Status Update on Elephant

Tim Beyne[1], Yu Long Chen[1], Christoph Dobraunig[2], and Bart Mennink[3]

[1] KU Leuven and imec-COSIC, Leuven, Belgium
[2] Intel Labs, USA
[3] Radboud University, Nijmegen, The Netherlands
elephant@cs.ru.nl

September 26th, 2022

1 Elephant v2

We first briefly summarize the Elephant authenticated encryption scheme. The mode of Elephant is a nonce-based encrypt-then-MAC construction, where encryption is performed using counter mode and message authentication using a variant of the protected counter sum [2, 29] MAC function. Both modes internally use a cryptographic permutation masked using LFSRs, akin to the masked Even-Mansour construction of Granger et al. [16]. The current Elephant v2 is depicted in Figure 1.

The Elephant v2 mode is permutation-based, only evaluates this permutation in the forward direction, and it is highly parallelizable. Whereas permutations are a popular approach in lightweight cryptography (6 finalists are permutation-based), Elephant v2 is unique in its parallelizability (it is the only finalist satisfying this property). Due to this paralellizability, there is no need for large permutations, we can go as small as 160-bit permutations while still matching the security goals recommended by the NIST lightweight call [30]. In detail, the Elephant family consists of three members:

1. Dumbo: Elephant-Spongent-π[160]. This instance achieves 112-bit security provided that the online complexity is at most around 2^{46} blocks. This instance is particularly well-suited for hardware, as Spongent [10] itself is;
2. Jumbo: Elephant-Spongent-π[176]. This is a slightly more conservative instance of Elephant, and achieves 127-bit security under the same conditions on the online complexity. We note, in particular, that Spongent-π[176] is ISO/IEC standardized [10, 20];
3. Delirium: Elephant-Keccak-f[200]. This variant performs well in both software and hardware. It also achieves 127-bit security, with a higher bound of around 2^{70} blocks on the online complexity. The permutation is the smallest instance that is specified in the NIST SHA-3 standard [3, 15] that fits our needs.

2 New Proofs Supporting the Security Claims

Elephant v1 [6] and v1.1 [8], the predecessors of the current Elephant v2 [7], already came with a tight generic security analysis. The security analysis (of

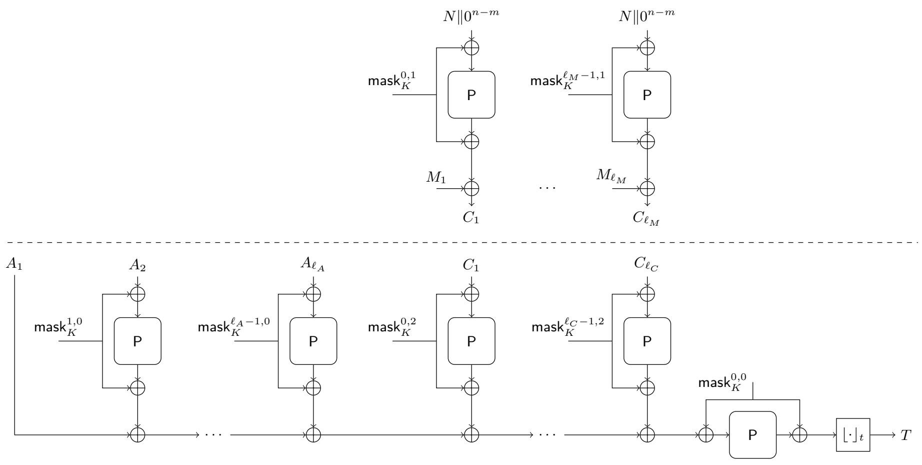

Fig. 1: Depiction of Elephant v2. For the encryption part (top): message is padded as $M_1 \ldots M_{\ell_M} \xleftarrow{n} M$, and ciphertext equals $C = \lfloor C_1 \ldots C_{\ell_M} \rfloor_{|M|}$. For the authentication part (bottom): nonce and associated data are padded as $A_1 \ldots A_{\ell_A} \xleftarrow{n} N\|A\|1$, and ciphertext is padded as $C_1 \ldots C_{\ell_C} \xleftarrow{n} C\|1$.

v1/v1.1) has been published in [4]. The updated specification [7] included a security proof of Elephant v2. In detail, in this specification, the following is proven about the mode of Elephant v2 (in the ideal permutation model):

– confidentiality in the nonce-respecting model;
– authenticity in the nonce-respecting model;
– authenticity in the nonce-misuse model.

In a recent work, multi-user security of Elephant v2 has been analyzed [5], and it was demonstrated that above three security properties *also hold in the multi-user setting* (up to a logical and negligible security loss).

3 Overview of Third-Party Analysis and Implications

Since the original introduction of Elephant, various third-party analyses have appeared. We distinguish between generic attacks on the Elephant mode in Section 3.1, dedicated analysis of Elephant in Section 3.2, and specific analysis on Spongent and Keccak in Section 3.3.

3.1 Generic Cryptanalysis

Bonnetain and Jaques [12] considered quantum security of the Elephant mode (both v1.1 and v2), in the setting where the adversary has classical access to the construction but can make quantum evaluations of the primitive. Although the attack is very interesting, it is not a threat for Elephant: the quantum key recovery attack requires more quantum operations than a direct key search.

3.2 Dedicated Analysis

Zhou et al. [41] derived an interpolation attack against round-reduced Keccak-f, and applied it to Delirium. Their analysis targets a round-reduced version of the encryption of Delirium, where the number of rounds of the used variant of the Keccak permutation is reduced from the specified 18 rounds to 8 rounds. The time complexity of the attack is estimated as $2^{98.3}$ XOR operations, the memory complexity is 2^{70} bits, and the required amount of data 2^{70} blocks. The attack makes use of the fact that an affine space of dimension 65 sums to zero after 6 rounds, which are then followed by 2 rounds for key recovery. Since the encryption of Elephant v1.1 and Elephant v2 are the same (the sole change is in the authentication), the analysis is applicable in a similar manner. We note that algebraic attacks on 8-round Delirium were anticipated in the original Elephant specification [6, 8, Section 5.3].

Vialar presented a side-channel key recovery attack against Dumbo [36], where he uses correlation power analysis to attack the first round of the underlying Spongent permutation. In a nutshell, it recovers the mask $\mathsf{mask}_K^{1,0}$ that is fed into the first round of the permutation, and recovers the key from that. The author also extends the work to Jumbo. Ultimately, this attack is mostly an attack on the unprotected Spongent permutation. In our original specification [6–8], we do not claim provable leakage resilience. When implementing Elephant, it is important to use proper protection of the underlying primitives.

Takemoto et al. [35] present a malicious implementation of Elephant (hardware Trojan) that makes side-channel attacks easier. This design does not have implications on the security of non-malicious implementations.

In [21], a statistical fault attack on an unprotected implementation of Elephant is shown. Since Elephant is not a tamper-resilient design, fault attacks on unprotected implementations are possible. If fault attacks are a threat, it is important to account for implementation-level countermeasures.

3.3 Spongent and Keccak Permutation

An extensive summary of the state-of-the-art cryptanalysis of the Spongent and Keccak permutation is included in the original specification [7, Section 5.3 and Section 5.4]. In this section, we will only discuss *new* articles investing the strength of these two permutation families.

To the best of our knowledge, only two *new* articles investigating the strength of the Spongent permutation have appeared [33, 34]. Sun, Wang and Wang [34] search for bit-based division properties in round-reduced versions of the Spongent permutation. Schrottenloher and Stevens [33] present improved meet-in-the-middle attacks on reduced-round Spongent. These works form no threat to Dumbo and Jumbo (neither v1.1 nor v2).

Furthermore, various works investigating the strength of the Keccak permutation have appeared [9, 11, 14, 17–19, 22–28, 31, 37–40]. These only consider round-reduced versions of the Keccak permutation or usage of this permutation in different modes. They form no threat to Delirium (neither for v1.1 nor for v2).

4 New Implementations

A reference implementation of Dumbo, Jumbo, and Delirium written in C99 can be found at `https://github.com/TimBeyne/Elephant`. We have released an additional parallelized reference implementation for Delirium.[4] This implementation processes up to eight blocks in parallel using an optimized parallel Keccak-f[200] implementation. The Keccak-f[200] implementation was generated using the *KeccakTools* package, by making suitable modifications to the Keccak-f[1600] parameters. Thus, the same strategy can be directly applied to obtain implementations with varying levels of parallelism suited to the target word size.

In the remainder of this section, we discuss new third-party software implementations in Section 4.1, and hardware implementations in Section 4.2, including ones that protect against side channel attacks.

4.1 Software Implementations

Campos et al. [13] provided a C implementation of Delirium which exploits parallelism on platforms with 32-bit words. For messages longer than 64 bytes, their implementation achieves a speedup of about 1.5 to 2 over the original reference implementation. However, we remark that their parallel Keccak-f[200] implementation is based on a reference rather than an optimized implementation. Our own experiments with the new parallel reference implementation suggest that this can have a strong performance impact.

Belaïd et al. [1] introduced a tool to automatically generate masked (and unmasked) bitsliced implementations. They apply the tool to several primitives, including Spongent-π. Their work illustrates that, despite the hardware-oriented nature of Spongent-π, reasonable software performance can be obtained by means of bitslicing. Due to its inherent parallelism, the Elephant mode is a good match for such implementations. The implementation in [1] processes up to eight blocks in parallel, but the target word size (32 bits) would allow increasing this to up to 32 blocks. We also note that there is significant room for improvement in the efficiency of bitsliced implementations of Spongent-π, because alternative representations of Spongent-π could be exploited (as for PRESENT [32]).

4.2 Hardware Implementations

The ATHENa (Automated Tools for Hardware EvaluatioN) project of the GMU summarizes two hardware implementations of Elephant, one by Mueller and Moradi (see `https://github.com/Chair-for-Security-Engineering/LWC-Masking`) and one by Haeussler, Gaj, and Kaps (see `https://github.com/GMUCERG/Elephant`). We refer to `https://cryptography.gmu.edu/athena/LWC/Lab_Implementation_Matching_HW.html` for more information about the evaluation of these hardware implementations.

[4] Available at `https://github.com/TimBeyne/Elephant`.

5 Target Applications

In general, the target application of Elephant is lightweight cryptography with a small footprint, while still providing the option to speed up using parallel implementations. Note that, indeed, Elephant is the candidate in the competition with the *smallest cryptographic primitive*, while still achieving comparable security, and the *only* remaining candidate offering parallelizability. This, together with the different levels of security that the mode achieves, additionally puts Elephant v2 at advantage over the current NIST standards.

References

1. Belaïd, S., Dagand, P., Mercadier, D., Rivain, M., Wintersdorff, R.: Tornado: Automatic Generation of Probing-Secure Masked Bitsliced Implementations. In: Canteaut, A., Ishai, Y. (eds.) EUROCRYPT 2020, Part III. LNCS, vol. 12107, pp. 311–341. Springer (2020)
2. Bernstein, D.J.: How to Stretch Random Functions: The Security of Protected Counter Sums. J. Cryptology 12(3), 185–192 (1999)
3. Bertoni, G., Daemen, J., Peeters, M., Van Assche, G.: The Keccak reference (January 2011)
4. Beyne, T., Chen, Y.L., Dobraunig, C., Mennink, B.: Dumbo, Jumbo, and Delirium: Parallel Authenticated Encryption for the Lightweight Circus. IACR Transactions on Symmetric Cryptology 2020(S1), 5–30 (2020)
5. Beyne, T., Chen, Y.L., Dobraunig, C., Mennink, B.: Multi-user Security of the Elephant v2 Authenticated Encryption Mode. In: AlTawy, R., Hülsing, A. (eds.) SAC 2021. LNCS, vol. 13203, pp. 155–178. Springer (2021)
6. Beyne, T., Chen, Y., Dobraunig, C., Mennink, B.: Elephant v1. First Round Candidate in NIST Lightweight Cryptography (February 25, 2019)
7. Beyne, T., Chen, Y., Dobraunig, C., Mennink, B.: Elephant v2. Finalist in NIST Lightweight Cryptography (May 17, 2021)
8. Beyne, T., Chen, Y., Dobraunig, C., Mennink, B.: Elephant v1.1. Second Round Candidate in NIST Lightweight Cryptography (September 27, 2019)
9. Bi, W., Dong, X., Li, Z., Zong, R., Wang, X.: MILP-aided cube-attack-like cryptanalysis on Keccak Keyed modes. Des. Codes Cryptogr. 87(6), 1271–1296 (2019)
10. Bogdanov, A., Knezevic, M., Leander, G., Toz, D., Varici, K., Verbauwhede, I.: Spongent: A Lightweight Hash Function. In: Preneel, B., Takagi, T. (eds.) CHES 2011. LNCS, vol. 6917, pp. 312–325. Springer (2011)
11. Boissier, R.H., Noûs, C., Rotella, Y.: Algebraic Collision Attacks on Keccak. IACR Transactions on Symmetric Cryptology 2021(1), 239–268 (2021)
12. Bonnetain, X., Jaques, S.: Quantum Period Finding against Symmetric Primitives in Practice. IACR Transactions on Cryptographic Hardware and Embedded Systems. 2022(1), 1–27 (2022)
13. Campos, F., Jellema, L., Lemmen, M., Müller, L., Sprenkels, D., Viguier, B.: Assembly or optimized C for lightweight cryptography on risc-v? In: Krenn, S., Shulman, H., Vaudenay, S. (eds.) CANS 2020. LNCS, vol. 12579, pp. 526–545. Springer (2020)
14. Chen, Y., Gao, X.: Quantum Algorithms for Boolean Equation Solving and Quantum Algebraic Attack on Cryptosystems. Cryptology ePrint Archive, Report 2018/008 (2018)

15. FIPS 202: SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions (August 2015)
16. Granger, R., Jovanovic, P., Mennink, B., Neves, S.: Improved Masking for Tweakable Blockciphers with Applications to Authenticated Encryption. In: Fischlin, M., Coron, J. (eds.) EUROCRYPT 2016. LNCS, vol. 9665, pp. 263–293. Springer (2016)
17. Guo, J., Liao, G., Liu, G., Liu, M., Qiao, K., Song, L.: Practical Collision Attacks against Round-Reduced SHA-3. J. Cryptology 33(1), 228–270 (2020)
18. Guo, J., Liu, G., Song, L., Tu, Y.: Exploring SAT for Cryptanalysis: (Quantum) Collision Attacks against 6-Round SHA-3. IACR Cryptol. ePrint Arch. p. 184 (2022)
19. He, L., Lin, X., Yu, H.: Improved Preimage Attacks on 4-Round Keccak-224/256. IACR Transactions on Symmetric Cryptology 2021(1), 217–238 (2021)
20. ISO/IEC 29192-5:2016. Information technology – Security techniques – Lightweight cryptography – Part 5: Hash-functions (2016)
21. Joshi, P., Mazumdar, B.: Single Event Transient Fault Analysis of ELEPHANT Cipher. CoRR abs/2106.09536 (2021)
22. Kumar, R., Mittal, N., Singh, S.: Cryptanalysis of 2 Round Keccak-384. In: Chakraborty, D., Iwata, T. (eds.) INDOCRYPT 2018. LNCS, vol. 11356, pp. 120–133. Springer (2018)
23. Kumar, R., Rajasree, M.S., AlKhzaimi, H.: Cryptanalysis of 1-Round KECCAK. In: Joux, A., Nitaj, A., Rachidi, T. (eds.) AFRICACRYPT 2018. LNCS, vol. 10831, pp. 124–137. Springer (2018)
24. Li, T., Sun, Y.: Preimage Attacks on Round-Reduced Keccak-224/256 via an Allocating Approach. In: Ishai, Y., Rijmen, V. (eds.) EUROCRYPT 2019, Part III. LNCS, vol. 11478, pp. 556–584. Springer (2019)
25. Li, Z., Dong, X., Bi, W., Jia, K., Wang, X., Meier, W.: New Conditional Cube Attack on Keccak Keyed Modes. IACR Transactions on Symmetric Cryptology 2019(2), 94–124 (2019)
26. Lin, X., He, L., Yu, H.: Improved Preimage Attacks on 3-Round Keccak-224/256. IACR Transactions on Symmetric Cryptology 2021(3), 84–101 (2021)
27. Liu, F., Cao, Z., Wang, G.: Finding Ordinary Cube Variables for Keccak-MAC with Greedy Algorithm. In: Attrapadung, N., Yagi, T. (eds.) IWSEC 2019. LNCS, vol. 11689, pp. 287–305. Springer (2019)
28. Liu, F., Isobe, T., Meier, W., Yang, Z.: Algebraic Attacks on Round-Reduced Keccak. In: Baek, J., Ruj, S. (eds.) ACISP 2021. LNCS, vol. 13083, pp. 91–110. Springer (2021)
29. Luykx, A., Preneel, B., Tischhauser, E., Yasuda, K.: A MAC Mode for Lightweight Block Ciphers. In: Peyrin, T. (ed.) FSE 2016. LNCS, vol. 9783, pp. 43–59. Springer (2016)
30. National Institute of Standards and Technology (NIST): Submission requirements and evaluation criteria for the lightweight cryptography standardization process (Aug 2018)
31. Rajasree, M.S.: Cryptanalysis of round-reduced KECCAK using non-linear structures. In: Hao, F., Ruj, S., Gupta, S.S. (eds.) INDOCRYPT 2019s. LNCS, vol. 11898, pp. 175–192. Springer (2019)
32. Reis, T.B.S., Aranha, D.F., López-Hernández, J.C.: PRESENT Runs Fast - Efficient and Secure Implementation in Software. In: Fischer, W., Homma, N. (eds.) CHES 2017. LNCS, vol. 10529, pp. 644–664. Springer (2017)

33. Schrottenloher, A., Stevens, M.: Simplified MITM Modeling for Permutations: New (Quantum) Attacks. In: Advances in Cryptology - CRYPTO 2022. LNCS, Springer (2022), to appear
34. Sun, L., Wang, W., Wang, M.: MILP-aided bit-based division property for primitives with non-bit-permutation linear layers. IET Inf. Secur. 14(1), 12–20 (2020)
35. Takemoto, S., Ikezaki, Y., Nozaki, Y., Yoshikawa, M.: Hardware Trojan for Lightweight Cryptoraphy Elephant. In: GCCE 2021. pp. 944–945. IEEE (2021)
36. Vialar, L.: Fast Side-Channel Key-Recovery Attack against Elephant Dumbo. Cryptology ePrint Archive, Report 2022/446 (2022)
37. Wang, R., Li, X., Gao, J., Li, H., Wang, B.: Quantum Rotational Cryptanalysis for Preimage Recovery of Round-Reduced Keccak. IACR Cryptol. ePrint Arch. p. 13 (2022)
38. Wei, C., Wu, C., Fu, X., Dong, X., He, K., Hong, J., Wang, X.: Preimage Attacks on 4-Round Keccak by Solving Multivariate Quadratic Systems. In: Park, J.H., Seo, S. (eds.) ICISC 2021. LNCS, vol. 13218, pp. 195–216. Springer (2021)
39. Zhao, Z., Chen, S., Wang, M., Wang, W.: Improved cube-attack-like cryptanalysis of reduced-round Ketje-Jr and Keccak-MAC. Inf. Process. Lett. 171, 106124 (2021)
40. Zhou, H., Li, Z., Dong, X., Jia, K., Meier, W., Ashur, T.: Practical Key-Recovery Attacks On Round-Reduced Ketje Jr, Xoodoo-AE And Xoodyak. Comput. J. 63(8), 1231–1246 (2020)
41. Zhou, H., Zong, R., Dong, X., Jia, K., Meier, W.: Interpolation Attacks on Round-Reduced Elephant, Kravatte and Xoofff. Comput. J. 64(4), 628–638 (2021)