms ASCON128 and Xoodyak by a factor of 3.21 and 2.61, respectively (encryption of 128 bytes of data and the same amount of associated data). Some preliminary implementation results were presented at the NIST Workshop on Lightweight Cryptography, which took place in May 2022. The final version of this paper containing the most recent results has been submitted to IACR TCHES and is, at the time of writing this note (i.e. September 2022), undergoing a minor revision to address the comments of the reviewers.

The paper of Blanc et al. [BLLG+22], which was published very recently, contains benchmarking results for 20 AEAD algorithms, including all 10 NIST finalists, for 8-bit AVR Atmega128, 16-bit MSP430, and 32-bit ARM Cortex-M3 platforms. These benchmarks are based on the reference and optimized implementations from the designer teams and third-party developers. Though most of these implementations have already been evaluated by other benchmarking initiatives before, some of the results are original and interesting because they shed some new light on the performance of the final-round candidates. In particular, this paper is the first to provide benchmarking results for a 16-bit platform, namely the MSP430F1611 from Texas Instruments. SCHWAEMM256-128 does not only have the smallest code size on MSP430, but is also the by far fastest AEAD algorithm among the ten finalists. However, all these results are based on C implementations without Assembly optimizations for the performance-critical parts.

A team composed of students and research staff of the University of Luxembourg developed optimized implementations of the five final-round candidates ASCON128, GIFT-COFB, TinyJambu v2, SCHWAEMM256-128, and Xoodyak for the 16-bit MSP430 platform. These implementations come with optimized Assembly code for the underlying primitive, which is a permutation in the case of ASCON, TinyJambu, SCHWAEMM, and Xoodyak, and a block cipher for GIFT-COFB. Though this project is still work in progress, the Assembler components are already available online on GitHub at https://github.com/johgrolux/aead430. The execution times for the permutations indicate that SCHWAEMM256-128 will be roughly twice as fast as Xoodyak and about 2.5 times faster than ASCON128. More detailed benchmarking results will most likely become available in October 2022.

In his PhD thesis [Alr21], Alrowaithy investigated the performance of all second round candidates on 8-bit micro-controllers. We reproduce part of their conclusion [Alr21, page 148] below.

> *"Finally, trying to find a primitive that does well on all three performance metrics is challenging. For example, primitives that are optimised for speed incur extra code size or memory usage. The only primitive we found to show a good balance between the three metrics is SPARKLE."*

In addition to the above-mentioned microcontroller implementations, there exist also some implementations of SPARKLE, SCHWAEMM, and ESCH for more powerful CPUs and even Graphics Processing Units (GPUs). For example, Anna Weine developed a vector implementation of the SPARKLE384 permutation using AVX2 instructions, which is available on the GitHub repository of the SPARKLE group. Lee et al. describe in [LJS+22] an efficient implementation of ESCH and some other final-round hash candidates for an RTX 3080 GPU. According to the results presented in Figure 8 of [LJS+22], ESCH reaches a roughly two times higher throughput than the hash functions of the candidates ASCON and Xoodyak.

Comparison with AES-GCM. One of the core requirements for AEAD candidates mentioned in NIST's call for proposals is that the submitted algorithms should "perform significantly better" in constrained environments than the current NIST standards (i.e. AES-GCM). SCHWAEMM clearly over-achieves this requirement; for example, according to NIST's official second-round benchmarking results, SCHWAEMM256-128 is able to encrypt 128 bytes of data more than five times faster than AES-GCM on each of the three main evaluation platforms, which are AVR ATmega, ARM Cortex-M0, and ARM Cortex-M4.

It is sometimes claimed that the software performance of AES-GCM is "good enough" for many or even most IoT applications. However, this view is not the consensus, and may stem from a form of "survivor bias." While it is true that the developers of applications which actually use AES-GCM obviously found a way to deal with the (by today's standards) mediocre software performance of AES-GCM, it should also be taken into account that there exist a large number of IoT applications that do not use the AES, typically for performance reasons. In many cases, the developers of the affected applications resort to some obscure and insecure encryption techniques or, even worse, do not use any encryption algorithm at all (see e.g. [WAF17]). Therefore, bad software execution times are not only a performance problem, but also constitute a massive security problem for the whole IoT ecosystem. Having a lightweight AEAD standard that is five times more efficient than AES-GCM will contribute to make strong cryptography much more viable for applications that, due to performance reasons, use currently either some weak encryption algorithm or no encryption at all.

State of the Cryptanalysis

What we established. By design, the algorithms in the SPARKLE suite are secure from single trail differential and linear cryptanalysis, with some security margin. Indeed, the structure of the permutations combined with the specifics of the modes of operation we chose allowed us to prove strong bounds on the probabilities of the relevant differential and linear trails.

At its core, the SPARKLE permutations have a modular structure, which means that we can gain precise insights into their properties by studying the non-linear component: the Alzette ARX-box.

Third party findings. Besides our own cryptanalysis, there have been a decent amount of external analysis of SPARKLE and its component Alzette recently. All existing attacks are only on *round-reduced variants* of SPARKLE, so they further contribute to the understanding and strengthen the confidence in the security of our design.

In [SS22], the authors presented improved *guess-and-determine* distinguishers on SPARKLE covering 4 steps of SPARKLE256 and SPARKLE384 and 5 steps of SPARKLE512 with practical complexity.

By using the tools published in [HW19] and [HW20], Huang et al. proved in [HXW21] improved bounds on the differential and linear properties of Alzette. We summarize their findings in Table 1, comparing those bounds with our own findings listed in Table 3.2 of the specification of the SPARKLE finalist. As we can see, our initial analysis is confirmed, and the upper bounds we used were pessimistic from our perspective (which further strengthen our confidence in our design). The authors of [HXW21] also analyzed the probabilities of *rotational-XOR differential* probabilities of all instances of Alzette used in SPARKLE. Those probabilities (over one iteration of Alzette) were reported to be in the range $[2^{-52.66}, 2^{-37.66}]$ depending on the constant c_i (see Tables 3 and 6 in [HXW21]).

Table 1: Differential and linear bounds for Alzette. In each row, the first line shows $-\log_2 p$, where p is the maximum expected differential trail probability for the differential case and the second line shows $-\log_2 c$, where c is the maximum expected absolute linear trail correlation for the linear case. The value set in parenthesis corresponds to the maximum absolute correlation of the linear hull taking clustering into account, derived by experimental verification.

Ref.	1	2	3	4	5	6	7	8	9	10	11	12
our specification	0	1	2	6	10	18	26	≥ 32	≥ 36	≥ 42	≥ 46	≥ 52
	0	0	1	2	5	8	13 (11.64)	17 (15.79)	–	–	–	–
[HXW21]	0	1	2	6	10	18	26	34	40	46	51	≥ 55
	0	0	1	2	5	8	13	17	19	22	–	–

In [LSL21], the authors introduced a generalization of differential-linear cryptanalysis, so-called *rotational differential-linear* attacks. The general idea of those attacks is to replace the differential part of a differential-linear distinuisher by a rotational-XOR difference. They analyzed (rotational) differential-linear distinguishers of Alzette with the constant c_0. More precisely, the authors found a differential-linear distinguisher over one iteration of Alzette with correlation $2^{-0.27}$ (experimental verification yielded $2^{-0.1}$), and a rotational differential-linear distinguisher over one iteration of Alzette with correlation $2^{-11.37}$ (experimental verification yielded $2^{-7.35}$).

In [NSLL22], the authors further extended the analysis conducted in [LSL21] with respect to (rotational) differential-linear attacks. Over one iteration of Alzette, they showed a deterministic differential-linear distinguisher and a rotational differential-linear distinguisher of correlation $2^{-5.57}$ (experimental verification yielded $2^{-3.14}$). For two iterations of Alzette, the authors presented a differential-linear distinguisher with correlation $-2^{-8.24}$ (experimental verification yielded $-2^{-5.50}$).

In the work [XLJ+22], the authors presented a refined method to compute probabilities of differential characteristics over ARX ciphers, without using the Markov assumption. For Alzette, they showed that their method can yield more accurate differential probabilities than what is obtained by using the Markov assumption. When applying their approach to Alzette, they were able to find some 4-round differential trails for which the probability estimates obtained using the usual tools were off, either because the probability is in fact much lower (0 instead of 2^{-23} and 2^{-38}), or very slightly underestimated (2^{-22} instead of 2^{-23}). However, these pathological cases are of little importance in practice as the best 4-round trail has a probability of 2^{-6}.

The Bachelor thesis [Spe22] analyzed the security of one round of Sparkle for keyed hashing with respect to differential collision attacks using 2-dimensional querysets. The 2-dimensional queryset with highest differential probability found has differential probability of $1.87 \cdot 2^{-5}$. Further, that work analyzed the existence of dominant linear trails over a 5-round version (instead of the original 4-round version) of Alzette. As a result, there are three dominant linear trails with absolute correlation of $2^{-5}, 2^{-7}$, and 2^{-9}, respectively.

Security margin. Overall, while the third party analysis presented above gives us new information about the behaviour of Sparkle in contexts we had not investigated, we are happy to say that it does not affect the security margin of our algorithms. Indeed, while the attacks against some round-reduced variants now have improved complexities, they do not cover more rounds overall.

Bibliography

[Alr21] Majed Humaid Alrowaithy. *Performance-efficient cryptographic primitives in constrained devices*. PhD thesis, Newcastle University, 2021.

[BLLG$^+$22] Soline Blanc, Abdelkader Lahmadi, Kévin Le Gouguec, Marine Minier, and Lama Sleem. Benchmarking of lightweight cryptographic algorithms for wireless IoT networks. *Wireless Networks*, ??(??):??–??, 2022.

[CGM$^+$22] Hao Cheng, Johann Großschädl, Ben Marshall, Dan Page, and Thinh Pham. RISC-V instruction set extensions for lightweight symmetric cryptography. 5th NIST Workshop on Lightweight Cryptography (LWC 2022), https://csrc.nist.gov/csrc/media/Events/2022/lightweight-cryptography-workshop-2022/documents/papers/risc-v-instruction-set-extensions-for-lightweight-symmetric-cryptography.pdf, 2022.

[HW19] Mingjiang Huang and Liming Wang. Automatic tool for searching for differential characteristics in ARX ciphers and applications. In Feng Hao, Sushmita Ruj, and Sourav Sen Gupta, editors, *INDOCRYPT 2019*, volume 11898 of *Lecture Notes in Computer Science*, pages 115–138. Springer, 2019.

[HW20] Mingjiang Huang and Liming Wang. Automatic search for the linear (hull) characteristics of ARX ciphers: Applied to speck, sparx, chaskey, and CHAM-64. *Secur. Commun. Networks*, 2020:4898612:1–4898612:14, 2020.

[HXW21] Mingjiang Huang, Zhen Xu, and Liming Wang. On the probability and automatic search of rotational-XOR cryptanalysis on ARX ciphers. *The Computer Journal*, 2021.

[LJS$^+$22] Wai-Kong Lee, Kyungbae Jang, Gyeongju Song, Hyunji Kim, Seong Oun Hwang, and Hwajeong Seo. Efficient implementation of lightweight hash functions on gpu and quantum computers for iot applications. *IEEE Access*, 10:59661–59674, 2022.

[LSL21] Yunwen Liu, Siwei Sun, and Chao Li. Rotational cryptanalysis from a differential-linear perspective - practical distinguishers for round-reduced FRIET, Xoodoo, and Alzette. In Anne Canteaut and François-Xavier Standaert, editors, *EUROCRYPT 2021, Part I*, volume 12696 of *Lecture Notes in Computer Science*, pages 741–770. Springer, 2021.

[NSLL22] Zhongfeng Niu, Siwei Sun, Yunwen Liu, and Chao Li. Rotational differential-linear distinguishers of ARX ciphers with arbitrary output linear masks. In Yevgeniy Dodis and Thomas Shrimpton, editors, *CRYPTO 2022*, Lecture Notes in Computer Science. Springer, 2022. (to appear).

[Spe22] Ties Speel. Cryptanalysis of SPARKLE's ARX-box Alzette, 2022. Bachelor Thesis, Radboud University.

[SS22] André Schrottenloher and Marc Stevens. Simplified MITM modeling for permutations: New (quantum) attacks. In Yevgeniy Dodis and Thomas Shrimpton, editors, *CRYPTO 2022*, Lecture Notes in Computer Science. Springer, 2022. to appear.

[WAF17] Daniel Wood, Noah J. Apthorpe, and Nick Feamster. Cleartext data transmissions in consumer IoT medical devices. In *Proceedings of the 2017 Workshop on Internet of Things Security and Privacy, IoT S&P@CCS, Dallas, TX, USA, November 03, 2017*, pages 7–12. ACM, 2017.

[XLJ+22] Zheng Xu, Yongqiang Li, Lin Jiao, Mingsheng Wang, and Willi Meier. Do NOT misuse the markov cipher assumption - automatic search for differential and impossible differential characteristics in ARX ciphers. *IACR Cryptol. ePrint Arch.*, page 135, 2022.