Endorsing AEAD and CTR modes for Keccak

NIST's pending update of FIPS 202 and revision of NIST SP 800-185

Markku-Juhani O. Saarinen

Tampere University, Finland
markku-juhani.saarinen@tuni.fi
October 7, 2024

Abstract. We support NIST's potential plan to specify SHA-3 derived functions ("Keccak Modes") for Authenticated Encryption with Associated Data (AEAD). We offer security and performance arguments for a Keccak-based AEAD as an excellent backup and a way to overcome the limitations of AES-GCM, the main current NIST-specified AEAD. We also suggest standardizing parallelizable counter modes for Keccak and allowing their use for encryption, and in DRBGs, MACs, and XOFs.

Keywords: Keccak · FIPS 202 · NIST SP 800-185 · AEAD · Counter Mode

Motivation

This note is a response to NIST's notice[1] about additional SHA-3 derived functions, dated September 4, 2024.

The SHA-3 and SHAKE functions defined in FIPS 202 [26] and the derived functions in NIST SP 800-185 [22] are all based on the 1600-bit keyless permutation KECCAK-$p[1600, 24]$. As noted in FIPS 202 itself, these functions *"[..] can be considered as modes of operation (modes) of the KECCAK-$p[1600, 24]$ permutation."* Hence we call these potential new SHA-3 derived functions simply as "modes".

Background: Why investing in Keccak hardware now makes sense. The main Post-Quantum Cryptography standards ML-KEM [29] and ML-DSA [28] make extensive use of SHA-3 standards, especially SHAKE. Current processor-based implementations of ML-KEM and ML-DSA on microcontroller and vector architectures spend well over 50% of their cycles just computing the KECCAK-$p[1600, 24]$ permutation [21, 39].

A single invocation of KECCAK-$p[1600, 24]$ requires thousands of cycles to compute on typical embedded and application-class processors, while a straightforward hardware module can accomplish the same task in 24 cycles [33]. As a consequence, the performance of PQC implementations can be almost doubled just by offering dedicated KECCAK acceleration. The acceleration for SHAKE parameter sets of hash-based signature standards SLH-DSA, XMSS, LMS [10, 30] is, of course, even more significant (perhaps 10×.)

Hence, there currently exists a strong motivation for the inclusion of powerful KECCAK acceleration either as a memory-mapped peripheral (for lower-end systems) or as an "all-rounds" instruction in future processor architectures [34]. Note that partial SHA-3 support, such as FEAT_SHA3 instructions in ARM [2], only accelerates a part of a single round, resulting in less significant gains. With increased architectural support, we can expect KECCAK-based AEAD schemes to clearly outperform their AES counterparts, as they do in pure hardware.

[1] NIST Proposes to Update FIPS 202, "SHA-3 Standard" and Revise SP 800-185, "SHA-3 Derived Functions". https://csrc.nist.gov/News/2024/proposal-to-update-fips-202-and-revise-sp-800-185

NIST Already Uses it Pt. 1: Counter Modes for Keccak

Output generation in SHAKE [26], cSHAKE, and TupleHashXOF [22] is sequential due to state chaining from block N to block $N+1$. While it is easy to run SHAKE in an *ad hoc* counter mode by simply concatenating a seed with a counter to generate blocks of output (single KECCAK-$p[1600, 24]$ generates 168 bytes in SHAKE128 and 136 bytes for in SHAKE256), the use of such a system for encryption or random bit generation is not presently allowed by NIST. We suggest standardizing counter mode output and explicitly allowing it to be used for encryption (analogous to AES-CTR [14]), Deterministic Random Bit Generation [3], XOFs, and MACs [22]. For key-derivation functions (KDFs), NIST already describes Keccak-based counter mode [9] – however this KMAC/cSHAKE-based mode seems to require some KECCAK-$p[1600, 24]$ invocations that are not strictly necessary.

We note that lattice-based PQC standards [28, 29] already extensively use the SHAKE XOF for random seed extension and also implicitly describe counter modes for SHAKE. Counters and indices are concatenated with seed inputs for SHAKE-based sampling functions in ML-KEM[29] ($\mathsf{SampleNTT}$, $\mathsf{SamplePolyCBD}_\eta$) and in ML-DSA[28] ($\mathsf{RejNTTPoly}$, $\mathsf{RejBoundedPoly}$). Data-parallelized KECCAK is used to implement these operations in the original AVX2 code [36, 37], as well as in ARM [4], and RISC-V [39] implementations.

NIST Already Uses it Pt. 2: Permutation AEAD Modes

Despite a solid theoretical framework for using permutations for encryption and Authenticated Encryption with Associated Data (AEAD) that predates the SHA-3 standard itself [1, 7], no such mode is currently offered based on the KECCAK permutation.

In the meantime, NIST has selected [38] the permutation-based ASCON family [13] as the upcoming lightweight cryptography standard. The permutation p in ASCON has many similarities to the KECCAK permutation (and was clearly inspired by it) but is made "lightweight" by being only 320 bits in size, compared to 1600 bits of KECCAK and having a reduced number of rounds. The AEAD mode of ASCON is based on the MONKEYDUPLEX construction [8], which was originally proposed for use with the KECCAK permutation.

In addition to a KECCAK mode that is analogous to the ASCON's AEAD mode, we suggest standardizing a parallelizable AEAD mode. NIST may also consider abstract APIs that allow "sessions" that simultaneously provide transcripts of communications and allow lightweight full-duplex protocols [19, 32].

Keccak AEADs: A Safe Alternative to AES AEADs

Limitations of AES. Essentially, all AES [27] modes are subject to a $\approx 2^{61}$-block "birthday bound" for encryption under a given secret key; the wide permutation size of KECCAK allows more long-lived keys. Furthermore, there is sufficient capacity in the KECCAK permutation to accommodate long nonces/IVs together with long sequence numbers. Currently the compromise is often at $\mathsf{nonce} + \mathsf{ctr} = 96 + 32 = 128$ in GCM[16] and CCM[15]. This is one of the reasons why AES-GCM keys are limited to 2^{32} blocks [20, 24].

Keccak seems more secure in the long run. After more than 15 years of intense cryptanalysis, the security margin of KECCAK-$p[1600, 24]$ remains very large. The best relevant attacks apply to at most seven of 24 rounds [18, 23, 35], and halving the number of rounds to 12 would still offer a reasonable security margin [5]. AES, on the other hand, has hardly any security margin left, as is apparent in NIST's own 2021 review [25].

There has been a suggestion to standardize Rijndael with 256-bit block size [17] to address the limitations of AES. This variant has not been cryptanalyzed much since it was proposed in the late 1990s [12]. It can be argued that the 256-bit Rijndael round function

is in some ways "weaker" than the 128-bit round function used in AES, requiring more rounds to reach the same basic random-indistinguishability properties. "Rijndael-256" is likely to require more rounds in addition to a redesigned key schedule. More research effort would be required to reach the same level of confidence in the security of the redesigned "Rijndael-256" that is already enjoyed by KECCAK-$p[1600, 24]$.

Overall, KECCAK-based AEADs offer higher security guarantees than AES-GCM or other AES-based AEADs. This is true for both confidentiality and integrity protection. The authentication tag produced by MONKEYDUPLEX or similar KECCAK-based AEAD modes maps to its actual security level, which is not the case with GCM beyond 64 bits.

Energy Efficiency, Critical Path Length, and Side-Channel Security. A single Keccak permutation is larger than an AES module but "performs the work" of $136/16 = 8.5$ AES-256 invocations or $168/16 = 10.5$ AES-128 invocations. The finite field multiplication in GCM also requires power. The basic hardware efficiency metrics of KECCAK are superior to most other symmetric schemes, including AES. The critical path of AES is made relatively long and inefficient mainly by the complexity of its S-Boxes [31]. Producing each output bit requires fewer logical operations (gates) with KECCAK modes than even with a 10-round AES-128. These S-Boxes also make the constant-time implementation of AES cumbersome compared to the χ function of KECCAK on "pure" software targets.

Most experts agree that the KECCAK permutation is relatively straightforward to protect against power- and emissions-based side-channel attacks. This was one of the original design considerations of KECCAK [6, 11], and there has been much subsequent work. Both ML-KEM and ML-DSA process secret variables using SHAKE, so a secure hardware module is likely to contain a side-channel secure KECCAK in any case.

Bibliography

[1] Elena Andreeva, Joan Daemen, Bart Mennink, and Gilles Van Assche. Security of keyed sponge constructions using a modular proof approach. In Gregor Leander, editor, *Fast Software Encryption - 22nd International Workshop, FSE 2015, Istanbul, Turkey, March 8-11, 2015, Revised Selected Papers*, volume 9054 of *Lecture Notes in Computer Science*, pages 364–384. Springer, 2015. URL: https://doi.org/10.1007/978-3-662-48116-5_18.

[2] ARM. Arm A64 instruction set for A-profile architecture. Guide DDI 0602 (ID092424), ARM, September 2024. URL: https://developer.arm.com/documentation/ddi0602/latest/.

[3] Elaine Barker and John Kelsey. Recommendation for random number generation using deterministic random bit generators. Special Publication SP 800-90A Revision 1, NIST, June 2015. URL: https://doi.org/10.6028/NIST.SP.800-90Ar1.

[4] Hanno Becker, Vincent Hwang, Matthias J. Kannwischer, Bo-Yin Yang, and Shang-Yi Yang. Neon NTT: faster dilithium, kyber, and saber on cortex-a72 and apple M1. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2022(1):221–244, 2022. URL: https://doi.org/10.46586/tches.v2022.i1.221-244.

[5] Guido Bertoni, Joan Daemen, Seth Hoffert, Michaël Peeters, Gilles Van Assche, Ronny Van Keer, and Benoît Viguier. TurboSHAKE. Cryptology ePrint Archive, Paper 2023/342, 2023. URL: https://eprint.iacr.org/2023/342.

[6] Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. Building power analysis resistant implementations of Keccak. August 2010. URL: https://csrc.nist.gov/Events/2010/The-Second-SHA-3-Candidate-Conference.

[7] Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. Duplexing the sponge: Single-pass authenticated encryption and other applications. In Ali Miri and Serge Vaudenay, editors, *Selected Areas in Cryptography - 18th International Workshop, SAC 2011, Toronto, ON, Canada, August 11-12, 2011, Revised Selected Papers*, volume 7118 of *Lecture Notes in Computer Science*, pages 320–337. Springer, 2011. URL: https://doi.org/10.1007/978-3-642-28496-0_19.

[8] Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. Permutation-based encryption, authentication and authenticated encryption. DIAC 2012 Workshop. DIAC - Directions in Authenticated Ciphers (no official proceedings). July 05 - 06, 2012, Stockholm, Sweden, July 2012. URL: https://keccak.team/fles/KeccakDIAC2012.pdf.

[9] Lily Chen. Recommendation for key derivation using pseudorandom functions. Special Publication SP 800-108r1-upd1, NIST, August 2022. URL: https://doi.org/10.6028/NIST.SP.800-108r1-upd1.

[10] David A. Cooper, Daniel C. Apon, Quynh H. Dang, Michael S. Davidson, Morris J. Dworkin, and Carl A. Miller. Recommendation for stateful hash-based signature schemes. Special Publication SP 800-208, NIST, October 2020. URL: https://doi.org/10.6028/NIST.SP.800-208.

[11] Joan Daemen. Changing of the guards: A simple and efficient method for achieving uniformity in threshold sharing. In Wieland Fischer and Naofumi Homma, editors, *Cryptographic Hardware and Embedded Systems - CHES 2017 - 19th International Conference, Taipei, Taiwan, September 25-28, 2017, Proceedings*, volume 10529 of *Lecture Notes in Computer Science*, pages 137–153. Springer, 2017. URL: https://doi.org/10.1007/978-3-319-66787-4_7.

[12] Joan Daemen and Vincent Rijmen. *The Design of Rijndael - The Advanced Encryption Standard (AES), Second Edition.* Information Security and Cryptography. Springer, 2020. URL: https://doi.org/10.1007/978-3-662-60769-5.

[13] Christoph Dobraunig, Maria Eichlseder, Florian Mendel, and Martin Schläffer. Ascon v1.2: Lightweight authenticated encryption and hashing. *J. Cryptol.*, 34(3):33, 2021. URL: https://doi.org/10.1007/s00145-021-09398-9.

[14] Morris J. Dworkin. Recommendation for block cipher modes of operation: Methods and techniques. Special Publication SP 800-38A, NIST, December 2001. URL: https://doi.org/10.6028/NIST.SP.800-38A.

[15] Morris J. Dworkin. Recommendation for block cipher modes of operation: The CCM mode for authentication and confidentiality. Special Publication SP 800-38C, NIST, May 2004. URL: https://doi.org/10.6028/NIST.SP.800-38C.

[16] Morris J. Dworkin. Recommendation for block cipher modes of operation: Galois/Counter Mode (GCM) and GMAC. Special Publication SP 800-38D, NIST, November 2007. URL: https://doi.org/10.6028/NIST.SP.800-38D.

[17] Morris J. Dworkin. NIST plans. Posting on ciphermodes-forum on Aug 9, 2024, August 2024. URL: https://groups.google.com/a/list.nist.gov/g/ciphermodes-forum/c/D5qni2KDoms/m/1UCxIBCVBAAJ.

[18] Jian Guo, Guozhen Liu, Ling Song, and Yi Tu. Exploring SAT for cryptanalysis: (quantum) collision attacks against 6-round SHA-3. In Shweta Agrawal and

Dongdai Lin, editors, *Advances in Cryptology - ASIACRYPT 2022 - 28th International Conference on the Theory and Application of Cryptology and Information Security, Taipei, Taiwan, December 5-9, 2022, Proceedings, Part III*, volume 13793 of *Lecture Notes in Computer Science*, pages 645–674. Springer, 2022. URL: https://doi.org/10.1007/978-3-031-22969-5_22.

[19] Mike Hamburg. The STROBE protocol framework. Cryptology ePrint Archive, Paper 2017/003, 2017. Real World Cryptography 2017. URL: https://eprint.iacr.org/2017/003.

[20] Tetsu Iwata, Keisuke Ohashi, and Kazuhiko Minematsu. Breaking and repairing GCM security proofs. In Reihaneh Safavi-Naini and Ran Canetti, editors, *Advances in Cryptology - CRYPTO 2012 - 32nd Annual Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2012. Proceedings*, volume 7417 of *Lecture Notes in Computer Science*, pages 31–49. Springer, 2012. URL: https://doi.org/10.1007/978-3-642-32009-5_3.

[21] Matthias J. Kannwischer, Richard Petri, Joost Rijneveld, Peter Schwabe, and Ko Stoffelen. PQM4: Post-quantum crypto library for the ARM Cortex-M4. Cryptology ePrint Archive, Paper 2019/844, 2019. Updated library: https://github.com/mupq/pqm4. URL: https://eprint.iacr.org/2019/844.

[22] John Kelsey, Shu jen Chang, and Ray Perlner. SHA-3 derived functions: cSHAKE, KMAC, TupleHash and ParallelHash. Special Publication SP 800-185, NIST, December 2016. URL: https://doi.org/10.6028/NIST.SP.800-185.

[23] Zheng Li, Xiaoyang Dong, Wenquan Bi, Keting Jia, Xiaoyun Wang, and Willi Meier. New conditional cube attack on Keccak keyed modes. *IACR Trans. Symmetric Cryptol.*, 2019(2):94–124, 2019. URL: https://doi.org/10.13154/tosc.v2019.i2.94-124.

[24] Atul Luykx and Kenneth G. Paterson. Limits on authenticated encryption use in TLS. Cryptology ePrint Archive, Paper 2024/051, 2024. (Preprint originally published in 2017). URL: https://eprint.iacr.org/2024/051.

[25] Nicky Mouha. Review of the advanced encryption standard. NIST Internal Report NIST IR 8319, NIST, July 2021. URL: https://doi.org/10.6028/NIST.IR.8319.

[26] NIST. SHA-3 standard: Permutation-based hash and extendable-output functions. Federal Information Processing Standards Publication FIPS 202, NIST, August 2015. URL: https://doi.org/10.6028/NIST.FIPS.202.

[27] NIST. Advanced encryption standard (AES). Federal Information Processing Standards Publication FIPS 197 Update 1, NIST, May 2023. Minor update to original published in 2001. URL: https://doi.org/10.6028/NIST.FIPS.197-upd1.

[28] NIST. Module-lattice-based digital signature standard. Federal Information Processing Standards Publication FIPS 204, NIST, August 2024. URL: https://doi.org/10.6028/NIST.FIPS.204.

[29] NIST. Module-lattice-based key-encapsulation mechanism standard. Federal Information Processing Standards Publication FIPS 203, NIST, August 2024. URL: https://doi.org/10.6028/NIST.FIPS.203.

[30] NIST. Stateless hash-based digital signature standard. Federal Information Processing Standards Publication FIPS 205, NIST, August 2024. URL: https://doi.org/10.6028/NIST.FIPS.205.

[31] Dag Arne Osvik and David Canright. A more compact AES, and more. Cryptology ePrint Archive, Paper 2024/1076, 2024. URL: https://eprint.iacr.org/2024/1076.

[32] Markku-Juhani O. Saarinen. Beyond modes: Building a secure record protocol from a cryptographic sponge permutation. In Josh Benaloh, editor, *Topics in Cryptology - CT-RSA 2014 - The Cryptographer's Track at the RSA Conference 2014, San Francisco, CA, USA, February 25-28, 2014. Proceedings*, volume 8366 of *Lecture Notes in Computer Science*, pages 270–285. Springer, 2014. URL: https://doi.org/10.1007/978-3-319-04852-9_14.

[33] Markku-Juhani O. Saarinen. Accelerating SLH-DSA by two orders of magnitude with a single hash unit. In Leonid Reyzin and Douglas Stebila, editors, *Advances in Cryptology - CRYPTO 2024 - 44th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2024, Proceedings, Part I*, volume 14920 of *Lecture Notes in Computer Science*, pages 276–304. Springer, 2024. URL: https://doi.org/10.1007/978-3-031-68376-3_9.

[34] Markku-Juhani O. Saarinen, G. Richard Newell, and Nicolas Brunie. RISC-V cryptography evolution: High assurance cryptography (HAC TG), post-quantum cryptography (PQC TG). Talk at IACR RWC – Raal World Cryptography Symposium 2024, March 25, Toronto Canada, March 2024. URL: https://iacr.org/submit/files/slides/2024/rwc/rwc2024/75/slides.pdf.

[35] Ling Song, Guohong Liao, and Jian Guo. Non-full sbox linearization: Applications to collision attacks on round-reduced Keccak. In Jonathan Katz and Hovav Shacham, editors, *Advances in Cryptology - CRYPTO 2017 - 37th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 20-24, 2017, Proceedings, Part II*, volume 10402 of *Lecture Notes in Computer Science*, pages 428–451. Springer, 2017. URL: https://doi.org/10.1007/978-3-319-63715-0_15.

[36] Dilithium Team. Dilithium – official reference implementation, 2024. Matches FIPS 204. Author viewed commit cbcd875 in October 2024. URL: https://github.com/pq-crystals/dilithium/.

[37] Kyber Team. Kyber – official reference implementation, 2024. Matches FIPS 203. Author viewed commit 10b478f in October 2024. URL: https://github.com/pq-crystals/kyber.

[38] Meltem Sönmez Turan, Kerry McKay, Donghoon Chang, Lawrence E. Bassham, Jinkeon Kang, Noah D. Waller, John M. Kelsey, and Deukjo Hong. Status report on the final round of the NIST lightweight cryptography standardization process. NIST Internal Report NIST IR 8454, NIST, June 2023. URL: https://doi.org/10.6028/NIST.IR.8454.

[39] Jipeng Zhang, Junhao Huang, Yuxing Yan, and Çetin Kaya Koç. Optimized software implementation of Keccak, Kyber, and Dilithium on RV{32,64}IM{B}{V}. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2025(1):to appear, 2024. URL: https://eprint.iacr.org/2024/1515.