Security Evaluation of SHA-3

Report prepared for the CRYPTREC Project

Itai Dinur
Département d'Informatique, École Normale Supérieure,
Paris, France

February 26, 2015

Abstract

Draft FIPS PUB 202 [32] specifies the Secure Hash Algorithm-3 (SHA-3) family of functions. The SHA-3 functions are based on instances of the Keccak algorithm that NIST selected as the winner of the SHA-3 cryptographic hash algorithm competition. The SHA-3 family consists of four cryptographic hash functions, SHA3-224, SHA3-256, SHA3-384, and SHA3-512, and two extendable-output functions (XOFs), SHAKE128 and SHAKE256.

This evaluation report summarizes the cryptanalysis published during the SHA-3 competition and afterwards on the Keccak algorithm.

Contents

Chapter 1

Executive Summary

1.1 SHA-3 and Keccak

Keccak is the winner of the NIST SHA-3 competition. It is a family of functions that can be used to generate arbitrary output length from an input of variable size. Keccak uses a mode of operation called the *sponge construction* (formally defined in [7]), which builds a function mapping variable-length input to variable-length output using a fixed-length permutation and a padding rule. The Keccak permutation works on a state of 1600 bits[1] using a sequence of 24 rounds. Four instances of Keccak, with output lengths 224, 256, 384 and 512 bits, were submitted to the SHA-3 competition [9].

Draft FIPS PUB 202 [32] specifies the actual SHA-3 family, containing 6 functions, all of which are directly based on Keccak instances (where the only difference between the SHA-3 functions and the underlying Keccak instances is the concatenation of a few bits to the message). The SHA-3 family contains 4 cryptographic hash functions, named according to their output length: SHA3-224, SHA3-256, SHA3-384, and SHA3-512. These 4 hash functions are directly based on the 4 Keccak instances that were submitted to the SHA-3 competition (Keccak[448], Keccak[512], Keccak[768] and Keccak[1024], respectively).[2]

Draft FIPS PUB 202 also specifies two extendable-output functions (XOFs), SHAKE128 and SHAKE256. These functions use the Keccak family of sponge functions to offer additional flexibility, and in particular, can be defined with an arbitrary output length d. However, the two XOFs are not based on the 4 Keccak instances that were submitted to the SHA-3 competition, but rather on Keccak instances with different parameters. In fact, each choice of XOF

[1] Keccak is also defined on smaller states, but these instances are not used for SHA-3, and are out of the scope of this report.

[2] In this report, we name the Keccak instances according to Draft FIPS PUB 202, but note that these instances were typically named Keccak-n in previous analysis, where n specifies the output length (e.g., Keccak[448] was previous named Keccak-224).

(SHAKE128 or SHAKE256) and a value of d can be viewed as an instantiation of a separate Keccak instance.

1.2 Security of SHA-3

1.2.1 Formal Security Claims

Standard applications of hash functions typically require core security properties of *collision resistance*, *preimage resistance*, and *second preimage resistance*. Given a hash function H with an n-bit output, these core security properties are defined below.

1. **Collision resistance**: It should be difficult to find a pair of different messages m_1 and m_2 such that $H(m_1) = H(m_2)$.
2. **Preimage resistance**: Given an arbitrary n-bit value x, it should be difficult to find any message m such that $H(m) = x$.
3. **Second preimage resistance**: Given message m_1, it should be difficult to find any different message m_2 such that $H(m_1) = H(m_2)$.

The security strengths of the 6 SHA-3 functions as claimed in Draft FIPS PUB 202 are summarized in Table 1.1.

Function	Output Size	Collision Resistance	Preimage Resistance	Second Preimage Resistance
SHA3-224	224	112	224	224
SHA3-256	256	128	256	256
SHA3-348	348	192	348	348
SHA3-512	512	256	512	512
SHAKE128	d	$min(d/2, 128)$	$min(d, 128)$	$min(d, 128)$
SHAKE256	d	$min(d/2, 256)$	$min(d, 256)$	$min(d, 256)$

Table 1.1: Security Strengths of SHA-3 Functions (in bits)

1.2.2 Security Analysis

When analyzing the security of a hash function, one typically starts by considering its mode of operation, namely, the sponge construction in case of SHA-3. The sponge construction provides formal proofs on security for any hash function [4], assuming that its internal permutation has no apparent weakness (it is "ideal"). In particular, these proofs of security apply to all Keccak instances, assuming that Keccak's internal permutation is ideal. Since the six SHA-3

functions are based on instances of Keccak, the security proofs of [4] apply to the SHA-3 functions as well, namely, the functions provide the security level claimed in Table 1.1, assuming that Keccak's internal permutation is ideal. On the other hand, there is no formal proof of security for Keccak's internal permutation,[3] and therefore attacks on Keccak (and on SHA-3) focus on trying to find weaknesses in its permutation.

Cryptanalysis of SHA-3 is roughly divided into three types.

1. **Type 1**: Attacks on the core security properties of the hash function, as given in Table 1.1. Most applications rely on core security properties, and these are considered as the most important security properties of any hash function.

2. **Type 2**: Distinguishers that find a property of the hash function which distinguishes it from a random function (such as a bias in its output). Although weaker than attacks of type 1, more general randomness properties of hash functions are required by some applications, and it is important to ensure that the hash function does not exhibit undesired properties in general.

3. **Type 3**: Distinguishers that find a property of the internal permutation of SHA-3 which distinguishes it from a random permutation. Since SHA-3 uses the internal permutation in a specific way (e.g., by restricting part of the permutation's input), a distinguisher of type 3 does not necessarily lead to a distinguisher on the hash function (of type 2) that can be observed at the output. Excluding some exceptions, distinguishers of this type have little influence on the security of the hash function in practice.

We note that the theoretical attack models for key-less hash functions are generally not well-defined. This is especially true for cryptanalysis of types 2 and 3, as one needs to be careful with the definition of a "distinguisher". However, in this report we generally avoid these issues, focusing on the significance of published cryptanalysis to the security of the hash function in practice.

While this report considers all 3 types of cryptanalysis, we note that Draft FIPS PUB 202 formalizes security claims for SHA-3 only against attacks of type 1 (as summarized in Table 1.1). It should be further noted that there are relations between the three types of cryptanalysis, and while the classification above helps to understand the impact of a given attack (or distinguisher), it is also important to examine its internal details. For example, some differential distinguishers of type 3 can be converted into collision attacks on type 1 when combined with additional cryptanalytic techniques. This is discussed in more detail in Chapter 2.

We stress that this report evaluates the security of the SHA-3 instances, but does not deal with how to embed them into various applications.

[3] Keccak is not different in this aspect from any modern hash function.

Attacks on Core Security Properties

Since there is no published attack that breaks the core security properties of Keccak (or the properties of SHA-3, as specified in Table 1.1), its cryptanalysis focuses on weaker instances in which the number of permutation rounds is reduced from the full 24. The security analysis of reduced-round variants of iterated cryptosystems is common practice, where the aim is to study the security margin of the cipher against dedicated attacks.

The largest number of rounds that can be attacked for the SHA-3 instances according to published analysis are given in Table 1.2. Note that this table only includes results on the 4 hash functions, but does not contain results on the two XOFs, SHAKE128 and SHAKE256. Indeed, the Keccak instances that correspond to the XOFs were not submitted to the SHA-3 competition, and their specific parameters did not undergo direct public analysis.

The results presented in table 1.2 are subject to two remarks, which will be further discussed in Chapter 2: first, the original cited results were obtained on Keccak instances, rather than on the SHA-3 functions derived from them. However, as the only difference between the SHA-3 functions and the corresponding Keccak instances is a small number of bits that are concatenated to the message, the original attacks on round-reduced Keccak generally apply to SHA-3. The second remark is that Table 1.2 does not include small optimizations of exhaustive search (such as [3], which we consider as a distinguisher of type 2, as described in Chapter 2).

Function	Attack Type	Number of Rounds Attacked	Reference
SHA3-224	Collision	4	[18, 20]
SHA3-224	(Second) Preimage	4	[28]
SHA3-256	Collision	5	[19]
SHA3-256	(Second) Preimage	4	[28]
SHA3-348	Collision	4	[19]
SHA3-348	(Second) Preimage	4	[28]
SHA3-512	Collision	3	[19]
SHA3-512	(Second) Preimage	4	[28]

Table 1.2: Number of Rounds that can be Attacked for SHA-3 Functions

Distinguishers on the Hash Function

General distinguishers on round-reduced SHA-3 can reach a few more rounds compared to attacks on its core security properties. However, there is no published distinguisher that can reach 10 rounds (or more) of any of the SHA-3 instances.

Distinguishers on the Permutation

Since the publication of Keccak (SHA-3), several distinguishers on (round-reduced) variants of its permutation were proposed. However, it is rather difficult to compare the various types of distinguishers on the permutation, as they differ in complexities and the potential effect on the security of the hash function. For example, some differential distinguishers for a small number of rounds (such as ones published in [22, 31]) have low complexities, and were shown to yield actual collision attacks on round-reduced SHA-3 in [18, 20]. On the other hand, zero-sum distinguishers [2], can potentially reach a very large number of rounds, but these have very high complexity, and it is highly unlikely that they would have any effect on the security of the hash function, as it is not clear how to exploit them to detect any unexpected property of the output.

1.2.3 Conclusions

As demonstrated in Table 1.2, the security margin of the 4 SHA-3 hash functions against known attacks is remarkable. Indeed, when considering the core security properties of hash functions, current cryptanalysis techniques can only break up to 5 out of full 24 rounds (about 21%) of the SHA-3 hash functions. While weaker types of distinguishers on the hash functions can target a few more rounds, their security margin remains very large. Other types of distinguishers on the internal SHA-3 permutation can target a relatively large number of rounds, but they have no influence on the security of the actual SHA-3 hash functions. To conclude, after more than 6 year of extensive analysis, there is very good confidence in the security of the 4 SHA-3 hash functions.

The confidence in the security of the SHA-3 XOFs is somewhat lower, as their underlying Keccak instances did not undergo direct public analysis during or after the SHA-3 competition. However, similarly to the 4 SHA-3 hash functions, the security of the XOFs is based on the strength of the Keccak permutation, which has undergone extensive analysis. Therefore, the security margin of the XOFs against dedicated attacks should generally be similar to the margin of the 4 SHA-3 hash functions.

Chapter 2

Summary of Cryptanalysis of SHA-3

This chapter summarizes the published cryptanalysis of the SHA-3 hash functions during and after the SHA-3 competition. We start with a brief description of SHA-3 and additional preliminaries. Then, the summary of cryptanalysis is divided into categories according to the applied techniques.

2.1 Preliminaries

In this section, we give a brief description of SHA-3, and outline the difference between SHA-3 and the underlying Keccak instances from the cryptanalytic perspective. Then, we give a short summary on generic attacks on SHA-3.

2.1.1 Description of Keccak

We give with a brief description of SHA-3 and the underlying Keccak sponge function. For more details, refer to the Draft FIPS PUB 202 [32].

The sponge construction [7] works on a state of b bits, which is split into two parts (see Figure 2.1): the first part contains the first r (rate) bits of the state and the second part contains the last $c = b - r$ (capacity) bits of the state.

Given a message, it is first padded using a padding rule, then cut into r-bit blocks, and the b state bits are initialized to zero. The sponge construction then processes the message in two phases: In the absorbing phase, the message blocks are processed iteratively by XORing each block into the first r bits of the current state, and then applying a fixed permutation on the value of the b-bit state. After processing all the blocks, the sponge construction switches to the squeezing phase. In this phase, n output bits are produced iteratively,

where in each iteration the first r bits of the state are returned as output and the permutation is applied.

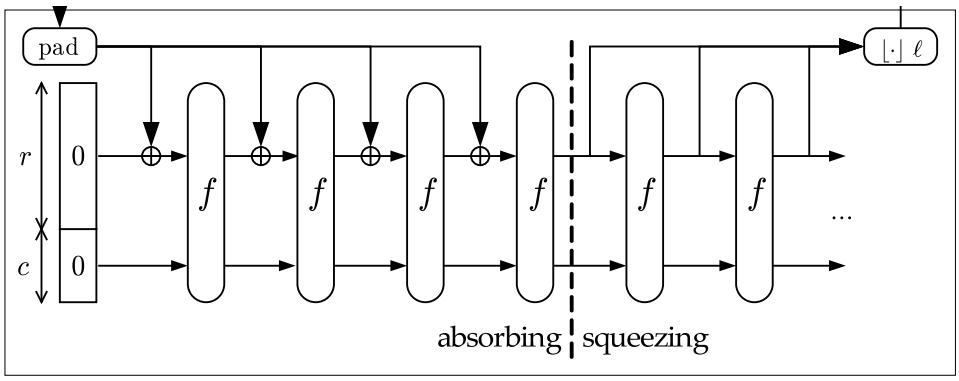


Figure 2.1: The Sponge Construction [6]

The Keccak hash function uses multi-rate padding: given a message, it first appends a single 1 bit. Then, it appends the minimum number of 0 bits followed by a single 1 bit, such that the length of the result is a multiple of r. Thus, multi-rate padding appends at least 2 bits and at most $r + 1$ bits.

The Keccak instances which are used in SHA-3 have $b = 1600$. The 1600-bit state can be viewed as a 3-dimensional array of bits, A[5][5][64], and each state bit is associated with 3 integer coordinates, A[x][y][z], where x and y are taken modulo 5, and z is taken modulo 64. Keccak uses the following naming conventions, which are helpful in describing the permutation:

- A row is a set of 5 bits with constant y and z coordinates, i.e. $A[*][y][z]$.
- A column is a set of 5 bits with constant x and z coordinates, i.e. $A[x][*][z]$.
- A lane is a set of 64 bits with constant x and y coordinates, i.e. $A[x][y][*]$.
- A slice is a set of 25 bits with a constant z coordinate, i.e. $A[*][*][z]$.

The Keccak permutation with $b = 1600$ (which is the only permutation relevant to this report) consists of 24 rounds, each round consists of five mappings $R = \iota \circ \chi \circ \pi \circ \rho \circ \theta$. The five mappings are given below, for each x,y, and z:

1. θ is a linear map, which adds (over $GF(2)$) to each bit in a column, the parity of two other columns.

   θ: $A[x][y][z] \leftarrow A[x][y][z] \oplus \sum_{y'=0}^{4} A[x-1][y'][z] \oplus \sum_{y'=0}^{4} A[x+1][y'][z-1]$

We note that the inverse mapping, θ^{-1}, which is more complicated and provides much faster diffusion: for θ^{-1}, flipping the value of any input bit, flips the value of more than half of the output bits.

2. ρ rotates the bits within each lane by $T(x, y)$, which is a predefined constant for each lane.
   ρ: $A[x][y][z] \leftarrow A[x][y][z + T(x, y)]$

3. π reorders the lanes.
   π: $A[x][y][z] \leftarrow A[x'][y'][z]$, where $\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 2 & 3 \end{pmatrix} \cdot \begin{pmatrix} x' \\ y' \end{pmatrix}$

4. χ is the only non-linear mapping of Keccak, working on each of the 320 rows independently.
   χ: $A[x][y][z] \leftarrow A[x][y][z] \oplus ((\neg A[x + 1][y][z]) \wedge A[x + 2][y][z])$
   Since χ works on each row independently, in can be viewed as an Sbox layer which simultaneously applies the same 5 bits to 5 bits Sbox to the 320 rows of the state. We note that the Sbox function is an invertible mapping, the algebraic degree of each output bit of χ as a polynomial in the five input bits is only 2. We also note that the algebraic degree the inverse mapping χ^{-1} is 3.

5. ι adds a round constant to the state.
   ι: $A \leftarrow A \oplus RC[i_r]$

The first three Keccak round mappings are linear, and we denote their composition by $L \triangleq \rho \circ \pi \circ \theta$.

2.1.2 Description of SHA-3

The four SHA-3 hash functions are defined from Keccak[c] (where c denotes the capacity), by appending two bits (domain separation) to the message and by specifying the length of the output, as follows:

$$SHA3\text{–}224(M) = Keccak[448](M||01, 224)$$
$$SHA3\text{–}256(M) = Keccak[512](M||01, 256)$$
$$SHA3\text{–}384(M) = Keccak[768](M||01, 384)$$
$$SHA3\text{–}512(M) = Keccak[1024](M||01, 512)$$

The two SHA-3 XOFs can be defined directly from Keccak, as follows:

$$SHAKE128(M, d) = Keccak[256](M||1111, d)$$
$$SHAKE256(M, d) = Keccak[512](M||1111, d)$$

2.1.3 The Difference Between SHA-3 and Underlying Keccak Instances

As described above, the SHA-3 instances are derived directly from Keccak instances by concatenating a few bits to the input message of the underlying Keccak instances. As the SHA-3 instances are also instances of Keccak, from a cryptanalytic point of view they are at least as strong as the underlying Keccak instances. Namely, any attack on a SHA-3 instance is an attack on the underlying Keccak instance, but an attack on a Keccak instance cannot necessarily be applied to the corresponding SHA-3 instance. In particular, the bits concatenated to the message reduce the number of degrees of freedom available to the attacker in the last message block, and this could potentially counter attacks on Keccak instances which target this block (all the attacks on Keccak instances published to date target the last message block). This implies that one needs to reconsider the published analysis on Keccak and check whether it also applies to SHA-3. In general, since only a few bits are concatenated to the SHA-3 message, their effect is expected to be negligible in most cases. Indeed, we have verified that the techniques of Sections 2.2, 2.4 and 2.5 are applicable to SHA-3 (essentially) in the same way they are applicable to the underlying SHA-3 instances. The analysis of the techniques of Section 2.3 is somewhat more involved and requires additional work, but we stress that such analysis will not change our conclusion, which asserts that the Keccak instances resist these techniques, and therefore also the instances of SHA-3.

We note that as SHA-3 uses the Keccak permutation, there is no difference between them in terms of cryptanalysis of type 3 (distinguishers on the permutation, as defined in Chapter 2), and the differences above only apply to cryptanalysis of types 1 and 2.

2.1.4 Generic Attacks

Before evaluating the resistance of the SHA-3 functions against dedicated attacks, we state their security against generic attacks that are applicable to any hash function, and against generic attacks that are applicable to any sponge function: Any hash function with an n-bit output provides (at most) $n/2$-bit security against collision attacks and n-bit security against preimage and second preimage attacks. When considering a sponge function with capacity c, it provides $c/2$-bit security against generic attacks, unless the hash function itself provides less security (see [7]). Therefore, the security of the SHA-3 instances against generic collisions attacks and against (second) preimage attacks in Table 1.1 is calculated according to the formulas $min(n/2, c/2)$ and $min(n, c/2)$, respectively.

2.2 Differential Techniques

Differential cryptanalysis of hash functions analyzes the propagation of differences (typically XOR differences - over $GF(2)$) of message pairs m_1 and m_2 with a fixed input difference Δ_{in} inside the internal components of the hash function. A differential characteristic is a specific way in which Δ_{in} propagates to an output difference Δ_{out}, and it is associated with a probability p which is an estimation of the probability that an arbitrary message pair with input difference Δ_{in} will have an output difference of Δ_{out}.

The main application of differential cryptanalysis to hash functions is in collision attacks, where the attacker tries to find a characteristic in which $\Delta_{out} = 0$ with a (relatively) high probability p. Then, by trying about p^{-1} arbitrary message pairs with corresponding input difference Δ_{in}, the attacker is likely to encounter a collision, i.e., a message pair m_1 and $m_1 \oplus \Delta_{in}$ such that $H(m_1) = H(m_1 \oplus \Delta_{in})$. In order to protect against differential cryptanalysis, it is generally expected that a hash function exhibits no high probability characteristics, i.e., it is desirable to upper bound the probability of any differential characteristic for the hash function.

Differential cryptanalysis of Keccak starts with analysis of its main building block, namely the Keccak permutation. It is important to note that general differential characteristics of high probability for the Keccak permutation are considered distinguishers of type 3 (as defined in Chapter 1). In order to attack an actual Keccak hash function instance, the input difference Δ_{in} of the characteristic should fit into the initial state of Keccak, namely, the input difference on the c capacity bits (which are not under control of the attacker) should be set to 0.[1] Assuming that this is the case, then if the output difference Δ_{out} on the output bits is zero, the characteristic may lead to a collision attack (cryptanalysis of type 1). Otherwise, it can be considered a distinguisher on the hash function (cryptanalysis of type 2), as it may be exploited to generate message pairs with a fixed (non-zero) output difference Δ_{out} more efficiently than expected from a random function. Generally, in order to find the best differential attacks, the cryptanalyst searches for differential characteristics of the highest possible probability.

In the rest of this section, we describe the known results regarding search for differential characteristics for round-reduced Keccak permutation. Then, we describe results that use these characteristics in order to attack the actual instances of Keccak.

[1] In this report, we mainly consider collisions between short single-block messages for which the difference of the c capacity bits must be set to zero. This is not necessarily the case in collisions between multi-block messages which differ in more than one message block, but such messages were not considered in previous collision attacks on Keccak.

2.2.1 Search for Differential Characteristics

Table 2.1 gives the currently best known results with respect to differential characteristics for round-reduced Keccak permutation. More details on these results are given below.

Rounds	Best Known Probability [reference]	Upper bound [reference]
3	2^{-32} [22]	2^{-32} [16]
4	2^{-134} [16]	-
5	2^{-510} [31]	-
6	2^{-1360} [9]	2^{-74} [16]
24	-	2^{-296} [16]

Table 2.1: Probability of Best Known Differential Characteristics and Upper Bounds for Round-Reduced Keccak Permutation

As many related hash function designs, the permutation contains a non-linear mapping, χ, which ensures that differential characteristics of high Hamming weight (namely, a large number of non-zero difference bits, or "active bits") have low probability, and linear mappings (in particular θ) that provide diffusion, i.e., ensure that a characteristic over several rounds has high Hamming weight. The combination of these two types of mappings allows to upper bound the probability of a differential characteristic over several rounds, as initially done in the Keccak submission to the SHA-3 competition [9].

While several of the concrete results of [9] were improved since its publication (mostly in [16]), it defined several important notions that were used in subsequent papers both to search for differential characteristics and to upper bound their probabilities. One of these notions is the *column parity kernel* or *CP-kernel*, which describes a special set of states (or state differences) in which the parity of all 320 columns is zero. As the θ mapping adds to each bit in a column, the parity of two other columns, it acts as an identity on the CP-kernel states and does increase its Hamming weight. On the other hand, [9] lower bounded the branch number of θ (the sum of Hamming weights of input and output) for states that are not in the CP-kernel. These techniques were used to search for characteristics with high probability over a few rounds of the permutation by limiting the search to characteristics that stay in the CP-kernel and a small number of low Hamming weight ones that are not in the CP-kernel.

The INDOCRYPT 2011 paper [31] by Naya-Plasencia et al. introduced the notion of a *double kernel*, which describes a state difference (or a differential characteristic) that remains inside the CP-kernel of θ for two consecutive rounds. Naya-Plasencia et al. then introduced an algorithm which efficiently finds double-kernel differential characteristics, and extended them to more rounds (the extensions for more than 2 rounds are outside the CP-kernel). However, once out of the CP-kernel, the Hamming weight of the state differ-

0000000000008000	0000000000000000	0000000000000000	0000000000000000	0000000000000000
0000000000000000	0000000000000000	0000000000000000	0000000000000000	0000200000000000
0000000000008000	0000000000000000	0000000000000000	0000000000000000	0000000000000000
0000000000000000	0000000000000000	0000000000000000	0000000000000000	0000000000000000
0000000000000000	0000000000000000	0000000000000000	0000000000000000	0000200000000000

The state is described as a matrix of 5×5 lanes of 64 bits, ordered from left to right, where each lane is given in hexadecimal using the little-endian format.
Example 1: A state in the column parity kernel with Hamming weight 4

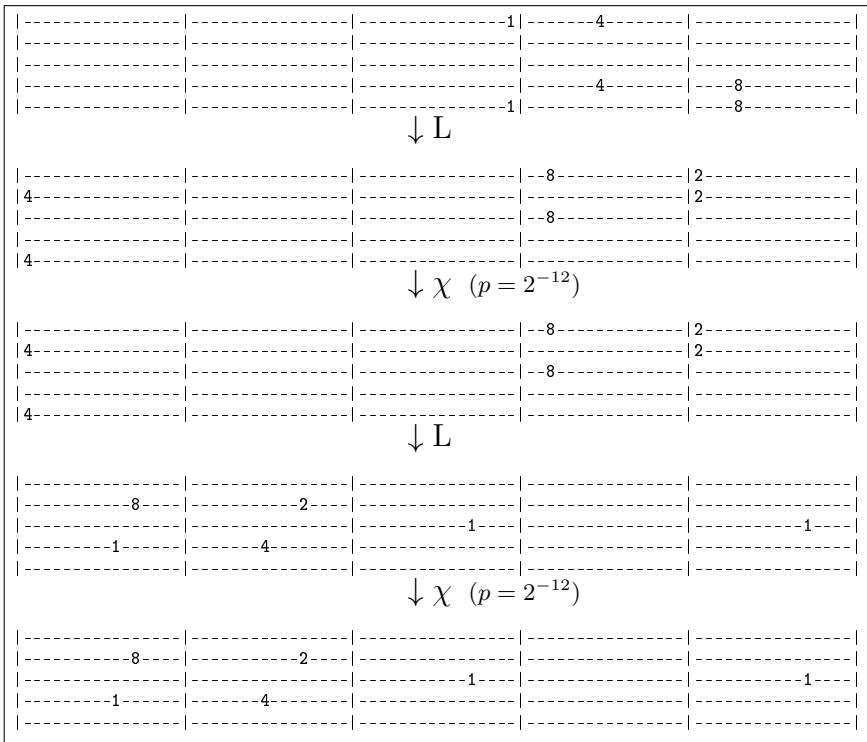
ences of the characteristic increases rapidly and the probability of characteristics which are longer than 3 rounds are significantly lower.

The FSE 2012 paper [22] by Duc et al. studied differential properties of the Keccak permutation, independently of [31]. The main motivation of [22] was to apply the rebound attack (originally published in [27]) to the Keccak permutation, whereas this attack was typically applied to AES-based permutations. The main idea of the rebound attack is to efficiently connect several differential characteristics for the permutation of the hash function by initiating the search for conforming pairs that follow the characteristics from the middle of the permutation and exploiting available degrees of freedom.

The main result of [22] is an 8-round distinguisher which requires a workload of about 2^{491}. It is important to note that while standard high-probability differential characteristics have a practical influence on the security of Keccak, rebound-based distinguishers of the type presented in [22] seem to be much less related to the security of the Keccak hash function instances (namely, they are strictly distinguishers of type 3). The reason for this is that search for conforming pairs from the middle of the permutation does not take into account the constraints imposed on the c capacity bits in the initial state of the characteristic, and they seem to be inapplicable to Keccak hash function instances (regardless of the workload they require).

Unlike rebound distinguishers, the individual standard differential characteristics presented in [22] are more related to the security of Keccak hash function instances. Independently of [31], Duc et al. described a different characteristic search algorithm for the Keccak permutation, and used it to find actual characteristics for up to 5 permutation rounds. All of these characteristics are extensions of shorter low-Hamming weight characteristics that stay in the CP-kernel for one or two rounds (i.e. double-kernel differential characteristics).

Finally, the FSE 2012 paper [16] by Daemen et al. presented a refined algorithm for search of differential characteristics for the Keccak permutation. The algorithm extended the work of [22] and exploited symmetries and additional properties of the Keccak mappings in order to represent its state in a compact way, which allows enumerating all 3-round characteristics with probability as low as 2^{-36}. As a result, [16] was able to formally prove that the 3-round characteristic of probability 2^{-32} found in [22] is the highest probability characteristic



A zero hexideciamal character is shown as '-'. The inputs to θ (and to L) in the 2 rounds are differences which are in the CP-kernel. As ι does not influence the state difference, it is not specified.

Characteristic 1: A (2-round) double-kernel characteristic with probability $p = 2^{-24}$

for 3 rounds of the Keccak permutation. The search for 3 round characteristics was partially extended to 4 rounds, and allowed Daemen et al. to find the best known 4-round differential characteristic. Furthermore, by considering concatenations of 3-round characteristics, [16] also proved that there exists no differential characteristic with probability higher than 2^{-74} for 6 rounds of the Keccak permutation. This upper bound trivially extends to the full 24-round permutation by considering concatenations of four 6-round characteristics, and obtaining the upper bound $2^{-74 \cdot 4} = 2^{-296}$.

2.2.2 Exploiting Differential Characteristics to Attack the Hash Function

The INDOCRYPT 2011 paper [31] presented first results on the actual round-reduced Keccak hash function instances Keccak[448] and Keccak[512] (which are similar to SHA3-224 and SHA3-256, respectively). In particular, it presented practical collision attacks on 2 rounds of the hash functions using single-block message pairs (or messages of arbitrary length which differ only in the last block) by exploiting double-kernel characteristics in which Δ_{in} is zero in the capacity bits and Δ_{out} is zero on the output bits (satisfying the constraints described at the beginning of the section). The characteristics were also extended to 3 and 4 rounds (with non-zero output difference) to give distinguishers on the hash function instances. Note that results were not obtained for KECCAK[768] and KECCAK[1024] (SHA3-384 and SHA3-512). The reason for this is that the value of c is much larger in these instances, and it is significantly more difficult to find good characteristics in which Δ_{in} is zero in all the capacity bits.

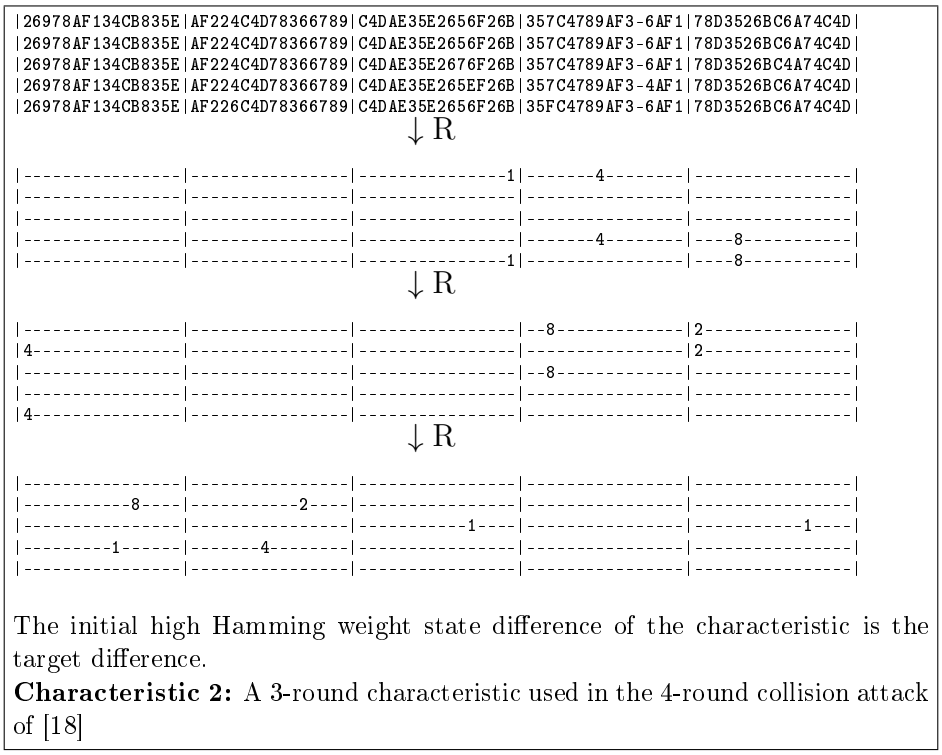
The FSE 2012 paper [18] by Dinur et al. presented practical collision attacks on 4 rounds of the Keccak hash function instances Keccak[448] and Keccak[512] using single-block message pairs. It also devised distinguishers on 5 rounds of these hash function instances.

The starting point of the technique of [18] was the double kernel characteristics presented in [31] and [22]. Such a 2-round characteristic of high probability was extended by one round backwards to form a 3-round characteristic (as done in [22]). However, due to the very fast diffusion of θ^{-1}, the initial difference of the characteristic has a high Hamming weight. Therefore, it does not satisfy the constraint that Δ_{in} is zero in the capacity bits, and cannot be directly used to attack the hash function in the way that 2-round characteristics were used in [31]. On the other hand, extending it backwards by an additional round in a standard way would have a very high cost in probability due to the high Hamming weight of its initial difference.

The main idea of [18] was to link the initial state of the extended 3-round characteristic (called the *target difference*) to the initial state of the hash function (Keccak[448] or Keccak[512]) using another round of the permutation with an algorithm called the *target difference algorithm*. The 4-round collision attack

of [18] is thus composed of two parts, where in the first part the target difference algorithm is executed in order to obtain a large set of message pairs that satisfy the target difference after the first round with probability 1. The second part of the attack is a standard differential attack in which the message pairs (returned by the target difference algorithm) are evaluated in order to find a pair whose difference evolves according to the 3-round characteristic (whose starting state is the target difference).

Internally, the target difference algorithm uses algebraic techniques to find message pairs that satisfy the target difference after 1 round. It exploits the relatively low (quadratic) algebraic degree of the permutation, and utilizes the abundance of degrees of freedom available in Keccak[448] and Keccak[512] (namely, the large value of r). One of the reasons for which the technique was not applied to KECCAK[768] and KECCAK[1024] (SHA3-384 and SHA3-512) is the reduced value of r for these instances.



The initial high Hamming weight state difference of the characteristic is the target difference.

Characteristic 2: A 3-round characteristic used in the 4-round collision attack of [18]

Finally, we also mention the more recent AFRICACRYPT 2014 paper [17]. This paper built on the target difference algorithm of [18], and combined it with longer (and truncated) differential characteristics in order to obtain a distinguisher on 6 rounds of Keccak[448] and Keccak[512].

2.2.3 Discussion

There are interesting open questions regarding the security of Keccak with respect to differential cryptanalysis. In particular, the upper bound on the probability of differential characteristics matches the lower bound for (up to) 3 rounds of the permutation, while for 4 rounds (and more) there is a big gap between the best known differential characteristic and the proven upper bound (see Table 2.1). This gap is attributed to the large state of the permutation and the difficulty to group and analyze together many differential characteristics (as typically done in AES-based designs) using a compact representation of the state (the paper [8] explicitly addressed this issue). While improved algorithms that search for differential characteristics were introduced in [16], they were only able to obtain optimal results for up to 3 rounds, and it is likely that better differential characteristics exist already for 4 rounds. Newly found characteristics could push existing attacks on the Keccak hash functions by an additional round or even two rounds, but any improvement beyond this would be very surprising. For the full 24-round Keccak permutation, the proven upper bound in [16] is 2^{-296}, but it seems highly likely that the probability of the actual best characteristic is significantly lower, and reducing this bound is another interesting open problem.

An additional potential direction to improve existing differential attacks is to extend the target difference algorithm of [18] to more than 1 round. For example, one can try to place a 3-round differential characteristic in rounds 2–5 of the permutation and try to mount a 5-round collision attack on the hash function by running the (improved) target difference algorithm to obtain message pairs that conform to the target difference after 2 rounds. However, the potential of this technique in improving existing attacks seems rather limited, as the original target difference algorithm exploited the simplicity of a single permutation round (and in particular its low algebraic degree), while analyzing more rounds seems to be significantly more complex.

As summarized in this section, the security margin of the four Keccak hash functions that were submitted to the SHA-3 competition against differential cryptanalysis is very big: while the Keccak instances use a 24-round permutation, collision attacks can break (some of) these instances only when their internal permutation is reduced to 4 rounds, and weaker distinguishers can only reach up to 6 rounds. Similar conclusion apply to the corresponding SHA-3 hash functions. Even when considering more exotic attack models which target the permutation rather than a concrete hash function instance, there are published results only on up to 8 rounds [22]. This also gives confidence in the security of the two SHA-3 XOFs against differential cryptanalysis, even through they were not explicitly analyzed. We conclude that while there is some room for improving existing differential techniques, these potential improvements are limited and should not threaten the security of any SHA-3 instance.

2.3 Techniques Based on Rotational and Symmetric Properties

Four out of five Keccak mappings were designed to preserve a special symmetric property which allows some performance optimizations and also helps to analyze its security against some cryptanalytic attacks (e.g., some of the optimizations used to obtain differential bounds in [16] exploit this property). The property asserts that four out of the five internal mappings of Keccak (all but ι), are translation invariant in the direction of the z axis. Namely, if one state A is the rotation of another state A' with respect to the z-axis (i.e., satisfies $A'[x][y][z] = A[x][y][z+i]$, for some value of i), then applying to them any of the θ, ρ, π, χ operations, maintains this property. The translation invariance property is broken by the fifth mapping ι, which adds constants to the Keccak state that are not translation invariant (for any $0 < i < 64$). However, the constants added by ι are of a low Hamming weight, and therefore they have a limited effect in the first few rounds of the permutation. This observation gives rise to some interesting properties of the reduced Keccak permutation, and was exploited in several publications, as noted below.

The FSE 2013 paper [28] by Morawiecki et al. used rotation cryptanalysis (formally introduced in [24]) of round-reduced Keccak to speed up preimage search on (up to) 4 rounds. The basic idea of [28] was to evaluate the reduced permutation on one single-block message and record its output. Denote the initial state of the evaluated message by A. Then, based on the translation invariance property which is not completely ruined by ι, it is possible to predict several (but not all) output bits for (up to) additional 63 rotated single-block messages, whose initial states A' are rotated variants of A, namely, $A'[x][y][z] = A[x][y][z+i]$ for $i \in \{1, 2, \ldots, 63\}$. The predicted output bits for these 63 additional messages allow to disqualify messages whose output bits do not match the desired preimage bits, and actually evaluate the permutation only when a match on these bits is guaranteed. This technique allowed [28] to save a factor of up to 64 in preimage search on up to 4-rounds of the 4 Keccak hash functions (see Table 1.2). The analysis was also extended to 5 rounds of the permutation to obtain some non-random properties (namely, cryptanalysis of type 3).

The FSE 2013 paper [19] by Dinur et al. exploited the translation invariance property of the four Keccak mappings in a different way to mount collision attacks. The main idea of [19] was to select a symmetric state A such that $A[x][y][z] = A[x][y][z+i]$ for some fixed value of $0 < i < 64$. Since operations on the z axis are performed modulo 64, this implies that i divides 64, or $i \in \{1, 2, 4, 8, 16, 32\}$. The translation invariance property implies that the symmetry is preserved by the mappings θ, ρ, π, χ, but is ruined by ι. However, since the constants added by ι are of low Hamming weight, the state remains close to symmetric in the first few rounds of the Keccak permutation, namely, equalities of the form $A[x][y][z] = A[x][y][z+i]$ are preserved for many bits of

the state, while for a minority of the bits $A[x][y][z] \oplus A[x][y][z+i] = 1$.

```
|169D169D169D169D|A965A965A965A965|3EC73EC73EC73EC7|9025902590259025|C264C264C264C264|
|A34BA34BA34BA34B|0F330F330F330F33|4902490249024902|3D683D683D683D68|613D613D613D613D|
|C684C684C684C684|B368B368B368B368|589B589B589B589B|5F335F335F335F33|E27AE27AE27AE27A|
|22E822E822E822E8|3D583D583D583D58|B37AB37AB37AB37A|1047104710471047|D525D525D525D525|
|60F360F360F360F3|C3E4C3E4C3E4C3E4|37FA37FA37FA37FA|8193819381938193|69BA69BA69BA69BA|
```

Each lane of the state consists of 4 repetitions of a 16-bit word.
Example 2: A symmetric state with $i = 16$

The technique used in [19] to analyze symmetric relations over several rounds of the permutation is called *generalized internal differential cryptanalysis*, generalizing the original method published in [33]. While standard differential cryptanalysis considers two different plaintexts, and analyzes the evolution of the difference between them, internal differential cryptanalysis considers only one plaintext, and follows the statistical evolution of the differences between its parts. In the case of Keccak, internal differential cryptanalysis is used in [19] to track the statistical evolution of almost symmetric states through the first few rounds of Keccak.

When symmetric relations are carried over several rounds to the output of the hash function, they reduce its effective size, and this technique is used in [19] to speed up collision search. The attack works by producing arbitrary (nearly) symmetric outputs starting from arbitrary symmetric states, but as the number of (nearly) symmetric outputs is significantly smaller than the total number of possible outputs, a collision is expected to occur much faster compared to a standard birthday attack. This is called a *squeeze attack* in [19], since it forces many (symmetric) inputs to squeeze into a small subset of possible (nearly symmetric) outputs in which collisions are more likely.

For example, assuming $i = 16$, in a symmetric state, $A[x][y][z]$ determines $A[x][y][z+16]$, $A[x][y][z+32]$ and $A[x][y][z+48]$. Thus, the effective output size of a symmetric SHA3-256 output is significantly reduced from 256 to $256/4 = 64$ bits, and the birthday bound is reduced from $2^{256/2} = 2^{128}$ to $2^{64/2} = 2^{32}$. When considering nearly symmetric states, the effective output size is larger than 64 bits, but it can still be significantly smaller than 256 (depending on how many nearly symmetric states are possible), and the collision attack may still require less than 2^{128} effort.

The main results of [19] (summarized in Table 1.2) include practical collision attacks on Keccak[768] and Keccak[1024], a 4-round collision attack on Keccak[768] with complexity 2^{147} and a 5-round collision attack on Keccak[512] with complexity of about 2^{115}.

We also mention the AFRICACRYPT 2014 paper [25], which used similar techniques to [19] to obtain a distinguisher on 6 rounds of the permutation (namely, cryptanalysis of type 3).

We note that most of the attacks mentioned in this section[2] exploit the fact that the c bits of capacity in the initial state of Keccak (which are not under control of the attacker) are set to zero. This allowed choosing (about) 64 rotated variants of an arbitrary initial state of Keccak in [28], and allowed choosing symmetric initial states in [19]. This also implies that these techniques can only be applied using 1-block messages.

2.3.1 Discussion

As noted in [19], there is an interesting relation between standard differential cryptanalysis and generalized internal differential cryptanalysis used to analyze symmetric relations over several rounds of the permutation. Roughly, the asymmetric relations in a given state propagate in a similar way to non-zero difference relations of two states in differential cryptanalysis. In general, this implies that the strong resistance of the Keccak permutation to differential cryptanalysis should also ensure its resistance to the type of attacks described in this section. However, the analysis of rotational and symmetric properties is somewhat more involved compared to differential cryptanalysis due to effect of the round constants (which are ignored in differential cryptanalysis).

We conclude that, in several cases, the techniques described in this section yield somewhat more efficient attacks on Keccak compared to differential techniques. Furthermore, as these untraditional cryptanalytic techniques are less understood and involve more complex analysis, there seems to be more room for improved attacks compared to differential techniques. However, the very large security margin of Keccak against existing cryptanalysis based on rotational and symmetric properties and their relation to differential cryptanalysis should ensure that they do not threaten the security of any concrete Keccak instance.

Finally, we note that the four SHA-3 hash functions (SHA3-224, SHA3-256, SHA3-385 and SHA3-512) seem to resist cryptanalysis based on rotational and symmetric properties somewhat better than the corresponding underlying Keccak instances (Keccak[448], Keccak[512], Keccak[768] and Keccak[1024]). The reason for this is that the additional bits added to the final block of the message in SHA-3 limit the ability of the attacker to choose rotational variants of a given state (in the attacks of [28]), or symmetric states (in the attacks of [19]). However, this only strengthens the conclusions of this section, regarding the strong resistance of the SHA-3 instances to cryptanalysis based on rotational properties and symmetry (although the the exact complexities of the attacks described in this section on the four Keccak instances may require reevaluation for the four SHA-3 hash functions).

[2] Not including the 5-round collision attack of [19] on Keccak[512].

2.4 Algebraic Techniques

In Section 2.2, we mentioned the work of [18, 20], which combined differential and algebraic techniques. In this section, we fucus on published algebraic cryptanalytic techniques on Keccak that are based on high order differential cryptanalysis [26]. The most important property of Keccak for such cryptanalysis, is the fact that the algebraic degree of each output bit of its round function over $GF(2)$ is only 2 in the input bits, due to the (only) non-linear map χ. This implies that the degree of each output bit of m Keccak permutation rounds is at most 2^m in the input bits. When considering m permutation rounds, the result of a high order differentiation over an affine subspace of inputs of dimension $m+1$ is zero for all output bits. Over $GF(2)$, where addition and subtraction are the same operation, such high order differentiation simply reduces to summing (modulo 2) the output bits of the m permutation rounds over the subspace of size 2^{m+1}. This simple analysis was published in the Keccak submission document, which gave upper bounds on the algebraic degree of monomials in the algebraic normal form (ANF) of the permutation, as function of its number of rounds. Then, based on experimental results, a more detailed analysis of the ANF of Keccak was given in [1]. It showed that it is possible to choose smaller subspaces for 3 and 4 rounds of Keccak to obtain a zero sum, but the general conclusion of [1] regarding the algebraic strength of Keccak was similar to the submission document.

In order to exploit the simple high order differential property to obtain distinguishers on Keccak instances (cryptanalysis of type 2), consider single block messages where the desired subspace of dimension $m+1$ is taken from the message bits (rather than from the bits of the permutation which are not under control of the attacker). The attacker needs to evaluate only $2^{m+1}-1$ messages in this subspace in order to obtain the output of the missing message "for free" (as the sum of outputs of all messages is zero). Thus, the attacker can obtain a small speedup of $(2^{m+1}-1)/2^{m+1}$ when searching for preimages (or second preimages) for a Keccak instance with permutation reduced to m rounds, assuming that the security level of the instance (the complexity of the generic preimage attack) is more than 2^{m+1}. This basic idea was extended by Bernstein in [3] by considering larger subspaces and predicting more outputs using fast polynomial evaluation techniques. These techniques give somewhat better improvement factors as they reduce the amortized computation time that the attacker has to perform per output, in exchange of a large amount of memory. The maximal value of m for which this technique is applicable is determined by the security level of the hash function instance against preimage attacks, reaching up to 8 rounds of Keccak[1024] (where the improvement factor is about $\sqrt{2}$, or $1/2$ of a bit).

The reason for which we consider the techniques of [3] as distinguishers on Keccak (cryptanalysis of type 2), rather than attacks on the hash function (cryptanalysis of type 1) is that they are essentially (optimized) variants of ex-

haustive search. In other words, in order to find a preimage for a Keccak instance with a preimage security level of b bits using these techniques, the attacker has to perform at least 2^b basic bit operations, and the speedup is obtained only by reducing the number of basic operations performed in an evaluation of the Keccak permutation. Such variants of exhaustive search are comparable to biclique cryptanalysis [11], and we consider them here as distinguishers on the hash function due to their inherent limitations. However, we stress that this is a matter of definition, and the classification of cryptanalysis of this type has been a matter of controversy in the cryptographic community.

A related form of cryptanalysis (which actually preceded [3]) is called zero-sum distinguishers, originally introduced in [2] by Aumasson and Meier. The objective of distinguishers of this type is to find a zero-sum set, defined as a set of inputs to the permutation whose sum is zero modulo 2, and whose image set sum is also zero modulo 2. Obviously, the linear subspaces selected by Bernstein in [3] are zero-sum sets (although they were used in a different context), but these reach a limited number of rounds. The idea of Aumasson and Meier was to find zero-sums for more rounds of the permutation (rather than the hash function) by selecting the linear subspace in the middle of the permutation, and evaluating it both forwards and backwards. Namely, if the m-round permutation consists of the round sequence $R_m \circ \ldots \circ R_2 \circ R_1$, then the linear subspace is selected from some middle round $1 < i < m$, and the permutation is partitioned into an inverse sub-permutation $R_1^{-1} \circ \ldots \circ R_{i-1}^{-1} \circ R_i^{-1}$ and the remaining sub-permutation $R_{i+1} \circ \ldots \circ R_{m-1} \circ R_m$. The sum of evaluations of both the forwards and backwards sub-permutations is zero if the algebraic degree of the sub-permutations is smaller than the dimension of the subspace, and in this case the set of inputs to R_1 is a zero-sum set. A relevant fact for this analysis is that the algebraic degree of an inverse-round of the permutation is 3, which is the degree of χ^{-1} (whereas the algebraic degree of χ is only 2).

As the subspace which is used to construct the distinguisher is selected from the middle of the permutation, it is clear that the obtained distinguisher is inapplicable to the hash function instances, since the inputs to the first round R_1 do not adhere to the restrictions of SHA-1 on the c capacity bits. Thus, zero-sum distinguishers are considered as cryptanalysis of type 3 and it seems highly unlikely that they would have any practical influence on the security of the hash function instances. The original zero-sum distinguishers of [2] reached 16 rounds of the Keccak permutation (with very high complexity of about 2^{1024}), and was later extended to more rounds in [12, 13, 14, 21], mainly using improved bounds on the algebraic degree of the sequence of inverse sub-permutation rounds. We note that despite their high complexity and inapplicability to the actual Keccak hash function instances, zero-sum distinguishers were the main reason that the Keccak team increased the number of permutation rounds to 24 from the original number of 18 [5]. Nevertheless, the most optimized analysis [14, 21] is able to

obtain distinguishers for the full 24-round permutation (with huge complexity of at least 2^{1579}).

2.4.1 Discussion

As detailed in this section, when considering cryptanalysis of types 2 and 3, algebraic techniques reach the most number of rounds compared to other cryptanalytic techniques published up to this point. Indeed, when considering distinguishers on the hash function (cryptanalysis of type 2), the cryptanalysis of [3] reaches (up to) 8 rounds. Perhaps, there is a small room for improvement, but as the results of [3] are inherently limited by the algebraic degree of the Keccak permutation, we do not expect such improvement to exceed 1 or 2 additional rounds.

As for distinguishers on the permutation (cryptanalysis of type 3), the cryptanalysis of [14, 21] reaches the full 24 rounds. However, we stress again that while distinguishers on the hash function (cryptanalysis of type 2) may be relevant for some applications, zero-sum distinguishers on the permutation pose no threat to the practical security of any Keccak instance in any application. Thus, the interest in zero-sum distinguishers, which push cryptanalysis of type 3 to the extreme, remains purely academical.

We conclude that the algebraic techniques analyzed in this section do not threaten the security of any SHA-3 instance.

2.5 Additional Techniques

In this section, we summarize additional cryptanalytic techniques that were applied to Keccak and were not described so far.

We first mention the analysis of [31], which (in addition to describing differential techniques, as summarized in Section 2.2) described preimage attacks for 2 rounds of Keccak[448] and Keccak[512]. These attacks are essentially based on the partial diffusion properties of 2-round Keccak, and it seems rather difficult to extend them to more rounds, where full diffusion is achieved. Furthermore, these techniques cannot be applied (with practical complexity) to Keccak[768] and Keccak[1024], due to the limited number of degrees of freedom available to the attacker in the final message block. Interestingly, the attacks of [31] remain the only practical preimage attacks on reduced Keccak instances, as the subsequent analysis of [28] is faster than exhaustive search only by a factor of (up to) 64 (although it can attack more rounds and is applicable to additional Keccak instances).

Finally, we mention generic tools which were applied to Keccak in order to obtain distinguishers and mount preimage attacks. These tool include a "triangulation" tool used in [2] and SAT solvers, used in [23, 29]. Generally speaking, these tools give some interesting heuristic results when applied to a

very small number of Keccak rounds (e.g., 2 and 3-round preimage attacks on Keccak instances with very small output lengths), but it is not clear how to extend these results to more rounds of Keccak and these techniques seem to have a rather limited significance.

Bibliography

[1] Jean-Philippe Aumasson and Dmitry Khovratovich. First analysis of keccak, 2009.

[2] Jean-Philippe Aumasson and Willi Meier. Zero-sum distinguishers for reduced Keccak-f and for the core functions of Luffa and Hamsi. NIST mailing list, 2009.

[3] Daniel J. Bernstein. Second preimages for 6 (7? (8??)) rounds of keccak? NIST mailing list, 2010.

[4] Guido Bertoni, Joan Daemen, Michael Peeters, and Gilles Van Assche. On the indifferentiability of the sponge construction. In Nigel P. Smart, editor, *Advances in Cryptology - EUROCRYPT 2008, 27th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Istanbul, Turkey, April 13-17, 2008. Proceedings*, volume 4965 of *Lecture Notes in Computer Science*, pages 181–197. Springer, 2008.

[5] Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. Note on zero-sum distinguishers of Keccak-f. NIST mailing list, 2010.

[6] Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. The Sponge Functions Corner. http://sponge.noekeon.org/, 2010.

[7] Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. Cryptographic sponge functions. http://sponge.noekeon.org/CSF-0.1.pdf, 2011.

[8] Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. On alignment in Keccak, 2011.

[9] Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. The Keccak SHA-3 submission. Submission to NIST (Round 3), 2011.

[10] Alex Biryukov, Guang Gong, and Douglas R. Stinson, editors. *Selected Areas in Cryptography - 17th International Workshop, SAC 2010, Waterloo, Ontario, Canada, August 12-13, 2010, Revised Selected Papers*, volume 6544 of *Lecture Notes in Computer Science*. Springer, 2011.

[11] Andrey Bogdanov, Dmitry Khovratovich, and Christian Rechberger. Biclique cryptanalysis of the full AES. In Dong Hoon Lee and Xiaoyun Wang, editors, *Advances in Cryptology - ASIACRYPT 2011 - 17th International Conference on the Theory and Application of Cryptology and Information Security, Seoul, South Korea, December 4-8, 2011. Proceedings*, volume 7073 of *Lecture Notes in Computer Science*, pages 344–371. Springer, 2011.

[12] Christina Boura and Anne Canteaut. A zero-sum property for the KECCAK-f permutation with 18 rounds. In *IEEE International Symposium on Information Theory, ISIT 2010, June 13-18, 2010, Austin, Texas, USA, Proceedings*, pages 2488–2492. IEEE, 2010.

[13] Christina Boura and Anne Canteaut. Zero-Sum Distinguishers for Iterated Permutations and Application to Keccak-f and Hamsi-256. In Biryukov et al. [10], pages 1–17.

[14] Christina Boura, Anne Canteaut, and Christophe De Cannière. Higher-Order Differential Properties of Keccak and *Luffa*. In Antoine Joux, editor, *Fast Software Encryption - 18th International Workshop, FSE 2011, Lyngby, Denmark, February 13-16, 2011, Revised Selected Papers*, volume 6733 of *Lecture Notes in Computer Science*, pages 252–269. Springer, 2011.

[15] Anne Canteaut, editor. *Fast Software Encryption - 19th International Workshop, FSE 2012, Washington, DC, USA, March 19-21, 2012. Revised Selected Papers*, volume 7549 of *Lecture Notes in Computer Science*. Springer, 2012.

[16] Joan Daemen and Gilles Van Assche. Differential Propagation Analysis of Keccak. In Canteaut [15], pages 422–441.

[17] Sourav Das and Willi Meier. Differential Biases in Reduced-Round Keccak. In Pointcheval and Vergnaud [34], pages 69–87.

[18] Itai Dinur, Orr Dunkelman, and Adi Shamir. New Attacks on Keccak-224 and Keccak-256. In Canteaut [15], pages 442–461.

[19] Itai Dinur, Orr Dunkelman, and Adi Shamir. Collision Attacks on Up to 5 Rounds of SHA-3 Using Generalized Internal Differentials. In Moriai [30], pages 219–240.

[20] Itai Dinur, Orr Dunkelman, and Adi Shamir. Improved Practical Attacks on Round-Reduced Keccak. *J. Cryptology*, 27(2):183–209, 2014.

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

[22] Alexandre Duc, Jian Guo, Thomas Peyrin, and Lei Wei. Unaligned rebound attack: Application to keccak. In Canteaut [15], pages 402–421.

[23] Ekawat Homsirikamol, Pawel Morawiecki, Marcin Rogawski, and Marian Srebrny. Security margin evaluation of SHA-3 contest finalists through sat-based attacks. In Agostino Cortesi, Nabendu Chaki, Khalid Saeed, and Slawomir T. Wierzchon, editors, *Computer Information Systems and Industrial Management - 11th IFIP TC 8 International Conference, CISIM 2012, Venice, Italy, September 26-28, 2012. Proceedings*, volume 7564 of *Lecture Notes in Computer Science*, pages 56–67. Springer, 2012.

[24] Dmitry Khovratovich and Ivica Nikolic. Rotational Cryptanalysis of ARX. In Seokhie Hong and Tetsu Iwata, editors, *Fast Software Encryption, 17th International Workshop, FSE 2010, Seoul, Korea, February 7-10, 2010, Revised Selected Papers*, volume 6147 of *Lecture Notes in Computer Science*, pages 333–346. Springer, 2010.

[25] Sukhendu Kuila, Dhiman Saha, Madhumangal Pal, and Dipanwita Roy Chowdhury. Practical Distinguishers against 6-Round Keccak-f Exploiting Self-Symmetry. In Pointcheval and Vergnaud [34], pages 88–108.

[26] Xuejia Lai. Higher Order Derivatives and Differential Cryptanalysis. *Communications and Cryptography*, 276:227–233, 1994.

[27] Florian Mendel, Christian Rechberger, Martin Schläffer, and Søren S. Thomsen. The rebound attack: Cryptanalysis of reduced whirlpool and grøstl. In Orr Dunkelman, editor, *Fast Software Encryption, 16th International Workshop, FSE 2009, Leuven, Belgium, February 22-25, 2009, Revised Selected Papers*, volume 5665 of *Lecture Notes in Computer Science*, pages 260–276. Springer, 2009.

[28] Pawel Morawiecki, Josef Pieprzyk, and Marian Srebrny. Rotational Cryptanalysis of Round-Reduced Keccak. In Moriai [30], pages 241–262.

[29] Pawel Morawiecki and Marian Srebrny. A SAT-based preimage analysis of reduced KECCAK hash functions. Cryptology ePrint Archive, Report 2010/285, 2010.

[30] Shiho Moriai, editor. *Fast Software Encryption - 20th International Workshop, FSE 2013, Singapore, March 11-13, 2013. Revised Selected Papers*, volume 8424 of *Lecture Notes in Computer Science*. Springer, 2014.

[31] María Naya-Plasencia, Andrea Röck, and Willi Meier. Practical Analysis of Reduced-Round Keccak. In Daniel J. Bernstein and Sanjit Chatterjee, editors, *Progress in Cryptology - INDOCRYPT 2011 - 12th International Conference on Cryptology in India, Chennai, India, December 11-14, 2011. Proceedings*, volume 7107 of *Lecture Notes in Computer Science*, pages 236–254. Springer, 2011.

[32] National Institute of Standards and Technology. DRAFT FIPS PUB 202, 2014.

[33] Thomas Peyrin. Improved Differential Attacks for ECHO and Grøstl. In Tal Rabin, editor, *Advances in Cryptology - CRYPTO 2010, 30th Annual Cryptology Conference, Santa Barbara, CA, USA, August 15-19, 2010. Proceedings*, volume 6223 of *Lecture Notes in Computer Science*, pages 370–392. Springer, 2010.

[34] David Pointcheval and Damien Vergnaud, editors. *Progress in Cryptology - AFRICACRYPT 2014 - 7th International Conference on Cryptology in Africa, Marrakesh, Morocco, May 28-30, 2014. Proceedings*, volume 8469 of *Lecture Notes in Computer Science*. Springer, 2014.