TurboSHAKE

Guido Bertoni[1], Joan Daemen[2], Seth Hoffert, Michaël Peeters[3], Gilles Van Assche[3], Ronny Van Keer[3] and Benoît Viguier[4]

[1] Security Pattern, Italy
[2] Radboud University, The Netherlands
[3] STMicroelectronics, Belgium
[4] ABN AMRO Bank, The Netherlands

Abstract. In a recent presentation, we promoted the use of 12-round instances of KECCAK, collectively called "TurboSHAKE", in post-quantum cryptographic schemes, but without defining them further. The goal of this note is to fill this gap: The definition of the TurboSHAKE family simply consists in exposing and generalizing the primitive already defined inside KANGAROOTWELVE.

Keywords: symmetric cryptography, hashing, Keccak

Cryptography involves careful trade-offs between performance and security. In symmetric cryptography, an important such trade-off is the choice of the number of rounds, which on the one hand is proportional to the amount of time taken to evaluate a primitive and on the other needs to be high enough to provide safety margin against possible progresses in attacks. Ideally, this choice is driven by cryptanalysis on round-reduced versions, but cryptanalysis requires hard work by cryptographic experts. Fortunately, KECCAK has received quite a large amount of cryptanalysis since its publication; in fact, KECCAK has seen more scientific publications on cryptanalysis than any other unbroken hash function to this date. In the light of these publications, we feel we can confidently propose to halve the number of rounds without compromising security.

It seems clear that a round-reduced version of the KECCAK sponge function can be useful in many cases. We proposed a few years ago an extendable output function (XOF), called KANGAROOTWELVE (or K12 for short), with the explicit goal of being able to build upon existing cryptanalysis, instead of creating a new design that would require fresh one [10]. More recently, for post-quantum cryptography, the NIST selected a number of public-key schemes that they will standardize and that use instances of KECCAK internally [62]. In some cases, the time spent by these schemes is dominated by the evaluation of the sponge function, and so logically we brought up the idea of reducing the number of rounds for this use-case [11]. Despite NIST's decision to stick to the nominal number of rounds, we believe the interest remains in general.

We observe that there are already quite a large number of named instances of KECCAK—besides the four SHA-3 hash functions and the two SHAKE XOFs in FIPS 202, the NIST SP 800-185 standard defines a few more [60, 61]. We therefore wish to keep the number of new definitions to a minimum. In fact, we are not defining anything really new: We are making the primitive inside K12 available and more broadly usable.

1 Specifications of TurboSHAKE

TurboSHAKE is a family of eXtendable Output Functions (XOF) parameterized by their capacity c, where the capacity directly relates to the claimed security level as detailed in Section 2. We restrict the capacity c to multiples of 8 not greater than 512.

A given instance, denoted TurboSHAKE[c], takes as input:

- a message M, a byte string of variable length, and
- a domain separation parameter D, a byte with a value in the range [$\texttt{0x01}, \ldots, \texttt{0x7F}$] in hexadecimal.

As a XOF, the output of TurboSHAKE[c] is unlimited, and the user can request as many output bits as desired. It can be used for traditional hashing simply by generating outputs of the desired digest size.

TurboSHAKE produces unrelated outputs on different tuples (c, M, D). For a given capacity, the value D is meant to provide domain separation, that is, for two different values $D_1 \neq D_2$, TurboSHAKE[c]$(\cdot, D_1)$ and TurboSHAKE[c]$(\cdot, D_2)$ act as two independent functions of M. We believe the range of D to be sufficient to cover all use cases.

Users that do not require multiple instances can take as default $D = \texttt{0x1F}$.

Named instances In addition, we define:

- TurboSHAKE128 as TurboSHAKE[$c = 256$], and
- TurboSHAKE256 as TurboSHAKE[$c = 512$].

Procedure To compute TurboSHAKE[c](M, D), proceed as follows. Let $R = 200 - c/8$ be the rate in bytes and f the Keccak-$p[1600, n_\mathrm{r} = 12]$ permutation [60].

1. Input preparation
   (a) Append to M the byte D, followed by the minimum number of bytes $\texttt{0x00}$ (possibly none) until $M' = M||D||\texttt{0x00}^*$ has a length that is multiple of R bytes.
   (b) Bitwise add (XOR) the byte $\texttt{0x80}$ into the last byte of M'.
   (c) Cut M' into m blocks of R bytes each, i.e., $M' = M_1|| \ldots ||M_m$.
2. Absorbing phase
   (a) Let $S = \texttt{0x00}^{200}$.
   (b) For each block M_i for $i = 1$ to m:
      i. Let $S \leftarrow f(S \oplus (M_i||\texttt{0x00}^{200-R}))$.
3. Squeezing phase
   (a) Repeat as long as necessary:
      i. Output the first R bytes of S.
      ii. Let $S \leftarrow f(S)$.
   (b) Truncate the output if longer than needed.

2 Security claim

We make a flat sponge claim [6] with c bits of claimed capacity in Claim 1. Informally, it means that TurboSHAKE shall offer the same security strength as a random oracle whenever that offers a strength below $c/2$ bits and a strength of $c/2$ bits in all other cases.

Claim 1 (Flat sponge claim [6]). *The success probability of any attack on TurboSHAKE[c] shall not be higher than the sum of that for a random oracle and* $1 - e^{-\frac{N^2}{2^{c+1}}}$, *with* N *the attack complexity in calls to* KECCAK-$p[1600, n_r = 12]$ *or its inverse. We exclude from the claim weaknesses due to the mere fact that the function can be described compactly and can be efficiently executed, e.g., the so-called random oracle implementation impossibility [52], as well as properties that cannot be modeled as a single-stage game [67].*

The flat sponge claim covers all attacks against TurboSHAKE[c] up to a given security strength of $c/2$ bits. Informally, saying that a cryptographic function has a security strength of s bits means that no attacks exist with complexity N and success probability p such that $N/p < 2^s$ [54]. For more details on the interpretation of the claim, we refer to [10, Section 4.1].

3 Rationale

In this section, we exhibit the equivalence with KECCAK reduced to 12 rounds, motivate our security claim and clarify its use in KANGAROOTWELVE.

Equivalence Consider the sponge function on top of the KECCAK-$p[1600, n_r = 12]$ permutation, with multi-rate padding pad10*1, capacity c and rate $r = 1600 - c$, as defined in the FIPS 202 standard [60], and let us call it $\mathcal{TS}_c$ for short, i.e.,

$$\mathcal{TS}_c \triangleq \text{SPONGE}[\text{KECCAK-}p[1600, n_r = 12], \text{pad}10^*1, r = 1600 - c] \ .$$

In comparison, note that the standard KECCAK is defined the same way, except for the number of rounds, i.e.,

$$\text{KECCAK}[c] = \text{SPONGE}[\text{KECCAK-}p[1600, n_r = 24], \text{pad}10^*1, r = 1600 - c] \ .$$

Then, TurboSHAKE[c](M, D) is equivalent to $\mathcal{TS}_c(M||\text{unpad}(D))$, where

- D and each byte of M is interpreted as a string of 8 bits, from the least to the most significant bit of the byte;
- unpad(D) removes the trailing bits '`0`' of D, if any, then the last bit '`1`' (e.g., unpad(`0x01`) is the empty string, unpad(`0x0B`) = '`110`').

Note that unpad(D) is not defined for strings of only zeroes, but that does not pose a problem for D as it has at least one '`1`'. We can view D as a string of bits that is padded with the pad10* padding rule. Here, $\mathcal{TS}_c$ uses the multi-rate padding rule pad10*1 instead. Multi-rate padding appends an initial '`1`'-bit, then zeroes and then a final '`1`'-bit. In the pseudocode in Section 1, the initial '`1`'-bit is contained in the encoding of D, and the final one is materialized by XORing `0x80` into the last byte of M'. Thanks to the fact that the last bit of parameter D is '`0`', setting the final bit of M' to '`1`'-bit corresponds with XORing `0x80` into the last byte of M'.

The default value for D, namely `0x1F`, is such that unpad(D) = '`1111`' and so TurboSHAKE128 and TurboSHAKE256 coincide with round-reduced SHAKE128 and SHAKE256, respectively.

Security Changing the number of rounds in the underlying permutation from 24 in the SHA-3 standard functions to 12 in TurboSHAKE implies a drastic reduction in safety margin. Still, TurboSHAKE is a reduced-round version of KECCAK and thereby directly benefits from all the cryptanalysis on the latter. There is ample evidence from third-party cryptanalysis that 12 rounds provides a comfortable security margin [2, 3, 41, 4, 15, 14, 59, 25, 16, 20, 26, 57, 40, 21, 56, 58, 19, 22, 37, 24, 55, 23, 36, 1, 30, 44, 71, 64, 68, 45, 34, 42, 69, 17, 38, 39, 78, 70, 77, 35, 51, 66, 49, 46, 43, 27, 12, 72, 50, 28, 13, 47, 31, 76, 74, 33, 32, 63, 29, 75, 65, 48], as well as from our own investigations [8, 7, 18, 9, 53].

We maintain a list of cryptanalysis results on our ciphers [5]. At the time of this writing, the best collision attack applicable to TurboSHAKE or to any SHA-3 instance works only when the permutation is reduced to 6 rounds [71, 29], and preimage attacks reach only 4 rounds [31, 74, 65]. Hence our proposal has a safety margin of 6 out of 12 rounds for collision and (second) preimage resistance.

Currently, the structural distinguisher that reaches the highest number of rounds is called SymSum and works on Keccak reduced to 9 rounds [72]. This distinguisher consider self-symmetric strings of bits, that is, strings of the form $X||X||Y||Y||\dots||Z||Z$, where X, Y and Z are 32-bit strings. The SymSum distinguisher produces a set of self-symmetric strings such that the set of corresponding outputs through 9-round Keccak sums to a self-symmetric string.

Finally, we limit the supported capacity to at most 512 bits, as we do not think that it makes much sense to claim more than 256 bits of security.

KangarooTwelve vs TurboSHAKE128 K12 is a XOF that is defined on top of the Keccak-$p[1600, n_r = 12]$ permutation [10, 73]. In its specifications, K12 uses a tree hash mode on top of a function called "F", which is exactly $\mathcal{TS}_{256}$, and hence K12 can be equivalently recast as *a mode on top of TurboSHAKE128* instead.

K12 uses TurboSHAKE128 with three values for D, namely, $D \in \{\texttt{0x06}, \texttt{0x07}, \texttt{0x0B}\}$. For a protocol that uses both K12 and TurboSHAKE128, it is therefore recommended to avoid using these three values for D.

Acknowledgments

We wish to thank all the researchers who analyzed Keccak and contributed to its understanding. This is what allowed us to confidently propose schemes based on round-reduced Keccak.

References

[1] Matthew Amy, Olivia Di Matteo, Vlad Gheorghiu, Michele Mosca, Alex Parent, and John M. Schanck, *Estimating the Cost of Generic Quantum Pre-image Attacks on SHA-2 and SHA-3*, Selected Areas in Cryptography, 2016, pp. 317–337.

[2] Jean-Philippe Aumasson and Dmitry Khovratovich, *First Analysis of Keccak*, NIST hash forum (2009).

[3] Jean-Philippe Aumasson and Willi Meier, *Zero-sum distinguishers for reduced Keccak-f and for the core functions of Luffa and Hamsi*, rump session of CHES, 2009.

[4] Daniel J. Bernstein, *Second preimages for 6 (7? (8??)) rounds of Keccak?*, NIST hash forum (2010).

[5] Guido Bertoni, Joan Daemen, Seth Hoffert, Michaël Peeters, Gilles Van Assche, and Ronny Van Keer, *Third-party cryptanalysis*, 2023, https://keccak.team/third_party.html.

[6] Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche, *Cryptographic sponge functions*, SHA-3 competition (round 3) (2011).

[7] ———, *On alignment in Keccak*, Ecrypt II Hash Workshop, 2011.

[8] ———, *The Keccak reference*, SHA-3 competition (round 3) (2011).

[9] ———, *The Making of Keccak*, Cryptologia **38** (2014), no. 1, 26–60.

[10] Guido Bertoni, Joan Daemen, Michaël Peeters, Gilles Van Assche, Ronny Van Keer, and Benoît Viguier, *KangarooTwelve: Fast Hashing Based on Keccak-p*, ACNS, 2018, pp. 400–418.

[11] Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche, *Twelve-round* KECCAK *for secure hashing*, NIST Fourth PQC Standardization Conference, 2022.

[12] Wenquan Bi, Xiaoyang Dong, Zheng Li, Rui Zong, and Xiaoyun Wang, *MILP-aided Cube-attack-like Cryptanalysis on Keccak Keyed Modes*, Des. Codes Cryptography **87** (2019), no. 6, 1271–1296.

[13] Rachelle Heim Boissier, Camille Noûs, and Yann Rotella, *Algebraic Collision Attacks on Keccak*, IACR Trans. Symmetric Cryptol. **2021** (2021), no. 1, 239–268.

[14] Christina Boura and Anne Canteaut, *A zero-sum property for the Keccak-f permutation with 18 rounds*, ISIT, 2010, pp. 2488–2492.

[15] ———, *Zero-Sum Distinguishers for Iterated Permutations and Application to Keccak-f and Hamsi-256*, Selected Areas in Cryptography, 2010, pp. 1–17.

[16] Christina Boura, Anne Canteaut, and Christophe De Cannière, *Higher-Order Differential Properties of Keccak and Luffa*, Fast Software Encryption, 2011, pp. 252–269.

[17] Yu-Ao Chen and Xiao-Shan Gao, *Quantum Algorithms for Boolean Equation Solving and Quantum Algebraic Attack on Cryptosystems*, IACR Cryptology ePrint Archive **2018** (2018), 8.

[18] Joan Daemen and Gilles Van Assche, *Differential Propagation Analysis of Keccak*, Fast Software Encryption, 2012, pp. 422–441.

[19] Sourav Das and Willi Meier, *Differential Biases in Reduced-Round Keccak*, Africacrypt, 2014, pp. 69–87.

[20] Itai Dinur, Orr Dunkelman, and Adi Shamir, *New Attacks on Keccak-224 and Keccak-256*, FSE, 2012, pp. 442–461.

[21] ———, *Collision Attacks on Up to 5 Rounds of SHA-3 Using Generalized Internal Differentials*, FSE, 2013, pp. 219–240.

[22] Itai Dinur, Paweł Morawiecki, Josef Pieprzyk, Marian Srebrny, and Michał Straus, *Practical Complexity Cube Attacks on Round-Reduced Keccak Sponge Function*, IACR Cryptology ePrint Archive **2014** (2014), 259.

[23] ———, *Cube Attacks and Cube-Attack-Like Cryptanalysis on the Round-Reduced Keccak Sponge Function*, Eurocrypt, 2015, pp. 733–761.

[24] Christoph Dobraunig, Maria Eichlseder, and Florian Mendel, *Heuristic Tool for Linear Cryptanalysis with Applications to CAESAR Candidates*, Asiacrypt, 2015, pp. 490–509.

[25] Ming Duan and Xuejia Lai, *Improved zero-sum distinguisher for full round Keccak-f permutation*, IACR Cryptology ePrint Archive **2011** (2011), 23.

[26] Alexandre Duc, Jian Guo, Thomas Peyrin, and Lei Wei, *Unaligned Rebound Attack: Application to Keccak*, Fast Software Encryption, 2012, pp. 402–421.

[27] Sergij V. Goncharov, *Using fuzzy bits and neural networks to partially invert few rounds of some cryptographic hash functions*, CoRR **abs/1901.02438** (2019).

[28] Jian Guo, Guohong Liao, Guozhen Liu, Meicheng Liu, Kexin Qiao, and Ling Song, *Practical Collision Attacks against Round-Reduced SHA-3*, J. Cryptol. **33** (2020), no. 1, 228–270.

[29] Jian Guo, Guozhen Liu, Ling Song, and Yi Tu, *Exploring SAT for Cryptanalysis: (Quantum) Collision Attacks Against 6-Round SHA-3*, Asiacrypt, 2022, pp. 645–674.

[30] Jian Guo, Meicheng Liu, and Ling Song, *Linear Structures: Applications to Cryptanalysis of Round-Reduced Keccak*, Asiacrypt, 2016, pp. 249–274.

[31] Le He, Xiaoen Lin, and Hongbo Yu, *Improved Preimage Attacks on 4-Round Keccak-224/256*, IACR Trans. Symmetric Cryptol. **2021** (2021), no. 1, 217–238.

[32] ———, *Improved Preimage Attacks on Round-Reduced Keccak-384/512 via Restricted Linear Structures*, IACR Cryptol. ePrint Arch. **2022** (2022), 788.

[33] Senyang Huang, Orna Agmon Ben-Yehuda, Orr Dunkelman, and Alexander Maximov, *Finding Collisions against 4-Round SHA-3-384 in Practical Time*, IACR Trans. Symmetric Cryptol. **2022** (2022), no. 3, 239–270.

[34] Senyang Huang, Xiaoyun Wang, Guangwu Xu, Meiqin Wang, and Jingyuan Zhao, *Conditional Cube Attack on Reduced-Round Keccak Sponge Function*, Eurocrypt, 2017, pp. 259–288.

[35] ———, *New Distinguisher on Reduced-Round Keccak Sponge Function*, IEICE Trans. Fundam. Electron. Commun. Comput. Sci. **102-A** (2019), no. 1, 242–250.

[36] Jérémy Jean and Ivica Nikolić, *Internal Differential Boomerangs: Practical Analysis of the Round-Reduced Keccak-f Permutation*, Fast Software Encryption, 2015, pp. 537–556.

[37] Sukhendu Kuila, Dhiman Saha, Madhumangal Pal, and Dipanwita Roy Chowdhury, *Practical Distinguishers against 6-Round Keccak-f Exploiting Self-Symmetry*, Africacrypt, 2014, pp. 88–108.

[38] Rajendra Kumar, Nikhil Mittal, and Shashank Singh, *Cryptanalysis of 2 Round Keccak-384*, Indocrypt, 2018, pp. 120–133.

[39] Rajendra Kumar, Mahesh Sreekumar Rajasree, and Hoda AlKhzaimi, *Cryptanalysis of 1-Round Keccak*, Africacrypt, 2018, pp. 124–137.

[40] Stefan Kölbl, Florian Mendel, Tomislav Nad, and Martin Schläffer, *Differential Cryptanalysis of Keccak Variants*, IMA Int. Conf., 2013, pp. 141–157.

[41] Joel Lathrop, *Cube Attacks on Cryptographic Hash Functions*, Master's thesis, Rochester Institute of Technology, 2009.

[42] Maolin Li and Lu Cheng, *Distinguishing Property for Full Round Keccak-f Permutation*, CISIS, 2017, pp. 639–646.

[43] Ting Li and Yao Sun, *Preimage Attacks on Round-Reduced Keccak-224/256 via an Allocating Approach*, Eurocrypt, 2019, pp. 556–584.

[44] Ting Li, Yao Sun, Maodong Liao, and Dingkang Wang, *Preimage Attacks on the Round-reduced Keccak with Cross-linear Structures*, IACR Trans. Symmetric Cryptol. **2017** (2017), no. 4, 39–57.

[45] Zheng Li, Wenquan Bi, Xiaoyang Dong, and Xiaoyun Wang, *Improved Conditional Cube Attacks on Keccak Keyed Modes with MILP Method*, Asiacrypt, 2017, pp. 99–127.

[46] 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** (2019), no. 2, 94–124.

[47] Xiaoen Lin, Le He, and Hongbo Yu, *Improved Preimage Attacks on 3-Round Keccak-224/256*, IACR Trans. Symmetric Cryptol. **2021** (2021), no. 3, 84–101.

[48] ———, *Practical Preimage Attack on 3-Round Keccak-256*, IACR Cryptol. ePrint Arch. **2023** (2023), 101.

[49] Fukang Liu, Zhenfu Cao, and Gaoli Wang, *Finding Ordinary Cube Variables for Keccak-MAC with Greedy Algorithm*, IWSEC, 2019, pp. 287–305.

[50] Fukang Liu, Takanori Isobe, Willi Meier, and Zhonghao Yang, *Algebraic Attacks on Round-Reduced Keccak/Xoodoo*, IACR Cryptol. ePrint Arch. **2020** (2020), 346.

[51] Guozhen Liu, Weidong Qiu, and Yi Tu, *New Techniques for Searching Differential Trails in Keccak*, IACR Trans. Symmetric Cryptol. **2019** (2019), no. 4, 407–437.

[52] Ueli M. Maurer, Renato Renner, and Clemens Holenstein, *Indifferentiability, Impossibility Results on Reductions, and Applications to the Random Oracle Methodology*, TCC, 2004, pp. 21–39.

[53] Silvia Mella, Joan Daemen, and Gilles Van Assche, *New techniques for trail bounds and application to differential trails in Keccak*, IACR Trans. Symmetric Cryptol. **2017** (2017), no. 1, 329–357.

[54] Daniele Micciancio and Michael Walter, *On the Bit Security of Cryptographic Primitives*, Eurocrypt, 2018, pp. 3–28.

[55] Paweł Morawiecki, *Malicious Keccak*, IACR Cryptology ePrint Archive **2015** (2015), 1085.

[56] Paweł Morawiecki, Josef Pieprzyk, and Marian Srebrny, *Rotational Cryptanalysis of Round-Reduced Keccak*, FSE, 2013, pp. 241–262.

[57] Paweł Morawiecki, Josef Pieprzyk, Marian Srebrny, and Michał Straus, *Preimage attacks on the round-reduced Keccak with the aid of differential cryptanalysis*, IACR Cryptology ePrint Archive **2013** (2013), 561.

[58] Paweł Morawiecki and Marian Srebrny, *A SAT-based preimage analysis of reduced Keccak hash functions*, Inf. Process. Lett. **113** (2013), no. 10-11, 392–397.

[59] María Naya-Plasencia, Andrea Röck, and Willi Meier, *Practical Analysis of Reduced-Round Keccak*, Indocrypt, 2011, pp. 236–254.

[60] NIST, *Federal information processing standard 202, SHA-3 standard: Permutation-based hash and extendable-output functions*, August 2015, http://dx.doi.org/10.6028/NIST.FIPS.202.

[61] ———, *NIST special publication 800-185, SHA-3 derived functions: cSHAKE, KMAC, TupleHash and ParallelHash*, December 2016, https://doi.org/10.6028/NIST.SP.800-185.

[62] ———, *Post-quantum cryptography, selected algorithms 2022*, 2022, https://csrc.nist.gov/Projects/post-quantum-cryptography/selected-algorithms-2022.

[63] Richard Preston, *Applying Grover's Algorithm to Hash Functions: A Software Perspective*, CoRR **abs/2202.10982** (2022).

[64] Kexin Qiao, Ling Song, Meicheng Liu, and Jian Guo, *New Collision Attacks on Round-Reduced Keccak*, Eurocrypt, 2017, pp. 216–243.

[65] Lingyue Qin, Jialiang Hua, Xiaoyang Dong, Hailun Yan, and Xiaoyun Wang, *Meet-in-the-Middle Preimage Attacks on Sponge-based Hashing*, IACR Cryptol. ePrint Arch. **2022** (2022), 1714.

[66] Mahesh Sreekumar Rajasree, *Cryptanalysis of Round-Reduced Keccak using Non-Linear Structures*, Indocrypt, 2019, pp. 175–192.

[67] Thomas Ristenpart, Hovav Shacham, and Thomas Shrimpton, *Careful with Composition: Limitations of the Indifferentiability Framework*, Eurocrypt, 2011, pp. 487–506.

[68] Dhiman Saha, Sukhendu Kuila, and Dipanwita Roy Chowdhury, *SymSum: Symmetric-Sum Distinguishers Against Round Reduced SHA3*, IACR Trans. Symmetric Cryptol. **2017** (2017), no. 1, 240–258.

[69] Ling Song and Jian Guo, *Cube-Attack-Like Cryptanalysis of Round-Reduced Keccak Using MILP*, IACR Trans. Symmetric Cryptol. **2018** (2018), no. 3, 182–214.

[70] Ling Song, Jian Guo, Danping Shi, and San Ling, *New MILP Modeling: Improved Conditional Cube Attacks on Keccak-Based Constructions*, Asiacrypt, 2018, pp. 65–95.

[71] Ling Song, Guohong Liao, and Jian Guo, *Non-Full Sbox Linearization: Applications to Collision Attacks on Round-Reduced Keccak*, Crypto, 2017, pp. 428–451.

[72] Sahiba Suryawanshi, Dhiman Saha, and Satyam Sachan, *New Results on the SymSum Distinguisher on Round-Reduced SHA3*, Africacrypt, 2020, pp. 132–151.

[73] Benoît Viguier, David Wong, Gilles Van Assche, Quynh Dang, and Joan Daemen, *KangarooTwelve*, Internet Research Task Force draft, August 2022, https://datatracker.ietf.org/doc/draft-irtf-cfrg-kangarootwelve/.

[74] Runsong Wang, Xuelian Li, Juntao Gao, Hui Li, and Baocang Wang, *Allocating Rotational Cryptanalysis based Preimage Attack on 4-round Keccak-224 for Quantum Setting*, IACR Cryptol. ePrint Arch. **2022** (2022), 977.

[75] ———, *Quantum rotational cryptanalysis for preimage recovery of round-reduced Keccak*, IACR Cryptol. ePrint Arch. **2022** (2022), 13.

[76] Congming Wei, Chenhao Wu, Ximing Fu, Xiaoyang Dong, Kai He, Jue Hong, and Xiaoyun Wang, *Preimage attacks on 4-round Keccak by solving multivariate quadratic systems*, IACR Cryptol. ePrint Arch. **2021** (2021), 732.

[77] Hailun Yan, Xuejia Lai, Lei Wang, Yu Yu, and Yiran Xing, *New zero-sum distinguishers on full 24-round Keccak-f using the division property*, IET Inf. Secur. **13** (2019), no. 5, 469–478.

[78] Chen-Dong Ye and Tian Tian, *New Insights into Divide-and-Conquer Attacks on the Round-Reduced Keccak-MAC*, IACR Cryptology ePrint Archive **2018** (2018), 059.