Forgery Attacks on Round-Reduced ICEPOLE-128

Christoph Dobraunig, Maria Eichlseder, and Florian Mendel

IAIK, Graz University of Technology, Austria
christoph.dobraunig@iaik.tugraz.at

Abstract. ICEPOLE is a family of authenticated encryptions schemes submitted to the ongoing CAESAR competition and in addition presented at CHES 2014. To justify the use of ICEPOLE, or to point out potential weaknesses, third-party cryptanalysis is needed. In this work, we evaluate the resistance of ICEPOLE-128 against forgery attacks. By using differential cryptanalysis, we are able to create forgeries from a known ciphertext-tag pair with a probability of $2^{-60.3}$ for a round-reduced version of ICEPOLE-128, where the last permutation is reduced to 4 (out of 6) rounds. This is a noticeable advantage compared to simply guessing the right tag, which works with a probability of 2^{-128}. As far as we know, this is the first published attack in a nonce-respecting setting on round-reduced versions of ICEPOLE-128.

Keywords: CAESAR, ICEPOLE, forgery, differential cryptanalysis

1 Introduction

ICEPOLE is a family of authenticated encryption schemes, which has been presented at CHES 2014 [17] and submitted to CAESAR [16]. CAESAR [18] is an open cryptographic competition aiming to find a suitable portfolio of authenticated encryption algorithms for many use cases. For the first round, 57 candidates have been submitted. Due to the open nature of CAESAR, those candidates have different design goals ranging from high-speed software designs to designs suitable for compact hardware implementations. This makes comparison of the submitted ciphers difficult, which is nevertheless necessary to determine the ciphers for the next rounds. However, all designs have one goal in common: security. Thus, as much security analysis as possible is needed to sort out weak CAESAR candidates and get insight in the security of the others.

The goal of authenticated encryption is to provide confidentiality and authenticity. Our attacks focus solely on the authenticity in a forgery attack. The goal is to manipulate known ciphertext-tag pairs in a way such that they are valid with a certain probability. For ICEPOLE-128, the intended number of bits of security with respect to authenticity is 128. Therefore we consider only attacks with a success probability above the generic 2^{-128} applicable.

The method of choice for our attacks is differential cryptanalysis [7]. To create forgeries, we need differential characteristics which hold with a high probability

The final publication is available at Springer via http://dx.doi.org/10.1007/978-3-319-31301-6_27

and fulfill certain constraints explained later. By using this technique, we are able to attack versions of ICEPOLE-128 where the last permutation is reduced to 4 out of 6 rounds. As far as we know, no forgery attacks have been performed on round-reduced versions of ICEPOLE. In addition, we have analyzed the main building block of ICEPOLE, its permutation. We were able to improve the results of the designers regarding high-probability characteristics for the ICEPOLE permutation without any additional constrains. Our results are summarized in Table 1. Note that we have verified the forgery for 3 rounds practically by using the reference implementation of ICEPOLE-128 submitted to CAESAR.

Table 1. Results for ICEPOLE.

	Type	Rounds	Probability
ICEPOLE-128	forgery	3/6	$2^{-14.8}$
		4/6	$2^{-60.3}$
ICEPOLE permutation	differential characteristic	5	$2^{-104.5}$
		6	$2^{-258.3}$

Related work. In the submission document [16], the designers bounded the minimum number of active S-boxes in a differential characteristic for the ICEPOLE permutation. They are able to show that for 3 rounds, the minimum number of active S-boxes is 9, and that there are no characteristics with 13 or fewer active S-boxes for 4 rounds. In addition, Morawiecki et al. [16] heuristically searched for differential characteristics. For 5 rounds, their best published differential characteristic has a probability of $2^{-186.2}$, and for 6 rounds $2^{-555.3}$.

Recently, Huang et al. [13] presented state-recovery attacks on ICEPOLE in a nonce-misuse scenario. They show that in this scenario, the internal state of ICEPOLE-128 and ICEPOLE-128a can be recovered with complexity 2^{46}, and the internal state of ICEPOLE-256a with complexity 2^{60}.

Outline. The remainder of the paper is organized as follows. We describe the design of ICEPOLE in Section 2. Afterwards, we give a high-level overview about the techniques used to find suitable differential characteristics in Section 3, followed by our attacks on round-reduced versions of ICEPOLE in Section 4. Finally, we conclude in Section 5.

2 Description of ICEPOLE

In this section, we give a short description of ICEPOLE-128 as it is specified in the CAESAR design document [16]. For more details about ICEPOLE-128 and the other members of the family, we refer to the CAESAR design document [16].

In case of disagreement between the specifications of ICEPOLE-128 in the CHES and CAESAR documents, we always stick to the version submitted to CAESAR and the available reference implementations of this version.

2.1 Mode of Operation

ICEPOLE uses a duplex-like [5] mode of operation, which operates on an internal state of 1280 bits. This state S is represented as 20 64-bit words $S[0..3][0..4]$. Bits on the same position of all 20 words are called slice. The encryption (as well as the decryption) can be split into the three subsequent phases: initialization, processing of data, and finalization (tag generation), and is shown in Fig. 1.

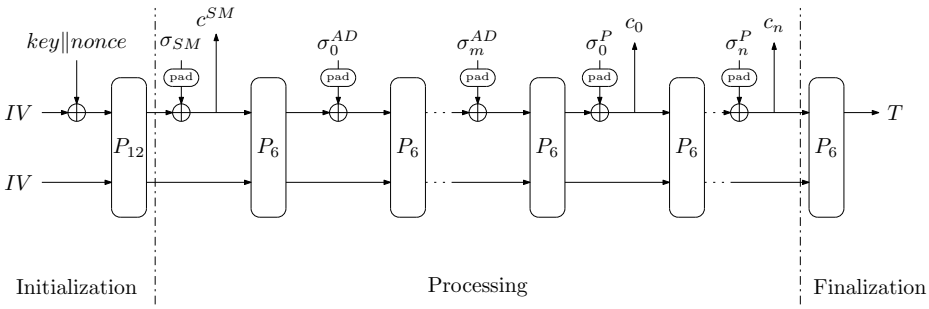

Fig. 1. Encryption of ICEPOLE-128.

Initialization. First, the state S is initialized with a constant value. Afterwards, the 128-bit key and the 128-bit nonce are xored to the internal state. Then, the 12-round variant P_{12} of the ICEPOLE permutation is applied.

Processing of Data. For processing, the associated data and the plaintext are split into 1024-bit blocks, with possibly smaller last blocks. Each of these blocks is padded to 1026 bits. The padding rule is to append a frame bit, followed by a single 1 and zeros until 1026 bits are reached.

After the initialization, the padded secret message number σ_{SM} is xored to the internal state and c_{SM} is extracted. Then, 6 rounds of the ICEPOLE permutation P_6 are applied. After the processing of the secret message number, the associated data blocks σ_i^{AD} are padded and injected, separated by the 6-round ICEPOLE permutation P_6. The plaintext blocks σ_i^P are processed in a similar way, except that ciphertext blocks c_i are extracted. For easier comparison with other sponge-based [3,4] primitives, we move the last permutation call P_6 (after the last plaintext block) to the finalization.

Finalization. Since we moved the last permutation call of the processing to the finalization, the finalization starts with calling P_6. Afterwards, the 128-bit tag T is extracted from the state:

$$T = S[0][0] \| S[1][0].$$

2.2 Permutation

Two variants of the ICEPOLE permutation are used: One with 6 rounds, P_6, and one with 12 rounds, P_{12}. Each round R consists of five steps, $R = \kappa \circ \psi \circ \pi \circ \rho \circ \mu$.

- $\boldsymbol{\mu}$: Mixing of every 20-bit slice through an MDS matrix over $GF(2^5)$.
- $\boldsymbol{\rho}$: Rotation within all 64-bit words.
- $\boldsymbol{\pi}$: Reordering of 64-bit words (words are swapped).
- $\boldsymbol{\psi}$: Parallel application of 256 identical 5-bit S-boxes.
- $\boldsymbol{\kappa}$: Constant addition.

For a detailed description of κ, ψ, π, ρ, and μ, we refer to the CAESAR design document [16].

3 Search for Differential Characteristics

As we will see later, the existence of differential characteristics holding with a high probability is crucial for our attacks on round-reduced ICEPOLE-128. Since ICEPOLE-128 is a bit-oriented construction, automatic search tools are helpful for finding complex differential characteristics with a high probability. ICEPOLE-128 has a rather big internal state of 1280 bits involving many operations per permutation round. Therefore, we have decided to use the guess-and-determine techniques already used for several attacks on hash functions [12,14,15] together with a greedy strategy, which has already been used to find differential characteristics with a high probability for SipHash [10].

We first describe the used concepts for representing differences within the used automatic search tool and propagating them in Section 3.1. Then, we give a high-level overview of our search strategy in Section 3.2.

3.1 Generalized Conditions and Propagation

To represent differential characteristics within the search tool, we have chosen generalized conditions [8]. These conditions are suitable for guess-and-determine-based searches, since they have a very high level of granularity. For instance, with a '`?`', it can be represented that no restrictions are given at some point of a search, or the value of a pair of bits can be completely determined, for instance by '`1`' denoting that both bits of the pair have to have the value 1. The complete set of all 16 generalized conditions can be found in Table 2.

Table 2. Generalized conditions [8].

x, x'	$0,0$	$1,0$	$0,1$	$1,1$
`?`	✓	✓	✓	✓
`-`	✓	–	–	✓
`x`	–	✓	✓	–
`0`	✓	–	–	–
`u`	–	✓	–	–
`n`	–	–	✓	–
`1`	–	–	–	✓
`#`	–	–	–	–

x, x'	$0,0$	$1,0$	$0,1$	$1,1$
`3`	✓	✓	–	–
`5`	✓	–	✓	–
`7`	✓	✓	✓	–
`A`	–	✓	–	✓
`B`	✓	✓	–	✓
`C`	–	–	✓	✓
`D`	✓	–	✓	✓
`E`	–	✓	✓	✓

Apart from the representation, the propagation of differences (or in this case of the generalized conditions) through the components of the ICEPOLE permutation has to be modeled. Here, we make the distinction between the linear part of one round, consisting of the application of μ, ρ, and π, and the nonlinear part ψ, which is the application of 256 5-bit S-boxes. The propagation for each S-box is done by exhaustively calculating all possible solutions for given input and output differences (basically look-ups in the difference distribution table). The propagation of the linear part of each round is modeled by techniques described in [11].

3.2 Search Strategy

On a high level, our search strategy can be split into the following two phases:

1. Search for a valid characteristic with a low number of active S-boxes.
2. Optimize the probability of the characteristic.

The first phase primarily serves to narrow the search space for the second phase. In this first phase, we search for truncated differentials with as few differentially active S-boxes as possible. In this context, an S-box is called active if there are differences on its inputs and outputs. The number of active S-boxes sets an upper bound on the best possible probability of a characteristic, since the maximum differential probability of the ICEPOLE S-box is 2^{-2}.

In the second phase, we search for the actual characteristic. In fact, just using the truncated differential and searching for the best assignment does not give us the best overall result. As we will see later, we search for characteristics having a special form, where a low number of active S-boxes does not necessarily give the best characteristic. Thus, we only fix the truncated differential for one or two rounds, leaving the other rounds completely undetermined, and search for high-probability characteristics by using the greedy algorithm presented in [10].

In summary, the first phase narrows the search space and gives us a good starting point for the second phase. Then, in the second phase, the actual characteristic is searched.

4 The Attack

The first thing to do when analyzing a cryptographic primitive is to find a promising point to attack. Thus, we explain the observations that have led to the attack on the finalization of ICEPOLE-128 using differential cryptanalysis in Section 4.1. After that, we discuss our first findings regarding forgeries in Section 4.2, and explain the trick leading to an improvement of the attack in Section 4.3. Finally, in Section 4.4, we show characteristics for 5 and 6 rounds of the ICEPOLE permutation which are not suitable for a forgery, but have a better probability than the best characteristics published by the designers [16].

4.1 Basic Attack Strategy

ICEPOLE uses a sponge-like mode of operation like several other CAESAR candidates including Ascon [9], Keyak [6], or NORX [1,2]. When comparing those Sponge constructions with ICEPOLE, it is noticeable that the last permutation, which separates the last plaintext injection from the extraction of the tag, has much fewer rounds in the case of ICEPOLE compared to the others.

For more detail, we have a closer look at the number of permutation rounds during the three different stages for the proposals of Ascon, ICEPOLE, Keyak, and NORX, which have the same security level of 128 bits. The number of rounds in each stage for these four primitives is given in Table 3. We can see that for Ascon and NORX, the number of rounds for data processing is reduced compared to finalization and initialization and for all three competitors of ICEPOLE, the finalization is equally strong as the initialization. In the case of ICEPOLE, the permutation used in the finalization has the same number of rounds as the permutation during the processing of data, and thus just half of the rounds of the permutation used during the initialization. So it is interesting to evaluate if the designers of the competitors of ICEPOLE have been overly conservative in the design of their respective finalization, or if their decision to invest more rounds can be justified.

Table 3. Permutation rounds for some sponge-like CAESAR candidates.

	Initialization	Data Processing	Finalization
Ascon	12	6	12
ICEPOLE	12	6	6
Keyak	12	12	12
NORX	8	4	8

4.2 Creating Forgeries

In this section, we first describe the principles of our attack on a high level. Afterwards, we discuss our preliminary results regarding suitable characteristics when just considering the 1024 bits of the ciphertext blocks to inject differences.

Attack Strategy. For creating forgeries with the help of differential characteristics, we have in principle two attack points in sponge-like constructions as ICEPOLE-128. We can either attack the data processing, or we can perform the attack on the finalization. In both cases, the key to a successful attack lies in the search for a suitable differential characteristic which holds with a high probability.

Fig. 2 shows how a forgery during the processing of the data works. This approach requires a differential characteristic with differences only in those parts of the state that can be modified with message blocks, while the rest of the state has to remain free of differences. In other words, we search for a characteristic capable of producing collisions on the internal state. If we have found such a characteristic with input difference Δ_0 that holds with probability 2^{-x}, we can create a forgery which succeeds with probability 2^{-x} as follows: Assume we know a valid ciphertext-tag pair consisting of two ciphertext blocks $(c_0\|c_1, T)$. Then, the ciphertext-tag pair $(c_0 \oplus \Delta_0\|c_1, T)$ is valid with probability 2^{-x}. Thus, a valid forgery can be created with complexity 2^x.

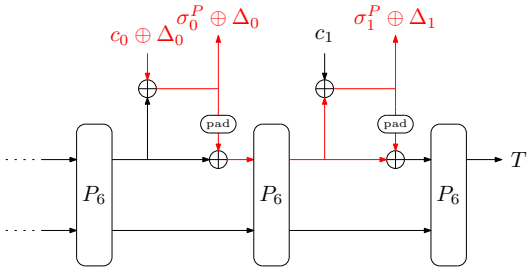

Fig. 2. Forgery during data processing.

The second option, attacking the finalization, is pictured in Fig. 3. In contrast to the previous attack, the requirements on a suitable characteristic can be relaxed. Here, we do not require a collision. It is sufficient that the difference Δ_1 for the tag T is known. The actual difference in the rest of the state does not matter in this attack. In other words, a forgery can be created from a known ciphertext-tag $(c_0\|c_1, T)$ by applying suitable differences to c_1 and T to get $(c_0\|c_1 \oplus \Delta_0, T \oplus \Delta_1)$.

In case of ICEPOLE, the permutation during the processing of the data and the finalization is equally strong. The requirements on suitable characteristics are

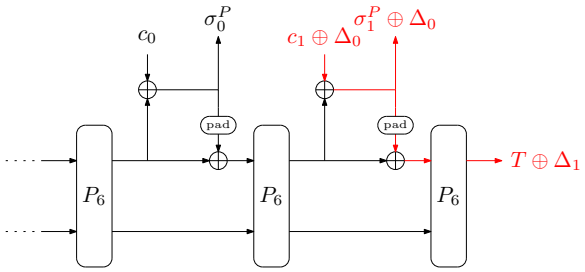

Fig. 3. Forgery during finalization.

less restrictive when attacking the finalization. Thus, attacks on the finalizations are easier to achieve. In addition, the fact that the linear layer is located before the application of the S-boxes comes in handy. ICEPOLE has a state size of 1280 bits. For the generation of the tag, only 128 bits of the 1280 bits are extracted. The other bits do not influence the tag. Since the S-boxes are located at the end of the permutation, 128 of the 256 S-boxes of the last round have no influence on the tag and therefore, do not contribute to the probability of creating a forgery. Moreover, the other 128 S-boxes of the last round only contribute a single bit, which also has a positive effect on the total probability.

Suitable Characteristics. As discussed before, we need characteristics with a good probability, where the input differences lie in the part of the state that can be controlled by a ciphertext block, and where as many of the active S-boxes as possible lie in parts which do not contribute to the probability. However, before we present our results, we describe the findings of the designers [16] and the results by Huang et al. [13].

The designers of ICEPOLE already searched for differential characteristics without any special restrictions. They have found characteristics for 3 rounds with probability $2^{-18.4}$, 4 rounds with $2^{-52.8}$, 5 rounds with $2^{-186.2}$ and 6 rounds with $2^{-555.5}$. Indeed, when considering that the last round of ICEPOLE only contributes partially to the probability, these results look promising from the perspective of an attacker. However, as already observed by Huang et al. [13], these characteristics cannot be used for attacks on the cipher. They showed that if only 1024 bits of a message block are considered suitable for introducing differences, it is impossible to find a 3-round path with 9 active S-boxes in the form 4-1-4. Moreover, they show that the minimum number of active S-boxes in the first round is 2 in this case.

Our search for suitable characteristics supports their result. If we just consider the 1024 bits of the message block suitable for differences, we can create forgeries for 3 rounds with probability $2^{-25.3}$ and, for 4 rounds with a probability close to 2^{-128}. However, in the next section, we explain how we improved the probability for the 4-round attack to $2^{-60.3}$ by exploiting the padding rule of the last processed plaintext block.

4.3 Exploiting the Padding

ICEPOLE uses at most 1024-bit message blocks, which are padded to 1026 bits by appending a frame bit, which is 0 for the last plaintext block, followed by a single 1 and as many zeros until 1026 bits are reached. So using, for instance, a 1016-bit block and a 1024-bit block (where the last byte fulfills the padding rule applied to the 1016-bit block) virtually flips a bit in an otherwise unaccessible part of the state. By using this trick, we are able to use characteristics where only one S-box is active in the first round.

With these improved differential characteristics, we are able to create forgeries for ICEPOLE-128 with the finalization reduced to 3 (out of 6) rounds with probability $2^{-14.8}$, and for 4 rounds (out of 6) with probability $2^{-60.3}$. The characteristics for the 3-round attack can be found in Table 4, and for the 4-round attack in Table 5 of Appendix A.

The 3-round attack on ICEPOLE-128 has been verified using the reference implementation ICEPOLE128v1 submitted to CAESAR with a modified number of rounds for permutation P_6. We fixed a random key at the beginning and encrypted random 1024-bit messages (last byte of messages has to be equal to the padding `0x2`) with random nonces to get 1024-bit ciphertexts. The forgeries are created by applying the difference shown in Table 4 to ciphertext and tag and discarding the last byte of the ciphertext. Removing the last byte of the ciphertext introduces a difference at bit 1026. Backed up by our experiments (2^{28} message-tag pairs), a forgery for round-reduced ICEPOLE-128, where the finalization is reduced to 3 out of 6 rounds, can be created with probability $2^{-11.7}$. For the 4-round attack, the probability is too low to be verified experimentally. However, parts of the used characteristic which have a high probability have been verified.

To introduce differences with the help of the padding, we can either extend or truncate known ciphertexts. As already discussed before, creating forgeries by truncating the last byte of the ciphertext only works if the last byte of the message before encryption equals the padding. Extending 1016-bit ciphertexts requires to guess 8 bits of the internal state correctly and hence decreases the probability by 2^{-8}. In the case of messages consisting of a fractional number of bytes, 1022-bit ciphertexts can be extended, leading to a decrease of 2^{-2}.

4.4 Characteristics for the Permutation

We also considered characteristics without any special restrictions. We have been able to improve the results published in the design documents. We have found a 5-round characteristic with an estimated probability of $2^{-104.5}$ and a 6-round characteristic with an estimated complexity of $2^{-258.4}$. The characteristics are given in Table 6 and Table 7 of Appendix A. Both characteristics are a perceptible improvement over the characteristics given in the design document [16], which have a probability of $2^{-186.2}$ and $2^{-555.3}$, respectively.

5 Conclusion

In this work, we have analyzed the resistance of ICEPOLE-128 against forgery attacks. Our attacks work for versions of ICEPOLE-128 where the permutation used during the finalization is reduced to 4 (out of 6) rounds. This means that ICEPOLE-128 has a security margin of 2 rounds, which is lower than the 3 rounds expected by the designers [16].

Acknowledgments. The work has been supported in part by the Austrian Science Fund (project P26494-N15) and by the Austrian Research Promotion Agency (FFG) and the Styrian Business Promotion Agency (SFG) under grant number 836628 (SeCoS).

References

1. Aumasson, J., Jovanovic, P., Neves, S.: NORX. Submission to the CAESAR competition: `http://competitions.cr.yp.to/round1/norxv1.pdf` (2014)
2. Aumasson, J., Jovanovic, P., Neves, S.: NORX: Parallel and Scalable AEAD. In: Kutylowski, M., Vaidya, J. (eds.) Computer Security – ESORICS 2014, Part II. LNCS, vol. 8713, pp. 19–36. Springer (2014)
3. Bertoni, G., Daemen, J., Peeters, M., Van Assche, G.: Sponge Functions. ECRYPT Hash Workshop 2007 (May 2007)
4. Bertoni, G., Daemen, J., Peeters, M., Van Assche, G.: On the Indifferentiability of the Sponge Construction. In: Smart, N.P. (ed.) Advances in Cryptology – EUROCRYPT 2008. LNCS, vol. 4965, pp. 181–197. Springer (2008)
5. Bertoni, G., Daemen, J., Peeters, M., Van Assche, G.: Duplexing the Sponge: Single-Pass Authenticated Encryption and Other Applications. In: Miri, A., Vaudenay, S. (eds.) Selected Areas in Cryptography – SAC 2011. LNCS, vol. 7118, pp. 320–337. Springer (2011)
6. Bertoni, G., Daemen, J., Peeters, M., Van Assche, G., Van Keer, R.: Keyak. Submission to the CAESAR competition: `http://competitions.cr.yp.to/round1/keyakv1.pdf` (2014)
7. Biham, E., Shamir, A.: Differential Cryptanalysis of DES-like Cryptosystems. J. Cryptology 4(1), 3–72 (1991)
8. De Cannière, C., Rechberger, C.: Finding SHA-1 Characteristics: General Results and Applications. In: Lai, X., Chen, K. (eds.) Advances in Cryptology – ASIACRYPT 2006. LNCS, vol. 4284, pp. 1–20. Springer (2006)
9. Dobraunig, C., Eichlseder, M., Mendel, F., Schläffer, M.: Ascon. Submission to the CAESAR competition: `http://competitions.cr.yp.to/round1/asconv1.pdf` (2014)
10. Dobraunig, C., Mendel, F., Schläffer, M.: Differential Cryptanalysis of SipHash. In: Joux, A., Youssef, A.M. (eds.) Selected Areas in Cryptography – SAC 2014. LNCS, vol. 8781, pp. 165–182. Springer (2014)
11. Eichlseder, M., Mendel, F., Nad, T., Rijmen, V., Schläffer, M.: Linear Propagation in Efficient Guess-and-Determine Attacks. In: Lilya Budaghyan, Tor Helleseth, M.G.P. (ed.) International Workshop on Coding and Cryptography. pp. 193–202 (2013)

12. Eichlseder, M., Mendel, F., Schläffer, M.: Branching heuristics in differential collision search with applications to SHA-512. In: Cid, C., Rechberger, C. (eds.) Fast Software Encryption – FSE 2014. LNCS, vol. 8540, pp. 473–488. Springer (2014)
13. Huang, T., Tjuawinata, I., Wu, H.: Differential-linear cryptanalysis of ICEPOLE. In: Leander, G. (ed.) Fast Software Encryption – FSE 2015. LNCS, vol. 9054, pp. 243–263. Springer (2015)
14. Mendel, F., Nad, T., Schläffer, M.: Finding SHA-2 Characteristics: Searching through a Minefield of Contradictions. In: Lee, D.H., Wang, X. (eds.) Advances in Cryptology – ASIACRYPT 2011. LNCS, vol. 7073, pp. 288–307. Springer (2011)
15. Mendel, F., Nad, T., Schläffer, M.: Improving Local Collisions: New Attacks on Reduced SHA-256. In: Johansson, T., Nguyen, P.Q. (eds.) Advances in Cryptology – EUROCRYPT 2013. LNCS, vol. 7881, pp. 262–278. Springer (2013)
16. Morawiecki, P., Gaj, K., Homsirikamol, E., Matusiewicz, K., Pieprzyk, J., Rogawski, M., Srebrny, M., Wójcik, M.: ICEPOLE. Submission to the CAESAR competition: `http://competitions.cr.yp.to/round1/icepolev1.pdf` (2014)
17. Morawiecki, P., Gaj, K., Homsirikamol, E., Matusiewicz, K., Pieprzyk, J., Rogawski, M., Srebrny, M., Wójcik, M.: ICEPOLE: High-Speed, Hardware-Oriented Authenticated Encryption. In: Batina, L., Robshaw, M. (eds.) Cryptographic Hardware and Embedded Systems – CHES 2014. LNCS, vol. 8731, pp. 392–413. Springer (2014)
18. The CAESAR committee: CAESAR: Competition for authenticated encryption: Security, applicability, and robustness (2014), `http://competitions.cr.yp.to/caesar.html`

A Differential Characteristics

Table 4. 3-round characteristic suitable for forgery with probability $2^{-14.8}$.

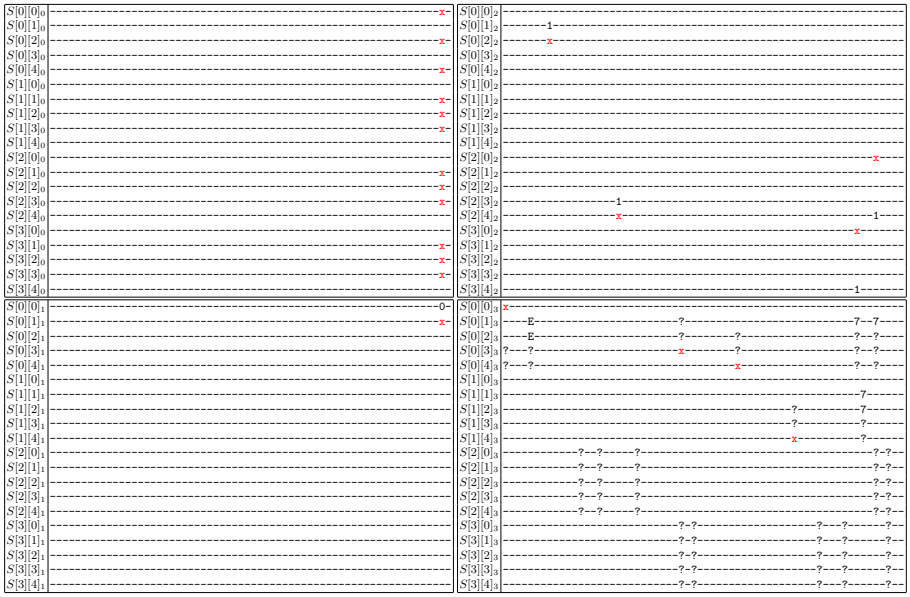

Table 5. 4-round characteristic suitable for forgery with probability $2^{-60.3}$.

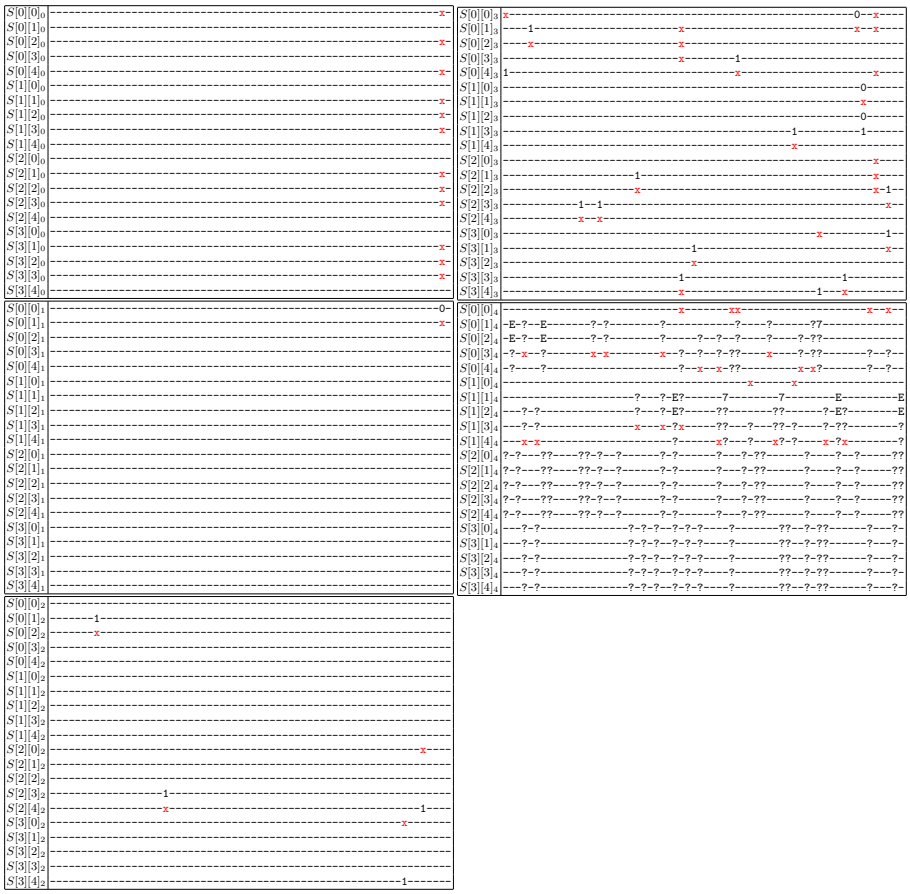

Table 6. 5-round characteristic with probability $2^{-104.5}$.

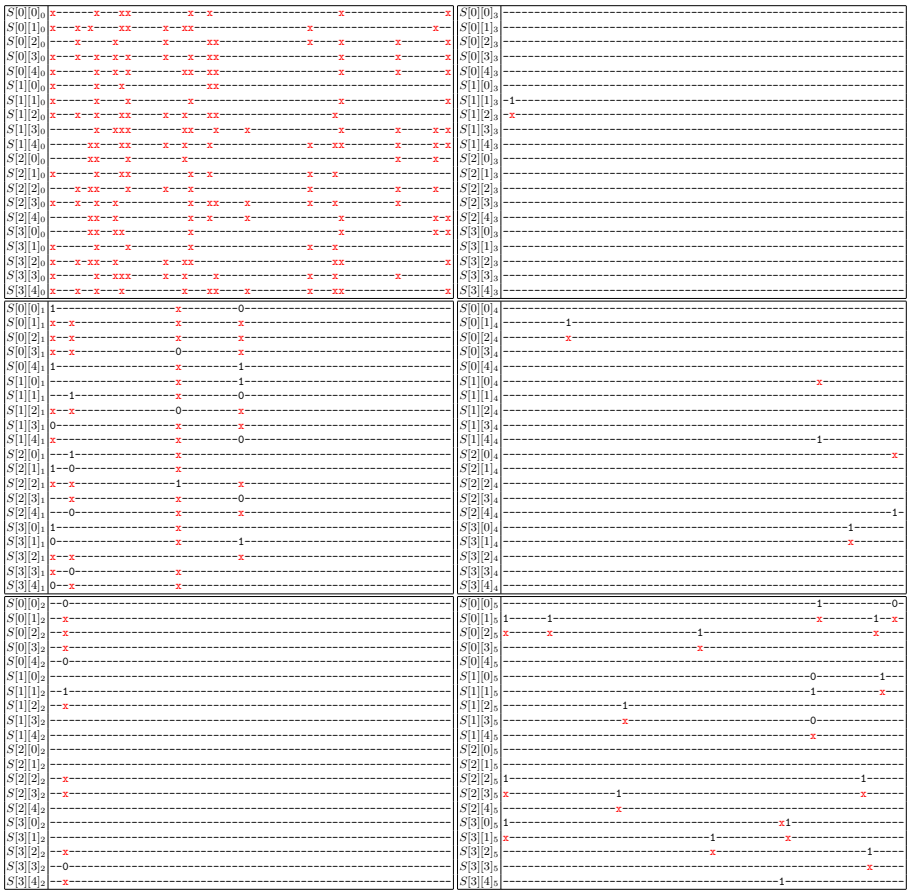

Table 7. 6-round characteristic with probability $2^{-258.3}$.