Designs, Codes and Cryptography
https://doi.org/10.1007/s10623-021-00850-2

# On the Resilience of Even-Mansour to Invariant Permutations

**Bart Mennink**[1] · **Samuel Neves**[2]

Received: 8 October 2020 / Revised: 26 January 2021 / Accepted: 3 February 2021


**Abstract**
Symmetric cryptographic primitives are often exposed to invariances: deterministic relations between plaintexts and ciphertexts that propagate through the primitive. Recent invariant subspace attacks have shown that these can be a serious issue. One way to mitigate invariant subspace attacks is at the primitive level, namely by proper use of round constants (Beierle et al., CRYPTO 2017). In this work, we investigate how to thwart invariance exploitation at the mode level, namely by assuring that a mode never evaluates its underlying primitive under any invariance. We first formalize the use of invariant cryptographic permutations from a security perspective, and analyze the Even-Mansour block cipher construction. We further demonstrate how the model composes, and apply it to the keyed sponge construction. The security analyses exactly pinpoint how the presence of linear invariances affects the bounds compared with analyses in the random permutation model. As such, they give an exact indication *how* invariances can be exploited. From a practical side, we apply the derived security bounds to the case where the Even-Mansour construction is instantiated with the 512-bit ChaCha permutation, and derive a distinguishing attack against Even-Mansour-ChaCha in $2^{128}$ queries, faster than the birthday bound. Comparable results are derived for instantiation using the 200-bit Keccak permutation without round constants (attack in $2^{50}$ queries), the 1024-bit CubeHash permutation (attack in $2^{256}$ queries), and the 384-bit Gimli permutation without round constants (attack in $2^{96}$ queries). The attacks do not invalidate the security of the permutations themselves, but rather they demonstrate the tightness of our bounds and confirm that care should be taken when employing a cryptographic primitive that has nontrivial linear invariances.

**Keywords** Permutation · Invariances · Security analysis · Indistinguishability · Attacks

**Mathematics Subject Classification** 94A60

Communicated by F. Mendel.

✉ Bart Mennink
b.mennink@cs.ru.nl

Samuel Neves
sneves@dei.uc.pt

[1] Digital Security Group, Radboud University, Nijmegen, The Netherlands

[2] CISUC, Department of Informatics Engineering, University of Coimbra, Coimbra, Portugal

Published online: 23 February 2021

## 1 Introduction

The core of symmetric cryptographic primitives and modes alike has been, historically, the block cipher. Its applications range from compression functions to authenticated encryption. For example, the AES block cipher [44] is at the core of AES-GCM [83], AES-CCM [104], AES-CBC [52], OCB [77], AEZ [67], and many other widely used modes.

Block ciphers usually consist of two main components: a (iterated) round function, which deals with input data, and a key schedule, which expands one key to multiple, random-looking, block-sized keys. Key material is regularly added to the data in-between evaluations of the round function. More recently, tweakable block ciphers [82] added a third input, the tweak, which in existing designs is essentially treated as key material [70].

A different approach has been gaining momentum in the last few years: permutation based cryptography. In this paradigm, there is no key; every part of the input is treated equally. Key material is then added only "at the top", as in the Even-Mansour construction [50,53], or the various keyed sponge modes [2,19,31,41,85]. This arguably makes designs cleaner and easier to analyze, as there are no separate processing lanes for key and data, and enables a clearer mixing of the key and the data.

For most permutation based modes of operation, security is argued in the random permutation model, meaning that the underlying permutation is assumed to be perfect. So as to meet this assumption as close as possible, permutations are often designed to prevent all kinds of structural defects, harmless as they may seem. This is the so-called "hermetic" design strategy. One usually accomplishes this by designing a strong iterated round function resistant to differential, linear, and other standard attacks, and adding round constants in-between evaluations of the round function. These constants achieve two goals:

- They make every round distinct, preventing internal differentials [94], slide attacks [22], and a defective cycle structure [86];
- They can prevent invariant subspace attacks, by breaking the symmetries that are preserved across round functions [9,79].

Many permutations follow this design pattern, such as the Keccak permutation [18,56], Ascon [49], PHOTON [66], Prøst [74], Gimli [16], Mixifer [99], or Xoodoo [40].

Other permutations, often single-purpose components of larger modes of operation, decide to omit such round constants, and let the mode of operation ensure that "bad" states are never reached. This is the case of Salsa20 [15] or NORX [4], in which invariance-breaking constants are simply added "at the top". This can result in simpler designs, at the cost of losing the random permutation model as a useful modeling tool.

### 1.1 Invariances

Consider the event that for a permutation $\mathsf{P} : \{0,1\}^n \to \{0,1\}^n$, there are one or more functions $\lambda : \{0,1\}^n \to \{0,1\}^n$ such that

$$\lambda \circ \mathsf{P}(x) = \mathsf{P} \circ \lambda(x) \tag{1}$$

for any $x \in \{0,1\}^n$. We say that $\mathsf{P}$ is *invariant* under $\lambda$, and call $\lambda$ an *invariance for* $\mathsf{P}$. The identity mapping is always an invariance for $\mathsf{P}$, but ideally, it is the only one, the reason being that non-trivial invariances expose non-random behavior of $\mathsf{P}$ and may allow and attacker to distinguish the primitive from random. Various examples of invariances for permutations are given in Sect. 2.

Invariances are inherent to symmetric cryptographic primitives. Barkan and Biham [8] studied invariances in AES under the guise of "self-duality". The work was revisited by Van Le [102] and Bouillaguet et al.[25]. A particularly fruitful invariance in the AES round $F_k$ is that of column rotation [102, Proposition 11]:

$$\lambda_i(F_k(x)) = F_{\lambda_i(k)}(\lambda_i(x)), \tag{2}$$

where $\lambda_i$ is any of the byte permutations

$$\begin{pmatrix} x_4 & x_8 & x_{12} & x_0 \\ x_5 & x_9 & x_{13} & x_1 \\ x_6 & x_{10} & x_{14} & x_2 \\ x_7 & x_{11} & x_{15} & x_3 \end{pmatrix}, \begin{pmatrix} x_8 & x_{12} & x_0 & x_4 \\ x_9 & x_{13} & x_1 & x_5 \\ x_{10} & x_{14} & x_2 & x_6 \\ x_{11} & x_{15} & x_3 & x_7 \end{pmatrix}, \begin{pmatrix} x_{12} & x_0 & x_4 & x_8 \\ x_{13} & x_1 & x_5 & x_9 \\ x_{14} & x_2 & x_6 & x_{10} \\ x_{15} & x_3 & x_7 & x_{11} \end{pmatrix}.$$

This property was exploited for attacking the ALRED family of authenticators [45] based on the keyless AES round [51, Sect. 4]. This property has also been used for attacks on various AES based ciphers and permutations, such as on PAES [105] by Jean et al.[71,72], on Simpira v1 [63] by Rønjom [97], and on Haraka v1 [76] by Jean [69]. Likewise, the permutations underlying ChaCha [13], BLAKE2 [5], and NORX [4] have similar invariances as the AES round. Notably, the cryptanalysis of NORX v2.0 [21,30] and the "chosen-IV" attacks on BLAKE2 [65] exploit these properties.

Invariant subspace attacks were formalized by Leander et al.[79] alongside their cryptanalysis of the PRINTcipher [75]. The notion received further formal analysis by Leander et al.[80] and Beierle et al.[9]. Todo et al.[100] expanded the notion to nonlinear invariances. Beyne generalized the analysis, and applied it to Midori-64 [20].

### 1.2 Dealing with Invariances

Basically, there are two ways to resolve the potential weaknesses caused by the presence of invariances:

- At the primitive level: avoid the presence of invariances. This is achieved by using more involved key schedules and round constants. Most invariant subspace attacks to date [20,27,64,69,79,80,97,100] exploit weaknesses at the primitive level. Beierle et al.[9] studied the issue of invariances in SPNs, with particular focus on the linear layer, and investigated the effect of round constants on the resistance against invariant subspace attacks;
- At the mode level: ensure that invariances are never inspected. In this case, the primitive must be "masked" at the mode level so that it is never evaluated for two values $x$ and $\lambda(x)$, where $\lambda$ is any of the non-trivial invariances of $\mathsf{P}$.

The former approach delivers primitives that may function as standalone objects to be used as a black-box by mode designers and implementers. The latter approach allows for simpler, more minimalistic primitives, but mode designers should pay attention to the fact that invariances are avoided. Theorists, however, typically ignore the issue of invariances entirely, simply assuming ideal primitives.

### 1.3 Invariances in Practice

The issue of invariances is not a purely theoretical matter. Invariances are inherent to symmetric cryptography, and some cryptographic permutations, including the ones underlying

Salsa20 [15], ChaCha [13], NORX [4], BLAKE2 [5], and CubeHash [14], do have certain invariances that make them easily distinguishable from their ideal counterparts. These invariances are not always attributed to flaws in the scheme: sometimes, the presence of invariances allows to obtain simpler and faster schemes. Yet, there are no available tools to study their behavior when used in permutation based modes. Given the popularity of ChaCha and AES-like building blocks to build new schemes, the case deserves proper formal understanding.

We stress that we do *not* advocate for general purpose primitive designers to design purposefully imperfect cryptographic primitives. Instead, our focus is on combined mode and primitive designers, noting that nowadays many cryptographic modes are designed alongside their single-purpose primitives. We have noticed this pattern in many CAESAR candidates [28], e.g., Prøst, NORX, ICEPOLE, Minalpher, PRIMATEs, and PAEQ. When designing a single-purpose permutation, as in these cases, it seems reasonable to handle properties that simplify or speed up the permutation by relying on the mode to protect against them (as is already the case, in a simpler setting, with Salsa20 and ChaCha).

### 1.4 Our Contribution

We present a structural analysis of how to handle the problem of invariant primitives at the mode level. Whereas Beierle et al. specify conditions on the use of round constants so as to *avoid* invariances, we describe how the security of masked primitive modes behaves *in the presence* of invariances. The work consists of a theoretical part (settling the model and deriving security bounds) and a practical part (translating the security bounds to attacks on concrete cryptographic permutations), as we will detail below.

#### Invariant Random Permutation Model

We first formalize a model for the use of invariant permutations in cryptographic modes. The model is fairly straightforward: instead of randomly drawing an $n$-bit permutation from the set of all $n$-bit permutations, one fixes a set of linear invariances $\Lambda$ and draws the permutation from the set of all $n$-bit permutations *that satisfy* (1) *for all* $\lambda \in \Lambda$. The model is given in Sect. 3.1. Note that the model is general; in fact, it covers more than the typical invariances we observe in Salsa20, ChaCha, NORX, and so on. We also point out that this practice is riskier than the hermetic design approach, as one needs to be sure that every allowed invariance is adequately covered by the analysis.

#### Analysis of Even-Mansour

Then, in Sect. 3.2 we analyze the plain single-key Even-Mansour construction [53] in the invariant permutation model. The tight security bound derived for this construction exactly pinpoints how the original (i.e., the random permutation model based) security bound deteriorates in the presence of invariances. Quite surprisingly: the loss is less trivial and more significant than initially thought. The reason for this is that a strong security bound can be derived *only if* for any input $x$ to the primitive $\mathsf{P}$, the value $\lambda(x)$ should never be evaluated by $\mathsf{P}$, *for any* $\lambda \in \Lambda$.

In detail, the analysis leads to a security bound in the invariant permutation model that is at most $|\Lambda|$ times the bound in the random permutation model plus $|\Lambda| - 1$ times a technical term $\mathbf{P}^*$ that bounds the probability of collisions over the invariances.

### Analysis of Keyed Sponge

The analysis of the Even-Mansour construction is particularly useful as it appears as building block in many security proofs. For example, security of the keyed sponge [2,19,31,41,59,85,90] and other MAC designs [87,89] can be reduced to the security of the Even-Mansour construction. In Sect. 3.4, we show how the security analysis of the Even-Mansour construction generalizes to a security analysis of the keyed sponge in the invariant permutation model.

### Instantiation

The term $\mathbf{P}^*$ is involved, and can be described as a combinatorial property of the invariances (refer to Theorem 1 for precise statement of the bound). Further derivation of the bound is then tied to the specific primitive and invariances associated with it. This, so far, leaves us with only a partial solution. In Sect. 4 we look at the remaining term in more detail for specific cryptographic primitives and invariances.

In Sect. 4.1 we consider additive invariances for the Salsa20 permutation. This permutation satisfies that it is invariant under parallel addition of 16 times the value $\Delta = 2^{31}$. We present a simple attack on Even-Mansour-Salsa20 in a constant number of construction queries that matches our security bound. The attack does not invalidate the security strength of Salsa20; it just confirms that its permutation should not be used in the Even-Mansour construction.

In Sect. 4.2 we consider linear invariances, noting that for a linear invariance $\lambda$, the probability term $\mathbf{P}^*$ can in turn be related with the rank of $\mathrm{id} \oplus \lambda$. This rank can be easily computed, and attacks can be mounted on the Even-Mansour construction based on the cryptographic permutations exhibiting that invariance. For the 512-bit Chacha [13] permutation, we obtain a distinguishing attack with a complexity of $2^{128}$ queries. For the 1600-bit Keccak permutation without round constants [18], a distinguishing attack in complexity $2^{400}$ queries is derived, and this attack scales down to smaller versions of Keccak, up to an attack in $2^{50}$ construction queries for 200-bit Keccak. We likewise derive a distinguishing attack for 1024-bit CubeHash [14] in $2^{256}$ construction queries, and for 384-bit Gimli without round constants [16] in $2^{96}$ construction queries. Again, none of these attacks invalidate the security claims of the primitives themselves. They rather demonstrate that one must be careful when using a cryptographic primitive that has invariances, and exemplify decisions that designers could conceivably make when designing a particular scheme.

The framework is likewise able to capture the attacks on NORXv2 [21,30], as well as the BLAKE2 [65] attacks with chosen IV. It can furthermore readily be extended to match the invariant subspaces found on Simpira v1 [97], or the attacks that rely on a block cipher hitting a particular set of bad keys.

### Discussion

The attacks on Even-Mansour instantiated with ChaCha, Keccak without round constants, CubeHash, and Gimli without round constants are well below the birthday bound on the corresponding state sizes. In Sect. 5.1, we consider different avenues to salvaging Even-Mansour. One approach is to consider tweakable variants of Even-Mansour, such as a generalization of the Masked Even-Mansour construction [62] that *masks* the key before it is added to the message. Details on this construction are given in Appendix A. In this appendix, we also explain how the results subsequently extend to the Offset Public Permutation (OPP) authenticated encryption scheme by Granger et al.[62]. Another approach is to consider 2-round

Even-Mansour. In the random permutation model, this construction has received solid investigation lately [23,32]. The attacks of Sect. 4 rely on the fact that an attacker may observe the structure of the inputs to and outputs of the cryptographic permutation, and by masking the input and output, or by adding an extra round, the attacker cannot observe this structure anymore, and the attack fails. Note that a minimal construction of 2-round Even-Mansour would take a single key, and add "round constants" to derive mutually slightly different round keys. Stretching the idea to multiple round Even-Mansour, this idea resembles the current practice of primitive level invariant subspace attack mitigation.

In Sect. 5.2, we discuss how our security analysis can be translated to multi-key security, where the attacker can query multiple instances of the constructions simultaneously. It appears that in this case, the attacker does not get any significant gain over the single-key setting. Finally, in Sect. 5.3, we informally discuss how our analyses may generalize to invariances that do not necessarily apply to the entire domain.

### 1.5 Notation

Throughout, parameters $\kappa, \tau, n \in \mathbb{N}$ denote key size, tweak size, and state size. The set of $n$-bit permutations is denoted $\mathsf{perm}(n)$. We denote by $\mathsf{tperm}(\mathcal{T}, n)$ the set of all families of $n$-bit permutations indexed by $t \in \mathcal{T}$. For $m \in \mathbb{N}$ such that $m \leq n$, we denote the falling factorial as $(n)_m = n(n-1)\cdots(n-m+1) = n!/(n-m)!$. For a finite set $\mathcal{X}$, we denote by $x \xleftarrow{\$} \mathcal{X}$ the uniformly random sampling of $x$ from $\mathcal{X}$.

## 2 Invariances

A permutation $\mathsf{P} \in \mathsf{perm}(n)$ is *invariant* under bijection $\lambda \in \mathsf{perm}(n)$ if for any $x \in \{0,1\}^n$,

$$\lambda \circ \mathsf{P}(x) = \mathsf{P} \circ \lambda(x)\,. \tag{3}$$

We call $\lambda$ an *invariance for* $\mathsf{P}$. Note that the identity function id is always an invariance for $\mathsf{P}$ (regardless of $\mathsf{P}$), and if $\mathsf{P}$ is invariant under $\lambda, \lambda'$, it is also invariant under $\lambda^{-1}$ and $\lambda \circ \lambda'$. In particular, the set of invariances for $\mathsf{P}$ forms a group $(\Lambda, \circ)$. For a group of invariances $\Lambda$, we write $\mathsf{perm}[\Lambda](n)$ as the set of all permutations that are invariant under $\Lambda$. In our work, we restrict our focus to *linear* invariances. We refer to Courtois [37,38] for research on nonlinear invariances.

In the remainder of this section, we discuss invariances for various theoretical and practical functions.

### 2.1 Random Permutation

If $\mathsf{P} : \{0,1\}^n \to \{0,1\}^n$ is a random permutation, i.e., a function that generates each response uniformly at random without replacement from $\{0,1\}^n$, with high probability $\Lambda = \{\mathrm{id}\}$. Stated differently: with high probability, this function has no invariances except for the trivial one.

### 2.2 Iterated Random Permutation

In the case $\mathsf{P} : \{0,1\}^n \to \{0,1\}^n$ is the repeated composition of the same round function $f^r(\cdot)$, we have the invariant

$$f \circ f^r = f^r \circ f \,.$$

In the case of an unknown round function, as in the case of block ciphers, finding pairs of inputs of the form $(x, f(x))$, i.e., *slid pairs*, is a core step of slide attacks [7,22]. Since $(x, f(x))$ implies $(P(x), f(P(x))$ at the output, recovering the key from such a slid pair is often efficiently achievable (e.g., KeeLoq [1]).

### 2.3 Salsa Family of Permutations

Salsa20 [15] is part of the eSTREAM portfolio of stream ciphers. Its core primitive is a permutation operating on a 16-word state, usually treated as a $4 \times 4$ matrix

$$\begin{pmatrix} x_0 & x_1 & x_2 & x_3 \\ x_4 & x_5 & x_6 & x_7 \\ x_8 & x_9 & x_{10} & x_{11} \\ x_{12} & x_{13} & x_{14} & x_{15} \end{pmatrix} .$$

The Salsa20 round consists of four parallel "quarter-round" operations, as follows:

$$\begin{aligned} x_0, x_4, x_8, x_{12} &= G(x_0, x_4, x_8, x_{12}) \,, \\ x_5, x_9, x_{13}, x_1 &= G(x_5, x_9, x_{13}, x_1) \,, \\ x_{10}, x_{14}, x_2, x_6 &= G(x_{10}, x_{14}, x_2, x_6) \,, \\ x_{15}, x_3, x_7, x_{11} &= G(x_{15}, x_3, x_7, x_{11}) \,, \end{aligned}$$

followed by a transposition of the state. In other words, Salsa20 consists of the composition of "column rounds" followed by "row rounds". The transposition is omitted after the last round.

Determining the invariances of Salsa20 is straightforward. The column rounds, when treating $G$ as a random permutation, are invariant under any permutation of the diagonals. Furthermore, the transposition forces the admissible invariant permutations to be restricted to rotations.

Besides this set of rotational invariances, Salsa20's $G$ has itself some invariant properties. In particular, $G$ is invariant with respect to addition (or xor) by $\{2^{31}\}^4$, as observed by Wagner [103], Robshaw, and Castro et al.[29].

It is interesting to note that Salsa20's predecessor, Salsa10 [12, §5], did not have any simple invariances, owing to its different quarter-round and the addition of a constant every two rounds. Salsa20's invariances were an explicit design choice.

#### 2.3.1 ChaCha, BLAKE2, and NORX

ChaCha [13] is a variant of Salsa20. It operates on the same $4 \times 4$ state as Salsa20, but the linear layer is slightly different. The ChaCha round first applies four parallel "quarter rounds"

$$\begin{aligned} x_0, x_4, x_8, x_{12} &= G(x_0, x_4, x_8, x_{12}) \,, \\ x_1, x_5, x_9, x_{13} &= G(x_1, x_5, x_9, x_{13}) \,, \\ x_2, x_6, x_{10}, x_{14} &= G(x_2, x_6, x_{10}, x_{14}) \,, \\ x_3, x_7, x_{11}, x_{15} &= G(x_3, x_7, x_{11}, x_{15}) \,, \end{aligned}$$

followed by a "shift rows" operation

$$\begin{pmatrix} x_0 & x_1 & x_2 & x_3 \\ x_4 & x_5 & x_6 & x_7 \\ x_8 & x_9 & x_{10} & x_{11} \\ x_{12} & x_{13} & x_{14} & x_{15} \end{pmatrix} \longmapsto \begin{pmatrix} x_0 & x_1 & x_2 & x_3 \\ x_5 & x_6 & x_7 & x_4 \\ x_{10} & x_{11} & x_8 & x_9 \\ x_{15} & x_{12} & x_{13} & x_{14} \end{pmatrix}.$$

The next round uses the same quarter-round, followed by the inverse of the "shift rows" operation.

The invariances here are similar to Salsa20. The quarter-round layer is invariant under any permutation of state columns. The linear layer, however, is only invariant under rotations of columns. Thus, ChaCha is invariant under rotations of columns.

Several other permutations based on the ChaCha, such as the ones in BLAKE2 [5] and NORX [4], share the same property. The rotational invariance of this round structure was implicitly used in the "chosen-IV" BLAKE2 attacks of Guo et al.[65], as well as explicitly described in the cryptanalysis of NORXv2 [21,30].

### 2.4 CubeHash

CubeHash [14] is an ARX permutation based SHA-3 candidate which, like Salsa20, had a highly symmetric round function. The underlying permutation works on a state of 32 words $x[0..31]$ of 32 bits each. It is invariant under the following 15 permutations of words:

$$(1,0,3,2,5,4,7,6,9,8,11,10,13,12,15,14,17,16,19,18,21,20,23,22,25,24,27,26,29,28,31,30),$$
$$(2,3,0,1,6,7,4,5,10,11,8,9,14,15,12,13,18,19,16,17,22,23,20,21,26,27,24,25,30,31,28,29),$$
$$(3,2,1,0,7,6,5,4,11,10,9,8,15,14,13,12,19,18,17,16,23,22,21,20,27,26,25,24,31,30,29,28),$$
$$(4,5,6,7,0,1,2,3,12,13,14,15,8,9,10,11,20,21,22,23,16,17,18,19,28,29,30,31,24,25,26,27),$$
$$(5,4,7,6,1,0,3,2,13,12,15,14,9,8,11,10,21,20,23,22,17,16,19,18,29,28,31,30,25,24,27,26),$$
$$(6,7,4,5,2,3,0,1,14,15,12,13,10,11,8,9,22,23,20,21,18,19,16,17,30,31,28,29,26,27,24,25),$$
$$(7,6,5,4,3,2,1,0,15,14,13,12,11,10,9,8,23,22,21,20,19,18,17,16,31,30,29,28,27,26,25,24),$$
$$(8,9,10,11,12,13,14,15,0,1,2,3,4,5,6,7,24,25,26,27,28,29,30,31,16,17,18,19,20,21,22,23),$$
$$(9,8,11,10,13,12,15,14,1,0,3,2,5,4,7,6,25,24,27,26,29,28,31,30,17,16,19,18,21,20,23,22),$$
$$(10,11,8,9,14,15,12,13,2,3,0,1,6,7,4,5,26,27,24,25,30,31,28,29,18,19,16,17,22,23,20,21),$$
$$(11,10,9,8,15,14,13,12,3,2,1,0,7,6,5,4,27,26,25,24,31,30,29,28,19,18,17,16,23,22,21,20),$$
$$(12,13,14,15,8,9,10,11,4,5,6,7,0,1,2,3,28,29,30,31,24,25,26,27,20,21,22,23,16,17,18,19),$$
$$(13,12,15,14,9,8,11,10,5,4,7,6,1,0,3,2,29,28,31,30,25,24,27,26,21,20,23,22,17,16,19,18),$$
$$(14,15,12,13,10,11,8,9,6,7,4,5,2,3,0,1,30,31,28,29,26,27,24,25,22,23,20,21,18,19,16,17),$$
$$(15,14,13,12,11,10,9,8,7,6,5,4,3,2,1,0,31,30,29,28,27,26,25,24,23,22,21,20,19,18,17,16).$$

These invariances, or "symmetric states", were first identified by Aumasson et al.[3] and later further classified by Ferguson et al.[55]. One of these invariances will be considered in detail in Sect. 4.2.3.

### 2.5 Bitsliced Designs

Keccak [18], Ascon [49], CBEAM [98], Mixifer [99], Xoodoo [40], and many others are part of a design principle that intentionally creates rotationally-invariant functions, and later adds constants as a symmetry-breaking step. As first exhibited by Daemen [39], such bitsliced designs are advantageous for practical reasons, both in implementation and performance tradeoffs, protection against side-channel attacks [42], as well as ease of analysis [98].

In the particular case of Keccak, whose state consists of a $5 \times 5$ matrix of $\{1, 2, 4, 8, 16, 32, 64\}$-bit words, the round function $\chi \circ \pi \circ \rho \circ \theta$ is invariant under the rotation of every word by the same amount. Comparable invariances occur for the other primitives.

### 2.6 Gimli

Gimli [16] is a recently proposed permutation operating on a $3 \times 4$ matrix of 32-bit words. Its round functions, without the constant additions, lead to invariance under the following permutations of state columns:

$$\begin{pmatrix} x_1 & x_0 & x_3 & x_2 \\ x_5 & x_4 & x_7 & x_6 \\ x_9 & x_8 & x_{11} & x_{10} \end{pmatrix}, \begin{pmatrix} x_2 & x_3 & x_0 & x_1 \\ x_6 & x_7 & x_4 & x_5 \\ x_{10} & x_{11} & x_8 & x_9 \end{pmatrix}, \begin{pmatrix} x_3 & x_2 & x_1 & x_0 \\ x_7 & x_6 & x_5 & x_4 \\ x_{11} & x_{10} & x_9 & x_8 \end{pmatrix}.$$

## 3 Secure Usage of Invariant Random Permutations

We consider the security of constructions reminiscent of the Even-Mansour construction [53]. The security model is outlined in Sect. 3.1, and Even-Mansour and its security in the invariant permutation model are stated in Sect. 3.2. The security proof is given in Sect. 3.3. This Even-Mansour construction and analysis is particularly useful as it appears as building block in many security proofs. In particular, by considering that the keyed sponge can be seen as a composition of the Even-Mansour construction, we demonstrate in Sect. 3.4 how the analysis of Even-Mansour in the invariant permutation model carries over to the keyed sponge.

We remark that the ideas generalize to other permutation based constructions, including multiple-round Even-Mansour [23] and generalizations of the Masked Even-Mansour construction [62]. See also Sect. 5.1.

### 3.1 Security Model

A distinguisher $\mathsf{D}$ is an algorithm that is given access to an oracle $\mathsf{O}$, written as $\mathsf{D}^{\mathsf{O}}$, and outputs a bit $b \in \{0, 1\}$. Its complexity is measured (solely) by the number of calls to its oracle. As such, we can consider it to be deterministic (as for any probabilistic distinguisher there is a deterministic one with at least the same success probability). The security of a construction $\mathsf{C}$ based on a primitive $\mathsf{P}$ is measured by the advantage of distinguishing $\mathsf{C}^{\mathsf{P}}$ from a random function $\mathsf{R}$ with the same interface and functionality as $\mathsf{C}$:

$$\Delta_{\mathsf{D}}\left(\mathsf{C}^{\mathsf{P}}, \mathsf{P}\ ;\ \mathsf{R}, \mathsf{P}\right) = \left|\mathbf{Pr}\left(1 \leftarrow \mathsf{D}^{\mathsf{C}^{\mathsf{P}}, \mathsf{P}}\right) - \mathbf{Pr}\left(1 \leftarrow \mathsf{D}^{\mathsf{R}, \mathsf{P}}\right)\right|, \tag{4}$$

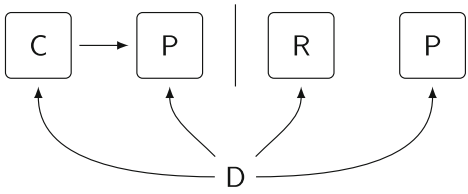


**Fig. 1** The indistinguishability model of (4)

where the randomness is taken over the distributions of $\mathsf{C}$, $\mathsf{P}$, and $\mathsf{R}$. The security distance of (4) is depicted in Fig. 1. In our work, $\mathsf{P}$ is always an invariant permutation from $\mathsf{perm}[\Lambda](n)$, unless explicitly stated otherwise.

Note that if $\Lambda = \{\mathrm{id}\}$, we retrieve indistinguishability in the random permutation model. Any caution that must be paid in the random permutation model must be paid in the invariant permutation model as well. For example, if a construction $\mathsf{C}$ is proven secure in the invariant permutation model, and it is subsequently instantiated using a concrete permutation, security may differ from the promised security statement and one must re-assess security of the scheme *in combination with* the permutation.

We will use the H-coefficient technique [33,92,93]. Consider any two oracles $\mathsf{O}_1$ or $\mathsf{O}_2$ and an information-theoretic deterministic distinguisher $\mathsf{D}$ trying to distinguish both. It can make a finite amount of queries, gathered in a view $\nu$. Denote by $X_1$ (resp. $X_2$) the probability distribution of views in interaction with $\mathsf{O}_1$ (resp. $\mathsf{O}_2$). Denote by $\mathcal{V}$ the set of "attainable views", i.e., views $\nu$ such that $\mathbf{Pr}\left(X_2 = \nu\right) > 0$.

**Lemma 1** (H-coefficient technique) *Consider a partition $\mathcal{V} = \mathcal{V}_{\mathrm{good}} \cup \mathcal{V}_{\mathrm{bad}}$ of the set of views into "good" and "bad" views. Let $\varepsilon \in [0, 1]$ be such that $\frac{\mathbf{Pr}(X_1 = \nu)}{\mathbf{Pr}(X_2 = \nu)} \geq 1 - \varepsilon$ for all $\nu \in \mathcal{V}_{\mathrm{good}}$. Then,*

$$\Delta_{\mathsf{D}}\left(\mathsf{O}_1\ ;\ \mathsf{O}_2\right) \leq \varepsilon + \mathbf{Pr}\left(X_2 \in \mathcal{V}_{\mathrm{bad}}\right)\ . \tag{5}$$

For view $\nu = \{(x_1, y_1), \ldots, (x_q, y_q)\}$ consisting of $q$ input/output tuples, we denote by $\mathsf{O} \vdash \nu$ the event that oracle $\mathsf{O}$ satisfies that $\mathsf{O}(x_i) = y_i$ for all $i = \{1, \ldots, q\}$.

### 3.2 Even-Mansour

We consider the plain single-key Even-Mansour construction $\mathsf{EM} : \{0,1\}^\kappa \times \{0,1\}^n \rightarrow \{0,1\}^n$, where $\kappa \leq n$ [53]:

$$\mathsf{EM}^{\mathsf{P}}(k, m) = \mathsf{P}(m \oplus k0^*) \oplus k0^* \tag{6}$$

(here, the number of appended zeros is, obviously, $n - \kappa$). In the random permutation model, where $\mathsf{P} \xleftarrow{\$} \mathsf{perm}(n)$, the Even-Mansour construction is known to be indistinguishable from a random permutation $\pi \xleftarrow{\$} \mathsf{perm}(n)$ up to $2qp/2^\kappa$, where the distinguisher makes $q$ construction and $p$ primitive queries [2,35,50,53,54,84,89]. We perform an analysis if $\mathsf{EM}$ is instantiated with a random invariant permutation $\mathsf{P}' \xleftarrow{\$} \mathsf{perm}[\Lambda](n)$, for a fixed $\Lambda$.

**Theorem 1** *Consider any fixed $\Lambda$. Let $k \xleftarrow{\$} \{0,1\}^\kappa$, $P' \xleftarrow{\$} perm[\Lambda](n)$, and $\pi \xleftarrow{\$} perm(n)$. Consider any distinguisher $\mathsf{D}$ making at most $q$ construction queries and $p$ primitive queries. We have,*

$$\Delta_D\left(EM_k^{P'}, P' \;;\; \pi, P'\right) \leq \frac{2|\Lambda|qp}{2^\kappa} + \max_{z=\{z_1,\ldots,z_q\}\subseteq\{0,1\}^n} \sum_{\lambda\in\Lambda\setminus\{\mathrm{id}\}} \sum_{z,z'\in z} \frac{2\#\left\{k \mid z \oplus k0^* = \lambda\left(z' \oplus k0^*\right)\right\}}{2^\kappa}.$$

We stress that in the summation, indeed, the case $z = z'$ is included.

Unlike intuition, the proof is more complicated than the ideal permutation based one, and contains various hurdles to be overcome. The proof is given in Sect. 3.3. In Sect. 4, the second part of the bound of Theorem 1 will be considered for specific invariant permutations, including the 512-bit ChaCha permutation [13], the {200, 400, 800, 1600}-bit Keccak permutation [18] with omitted round constants, the 1024-bit permutation of CubeHash [14], and the 384-bit permutation Gimli [16] with omitted round constants.

### 3.3 Proof of Theorem 1

Let $k \xleftarrow{\$} \{0,1\}^\kappa$, $\mathsf{P}' \xleftarrow{\$} \mathsf{perm}[\Lambda](\{0,1\}^n)$, and $\pi \xleftarrow{\$} \mathsf{perm}(n)$. Consider any fixed deterministic distinguisher $\mathsf{D}$ that has access to either $\mathsf{O}_1 = (\mathsf{EM}^{\mathsf{P}'}, \mathsf{P}')$ or $\mathsf{O}_2 = (\pi, \mathsf{P}')$. Gather its $q$ constructions queries in a view $\nu_\mathrm{c} = \{(m_1, c_1), \ldots, (m_q, c_q)\}$ and its $p$ primitive queries in a view $\nu_\mathrm{p} = \{(x_1, y_1), \ldots, (x_p, y_p)\}$. After $\mathsf{D}$'s interaction with its oracle but before it outputs its decision bit $b$, we reveal the random key $k$ (this simplifies the probability analysis in the H-coefficient technique later on). The complete view is denoted

$$\nu = (\nu_\mathrm{c}, \nu_\mathrm{p}, k).$$

We assume that $\mathsf{D}$ never repeats any query, hence $\nu_\mathrm{c}$ and $\nu_\mathrm{p}$ do not contain duplicate queries. We particularly assume that $\mathsf{D}$ never repeats any primitive query transformed over any $\lambda$.

A view $\nu$ is called *bad* if it satisfies one of the following conditions:

$$\begin{aligned}
\mathsf{BAD}_\mathrm{c} &\iff \exists\ (m,c) \in \nu_\mathrm{c}\,,\ 0\lambda \in \Lambda\setminus\{\mathrm{id}\} : \\
&\quad m \oplus k0^* = \lambda\left(m \oplus k0^*\right) \text{ or } c \oplus k0^* = \lambda\left(c \oplus k0^*\right), \\
\mathsf{BAD}_\mathrm{cc} &\iff \exists \text{ distinct } (m,c), (m',c') \in \nu_\mathrm{c}\,,\ \lambda \in \Lambda : \\
&\quad m \oplus k0^* = \lambda\left(m' \oplus k0^*\right) \text{ or } c \oplus k0^* = \lambda\left(c' \oplus k0^*\right), \\
\mathsf{BAD}_\mathrm{cp} &\iff \exists\ (m,c) \in \nu_\mathrm{c}\,,\ (x,y) \in \nu_\mathrm{p}\,,\ \lambda \in \Lambda : \\
&\quad m \oplus k0^* = \lambda\left(x\right) \text{ or } c \oplus k0^* = \lambda\left(y\right).
\end{aligned}$$

In the real world $\mathsf{O}_1$, any construction query $(m, c) \in \nu_\mathrm{c}$ corresponds to a primitive evaluation $(m \oplus k0^*, c \oplus k0^*)$ of $\mathsf{P}'$. Events $\mathsf{BAD}_\mathrm{cc}$ and $\mathsf{BAD}_\mathrm{cp}$ resemble much of what is typically considered in simple Even-Mansour proofs: they capture the event of collisions among construction queries (the former) or between a construction query and a primitive query (the latter). In current proof, however, they cover the event of a collision *even if transformed over any invariance* $\lambda \in \Lambda$.[1] New compared to classical Even-Mansour proofs is $\mathsf{BAD}_\mathrm{c}$. This event considers the case that a single query collides with itself non-trivially under a transformation of a $\lambda \in \Lambda\setminus\{\mathrm{id}\}$. Note that, by attainability of transcripts, we do not have to take into account collisions among primitive queries, not even under a transformation by any $\lambda \in \Lambda$.

[1] In a strict sense, $\mathsf{BAD}_\mathrm{cc}$ happens with probability 0 if $\lambda = \mathrm{id}$, as the distinguisher never repeats queries. We have left it in for completeness.

In below two lemmas, we will derive an upper bound on $\mathbf{Pr}\left(X_2 \in \mathcal{V}_{\text{bad}}\right)$ (in Lemma 2), and show that for any good view, $\mathbf{Pr}\left(X_1 = \nu\right) \geq \mathbf{Pr}\left(X_2 = \nu\right)$ (in Lemma 3). This completes the proof using Lemma 1 on the H-coefficient technique.

**Lemma 2** *We have*

$$\mathbf{Pr}\left(X_2 \in \mathcal{V}_{\text{bad}}\right) \leq \frac{2|\Lambda|qp}{2^\kappa} + \max_{z=\{z_1,\ldots,z_q\}\subseteq\{0,1\}^n} \sum_{\lambda\in\Lambda\setminus\{\text{id}\}} \sum_{z,z'\in z} \frac{2\#\left\{k \mid z \oplus k0^* = \lambda\left(z' \oplus k0^*\right)\right\}}{2^\kappa}.$$

***Proof*** The probability $\mathbf{Pr}\left(X_2 \in \mathcal{V}_{\text{bad}}\right)$ equals the probability that $\mathsf{BAD} := \mathsf{BAD}_{\text{c}} \vee \mathsf{BAD}_{\text{cc}} \vee \mathsf{BAD}_{\text{cp}}$ holds in the ideal world. By basic probability theory,

$$\mathbf{Pr}\left(X_2 \in \mathcal{V}_{\text{bad}}\right) = \mathbf{Pr}\left(\mathsf{BAD}\right) \leq \mathbf{Pr}\left(\mathsf{BAD}_{\text{c}}\right) + \mathbf{Pr}\left(\mathsf{BAD}_{\text{cc}}\right) + \mathbf{Pr}\left(\mathsf{BAD}_{\text{cp}}\right), \tag{7}$$

where we recall that in the ideal world, $k \xleftarrow{\$} \{0,1\}^\kappa$.

**$\mathsf{BAD}_{\text{c}}$.** Consider any query $(m,c) \in \nu_{\text{c}}$ and any $\lambda \in \Lambda\setminus\{\text{id}\}$. As $k \xleftarrow{\$} \{0,1\}^\kappa$, the bad event is set with probability at most

$$\frac{\#\left\{k \mid m \oplus k0^* = \lambda\left(m \oplus k0^*\right)\right\} + \#\left\{k \mid c \oplus k0^* = \lambda\left(c \oplus k0^*\right)\right\}}{2^\kappa}.$$

Taking the sum over all queries and all choices of $\lambda$, the bad event is set with probability at most

$$\begin{aligned}\mathbf{Pr}\left(\mathsf{BAD}_{\text{c}}\right) &\leq \sum_{(m,c)\in\nu_{\text{c}}} \sum_{\lambda\in\Lambda\setminus\{\text{id}\}} \frac{\#\left\{k \mid m \oplus k0^* = \lambda\left(m \oplus k0^*\right)\right\}}{2^\kappa} \\ &\quad + \sum_{(m,c)\in\nu_{\text{c}}} \sum_{\lambda\in\Lambda\setminus\{\text{id}\}} \frac{\#\left\{k \mid c \oplus k0^* = \lambda\left(c \oplus k0^*\right)\right\}}{2^\kappa} \\ &\leq \max_{z=\{z_1,\ldots,z_q\}\subseteq\{0,1\}^n} \sum_{\lambda\in\Lambda\setminus\{\text{id}\}} \sum_{z\in z} \frac{2\#\left\{k \mid z \oplus k0^* = \lambda\left(z \oplus k0^*\right)\right\}}{2^\kappa}.\end{aligned}$$

Note that the second bound is pretty tight. For each of the construction queries, the distinguisher either chooses $m$ and receives randomly generated $c$ or it chooses $c$ and receives randomly generated $m$. For bounding the bad event, we simply give it the power to choose *both m and c for every query*. As the distinguisher achieves highest advantage if it chooses all $m$'s at its own discretion, the sum over the $\#\left\{k \mid m \oplus k0^* = \lambda\left(m \oplus k0^*\right)\right\}$'s will dominate and the sum over the $\#\left\{k \mid c \oplus k0^* = \lambda\left(c \oplus k0^*\right)\right\}$'s will disappear. Our bound is at most a factor 2 larger.

**$\mathsf{BAD}_{\text{cc}}$.** Consider any distinct queries $(m,c), (m',c') \in \nu_{\text{c}}$ and any $\lambda \in \Lambda$. If $\lambda = \text{id}$ the event happens with probability 0; we consider $\lambda \in \Lambda\setminus\{\text{id}\}$. As $k \xleftarrow{\$} \{0,1\}^\kappa$, the bad event is set with probability at most

$$\frac{\#\left\{k \mid m \oplus k0^* = \lambda\left(m' \oplus k0^*\right)\right\} + \#\left\{k \mid c \oplus k0^* = \lambda\left(c' \oplus k0^*\right)\right\}}{2^\kappa}.$$

Taking the sum over all distinct queries and all choices of $\lambda$, the bad event is set with probability at most

$$\begin{aligned}\mathbf{Pr}\left(\mathsf{BAD}_{\mathrm{cc}}\right) \leq & \sum_{(m,c)\neq(m',c')\in\nu_{\mathrm{c}}} \sum_{\lambda\in\Lambda\setminus\{\mathrm{id}\}} \frac{\#\left\{k \mid m\oplus k0^* = \lambda\left(m'\oplus k0^*\right)\right\}}{2^\kappa} \\ & + \sum_{(m,c)\neq(m',c')\in\nu_{\mathrm{c}}} \sum_{\lambda\in\Lambda\setminus\{\mathrm{id}\}} \frac{\#\left\{k \mid c\oplus k0^* = \lambda\left(c'\oplus k0^*\right)\right\}}{2^\kappa} \\ \leq & \max_{z=\{z_1,\ldots,z_q\}\subseteq\{0,1\}^n} \sum_{\lambda\in\Lambda\setminus\{\mathrm{id}\}} \sum_{z\neq z'\in z} \frac{2\#\left\{k \mid z\oplus k0^* = \lambda\left(z'\oplus k0^*\right)\right\}}{2^\kappa}\,.\end{aligned}$$

**$\mathsf{BAD}_{\mathrm{cp}}$.** Consider any queries $(m, c) \in \nu_{\mathrm{c}}$, $(x, y) \in \nu_{\mathrm{p}}$, and any $\lambda \in \Lambda$. The bad event is set if

$$k0^* \in \{m \oplus \lambda(x)\,,\; c \oplus \lambda(y)\}\,. \tag{8}$$

As $k \xleftarrow{\$} \{0,1\}^\kappa$, Eq. (8) holds with probability at most $2/2^\kappa$. Taking the sum over all queries and all choices of $\lambda$, the bad event is set with probability at most

$$\mathbf{Pr}\left(\mathsf{BAD}_{\mathrm{cp}}\right) \leq \frac{2|\Lambda|qp}{2^\kappa}\,.$$

**Conclusion.** Adding the separate bounds in accordance with (7) gives

$$\begin{aligned}\mathbf{Pr}\left(\mathsf{BAD}\right) \leq & \frac{2|\Lambda|qp}{2^\kappa} \\ & + \max_{z=\{z_1,\ldots,z_q\}\subseteq\{0,1\}^n} \sum_{\lambda\in\Lambda\setminus\{\mathrm{id}\}} \sum_{z,z'\in z} \frac{2\#\left\{k \mid z\oplus k0^* = \lambda\left(z'\oplus k0^*\right)\right\}}{2^\kappa}\,,\end{aligned}$$

where in the summation the values $z, z'$ may be identical. □

**Lemma 3** *For any good view $\nu \in \mathcal{V}_{\mathrm{good}}$, $\mathbf{Pr}\left(X_1 = \nu\right) \geq \mathbf{Pr}\left(X_2 = \nu\right)$.*

***Proof*** In the real world $\mathsf{O}_1$, every tuple in $(\nu_{\mathrm{c}}, \nu_{\mathrm{p}})$ defines *exactly* one input-output tuple of $\mathsf{P}' \xleftarrow{\$} \mathsf{perm}[\Lambda](n)$. Therefore, we derive

$$\begin{aligned}\mathbf{Pr}\left(X_1 = \nu\right) = & \mathbf{Pr}\left(\mathsf{P}' \xleftarrow{\$} \mathsf{perm}[\Lambda](n)\;;\; \mathsf{EM}_k^{\mathsf{P}'} \vdash \nu_{\mathrm{c}} \wedge \mathsf{P}' \vdash \nu_{\mathrm{p}} \mid k\right) \\ & \cdot \mathbf{Pr}\left(k' \xleftarrow{\$} \{0,1\}^\kappa \;:\; k' = k\right) \\ = & \frac{1}{2^\kappa} \cdot \mathbf{Pr}\left(\mathsf{P}' \xleftarrow{\$} \mathsf{perm}[\Lambda](n)\;;\; \mathsf{P}' \vdash \bar{\nu}_{\mathrm{c}} \cup \nu_{\mathrm{p}}\right)\,,\end{aligned}$$

where $\bar{\nu}_{\mathrm{c}} = \{(m \oplus k0^*, c \oplus k0^*) \mid (m, c) \in \nu_{\mathrm{c}}\}$, and where $\bar{\nu}_{\mathrm{c}} \cup \nu_{\mathrm{p}}$ contains a total amount of $q + p$ tuples among which there are no input or output collisions even when the tuples are transformed over any $\lambda \in \Lambda$.

In the ideal world $\mathsf{O}_2$, we have

$$\begin{aligned}\mathbf{Pr}\left(X_2 = \nu\right) = & \mathbf{Pr}\left(\pi \xleftarrow{\$} \mathsf{perm}(n)\;;\; \pi \vdash \nu_{\mathrm{c}}\right) \\ & \cdot \mathbf{Pr}\left(\mathsf{P}' \xleftarrow{\$} \mathsf{perm}[\Lambda](n)\;;\; \mathsf{P}' \vdash \nu_{\mathrm{p}}\right) \\ & \cdot \mathbf{Pr}\left(k' \xleftarrow{\$} \{0,1\}^\kappa \;:\; k' = k\right) \\ = & \frac{1}{(2^n)_q \cdot 2^\kappa} \cdot \mathbf{Pr}\left(\mathsf{P}' \xleftarrow{\$} \mathsf{perm}[\Lambda](n)\;;\; \mathsf{P}' \vdash \nu_{\mathrm{p}}\right)\,.\end{aligned}$$

where $\nu_{\mathrm{p}}$ represents a list of $p$ tuples for which there are no input or outputs collisions even when transformed over any $\lambda \in \Lambda$.

We obtain that

$$\begin{aligned}
\frac{\mathbf{Pr}\left(X_1 = \nu\right)}{\mathbf{Pr}\left(X_2 = \nu\right)} &= \frac{\mathbf{Pr}\left(\mathsf{P}' \xleftarrow{\$} \mathsf{perm}[\Lambda](n)\ ;\ \mathsf{P}' \vdash \bar{\nu}_{\mathrm{c}} \cup \nu_{\mathrm{p}}\right)}{\frac{1}{(2^n)_q} \cdot \mathbf{Pr}\left(\mathsf{P}' \xleftarrow{\$} \mathsf{perm}[\Lambda](n)\ ;\ \mathsf{P}' \vdash \nu_{\mathrm{p}}\right)} \\
&= \frac{\mathbf{Pr}\left(\mathsf{P}' \xleftarrow{\$} \mathsf{perm}[\Lambda](n)\ ;\ \mathsf{P}' \vdash \bar{\nu}_{\mathrm{c}} \cup \nu_{\mathrm{p}} \;\middle|\; \mathsf{P}' \vdash \nu_{\mathrm{p}}\right)}{\frac{1}{(2^n)_q}} \\
&\geq \frac{\mathbf{Pr}\left(\mathsf{P} \xleftarrow{\$} \mathsf{perm}(n)\ ;\ \mathsf{P} \vdash \bar{\nu}_{\mathrm{c}} \cup \nu_{\mathrm{p}} \;\middle|\; \mathsf{P} \vdash \nu_{\mathrm{p}}\right)}{\frac{1}{(2^n)_q}} \\
&= \frac{\frac{1}{(2^n-p)_q}}{\frac{1}{(2^n)_q}} \geq 1\,,
\end{aligned}$$

where for the approximation we use that the list of tuples in $\bar{\nu}_{\mathrm{c}} \cup \nu_{\mathrm{p}}$ do not collide even when transformed over any $\lambda \in \Lambda$. □

### 3.4 Keyed Sponge

The sponge hash function construction [17] consists of a sequential application of a permutation on a state, where the permutation evaluations are interleaved with absorption of data into the outer part of the state or extraction of data from that part of the state. Security is determined by the *capacity*, the size of the state that is not involved in the absorption or extraction. The construction has led to various MAC designs based on a permutation [2,19,31,41,59,85,87,89,90], that also consist of a sequential application of a permutation, but with the initial state masked using a key. For the sake of exemplification, we will consider the full-state keyed sponge of Mennink et al.[85]. We remark that follow-up work considered a more general scheme with a more accurate bound [41], but the scheme of the former is easier to describe concisely. We will drop the adjective "full-state" for simplicity.

The keyed sponge (KS) is a message authentication code. On input of a message $M \in \{0,1\}^*$ and a parameter $\ell \in \mathbb{N}$, it outputs a tag $T \in \{0,1\}^\ell$. For identical messages but different parameters $\ell, \ell'$, the shortest response is a prefix of the longer one. Internally, it uses a permutation on an $n$-bit state split in an inner part of $c$ bits and an outer part of $r$ bits, in such a way that $c + r = n$. We furthermore assume that $\kappa = c$ (but see the proof of Theorem 2). The function is depicted in Fig. 2.

Security of the keyed sponge is measured by its "PRF security", a distance to a variable output length random oracle $\mathsf{R}$ [10], i.e., (4) with $\mathsf{R}$ as random function:

$$\Delta_{\mathsf{D}}\left(\mathsf{KS}_k^{\mathsf{P}}, \mathsf{P}\ ;\ \mathsf{R}, \mathsf{P}\right)\,.$$

Mennink et al.[85] proved the following security bound for KS.

**Theorem 2** *(**Mennink et al.** [85]) Let $k \xleftarrow{\$} \{0,1\}^\kappa$, $P \xleftarrow{\$} \mathsf{perm}(n)$, and $\mathsf{R}$ be a random oracle as described above. For any distinguisher $\mathsf{D}$ making at most $\sigma$ construction query blocks and*

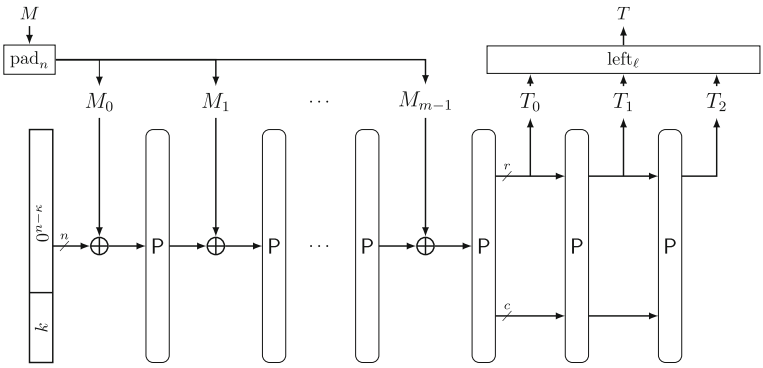


**Fig. 2** The keyed sponge (KS) of [85]

*p primitive queries,*

$$\Delta_D\left(KS_k^P, P\ ;\ R, P\right) \leq \frac{2\sigma^2}{2^n} + \frac{2\sigma^2}{2^c} + \frac{2\sigma\, p}{2^\kappa}\,. \tag{9}$$

***Proof*** *(sketch)* Consider any fixed deterministic distinguisher $\mathsf{D}$ that makes a total amount of $\sigma$ construction query blocks and $p$ primitive queries. Consider $\mathsf{EM}_k^\mathsf{P}$ of (6). By xoring the key $k$ twice in-between each two consecutive evaluations of $\mathsf{P}$,[2] we can observe that

$$\mathsf{KS}_k^\mathsf{P} = \mathsf{KS}_0^{\mathsf{EM}_k^\mathsf{P}}\,,$$

where the subscripts denote the key input. Therefore,

$$\begin{aligned}\Delta_\mathsf{D}\left(\mathsf{KS}_k^\mathsf{P}, \mathsf{P}\ ;\ \mathsf{R}, \mathsf{P}\right) &= \Delta_\mathsf{D}\left(\mathsf{KS}_0^{\mathsf{EM}_k^\mathsf{P}}, \mathsf{P}\ ;\ \mathsf{R}, \mathsf{P}\right)\\ &\leq \Delta_\mathsf{D}\left(\mathsf{KS}_0^\pi, \mathsf{P}\ ;\ \mathsf{R}, \mathsf{P}\right) + \Delta_{\mathsf{D}'}\left(\mathsf{EM}_k^\mathsf{P}, \mathsf{P}\ ;\ \pi, \mathsf{P}\right)\\ &= \Delta_\mathsf{D}\left(\mathsf{KS}_0^\pi\ ;\ \mathsf{R}\right) + \Delta_{\mathsf{D}'}\left(\mathsf{EM}_k^\mathsf{P}, \mathsf{P}\ ;\ \pi, \mathsf{P}\right)\,,\end{aligned} \tag{10}$$

where $\pi \xleftarrow{\$} \mathsf{perm}(n)$ is a random permutation, and $\mathsf{D}'$ is some distinguisher that makes at most $\sigma$ construction queries and $p$ primitive queries. For the first term of (10), Mennink et al. proved that (simplified)

$$\Delta_\mathsf{D}\left(\mathsf{KS}_0^\pi\ ;\ \mathsf{R}\right) \leq \frac{2\sigma^2}{2^n} + \frac{2\sigma^2}{2^c}\,. \tag{11}$$

The second term of (10) is at most $2\sigma\, p/2^\kappa$ (see the introductory text of Sect. 3.2).[3] □

We demonstrate how a security bound for $\mathsf{KS}$ based on an invariant permutation $\mathsf{P}' \xleftarrow{\$} \mathsf{perm}[\Lambda](n)$ is easily derived.

[2] If $\kappa \leq c$, a better bound can be derived if we would xor a $c$-bit dummy value $l$ instead, cf. [41].

[3] A smarter bound can be derived here, noting that in the $\mathsf{KS}$ construction the distinguisher has no full freedom over the entire input to $\mathsf{EM}_k^\mathsf{P}$ [41].

**Theorem 3** *Consider any fixed $\Lambda$. Let $k \xleftarrow{\$} \{0,1\}^\kappa$, $P' \xleftarrow{\$} \mathsf{perm}[\Lambda](n)$, and $R$ be a random oracle as described above. For any distinguisher $D$ making at most $\sigma$ construction query blocks and $p$ primitive queries,*

$$\Delta_D\left(KS_k^{P'}, P' \; ; \; R, P'\right) \leq \frac{2\sigma^2}{2^n} + \frac{2\sigma^2}{2^c} + \mathtt{BOUND}_{Thm.1}(\sigma, p)\,, \tag{12}$$

*where $\mathtt{BOUND}_{Thm.1}(\sigma, p)$ is the bound of Theorem 1 for distinguisher's parameters $\sigma$ and $p$.*

***Proof*** Consider any fixed deterministic distinguisher $\mathsf{D}$ that makes a total amount of $\sigma$ construction query blocks and $p$ primitive queries. Consider $\mathsf{EM}_k^{\mathsf{P}'}$ of (6). By xoring the key $k$ twice in-between each two consecutive evaluations of $\mathsf{P}'$, we can observe that

$$\mathsf{KS}_k^{\mathsf{P}'} = \mathsf{KS}_0^{\mathsf{EM}_k^{\mathsf{P}'}}\,,$$

where the subscripts denote the key input. Therefore,

$$\begin{aligned}
\Delta_{\mathsf{D}}\left(\mathsf{KS}_k^{\mathsf{P}'}, \mathsf{P}' \; ; \; \mathsf{R}, \mathsf{P}'\right) &= \Delta_{\mathsf{D}}\left(\mathsf{KS}_0^{\mathsf{EM}_k^{\mathsf{P}'}}, \mathsf{P}' \; ; \; \mathsf{R}, \mathsf{P}'\right) \\
&\leq \Delta_{\mathsf{D}}\left(\mathsf{KS}_0^{\pi}, \mathsf{P}' \; ; \; \mathsf{R}, \mathsf{P}'\right) + \Delta_{\mathsf{D}'}\left(\mathsf{EM}_k^{\mathsf{P}'}, \mathsf{P}' \; ; \; \pi, \mathsf{P}'\right) \\
&= \Delta_{\mathsf{D}}\left(\mathsf{KS}_0^{\pi} \; ; \; \mathsf{R}\right) + \Delta_{\mathsf{D}'}\left(\mathsf{EM}_k^{\mathsf{P}'}, \mathsf{P}' \; ; \; \pi, \mathsf{P}'\right)\,, \qquad (13)
\end{aligned}$$

where $\pi \xleftarrow{\$} \mathsf{perm}(n)$ is a random permutation, and $\mathsf{D}'$ is some distinguisher that makes at most $\sigma$ construction queries and $p$ primitive queries. The first term is bounded as (11) as before, and for the second term of (13) we rely on Theorem 1. □

## 4 Understanding Invariance Loss

The bound of Theorem 1 precisely demonstrates what aspect of the invariant permutation must be exploited in order to break the Even-Mansour construction: further analysis boils down to upper bounding the sizes of the sets $\#\{k \mid \ldots\}$. Focus on Even-Mansour with key size $\kappa = n$, and consider the task of upper bounding

$$\max_{\mathcal{z}=\{z_1,\ldots,z_q\}\subseteq\{0,1\}^n} \sum_{z,z'\in\mathcal{z}} \#\left\{k \mid z \oplus k = \lambda\left(z' \oplus k\right)\right\} \tag{14}$$

for any invariance $\lambda \neq \mathrm{id}$. In the specific case that $\lambda$ satisfies $\lambda(x \oplus y) = \lambda(x) \oplus \lambda(y)$, which always holds in our work as we consider linear invariances, we will resort to a rewritten version of (14) that will be easier to work with:

$$\max_{\mathcal{z}=\{z_1,\ldots,z_q\}\subseteq\{0,1\}^n} \sum_{z,z'\in\mathcal{z}} \#\left\{k \mid k \oplus \lambda(k) = z \oplus \lambda(z')\right\}\,. \tag{15}$$

Bounding this quantity seems to relate to upper bounding the set of weak keys $k$, but the goal is not quite the same. Rather, the current issue boils down to quantifying the image of $k \mapsto k \oplus \lambda(k)$ and the ability of an adversary in selecting $\mathcal{z} = \{z_1, \ldots, z_q\}$ so as to maximize the number of "hits": choices $z, z' \in \mathcal{z}$ such that $z \oplus \lambda(z')$ has many preimages over $(\mathrm{id} \oplus \lambda)^{-1}$.

We will consider the problem of bounding (14) for the case of additive invariances in Sect. 4.1, and derive a practical attack on Even-Mansour based on Salsa20. Then, in Sect. 4.2,

we will consider linear invariances (that satisfy $\lambda(x \oplus y) = \lambda(x) \oplus \lambda(y)$), and restrict our focus to bounding (15). We will investigate how to maximize this set, and subsequently show how the analysis applies to the 512-bit ChaCha permutation [13] (distinguishing attack in $2^{128}$ construction queries), the $n$-bit Keccak permutation without round constants [18] (distinguishing attack in $2^{n/4}$ construction queries), CubeHash [14] (distinguishing attack in $2^{256}$ construction queries), and Gimli without round constants [16] (distinguishing attack in $2^{96}$ construction queries).

### 4.1 Additive Invariances

As mentioned in Sect. 2.3, the Salsa20 permutation $\mathsf{P}'$ is invariant under addition of the constant $\Delta = 2^{31}$ to each of the 16 words. Call this invariance $\lambda^{\text{add}}$:

$$\lambda^{\text{add}} : k \mapsto k \oplus \{\Delta\}^{16} . \tag{16}$$

It is easy to verify that

$$\#\left\{k \mid z \oplus k = \lambda^{\text{add}}\left(z' \oplus k\right)\right\} = \begin{cases} 2^{\kappa} \text{ for } z' = z \oplus \{\Delta\}^{16} , \\ 0 \quad \text{otherwise} . \end{cases}$$

Based on this observation, we can describe a simple distinguishing attack on Even-Mansour-Salsa20 that succeeds in constant time.

1. Fix any $m, m'$ that satisfy $m \oplus m' = \{\Delta\}^{16}$. Query $m$ and $m'$ to the construction oracle to obtain the corresponding $c$ and $c'$;
2. For the real world $\mathsf{EM}^{\mathsf{P}'}$, we necessarily have

$$\begin{aligned} c \oplus k = \mathsf{P}'(m \oplus k) &= \mathsf{P}' \circ \lambda^{\text{add}}(m' \oplus k) \\ &= \lambda^{\text{add}} \circ \mathsf{P}'(m' \oplus k) = \lambda^{\text{add}}(c' \oplus k) = c' \oplus k \oplus \{\Delta\}^{16} . \end{aligned}$$

3. If, indeed, $c \oplus c' = \{\Delta\}^{16}$, $\mathsf{D}$ is with high probability interacting with $\mathsf{EM}_k^{\mathsf{P}'}$; otherwise, it is interacting with $\pi$.

The attentive reader will recognize this as an elementary differential distinguisher.

Concretely, the attack means the following for the second term of Theorem 1. Assume that $\Lambda = \{\lambda^{\text{add}}\}$. Above attack shows that there exists an adversary that makes $q = 2$ construction queries, and that can in this way select 2 values $\mathcal{z} = \{z_1, z_2\}$ such that

$$\sum_{z,z' \in \mathcal{z}} \frac{2\#\left\{k \mid z \oplus k0^* = \lambda^{\text{add}}\left(z' \oplus k0^*\right)\right\}}{2^{\kappa}} \approx 1 .$$

In other words, the bound of Theorem 1 becomes void, and as $q = 2$, this can be considered a practical attack.

### 4.2 Linear Invariances

For linear $\lambda$, quantifying the image of $k \mapsto k \oplus \lambda(k)$ is easy: if the rank of $\text{id} \oplus \lambda$ is $r$, its image has size $2^r$. Consequently, the lower the rank of $\text{id} \oplus \lambda$, the higher (15) can get. On the other hand, if $\lambda$ is a linear orthomorphism[4] there is no image size reduction.

[4] An orthomorphism is a bijection $\phi$ such that $\phi - \text{id}$ is also a bijection.

The second challenge is to choose $\mathcal{Z}$ in such a way as to hit as many elements of the image of $\mathrm{id} \oplus \lambda$ as possible. One way this can be accomplished is by random sampling $z \in \mathcal{Z}$ from linear combinations of a basis of $\mathrm{id} \oplus \lambda$. Query both $z$ and $\lambda^{-1}(z)$ if they are distinct. Given $q$ (not necessarily distinct) queries we have up to $\binom{q}{2}$ values, and the expected number of queries until every element of $\mathrm{id} \oplus \lambda$ is covered is a variant of the coupon collector's problem [57], and is given by $\left(\sqrt{8 \cdot 2^r \cdot H_{2^r} + 1} + 1\right)/2$, where $H_n$ is the $n$th harmonic number. We can approximate it by $\sqrt{8 \cdot (\log(2^r) + 0.5772156649\ldots) \cdot 2^r}$ or, more loosely, $\sqrt{8r} \cdot 2^{r/2}$.

In some cases a basis of $\mathrm{id} \oplus \lambda$ can be generated quite easily. For example, in the invariances covered below, it suffices to find the smallest set $\mathcal{Z} \subset \mathbb{F}_2^r$ such that $\{z \oplus z' : z, z' \in \mathcal{Z}\} = \mathbb{F}_2^r$. This is equivalent to a well-studied problem in coding theory: finding a minimal length code $[n, n-r]$ of co-dimension $r$ and covering radius 2 [36]. Explicit constructions of these codes are given in, e.g., [26,47,58]. Such sets are also known, in Galois geometry, as "1-saturating sets" [48,61], "saturated sets" [47,101], "dense sets" [24], or "spanning sets" [26]. Clark and Pedersen [34] give a particularly simple solution, described in Algorithm 1, for $r \geq 4$, of length $2 \cdot 2^{r/2} - 2$ for even $r$, and $3 \cdot 2^{\lfloor r/2 \rfloor} - 2$ for odd $r$.

**Algorithm 1** Clark-Pedersen saturating set

**Require:** $\mathbb{F}_2^n$, $n \geq 4$
**Ensure:** $|S| = 2^{n/2+1} - 2$
  $S \leftarrow \{0\}$ ▷ Vectors in $\mathbb{F}_2^n$ are identified by integers
  **for** $x \leftarrow 1$ to $2^{n/2} - 1$ **do**
    $S \leftarrow S \cup \{2 \cdot x + (\mathrm{hw}(x) + 1 \bmod 2)\}$ ▷ Ensure odd parity
  **for** $y \leftarrow 1$ to $2^{n/2-1+n \bmod 2} - 1$ **do**
    $w \leftarrow \mathrm{hw}(y)$ ▷ $\mathrm{hw}(\cdot)$ denotes Hamming weight
    **if** $w = 1$ **then**
      $S \leftarrow S \cup \{y \cdot 2^{n/2+1}\}$
      $S \leftarrow S \cup \{y \cdot 2^{n/2+1} + 1\}$
    **else**
      $S \leftarrow S \cup \{y \cdot 2^{n/2+1} + 2^{w-1}\}$
      $S \leftarrow S \cup \{y \cdot 2^{n/2+1} + 2^{w-1} + 1\}$
  **return** $S$

The above reasoning immediately translates to a distinguisher for the single-key Even-Mansour block cipher when used with a permutation carrying a linear invariance $\lambda$:

1. Query the oracle with $\mathcal{Z}$ such that as much—or all—possible values of $k \oplus \lambda(k)$ are covered by $z \oplus \lambda(z')$, with $z, z' \in \mathcal{Z}$. This can be accomplished by random sampling or by an appropriate covering code as described above;
2. If there are two queries such that $k \oplus \lambda(k) = m_i \oplus \lambda(m_j)$, then we also have

$$\begin{aligned} c_i \oplus k &= \mathsf{P}'(m_i \oplus k) \\ &= \mathsf{P}' \circ \lambda(m_j \oplus k) \\ &= \lambda \circ \mathsf{P}'(m_j \oplus k) \\ &= c_j \oplus k\,, \end{aligned}$$

   whence $m_i \oplus \lambda(m_j) = c_i \oplus \lambda(c_j)$ and $m_i \oplus c_i = \lambda(m_j \oplus c_j)$;
3. Such a pair will be found in approximately $2^{r/2}$ queries, where $r$ is the rank of $\mathrm{id} \oplus \lambda$. If $r$ is significantly smaller than $n$, this leads to a distinguisher that also yields the value of $(\mathrm{id} \oplus \lambda)(k)$.

In terms of Theorem 1, above attack rather aims at recovering $(\mathrm{id} \oplus \lambda)(k)$, which can possibly be done faster than recovering $k$ as the $\mathrm{id} \oplus \lambda$ has rank $r \leq \kappa$. Nevertheless, every image of $\mathrm{id} \oplus \lambda$ has $2^{\kappa - r}$ preimages, and after $q \approx 2^{r/2}$ queries, above attack maximizes

$$\sum_{z,z' \in \mathcal{Z}} \#\left\{k \mid k \oplus \lambda(k) = z \oplus \lambda(z')\right\}$$

to approximately $2^{\kappa}$. For the second term of Theorem 1, this implies that

$$\sum_{z,z' \in \mathcal{Z}} \frac{2\#\left\{k \mid z \oplus k0^* = \lambda\left(z' \oplus k0^*\right)\right\}}{2^{\kappa}} \approx 1\,.$$

As in Sect. 4.1, the bound of Theorem 1 becomes void. In this case, however, more construction queries are needed, namely $q \approx 2^{r/2}$. It depends on the construction and on the invariance whether $r$ is low enough for this attack to be considered practical.

#### 4.2.1 ChaCha

The invariance $\lambda : \{0,1\}^{512} \to \{0,1\}^{512}$ present in ChaCha [13] for which $\mathrm{id} \oplus \lambda$ has the lowest rank is the rotation of each 128-bit column by 2 positions (see also Sect. 2.3.1). The rank of $\mathrm{id} \oplus \lambda$ is thus 256, and the image of $k \oplus \lambda(k)$ can be given by two copies of $\mathbb{F}_2^{256}$ side by side. Using Algorithm 1, the attack succeeds in at most $2^{129} - 2$ queries.

#### 4.2.2 Keccak

We may consider the $n$-bit Keccak permutation [18], with $n \in \{200, 400, 800, 1600\}$, modified to omit the constant addition $\iota$. In this case, we have 25 parallel operations that swap one half of each word with the other half. The rank of this operation is $n/2$ and, as above, the attack would succeed in approximately $2^{n/4}$ queries.

#### 4.2.3 CubeHash

The 1024-bit state of CubeHash [14] is split into 32 words of 32 bits. The state can then be decomposed into 32 independent 32-bit vectors, corresponding to each bit of each word, and an invariance reminiscent of those in Sect. 2.4 is applied. In this case, we can consider the invariance corresponding to "swap each adjacent word", which has rank 512. The attack succeeds in approximately $2^{256}$ queries.

#### 4.2.4 Gimli

For Gimli [16], assuming omission of the round constants, one can decompose its 384-bit state into 96 independent 4-bit vectors, corresponding to each bit of each column, after which a permutation matrix $\pi$ is applied. From Sect. 2.6, we can conclude that this permutation $\pi$ is either of

$$\begin{pmatrix} 0&1&0&0\\ 1&0&0&0\\ 0&0&0&1\\ 0&0&1&0 \end{pmatrix}, \begin{pmatrix} 0&0&1&0\\ 0&0&0&1\\ 1&0&0&0\\ 0&1&0&0 \end{pmatrix}, \begin{pmatrix} 0&0&0&1\\ 0&0&1&0\\ 0&1&0&0\\ 1&0&0&0 \end{pmatrix}. \tag{17}$$

For each of these, the rank of $\mathrm{id} \oplus \pi$ equals 2, meaning that the rank of $\mathrm{id} \oplus \lambda$ is 192. The attack succeeds in approximately $2^{96}$ queries.

## 5 Conclusion

### 5.1 Salvaging Use of Even-Mansour

One way to mitigate the attacks on Even-Mansour of Sect. 4 is to consider a tweakable variant of Even-Mansour, such as a generalization of the Masked Even-Mansour construction [62]. This construction uses a tweak to mask the key before it is added to the message. In detail, it does not simply add $k0^*$ to the message (as in (6)), but rather it adds

$$\Phi(u, s)\,,$$

where $s$ is a tweak and $u$ is a subkey derived from the key using the permutation $\mathsf{P}$. This masking function can be assumed to eliminate any structure in

$$\Phi(u, s) \oplus \lambda \circ \Phi(u, s')\,.$$

We elaborate on this generalization of the Masked Even-Mansour construction, and prove security in the invariant permutation model, in Appendix A.1. In Appendix A.2, we subsequently explain how the analysis extends to security of the Offset Public Permutation (OPP) authenticated encryption scheme by Granger et al.[62]. For these constructions, it particularly becomes clear that *exploiting* invariances becomes significantly harder, if not impossible. On the downside, the security analysis becomes much more technically involved. A less (conceptually) costly way of salvaging Even-Mansour is to add the key using, say, $\mathbb{F}_{2^n}$ multiplication instead of xor, after which the typical invariance is no longer linear.

Another way to mitigate the attacks on Even-Mansour is to consider 2-round Even-Mansour [23,32]:

$$2\mathsf{EM}^{\mathsf{P}}(k, m) = \mathsf{P}(\mathsf{P}(m \oplus k0^*) \oplus \iota(k0^*)) \oplus k0^*\,, \tag{18}$$

where $\iota$ is an orthomorphism (i.e., both $\iota$ and $\iota \oplus \mathrm{id}$ are a permutation). An attacker that has oracle access to $2\mathsf{EM}^{\mathsf{P}'}$ for an invariant permutation $\mathsf{P}' \in \mathsf{perm}[\Lambda](n)$ is unable to observe the input-output pattern of $\mathsf{P}'$ and cannot mount attacks comparable to those of Sect. 4.

The attack could be slowed down as well by preventing the adversary to have full control over the input of the construction. For example, in the keyed sponge of Sect. 3.4, the data is absorbed over the entire state, but the original keyed sponge [2,19] maintains an inner part on which no data is absorbed. The adversary has to run the algorithm and wait for the inner part to fall into a symmetric state.

### 5.2 Multi-Key Security

In the random permutation model, the Even-Mansour construction is known to admit only a small loss: it is proven to be secure up to $(q^2+2qp)/2^\kappa$ [88]. The loss is caused by the possible presence of cross-collisions between construction queries for different keys. A similar loss occurs in analyses in the random invariant permutation model: the adversary succeeds if two evaluations for different keys, say $(m, c)$ for key $k$ and $(m', c')$ for key $k'$, collide under a transformation of any invariance $\lambda$:

$$m \oplus k0^* = \lambda\big(m' \oplus k'0^*\big) \text{ or } c \oplus k0^* = \lambda\big(c' \oplus k'0^*\big)\,.$$

The extra loss is marginal as for each query $m$ (or, in inverse direction, $c$) that the distinguisher makes, it has to tie itself to a specific key instance. As noted by Lee et al.[81], one may consider

this extra choice as a tweak, transforming the scheme into a specific form of tweaked Even-Mansour [35].

### 5.3 Extension to Different Primitives and Invariances

The treatment of Sect. 2 can be generalized to cover invariances that only apply to a subset $S \subset \{0,1\}^n$ of potential inputs. Or perhaps this subset is not known, but only the probability of randomly hitting it. Midori [6] is an interesting example of that kind. The high number of fixed points $(3, 7, 8, 9)$ of its S-box, combined with a simple binary diffusion matrix and poor choice of key schedule constants, results in a weak set of $2^{32}$ keys $\{0,1\}^{16\times 2}$ for which the cipher is entirely linear—and invariant—for inputs of the form $\{8,9\}^{16}$ [64].

Furthermore, Sect. 2 generalizes to arbitrary (not necessarily bijective) functions $\{0,1\}^n \to \{0,1\}^n$ verbatim. The generalization is particularly meaningful for block ciphers $\mathsf{E} : \{0,1\}^\kappa \times \{0,1\}^n \to \{0,1\}^n$, recalling the invariance in AES of (2) that has been exploited in cryptanalytic attacks of AES based ciphers [69,71,72,97], as explained in Sect. 1.1.

For example, one may consider NOEKEON without round constants [43], for which

$$\lambda \circ \mathsf{NOEKEON}(k, m) = \mathsf{NOEKEON}(\lambda(k), \lambda(m)) \,. \tag{19}$$

We stress that this condition does not apply to NOEKEON due to the presence of round constants.

Related to invariance subspace attacks are weak key attacks on block ciphers. An example of such a similarity is the complementation property of DES [46]:

$$\mathsf{E}(k, x) \oplus \Delta = \mathsf{E}(k \oplus \Delta, x \oplus \Delta) \,,$$

for $\Delta = 11\ldots11$.



## A Extension to Masked Even-Mansour

We will demonstrate how the results of Sect. 3 extend to more involved modes. We will do so by considering the Masked Even-Mansour construction in Appendix A.1, and the Offset Public Permutation authenticated encryption scheme in Appendix A.2.

### A.1 Generalized Masked Even-Mansour

We will consider an abstraction, formally a generalization, of the Masked Even-Mansour construction of Granger et al.[62]. The tweak space is now composed of $\{0,1\}^\tau \times \mathcal{T}$ for some space $\mathcal{T}$. Formally, we consider the tweakable block cipher $\mathsf{genMEM} : \{0,1\}^\kappa \times (\{0,1\}^\tau \times \mathcal{T}) \times \{0,1\}^n \to \{0,1\}^n$ based on a permutation $\mathsf{P} : \{0,1\}^n \to \{0,1\}^n$, where $\kappa + \tau \leq n$:

$$\mathsf{genMEM}^{\mathsf{P}}(k,(t,s),m) = \mathsf{P}(m \oplus \Phi(\mathsf{P}(kt0^*),s)) \oplus \Phi(\mathsf{P}(kt0^*),s)\,, \tag{20}$$

for some masking function $\Phi : \{0,1\}^n \times \mathcal{T} \to \{0,1\}^n$. We assume that $\Phi(u,s) = u$ for any $u \in \{0,1\}^n$ if and only if $s = 0$.

The original construction MEM of Granger et al.[62] can be retained by fixing $v \in \mathbb{N}$, fixing $v$ LFSRs $\varphi_0, \ldots, \varphi_{v-1}$, setting $\mathcal{T} = \mathbb{N}^v$, and defining

$$\Phi(u,\boldsymbol{s}) = \varphi_{v-1}^{s_{v-1}} \circ \cdots \circ \varphi_0^{s_0}(u)\,, \tag{21}$$

where $\boldsymbol{s} = (s_0, \ldots, s_{v-1}) \in \mathcal{T}$. In this construction, indeed, $\Phi(u,\boldsymbol{s}) = u$ for $\boldsymbol{s} = (0,\ldots,0)$, and the space $\mathcal{T}$ is further restricted in such a way that this is the only element (see [62]). The choice of MEM as our starting primitive is arbitrary, arguably better choices for the sake of generality would have been Tweakable Even-Mansour (TEM) [35] or XPX [84]. However, TEM is general in the sense that it takes an arbitrary mask based on a universal hash function independent of the permutation P, and XPX considers masks of the form $\delta_1 k \oplus \delta_2 \mathsf{P}(k)$ which suits generality but makes the proof more technical. Application-wise, as we will see in Appendix A.2, starting from MEM is more convenient.

The security of genMEM is highly dependent on restrictions put on the masking function. This was also the case for the original MEM, which for example becomes insecure if an adversary manages to find two distinct $\boldsymbol{s}, \boldsymbol{s'}$ such that

$$\varphi_{v-1}^{s_{v-1}} \circ \cdots \circ \varphi_0^{s_0} = \varphi_{v-1}^{s'_{v-1}} \circ \cdots \circ \varphi_0^{s'_0}\,.$$

Our abstraction of the masking function allows us to discard these peculiarities. Inspired by [62], we define *$\epsilon$-properness* of the masking function $\Phi$ as follows:

**Definition 1** The masking function $\Phi : \{0,1\}^n \times \mathcal{T} \to \{0,1\}^n$ is called *$\epsilon$-proper* if the family of functions

$$\{\Phi(u,\cdot) : \mathcal{T} \to \{0,1\}^n \mid u \in \{0,1\}^n\} \tag{22}$$

is $\epsilon$-universal and $\epsilon$-xor-universal, respectively:

$$\max_{s \in \mathcal{T},\ y \in \{0,1\}^n} \mathbf{Pr}\left(\Phi(u,s) = y\right) \le 2^{-\epsilon}\,,$$
$$\max_{s \neq s' \in \mathcal{T},\ y \in \{0,1\}^n} \mathbf{Pr}\left(\Phi(u,s) \oplus \Phi(u,s') = y\right) \le 2^{-\epsilon}\,,$$

where the randomness is taken over $u$.

We are ready to prove security of genMEM in the random invariant permutation model.

**Theorem 4** *Consider any fixed $\Lambda$. Let $k \xleftarrow{\$} \{0,1\}^\kappa$, $\mathsf{P} \xleftarrow{\$} \mathsf{perm}[\Lambda](n)$, and $\tilde{\pi} \xleftarrow{\$} \mathsf{tperm}(\{0,1\}^\tau \times \mathcal{T}, n)$. Let $\Phi$ be an $\epsilon$-proper masking function. Consider any distinguisher $\mathsf{D}$ making at most $q$ construction queries and $p$ primitive queries, that never queries its construction oracle in inverse direction for any tweak of the form $(t,0) \in \{0,1\}^\tau \times \mathcal{T}$. Among the $q$ construction queries, assume that there are at most $q'$ different tweaks $t \in \{0,1\}^\tau$, and each of these occurs at most $\ell$ times. We have,*

$$\begin{aligned}
&\Delta_{\mathsf{D}}\left(\mathsf{genMEM}_k^{\mathsf{P}'}, \mathsf{P}' \,;\, \tilde{\pi}, \mathsf{P}'\right) \\
&\quad \le |\Lambda| \binom{q}{2} \frac{3}{\min\{2^\epsilon, 2^n\}} + \frac{3|\Lambda|q(q+p)}{\min\{2^\epsilon, 2^n\}} + \frac{|\Lambda|p}{2^\kappa} + q' \cdot \sum_{\lambda \in \Lambda \setminus \{\mathrm{id}\}} \frac{\#\{u \mid u = \lambda(u)\}}{2^n} \\
&\qquad + \max_{z=(z_1,\ldots,z_{q'}) \subseteq \{0,1\}^\tau} \sum_{\lambda \in \Lambda \setminus \{\mathrm{id}\}} \sum_{z,z' \in z} \frac{\#\left\{k \mid k0^* \oplus 0^\kappa z 0^* = \lambda\left(k0^* \oplus 0^\kappa z' 0^*\right)\right\}}{2^\kappa}
\end{aligned}$$

$$+ q' \max_{\substack{\mathcal{Z}=\{z_1,\ldots,z_\ell\}\subseteq\{0,1\}^n \\ \mathcal{S}=\{s_1,\ldots,s_\ell\}\subseteq\mathcal{T}}} \sum_{\lambda\in\Lambda\setminus\{\mathrm{id}\}} \sum_{\substack{z,z'\in\mathcal{Z} \\ s,s'\in\mathcal{S}}} \frac{2\#\left\{u \mid z \oplus \Phi(u,s) = \lambda\left(z' \oplus \Phi(u,s')\right)\right\}}{2^n}$$

$$+ q' \cdot \max_{\substack{\mathcal{Z}=\{z_1,\ldots,z_\ell\}\subseteq\{0,1\}^n \\ \mathcal{S}=\{s_1,\ldots,s_\ell\}\subseteq\mathcal{T}\setminus\{0\}}} \sum_{\lambda\in\Lambda\setminus\{\mathrm{id}\}} \sum_{\substack{z\in\mathcal{Z} \\ s\in\mathcal{S}}} \frac{\#\{u \mid \Phi(u,s) \oplus z = \lambda(u)\}}{2^n} .$$

The proof is very similar to those in [62,84], and included in Appendix B. Again, for $\Lambda = \{\mathrm{id}\}$ the theorem gives security of $\mathsf{genMEM}^{\mathsf{P}}$ in the random permutation model, for $\mathsf{P} \xleftarrow{\$} \mathsf{perm}(n)$. In this case, it matches the security bound derived by Granger et al.[62] on $\mathsf{MEM}^{\mathsf{P}}$: $\frac{4.5q^2+3qp}{2^\epsilon} + \frac{p}{2^\kappa}$, where the authors used that $n \geq \epsilon$ and simplified their bound accordingly.

The analysis is more involved than that of Even-Mansour in Theorem 1 due to the fact that (i) the masking consists of evaluations of $\mathsf{P}$ and (ii) the distinguisher can influence the masking by the use of tweaks. These differences result in additional bad events in the analysis. The last two terms of the bound have a form of $q'$ times a maximum taken over $\ell$ values. If the number of choices of $t$ is evenly distributed, $q' \cdot \ell \approx q$ and the bound is pretty tight. If there are outliers, for example, $q - \ell$ tweaks occur 1 time and 1 tweak occurs $\ell$ times, a slightly more accurate bound can be derived straightforwardly from the proof. In typical applications of $\mathsf{genMEM}$, $t$ represents a nonce, and each nonce is used an approximately equal amount of times.

### A.2 Offset Public Permutation

Offset Public Permutation (OPP) is an authenticated encryption scheme by Granger et al.[62]. It is designed on top of the MEM tweakable block cipher, with the specific masking function of (21) for cleverly chosen LFSRs. It takes nonces of size $n - \kappa$ bits, and outputs tags of size $t \leq n$. The encryption function $\mathcal{E}_k$ of OPP gets as input a nonce $N$, associated data $A$, and a message $M$, and outputs a ciphertext $C$ and tag $T$. The function $\mathcal{E}_k$ for integral data is depicted in Fig. 3; we refer to [62] for the full version. Associated with $\mathcal{E}_k$ is a decryption function $\mathcal{D}_k$: it should satisfy the property that for any input tuple $(N, A, M)$ to $\mathcal{E}_k$,

$$\mathcal{D}_k(N, A, \mathcal{E}_k(N, A, M)) = M .$$

If $\mathcal{D}_k$ does not receive a valid input, it outputs a dedicated $\perp$ symbol.

Security of the authenticated encryption scheme is slightly more advanced. The distinguisher has access to both $\mathcal{E}_k$ and $\mathcal{D}_k$, with the restriction that it never relays the response of an encryption query to the decryption oracle. The encryption oracle should never be queried

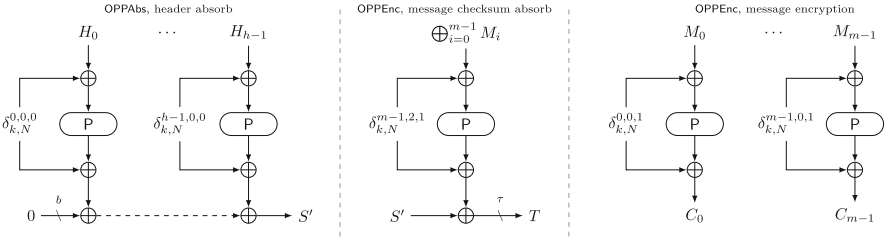


**Fig. 3** Offset Public Permutation (OPP) of [62] for integral data. Here, $\delta_{k,N}^{s_0,s_1,s_2}$ is short for $\Phi(\mathsf{P}(kN0^*), (s_0, s_1, s_2))$ with the function $\Phi$ of (21)

twice on the same nonce (there is no such restriction for couples of queries involving a decryption query). Security is measured by a distance to a random oracle $\mathsf{R}$ that for each input $M$ responds with a random string of size $|M|+t$, and a function that outputs a bot sign $\perp$ for each decryption query:

$$\Delta_{\mathsf{D}}\left(\mathcal{E}_k^{\mathsf{P}}, \mathcal{D}_k^{\mathsf{P}}, \mathsf{P}\ ;\ \mathsf{R}, \perp, \mathsf{P}\right),$$

where for simplicity we now assume the distinguisher can make one forgery attempt. Granger et al.[62] proved the following security bound for $\mathsf{OPP}$ based on a random permutation $\mathsf{P} \xleftarrow{\$} \mathsf{perm}(n)$:

$$\frac{2^{n-t}}{2^n-1} + \frac{4.5q^2+3qp}{2^\epsilon} + \frac{p}{2^\kappa},$$

where $q$ is the number of construction query blocks, and with $\epsilon = n$ (the masking of (21) is optimal). The proof idea is related to that of Theorem 2 but more advanced as a tweakable block cipher is involved. We demonstrate how a security bound for $\mathsf{OPP}$ based on an invariant permutation $\mathsf{P}' \xleftarrow{\$} \mathsf{perm}[\Lambda](n)$ is derived.

**Theorem 5** *Consider any fixed $\Lambda$. Let $k \xleftarrow{\$} \{0,1\}^\kappa$, $P' \xleftarrow{\$} perm[\Lambda](n)$, and $R$ be a random oracle as described above. Consider any distinguisher $D$ making at most $q'$ construction queries, each of length at most $\ell$ blocks and in total at most $q$ blocks, and $p$ primitive queries. We have,*

$$\Delta_D\left(\mathcal{E}_k^{P'}, \mathcal{D}_k^{P'}, P'\ ;\ R, \perp, P'\right) \le \frac{2^{n-t}}{2^n-1} + \mathtt{BOUND}_{Thm.4}(q, q', \ell, p), \tag{23}$$

*where $\mathtt{BOUND}_{Thm.4}(q, q', \ell, p)$ is the bound of Theorem 4 for distinguisher's parameters $(q, q', \ell, p)$, and with $\epsilon = n$.*

***Proof*** Consider any fixed deterministic distinguisher $\mathsf{D}$ that makes a total of $q$ construction query blocks and $p$ primitive queries. Consider $\mathsf{genMEM}_k^{\mathsf{P}'}$ of (20) with the masking function $\Phi$ of (21) and for $v = 3$, three LFSRs $\varphi_0$, $\varphi_1 = \varphi_0 \oplus \mathrm{id}$, and $\varphi_2 = \varphi_0^2 \oplus \varphi_0 \oplus \mathrm{id}$, and tweak space $\mathcal{T} = \{0, \ldots, \ell-1\} \times \{0, \ldots, 3\} \times \{0, 1\}$ (where $\ell$ denotes the maximum query length). The tweak $(0, 0, 0)$ is called in forward direction only, and we will be able to apply Theorem 4 later on.

We can observe that

$$\mathsf{OPP}_k^{\mathsf{P}'} = \Theta\mathsf{CB}_0^{\mathsf{genMEM}_k^{\mathsf{P}'}},$$

where $\Theta\mathsf{CB}$ is a tweakable block cipher based authenticated encryption scheme of Krovetz and Rogaway [77] (see [62] for details). Denote, for brevity, the encryption and decryption functions of $\Theta\mathsf{CB}$ by $\Theta\mathcal{E}$ and $\Theta\mathcal{D}$. We have:

$$\begin{aligned}
\Delta_{\mathsf{D}}\left(\mathcal{E}_k^{\mathsf{P}'}, \mathcal{D}_k^{\mathsf{P}'}, \mathsf{P}'\ ;\ \mathsf{R}, \perp, \mathsf{P}'\right) &= \Delta_{\mathsf{D}}\left(\Theta\mathcal{E}_0^{\mathsf{genMEM}_k^{\mathsf{P}'}}, \Theta\mathcal{D}_0^{\mathsf{genMEM}_k^{\mathsf{P}'}}, \mathsf{P}'\ ;\ \mathsf{R}, \perp, \mathsf{P}'\right) \\
&\le \Delta_{\mathsf{D}}\left(\Theta\mathcal{E}_0^{\tilde{\pi}}, \Theta\mathcal{D}_0^{\tilde{\pi}}, \mathsf{P}'\ ;\ \mathsf{R}, \perp, \mathsf{P}'\right) + \Delta_{\mathsf{D}'}\left(\mathsf{genMEM}_k^{\mathsf{P}'}, \mathsf{P}'\ ;\ \tilde{\pi}, \mathsf{P}'\right) \\
&\le \Delta_{\mathsf{D}}\left(\Theta\mathcal{E}_0^{\tilde{\pi}}, \Theta\mathcal{D}_0^{\tilde{\pi}}\ ;\ \mathsf{R}, \perp\right) + \Delta_{\mathsf{D}'}\left(\mathsf{genMEM}_k^{\mathsf{P}'}, \mathsf{P}'\ ;\ \tilde{\pi}, \mathsf{P}'\right),
\end{aligned} \tag{24}$$

where $\tilde{\pi} \xleftarrow{\$} \mathsf{tperm}(\{0,1\}^\tau \times \mathcal{T}, n)$ is a random tweakable permutation, and $\mathsf{D}'$ is some distinguisher that makes at most $q$ construction query blocks, for $q'$ distinct tweaks $t$ and

each tweak occurring at most $\ell$ times, and $p$ primitive queries. For the first term of (24), Krovetz and Rogaway [77] proved:

$$\Delta_{\mathsf{D}}\left(\Theta\mathcal{E}_0^{\tilde{\pi}}, \Theta\mathcal{D}_0^{\tilde{\pi}} \;;\; \mathsf{R}, \perp\right) \leq \frac{2^{n-t}}{2^n - 1}\,.$$

For the second term of (24) we rely on Theorem 4. □

## B Proof of Theorem 4

The proof follows [84] almost verbatim, but extends it in the presence of the invariances $\Lambda$.

Let $k \xleftarrow{\$} \{0,1\}^\kappa$, $\mathsf{P}' \xleftarrow{\$} \mathsf{perm}[\Lambda](\{0,1\}^n)$, and $\tilde{\pi} \xleftarrow{\$} \mathsf{tperm}(\{0,1\}^\tau \times \mathcal{T}, n)$. Consider any fixed deterministic distinguisher $\mathsf{D}$ that has access to either $\mathsf{O}_1 = (\mathsf{genMEM}^{\mathsf{P}'}, \mathsf{P}')$ or $\mathsf{O}_2 = (\tilde{\pi}, \mathsf{P}')$. Gather its $q$ constructions queries in a view $\nu_{\mathrm{c}} = \{(t_1, s_1, m_1, c_1), \ldots, (t_q, s_q, m_q, c_q)\}$ and its $p$ primitive queries in a view $\nu_{\mathrm{p}} = \{(x_1, y_1), \ldots, (x_p, y_p)\}$. Let $q'$ be the number of unique values $t_i$ in $\nu_{\mathrm{c}}$, and denote these unique values by $\{t'_1, \ldots, t'_{q'}\}$. We recall that any construction query of the form $(t, 0, m, c)$ *must have been made in forward direction*.

After $\mathsf{D}$'s interaction with its oracle but before it outputs its decision bit $b$, we reveal the random key $k$ and a tuple $\nu_{\mathrm{m}} = \{(w_1, u_1), \ldots, (w_{q'}, u_{q'})\}$. In the real world, this tuple is generated as $w_i = kt'_i0^*$ and $u_i = \mathsf{P}(w_i)$. In the ideal world the $w_i$'s are unchanged but the values $u_1, \ldots, u_{q'} \xleftarrow{\$} \{0,1\}^n$ are uniformly randomly drawn.

The complete view is denoted

$$\nu = (\nu_{\mathrm{c}}, \nu_{\mathrm{p}}, \nu_{\mathrm{m}}, k)\,.$$

We assume that $\mathsf{D}$ never repeats any query, hence $\nu_{\mathrm{c}}$ and $\nu_{\mathrm{p}}$ do not contain duplicate queries. We particularly assume that $\mathsf{D}$ never repeats any primitive query transformed over any $\lambda$. For $\nu_{\mathrm{m}}$, two tuples never have colliding inputs (by definition of the values $t'_i$) but they may have colliding outputs. Also, $\nu_{\mathrm{c}}$ satisfies the property that for any two tuples with $(t_i, s_i) = (t_{i'}, s_{i'})$, we have $m_i \neq m_{i'}$ and $c_i \neq c_{i'}$.

For any $t \in \{t_1, \ldots, t_q\}$, write $\nu_{\mathrm{m}}(t) = u$ as shortcut notation for the unique $u$ such that $(kt0^*, u) \in \nu_{\mathrm{m}}$. A view $\nu$ is called *bad* if it satisfies one of the following conditions:

$$\begin{aligned}
\mathsf{BAD}_{\mathrm{c}} &\iff \exists\,(t,s,m,c) \in \nu_{\mathrm{c}}\,,\ \lambda \in \Lambda\backslash\{\mathrm{id}\}: \\
&\quad m \oplus \Phi(\nu_{\mathrm{m}}(t), s) = \lambda\big(m \oplus \Phi(\nu_{\mathrm{m}}(t), s)\big) \text{ or} \\
&\quad c \oplus \Phi(\nu_{\mathrm{m}}(t), s) = \lambda\big(c \oplus \Phi(\nu_{\mathrm{m}}(t), s)\big)\,, \\
\mathsf{BAD}_{\mathrm{cc}} &\iff \exists \text{ distinct } (t,s,m,c), (t',s',m',c') \in \nu_{\mathrm{c}}\,,\ \lambda \in \Lambda: \\
&\quad m \oplus \Phi(\nu_{\mathrm{m}}(t), s) = \lambda\big(m' \oplus \Phi(\nu_{\mathrm{m}}(t'), s')\big) \text{ or} \\
&\quad c \oplus \Phi(\nu_{\mathrm{m}}(t), s) = \lambda\big(c' \oplus \Phi(\nu_{\mathrm{m}}(t'), s')\big)\,, \\
\mathsf{BAD}_{\mathrm{cm}} &\iff \exists\,(t,s,m,c) \in \nu_{\mathrm{c}}\,,\ (w,u) \in \nu_{\mathrm{m}}\,,\ \lambda \in \Lambda: \\
&\quad m \oplus \Phi(\nu_{\mathrm{m}}(t), s) = \lambda\big(w\big) \text{ or } c \oplus \Phi(\nu_{\mathrm{m}}(t), s) = \lambda\big(u\big)\,, \\
\mathsf{BAD}_{\mathrm{cp}} &\iff \exists\,(t,s,m,c) \in \nu_{\mathrm{c}}\,,\ (x,y) \in \nu_{\mathrm{p}}\,,\ \lambda \in \Lambda: \\
&\quad m \oplus \Phi(\nu_{\mathrm{m}}(t), s) = \lambda\big(x\big) \text{ or } c \oplus \Phi(\nu_{\mathrm{m}}(t), s) = \lambda\big(y\big)\,, \\
\mathsf{BAD}_{\mathrm{m}} &\iff \exists\,(w,u) \in \nu_{\mathrm{m}}\,,\ \lambda \in \Lambda\backslash\{\mathrm{id}\}: \\
&\quad w = \lambda\big(w\big) \text{ or } u = \lambda\big(u\big)\,, \\
\mathsf{BAD}_{\mathrm{mm}} &\iff \exists \text{ distinct } (w,u), (w',u') \in \nu_{\mathrm{m}}\,,\ \lambda \in \Lambda:
\end{aligned}$$

$$
\begin{aligned}
& w = \lambda\big(w'\big) \text{ or } u = \lambda\big(u'\big)\,, \\
\mathsf{BAD}_{\mathrm{mp}} \Longleftrightarrow\ & \exists\ (w,u) \in \nu_{\mathrm{m}}\,,\ (x,y) \in \nu_{\mathrm{p}}\,,\ \lambda \in \Lambda : \\
& w = \lambda\big(x\big) \text{ or } u = \lambda\big(y\big)\,.
\end{aligned}
$$

The bad events look technically involved, but in fact are not. Restricted to the case that $\lambda = \mathrm{id}$, the events capture that $\nu$ does not include any two evaluations of $\mathsf{P}$ with identical input or identical output. Stretching the analysis to arbitrary $\lambda$, we want that there are no collisions transformed over $\lambda$. This explains all bad events, except for the presence of $\mathsf{BAD}_{\mathrm{c}}$ and $\mathsf{BAD}_{\mathrm{m}}$: these are included as for $\lambda \neq \mathrm{id}$, a query may non-trivially collide with itself.

In below two lemmas, we will derive an upper bound on $\mathbf{Pr}\left(X_2 \in \mathcal{V}_{\mathrm{bad}}\right)$ (Lemma 4), and show that for any good view, $\mathbf{Pr}\left(X_1 = \nu\right) \geq \mathbf{Pr}\left(X_2 = \nu\right)$ (Lemma 5). This completes the proof using Lemma 1 on the H-coefficient technique.

**Lemma 4** *We have*

$$
\begin{aligned}
\mathbf{Pr}\left(X_2 \in \mathcal{V}_{\mathrm{bad}}\right) \leq\ & |\Lambda| \binom{q}{2} \frac{3}{\min\{2^\epsilon, 2^n\}} + \frac{3|\Lambda| q(q+p)}{\min\{2^\epsilon, 2^n\}} + \frac{|\Lambda| p}{2^\kappa} \\
& + q' \cdot \sum_{\lambda \in \Lambda \setminus \{\mathrm{id}\}} \frac{\#\{u \mid u = \lambda(u)\}}{2^n} \\
& + \max_{z = (z_1, \ldots, z_{q'}) \subseteq \{0,1\}^\tau} \sum_{\lambda \in \Lambda \setminus \{\mathrm{id}\}} \sum_{z, z' \in z} \frac{\#\big\{k \mid k0^* \oplus 0^\kappa z 0^* = \lambda\big(k0^* \oplus 0^\kappa z' 0^*\big)\big\}}{2^\kappa} \\
& + q' \max_{\substack{z = \{z_1, \ldots, z_\ell\} \subseteq \{0,1\}^n \\ s = \{s_1, \ldots, s_\ell\} \subseteq \mathcal{T}}} \sum_{\lambda \in \Lambda \setminus \{\mathrm{id}\}} \sum_{\substack{z, z' \in z \\ s, s' \in s}} \frac{2\#\big\{u \mid z \oplus \Phi(u,s) = \lambda\big(z' \oplus \Phi(u,s')\big)\big\}}{2^n} \\
& + q' \cdot \max_{\substack{z = \{z_1, \ldots, z_\ell\} \subseteq \{0,1\}^n \\ s = \{s_1, \ldots, s_\ell\} \subseteq \mathcal{T} \setminus \{0\}}} \sum_{\lambda \in \Lambda \setminus \{\mathrm{id}\}} \sum_{\substack{z \in z \\ s \in s}} \frac{\#\{u \mid \Phi(u,s) \oplus z = \lambda(u)\}}{2^n}\,.
\end{aligned}
$$

***Proof*** The probability $\mathbf{Pr}\left(X_2 \in \mathcal{V}_{\mathrm{bad}}\right)$ equals the probability that $\mathsf{BAD} := \mathsf{BAD}_{\mathrm{c}} \vee \mathsf{BAD}_{\mathrm{cc}} \vee \mathsf{BAD}_{\mathrm{cm}} \vee \mathsf{BAD}_{\mathrm{cp}} \vee \mathsf{BAD}_{\mathrm{m}} \vee \mathsf{BAD}_{\mathrm{mm}} \vee \mathsf{BAD}_{\mathrm{mp}}$ holds in the ideal world. By basic probability theory,

$$
\begin{aligned}
\mathbf{Pr}\left(X_2 \in \mathcal{V}_{\mathrm{bad}}\right) =& \mathbf{Pr}\left(\mathsf{BAD}\right) \\
\leq& \mathbf{Pr}\left(\mathsf{BAD}_{\mathrm{c}}\right) + \mathbf{Pr}\left(\mathsf{BAD}_{\mathrm{cc}}\right) + \mathbf{Pr}\left(\mathsf{BAD}_{\mathrm{cm}}\right) + \mathbf{Pr}\left(\mathsf{BAD}_{\mathrm{cp}}\right) \\
& + \mathbf{Pr}\left(\mathsf{BAD}_{\mathrm{m}}\right) + \mathbf{Pr}\left(\mathsf{BAD}_{\mathrm{mm}}\right) + \mathbf{Pr}\left(\mathsf{BAD}_{\mathrm{mp}}\right)\,, \qquad (25)
\end{aligned}
$$

where we recall that in the ideal world, $k \xleftarrow{\$} \{0,1\}^\kappa$ and $u_1, \ldots, u_{q'} \xleftarrow{\$} \{0,1\}^n$. In addition, for any construction query of the form $(t, 0, m, c)$, $c$ is randomly drawn from a set of size at least $2^n - q$ elements.

**$\mathsf{BAD}_{\mathbf{c}}$.** Consider any query $(t, s, m, c) \in \nu_{\mathrm{c}}$ and any $\lambda \in \Lambda \setminus \{\mathrm{id}\}$. As $\nu_{\mathrm{m}}(t) = u \xleftarrow{\$} \{0,1\}^n$ is a randomly generated $n$-bit value, the bad event is set with probability at most

$$
\frac{\#\big\{u \mid m \oplus \Phi(u,s) = \lambda\big(m \oplus \Phi(u,s)\big)\big\} + \#\big\{k \mid c \oplus \Phi(u,s) = \lambda\big(c \oplus \Phi(u,s)\big)\big\}}{2^n}\,.
$$

Taking the sum over all queries and all choices of $\lambda$, the bad event is set with probability at most

$$\begin{aligned}\mathbf{Pr}\left(\mathsf{BAD}_{\mathrm{c}}\right) \leq & \sum_{(t,s,m,c)\in\nu_{\mathrm{c}}} \sum_{\lambda\in\Lambda\setminus\{\mathrm{id}\}} \frac{\#\left\{k \mid c\oplus\Phi(u,s)=\lambda\left(c\oplus\Phi(u,s)\right)\right\}}{2^n} \\ & + \sum_{(t,s,m,c)\in\nu_{\mathrm{c}}} \sum_{\lambda\in\Lambda\setminus\{\mathrm{id}\}} \frac{\#\left\{k \mid c\oplus\Phi(u,s)=\lambda\left(c\oplus\Phi(u,s)\right)\right\}}{2^n} \\ \leq & q' \cdot \max_{\substack{z=\{z_1,\ldots,z_\ell\}\subseteq\{0,1\}^n \\ s=\{s_1,\ldots,s_\ell\}\subseteq\mathcal{T}}} \sum_{\lambda\in\Lambda\setminus\{\mathrm{id}\}} \sum_{\substack{z\in z \\ s\in s}} \frac{2\#\left\{u \mid z\oplus\Phi(u,s)=\lambda\left(z\oplus\Phi(u,s)\right)\right\}}{2^n}\,.\end{aligned}$$

Here, we made a similar simplification as in the proof of Lemma 2.

**BAD$_{\mathrm{cc}}$**. Consider any distinct queries $(t,s,m,c)$, $(t',s',m',c') \in \nu_{\mathrm{c}}$ and any $\lambda \in \Lambda$.

- If $t \neq t'$, then $\nu_{\mathrm{m}}(t)$ and $\nu_{\mathrm{m}}(t')$ are two independent randomly generated $n$-bit values and we argue based on the former. The bad event is set if

$$\Phi(\nu_{\mathrm{m}}(t),s) \in \{m\oplus\lambda\left(m'\oplus\Phi(\nu_{\mathrm{m}}(t'),s')\right), c\oplus\lambda\left(c'\oplus\Phi(\nu_{\mathrm{m}}(t'),s')\right)\}\,.$$

  As $\Phi$ is $\epsilon$-proper, it is $\epsilon$-universal (Definition 1), and the bad event is set with probability $2/2^{\epsilon}$;
- If $t = t'$, then $\nu_{\mathrm{m}}(t) = \nu_{\mathrm{m}}(t')$ is a single randomly generated $n$-bit value. In this case, either $s \neq s'$ or ($m \neq m'$ and $c \neq c'$).
  - If $\lambda = \mathrm{id}$, the bad event is set if

$$\Phi(\nu_{\mathrm{m}}(t),s)\oplus\Phi(\nu_{\mathrm{m}}(t),s') \in \{m\oplus m', c\oplus c'\}\,. \tag{26}$$

    As $\Phi$ is $\epsilon$-proper, it is $\epsilon$-xor-universal (Definition 1), and (26) holds with probability at most $2/2^{\epsilon}$;
  - If $\lambda \neq \mathrm{id}$, as $\nu_{\mathrm{m}}(t) = u \xleftarrow{\$} \{0,1\}^n$ is a randomly generated $n$-bit value, the bad event is set with probability at most

$$\frac{\#\left\{u \mid m\oplus\Phi(u,s)=\lambda\left(m'\oplus\Phi(u,s')\right)\right\}+\#\left\{u \mid c\oplus\Phi(u,s)=\lambda\left(c'\oplus\Phi(u,s')\right)\right\}}{2^n}\,.$$

Taking the sum over all distinct queries and all choices of $\lambda$, the bad event is set with probability at most

$$\begin{aligned}\mathbf{Pr}\left(\mathsf{BAD}_{\mathrm{cc}}\right) \leq & |\Lambda|\binom{q}{2}\frac{2}{2^{\epsilon}} \\ & + \sum_{\substack{(t,s,m,c)\neq(t',s',m',c')\in\nu_{\mathrm{c}} \\ t=t'}} \sum_{\lambda\in\Lambda\setminus\{\mathrm{id}\}} \frac{\#\left\{u \mid m\oplus\Phi(u,s)=\lambda\left(m'\oplus\Phi(u,s')\right)\right\}}{2^n} \\ & + \sum_{\substack{(t,s,m,c)\neq(t',s',m',c')\in\nu_{\mathrm{c}} \\ t=t'}} \sum_{\lambda\in\Lambda\setminus\{\mathrm{id}\}} \frac{\#\left\{u \mid c\oplus\Phi(u,s)=\lambda\left(c'\oplus\Phi(u,s')\right)\right\}}{2^n} \\ \leq & |\Lambda|\binom{q}{2}\frac{2}{2^{\epsilon}} \\ & + q' \max_{\substack{z=\{z_1,\ldots,z_\ell\}\subseteq\{0,1\}^n \\ s=\{s_1,\ldots,s_\ell\}\subseteq\mathcal{T}}} \sum_{\lambda\in\Lambda\setminus\{\mathrm{id}\}} \sum_{\substack{z,z'\in z \\ s,s'\in s \\ (z,s)\neq(z',s')}} \frac{2\#\left\{u \mid z\oplus\Phi(u,s)=\lambda\left(z'\oplus\Phi(u,s')\right)\right\}}{2^n}\,.\end{aligned}$$

Here, we made a similar simplification as before. We stress that in the summation, we have to include the case that $z = z'$ (but necessarily $s \neq s'$) and $s = s'$ (but necessarily $z \neq z'$).

**BAD$_{\mathbf{cm}}$**. Consider any queries $(t, s, m, c) \in \nu_{\mathrm{c}}$, $(w, u) \in \nu_{\mathrm{m}}$, and any $\lambda \in \Lambda$. For the first part of the bad event, $m \oplus \Phi(\nu_{\mathrm{m}}(t), s) = \lambda(w)$, we can observe that $\nu_{\mathrm{m}}(t) \xleftarrow{\$} \{0,1\}^n$ is independently generated from $w$. As $\Phi$ is $\epsilon$-proper, it is $\epsilon$-universal (Definition 1), and the bad event is set with probability $1/2^\epsilon$. Taking the sum over all queries and all choices of $\lambda$, the first part of the bad event is set with probability at most $\frac{|\Lambda|qq'}{2^\epsilon}$.

For the second part of the bad event, $c \oplus \Phi(\nu_{\mathrm{m}}(t), s) = \lambda(u)$, the analysis is more complicated. Let $w = kt'0^*$, such that $u = \nu_{\mathrm{m}}(t')$. The bad event is set if

$$\Phi(\nu_{\mathrm{m}}(t), s) \oplus \lambda(\nu_{\mathrm{m}}(t')) = c\,, \tag{27}$$

or equivalently

$$\Phi(\nu_{\mathrm{m}}(t), s) \oplus \lambda(\Phi(\nu_{\mathrm{m}}(t'), 0)) = c\,. \tag{28}$$

- If $t \neq t'$, then $\nu_{\mathrm{m}}(t)$ and $\nu_{\mathrm{m}}(t')$ are two independent randomly generated $n$-bit values and we argue based on the former. As $\Phi$ is $\epsilon$-proper, it is $\epsilon$-xor-universal (Definition 1), and Eq. (28) holds with probability at most $1/2^\epsilon$;
- If $t = t'$, then $\nu_{\mathrm{m}}(t) = \nu_{\mathrm{m}}(t')$ is a single randomly generated $n$-bit value.
  - If $s \neq 0$, we have the following two cases:
    - If $\lambda = \mathrm{id}$, as $\Phi$ is $\epsilon$-proper, it is $\epsilon$-universal (Definition 1), and (28) holds with probability at most $1/2^\epsilon$;
    - If $\lambda \neq \mathrm{id}$, as $\nu_{\mathrm{m}}(t) = u \xleftarrow{\$} \{0,1\}^n$, Eq. (27) holds with probability at most
    $$\frac{\#\{u \mid \Phi(u, s) \oplus c = \lambda(u)\}}{2^n}\,;$$
  - If $s = 0$, we have $\Phi(\nu_{\mathrm{m}}(t), s) = \Phi(\nu_{\mathrm{m}}(t), 0) = \nu_{\mathrm{m}}(t)$, and we cannot rely on the randomness of $\nu_{\mathrm{m}}(t)$. Instead, we use that in this case $c$ is randomly drawn from a set of size at least $2^n - q \geq 2^{n-1}$, and hence (28) holds with probability at most $2/2^n$.

Taking the sum over all queries and all choices of $\lambda$, the second part of the bad event is set with probability at most

$$\begin{aligned}
&\frac{2|\Lambda|qq'}{\min\{2^\epsilon, 2^n\}} + \sum_{(t,s,m,c)\in\nu_{\mathrm{c}}} \sum_{\lambda\in\Lambda\setminus\{\mathrm{id}\}} \frac{\#\{u \mid \Phi(u, s) \oplus c = \lambda(u)\}}{2^n} \\
&\leq \frac{2|\Lambda|qq'}{\min\{2^\epsilon, 2^n\}} + q' \cdot \max_{\substack{z=\{z_1,\ldots,z_\ell\}\subseteq\{0,1\}^n \\ s=\{s_1,\ldots,s_\ell\}\subseteq\mathcal{T}\setminus\{0\}}} \sum_{\lambda\in\Lambda\setminus\{\mathrm{id}\}} \sum_{\substack{z\in z \\ s\in s}} \frac{\#\{u \mid \Phi(u, s) \oplus z = \lambda(u)\}}{2^n}\,.
\end{aligned}$$

Here, we made a similar simplification as before.

Aggregating both cases, the entire bad event is set with probability at most

$$\begin{aligned}
\mathbf{Pr}\,(\mathsf{BAD}_{\mathrm{cm}}) \leq &\frac{3|\Lambda|qq'}{\min\{2^\epsilon, 2^n\}} \\
&+ q' \cdot \max_{\substack{z=\{z_1,\ldots,z_\ell\}\subseteq\{0,1\}^n \\ s=\{s_1,\ldots,s_\ell\}\subseteq\mathcal{T}\setminus\{0\}}} \sum_{\lambda\in\Lambda\setminus\{\mathrm{id}\}} \sum_{\substack{z\in z \\ s\in s}} \frac{\#\{u \mid \Phi(u, s) \oplus z = \lambda(u)\}}{2^n}\,.
\end{aligned}$$

**BAD$_{\mathbf{cp}}$**. Consider any queries $(t, s, m, c) \in \nu_{\mathrm{c}}$, $(x, y) \in \nu_{\mathrm{p}}$, and any $\lambda \in \Lambda$. The bad event is set if

$$\Phi(\nu_{\mathrm{m}}(t), s) \in \{m \oplus \lambda(x), c \oplus \lambda(y)\}.$$

As $\Phi$ is $\epsilon$-proper, it is $\epsilon$-universal (Definition 1), and the bad event is set with probability $2/2^{\epsilon}$. Taking the sum over all queries and all choices of $\lambda$, the bad event is set with probability at most

$$\mathbf{Pr}\left(\mathsf{BAD}_{\mathrm{cp}}\right) \leq \frac{2|\Lambda|qp}{2^{\epsilon}}.$$

**BAD$_{\mathbf{m}}$**. Consider any query $(w, u) \in \nu_{\mathrm{m}}$ and any $\lambda \in \Lambda\backslash\{\mathrm{id}\}$. Recall that $w = kt0^*$ for some $t$. The first part of the bad event, $w = \lambda(w)$, is set if

$$k0^* \oplus 0^{\kappa}t0^* = \lambda\left(k0^* \oplus 0^{\kappa}t0^*\right). \tag{29}$$

As $k \xleftarrow{\$} \{0,1\}^{\kappa}$, Eq. (29) holds with probability at most

$$\frac{\#\left\{k \mid k0^* \oplus 0^{\kappa}t0^* = \lambda\left(k0^* \oplus 0^{\kappa}t0^*\right)\right\}}{2^{\kappa}}.$$

The second part of the bad event, $u = \lambda(u)$, likewise holds with probability at most

$$\frac{\#\{u \mid u = \lambda(u)\}}{2^{n}},$$

as $u \xleftarrow{\$} \{0,1\}^{n}$.

Taking the sum over all queries and all choices of $\lambda$, the bad event is set with probability at most

$$\begin{aligned}
\mathbf{Pr}\left(\mathsf{BAD}_{\mathrm{m}}\right) \leq & q' \cdot \sum_{\lambda \in \Lambda\backslash\{\mathrm{id}\}} \frac{\#\{u \mid u = \lambda(u)\}}{2^{n}} \\
& + \sum_{(w,u)\in\nu_{\mathrm{m}}} \sum_{\lambda \in \Lambda\backslash\{\mathrm{id}\}} \frac{\#\left\{k \mid k0^* \oplus 0^{\kappa}t0^* = \lambda\left(k0^* \oplus 0^{\kappa}t0^*\right)\right\}}{2^{\kappa}} \\
\leq & q' \cdot \sum_{\lambda \in \Lambda\backslash\{\mathrm{id}\}} \frac{\#\{u \mid u = \lambda(u)\}}{2^{n}} \\
& + \max_{z=(z_1,\ldots,z_{q'})\subseteq\{0,1\}^{\tau}} \sum_{\lambda \in \Lambda\backslash\{\mathrm{id}\}} \sum_{z\in z} \frac{\#\left\{k \mid k0^* \oplus 0^{\kappa}z0^* = \lambda\left(k0^* \oplus 0^{\kappa}z0^*\right)\right\}}{2^{\kappa}},
\end{aligned}$$

where in fact the $w$'s are parsed as $kt0^*$, and we made a similar simplification as before.

**BAD$_{\mathbf{mm}}$**. Consider any distinct queries $(w, u)$, $(w', u') \in \nu_{\mathrm{m}}$ and any $\lambda \in \Lambda$. The second part of the bad event, $u = \lambda(u')$, happens with probability $1/2^{n}$. The first part of the bad event, $w = \lambda(w')$, happens with non-zero probability only if $\lambda \neq \mathrm{id}$; we consider $\lambda \in \Lambda\backslash\{\mathrm{id}\}$. Recall that $w = kt0^*$ for some $t$, and similarly for $w'$. The bad event is set if

$$k0^* \oplus 0^{\kappa}t0^* = \lambda\left(k0^* \oplus 0^{\kappa}t'0^*\right). \tag{30}$$

As $k \xleftarrow{\$} \{0,1\}^{\kappa}$, Eq. (30) holds with probability at most

$$\frac{\#\left\{k \mid k0^* \oplus 0^{\kappa}t0^* = \lambda\left(k0^* \oplus 0^{\kappa}t'0^*\right)\right\}}{2^{\kappa}}.$$

Taking the sum over all distinct queries and all choices of $\lambda$, the bad event is set with probability at most

$$\begin{aligned}\mathbf{Pr}\left(\mathsf{BAD}_{\mathrm{mm}}\right) \le & |\Lambda| \binom{q'}{2} \frac{1}{2^n} \\ & + \sum_{(w,u)\neq(w',u')\in\nu_{\mathrm{c}}} \sum_{\lambda\in\Lambda\setminus\{\mathrm{id}\}} \frac{\#\left\{k \mid k0^* \oplus 0^\kappa t0^* = \lambda\left(k0^* \oplus 0^\kappa t'0^*\right)\right\}}{2^\kappa} \\ \le & |\Lambda| \binom{q'}{2} \frac{1}{2^n} \\ & + \max_{z=(z_1,\ldots,z_{q'})\subseteq\{0,1\}^\tau} \sum_{\lambda\in\Lambda\setminus\{\mathrm{id}\}} \sum_{z\neq z'\in z} \frac{\#\left\{k \mid k0^* \oplus 0^\kappa z0^* = \lambda\left(k0^* \oplus 0^\kappa z'0^*\right)\right\}}{2^\kappa},\end{aligned}$$

where we made a similar simplification as before.

**$\mathsf{BAD}_{\mathbf{mp}}$.** Consider any query $(x, y) \in \nu_{\mathrm{p}}$ and any $\lambda \in \Lambda$. The first part of the bad event, $w = \lambda\left(x\right)$, happens if the first $\kappa$ bits $\lambda\left(x\right)$ equal $k$, which happens with probability $1/2^\kappa$. The second part of the bad event, $u = \lambda\left(y\right)$, happens with probability $q'/2^n$ (noting that we have to sum over all possible queries $(w, u) \in \nu_{\mathrm{m}}$). Taking the sum over all queries and all choices of $\lambda$, the bad event is set with probability at most

$$\mathbf{Pr}\left(\mathsf{BAD}_{\mathrm{mp}}\right) \le \frac{|\Lambda| p}{2^\kappa} + \frac{|\Lambda| q' p}{2^n}.$$

**Conclusion.** Adding the separate bounds in accordance with (25) gives

$$\begin{aligned}\mathbf{Pr}\left(\mathsf{BAD}\right) \le & |\Lambda| \binom{q}{2} \frac{3}{\min\{2^\epsilon, 2^n\}} + \frac{3|\Lambda| q(q+p)}{\min\{2^\epsilon, 2^n\}} + \frac{|\Lambda| p}{2^\kappa} \\ & + q' \cdot \sum_{\lambda\in\Lambda\setminus\{\mathrm{id}\}} \frac{\#\{u \mid u = \lambda(u)\}}{2^n} \\ & + \max_{z=(z_1,\ldots,z_{q'})\subseteq\{0,1\}^\tau} \sum_{\lambda\in\Lambda\setminus\{\mathrm{id}\}} \sum_{z,z'\in z} \frac{\#\left\{k \mid k0^* \oplus 0^\kappa z0^* = \lambda\left(k0^* \oplus 0^\kappa z'0^*\right)\right\}}{2^\kappa} \\ & + q' \max_{\substack{z=\{z_1,\ldots,z_\ell\}\subseteq\{0,1\}^n \\ s=\{s_1,\ldots,s_\ell\}\subseteq\mathcal{T}}} \sum_{\lambda\in\Lambda\setminus\{\mathrm{id}\}} \sum_{\substack{z,z'\in z \\ s,s'\in s}} \frac{2\#\left\{u \mid z \oplus \Phi(u,s) = \lambda\left(z' \oplus \Phi(u,s')\right)\right\}}{2^n} \\ & + q' \cdot \max_{\substack{z=\{z_1,\ldots,z_\ell\}\subseteq\{0,1\}^n \\ s=\{s_1,\ldots,s_\ell\}\subseteq\mathcal{T}\setminus\{0\}}} \sum_{\lambda\in\Lambda\setminus\{\mathrm{id}\}} \sum_{\substack{z\in z \\ s\in s}} \frac{\#\{u \mid \Phi(u,s) \oplus z = \lambda(u)\}}{2^n},\end{aligned}$$

where in the summations the values $z, z'$ and $s, s'$ may be identical, and we have simplified the bound by allowing a factor 2 for some of the terms. □

**Lemma 5** *For any good view $\nu \in \mathcal{V}_{\mathrm{good}}$, $\mathbf{Pr}\left(X_1 = \nu\right) \ge \mathbf{Pr}\left(X_2 = \nu\right)$.*

***Proof*** In the real world $\mathsf{O}_1$, every tuple in $(\nu_c, \nu_p, \nu_m)$ defines *exactly* one input-output tuple of $\mathsf{P}' \xleftarrow{\$} \mathsf{perm}[\Lambda](n)$. Therefore, we derive

$$\begin{aligned}\mathbf{Pr}\left(X_1 = \nu\right) =& \mathbf{Pr}\left(\mathsf{P}' \xleftarrow{\$} \mathsf{perm}[\Lambda](n)\ ;\ \mathsf{genMEM}_k^{\mathsf{P}'} \vdash \nu_c \wedge \mathsf{P}' \vdash \nu_p \cup \nu_m \mid k\right) \\ & \cdot \mathbf{Pr}\left(k' \xleftarrow{\$} \{0,1\}^\kappa\ :\ k' = k\right) \\ =& \frac{1}{2^\kappa} \cdot \mathbf{Pr}\left(\mathsf{P}' \xleftarrow{\$} \mathsf{perm}[\Lambda](n)\ ;\ \mathsf{P}' \vdash \bar{\nu}_c \cup \nu_p \cup \nu_m\right),\end{aligned}$$

where $\bar{\nu}_c = \{(m \oplus \Phi(\nu_m(t), s), c \oplus \Phi(\nu_m(t), s)) \mid (t, s, m, c) \in \nu_c\}$, and where $\bar{\nu}_c \cup \nu_p \cup \nu_m$ contains a total amount of $q + p + q'$ tuples among which there are no input or output collisions even when the tuples are transformed over any $\lambda \in \Lambda$.

In the ideal world $\mathsf{O}_2$, for $(t, s) \in \{0,1\}^\tau \times \mathcal{T}$, define $q_{(t,s)}$ to be the number of tuples in $\nu_c$ for tweak $(t, c)$. We have

$$\begin{aligned}\mathbf{Pr}\left(X_2 = \nu\right) &= \mathbf{Pr}\left(\tilde{\pi} \xleftarrow{\$} \mathsf{tperm}(\{0,1\}^\tau \times \mathcal{T}, n)\ ;\ \tilde{\pi} \vdash \nu_c\right) \\ &\quad \cdot \mathbf{Pr}\left(\mathsf{P}' \xleftarrow{\$} \mathsf{perm}[\Lambda](n)\ ;\ \mathsf{P}' \vdash \nu_p\right) \\ &\quad \cdot \mathbf{Pr}\left(u'_1, \ldots, u'_{q'} \xleftarrow{\$} \{0,1\}^n\ ;\ (u'_1, \ldots, u'_{q'}) = (u_1, \ldots, u_{q'})\right) \\ &\quad \cdot \mathbf{Pr}\left(k' \xleftarrow{\$} \{0,1\}^\kappa\ :\ k' = k\right) \\ &= \frac{1}{\prod_{(t,s)} (2^n)_{q_{(t,s)}} \cdot 2^{nq'+\kappa}} \cdot \mathbf{Pr}\left(\mathsf{P}' \xleftarrow{\$} \mathsf{perm}[\Lambda](n)\ ;\ \mathsf{P}' \vdash \nu_p\right) \\ &\leq \frac{1}{(2^n)_{q+q'} \cdot 2^\kappa} \cdot \mathbf{Pr}\left(\mathsf{P}' \xleftarrow{\$} \mathsf{perm}[\Lambda](n)\ ;\ \mathsf{P}' \vdash \nu_p\right).\end{aligned}$$

where $\nu_p$ represents a list of $p$ tuples for which there are no input or outputs collisions even when transformed over any $\lambda \in \Lambda$.

We obtain that

$$\begin{aligned}\frac{\mathbf{Pr}\left(X_1 = \nu\right)}{\mathbf{Pr}\left(X_2 = \nu\right)} &\geq \frac{\mathbf{Pr}\left(\mathsf{P}' \xleftarrow{\$} \mathsf{perm}[\Lambda](n)\ ;\ \mathsf{P}' \vdash \bar{\nu}_c \cup \nu_p \cup \nu_m\right)}{\frac{1}{(2^n)_{q+q'}} \cdot \mathbf{Pr}\left(\mathsf{P}' \xleftarrow{\$} \mathsf{perm}[\Lambda](n)\ ;\ \mathsf{P}' \vdash \nu_p\right)} \\ &= \frac{\mathbf{Pr}\left(\mathsf{P}' \xleftarrow{\$} \mathsf{perm}[\Lambda](n)\ ;\ \mathsf{P}' \vdash \bar{\nu}_c \cup \nu_p \cup \nu_m \mid \mathsf{P}' \vdash \nu_p\right)}{\frac{1}{(2^n)_{q+q'}}} \\ &\geq \frac{\mathbf{Pr}\left(\mathsf{P} \xleftarrow{\$} \mathsf{perm}(n)\ ;\ \mathsf{P} \vdash \bar{\nu}_c \cup \nu_p \cup \nu_m \mid \mathsf{P} \vdash \nu_p\right)}{\frac{1}{(2^n)_{q+q'}}} \\ &= \frac{\frac{1}{(2^n - p)_{q+q'}}}{\frac{1}{(2^n)_{q+q'}}} \geq 1.\end{aligned}$$

where for the second approximation we use that the list of tuples in $\bar{\nu}_c \cup \nu_p$ do not collide even when transformed over any $\lambda \in \Lambda$. □

## References


1. Aerts W., Biham E., Moitie D.D., Mulder E.D., Dunkelman O., Indesteege S., Keller N., Preneel B., Vandenbosch G.A.E., Verbauwhede I.: A practical attack on KeeLoq. J. Cryptol. **25**(1), 136–157 (2012).
2. Andreeva E., Daemen J., Mennink B., Van Assche G.: Security of Keyed Sponge Constructions Using a Modular Proof Approach. In: Leander [78], pp. 364–384.
3. Aumasson J., Brier E., Meier W., Naya-Plasencia M., Peyrin T.: Inside the Hypercube. In: C. Boyd, J.M.G. Nieto (eds.) ACISP 2009, *LNCS*, vol. 5594, pp. 202–213. Springer (2009).
4. Aumasson J., Jovanovic P., Neves S.: NORX: Parallel and Scalable AEAD. In: M. Kutylowski, J. Vaidya (eds.) ESORICS 2014, Part II, *LNCS*, vol. 8713, pp. 19–36. Springer (2014).
5. Aumasson J., Neves S., Wilcox-O'Hearn Z., Winnerlein C.: BLAKE2: Simpler, Smaller, Fast as MD5. In: M.J.J. Jr., M.E. Locasto, P. Mohassel, R. Safavi-Naini (eds.) ACNS 2013, *LNCS*, vol. 7954, pp. 119–135. Springer (2013).
6. Banik S., Bogdanov A., Isobe T., Shibutani K., Hiwatari H., Akishita T., Regazzoni F.: Midori: A Block Cipher for Low Energy. In: Iwata and Cheon [68], pp. 411–436.
7. Bar-On A., Biham E., Dunkelman O., Keller N.: Efficient slide attacks. J. Cryptol. **31**(3), 641–670 (2018).
8. Barkan E., Biham E.: In How Many Ways Can You Write Rijndael? In: Y. Zheng (ed.) ASIACRYPT 2002, *LNCS*, vol. 2501, pp. 160–175. Springer (2002).
9. Beierle C., Canteaut A., Leander G., Rotella Y.: Proving Resistance Against Invariant Attacks: How to Choose the Round Constants. In: J. Katz, H. Shacham (eds.) CRYPTO 2017, Part II, *LNCS*, vol. 10402, pp. 647–678. Springer (2017).
10. Bellare M., Rogaway P.: Random Oracles are Practical: A Paradigm for Designing Efficient Protocols. In: D.E. Denning, R. Pyle, R. Ganesan, R.S. Sandhu, V. Ashby (eds.) CCS '93, pp. 62–73. ACM (1993).
11. Benaloh J. (ed.): CT-RSA 2014, LNCS, vol. 8366. Springer, (2014).
12. Bernstein D.J.: Cache-timing attacks on AES (2004). http://cr.yp.to/papers.html#cachetiming. ID: cd9faae9bd5308c440df50fc26a517b4.
13. Bernstein D.J.: ChaCha, a variant of Salsa20. https://cr.yp.to/chacha.html (2008).
14. Bernstein D.J.: CubeHash specification (2.B.1) (2008). https://cubehash.cr.yp.to/submission.html.
15. Bernstein D.J.: The Salsa20 Family of Stream Ciphers. In: M.J.B. Robshaw, O. Billet (eds.) New Stream Cipher Designs - The eSTREAM Finalists, *LNCS*, vol. 4986, pp. 84–97. Springer (2008).
16. Bernstein D.J., Kölbl S., Lucks S., Massolino P.M.C., Mendel F., Nawaz K., Schneider T., Schwabe P., Standaert F., Todo Y., Viguier B.: Gimli : A Cross-Platform Permutation. In: W. Fischer, N. Homma (eds.) CHES 2017, *LNCS*, vol. 10529, pp. 299–320. Springer (2017).
17. Bertoni G., Daemen J., Peeters M., Van Assche G.: Sponge functions. Ecrypt Hash. Workshop **2007**, (2007).
18. Bertoni G., Daemen, J., Peeters, M., Van Assche, G.: The Keccak reference (2011). https://keccak.team/files/Keccak-reference-3.0.pdf
19. Bertoni G., Daemen J., Peeters M., Van Assche G.: On the security of the keyed sponge construction. Symmetric Key Encryption Workshop (SKEW 2011) (2011).
20. Beyne T.: Block Cipher Invariants as Eigenvectors of Correlation Matrices. In: T. Peyrin, S.D. Galbraith (eds.) ASIACRYPT 2018, Part I, *LNCS*, vol. 11272, pp. 3–31. Springer (2018).
21. Biryukov A., Udovenko A., Velichkov V.: Analysis of the NORX Core Permutation. Cryptology ePrint Archive, Report 2017/034 (2017).
22. Biryukov A., Wagner D.A.: Slide Attacks. In: L.R. Knudsen (ed.) FSE '99, *LNCS*, vol. 1636, pp. 245–259. Springer (1999).
23. Bogdanov A., Knudsen L.R., Leander G., Standaert F., Steinberger J.P., Tischhauser E.: Key-Alternating Ciphers in a Provable Setting: Encryption Using a Small Number of Public Permutations - (Extended Abstract). In: Pointcheval and Johansson [95], pp. 45–62.
24. Boros E., Szonyi T., Tichler K.: On defining sets for projective planes. Discrete Math. **303**(1–3), 17–31 (2005).
25. Bouillaguet C., Dunkelman O., Leurent G., Fouque P.: Another Look at Complementation Properties. In: S. Hong, T. Iwata (eds.) FSE 2010, *LNCS*, vol. 6147, pp. 347–364. Springer (2010).
26. Brualdi R.A., Pless V., Wilson R.M.: Short codes with a given covering radius. IEEE Trans. Inf. Theory **35**(1), 99–109 (1989).
27. Bulygin S., Walter M., Buchmann J.A.: Full analysis of PRINTcipher with respect to invariant subspace attack: efficient key recovery and countermeasures. Des. Codes Cryptogr **73**(3), 997–1022 (2014).
28. CAESAR: Competition for Authenticated Encryption: Security, Applicability, and Robustness (2018). http://competitions.cr.yp.to/caesar.html

29. Castro J.C.H., Estévez-Tapiador J.M., Quisquater J.: On the Salsa20 Core Function. In: K. Nyberg (ed.) FSE 2008, *LNCS*, vol. 5086, pp. 462–469. Springer (2008).
30. Chaigneau C., Fuhr T., Gilbert H., Jean J., Reinhard J.: Cryptanalysis of NORX v2.0. IACR Trans. Symmetric Cryptol. **2017**(1), 156–174 (2017).
31. Chang D., Dworkin M., Hong S., Kelsey J., Nandi M.: A keyed sponge construction with pseudorandomness in the standard model. NIST's 3rd SHA-3 Candidate Conference 2012 (2012).
32. Chen S., Lampe R., Lee J., Seurin Y., Steinberger J.P.: Minimizing the Two-Round Even-Mansour Cipher. In: J.A. Garay, R. Gennaro (eds.) CRYPTO 2014, Part I, *LNCS*, vol. 8616, pp. 39–56. Springer (2014).
33. Chen S., Steinberger J.P.: Tight Security Bounds for Key-Alternating Ciphers. In: P.Q. Nguyen, E. Oswald (eds.) EUROCRYPT 2014, *LNCS*, vol. 8441, pp. 327–350. Springer (2014).
34. Clark W.E., Pedersen J.: Sum-Free Sets in Vector Spaces over GF(2). J. Comb. Theory Ser. A **61**(2), 222–229 (1992).
35. Cogliati B., Lampe R., Seurin Y.: Tweaking Even-Mansour Ciphers. In: Gennaro and Robshaw [60], pp. 189–208.
36. Cohen G., Honkala I., Litsyn S., Lobstein A.: Covering Codes. North-Holland Mathematical Library. Elsevier Science, Amsterdam (1997).
37. Courtois N.T.: On the existence of non-linear invariants and algebraic polynomial constructive approach to backdoors in block ciphers. Cryptology ePrint Archive, Report 2018/807 (2018).
38. Courtois N.T.: Structural Nonlinear Invariant Attacks on T-310: Attacking Arbitrary Boolean Functions. Cryptology ePrint Archive, Report 2018/1242 (2018).
39. Daemen J.: Cipher and hash function design, strategies based on linear and differential cryptanalysis, PhD Thesis. KU Leuven (1995).
40. Daemen J., Hoffert S., Van Assche G., Van Keer R.: Xoodoo cookbook. Cryptology ePrint Archive, Report 2018/767 (2018).
41. Daemen J., Mennink B., Van Assche G.: Full-State Keyed Duplex with Built-In Multi-user Support. In: T. Takagi, T. Peyrin (eds.) ASIACRYPT 2017, Part II, *LNCS*, vol. 10625, pp. 606–637. Springer (2017).
42. Daemen J., Peeters M., Van Assche G.: Bitslice Ciphers and Power Analysis Attacks. In: B. Schneier (ed.) FSE 2000, *LNCS*, vol. 1978, pp. 134–149. Springer (2000).
43. Daemen J., Peeters M., Van Assche G., Rijmen V.: Nessie Proposal: Noekeon. http://gro.noekeon.org/Noekeon-spec.pdf (2000).
44. Daemen J., Rijmen V.: The Design of Rijndael: AES - The Advanced Encryption Standard. Springer, New York (2002).
45. Daemen J., Rijmen V.: The MAC function Pelican 2.0. Cryptology ePrint Archive, Report 2005/088 (2005).
46. Davies D.W.: Some Regular Properties of the 'Data Encryption Standard' Algorithm. In: Chaum D., Rivest R.L., Sherman A.T. (eds.) CRYPTO '82, pp. 89–96. Plenum Press, New York (1982).
47. Davydov A.A.: Constructions and families of covering codes and saturated sets of points in projective geometry. IEEE Trans. Inf. Theory **41**(6), 2071–2080 (1995).
48. Davydov A.A., Marcugini S., Pambianco F.: Minimal 1-saturating sets and complete caps in binary projective spaces. J. Comb. Theory Ser. A **113**(4), 647–663 (2006).
49. Dobraunig C., Eichlseder M., Mendel F., Schläffer M.: Ascon v1.2 (2016). Submission to CAESAR competition.
50. Dunkelman O., Keller N., Shamir A.: Minimalism in Cryptography: The Even-Mansour Scheme Revisited. In: Pointcheval and Johansson [95], pp. 336–354.
51. Dunkelman O., Keller N., Shamir A.: Almost universal forgery attacks on AES-based MAC's. Des. Codes Cryptogr. **76**(3), 431–449 (2015).
52. Ehrsam W.F., Meyer C.H., Smith J.L., Tuchman W.L.: Message verification and transmission error detection by block chaining (1978). US Patent 4,074,066.
53. Even S., Mansour Y.: A Construction of a Cipher From a Single Pseudorandom Permutation. In: H. Imai, R.L. Rivest, T. Matsumoto (eds.) ASIACRYPT '91, *LNCS*, vol. 739, pp. 210–224. Springer (1991).
54. Farshim P., Procter G.: The Related-Key Security of Iterated Even-Mansour Ciphers. In: Leander [78], pp. 342–363.
55. Ferguson N., Lucks S., McKay K.A.: Symmetric States and their Structure: Improved Analysis of CubeHash. Cryptology ePrint Archive, Report 2010/273 (2010).
56. FIPS 202: SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions (2015)
57. Flajolet P., Gardy D., Thimonier L.: Birthday paradox, coupon collectors, caching algorithms and self-organizing search. Discret. Appl. Math. **39**(3), 207–229 (1992). https://doi.org/10.1016/0166-218X(92)90177-C.

58. Gabidulin E.M., Davydov A.A., Tombak L.M.: Linear codes with covering radius 2 and other new covering codes. IEEE Trans. Inf. Theory **37**(1), 219–224 (1991).
59. Gazi P., Pietrzak K., Tessaro S.: The Exact PRF Security of Truncation: Tight Bounds for Keyed Sponges and Truncated CBC. In: Gennaro and Robshaw [60], pp. 368–387.
60. Gennaro R., Robshaw M. (eds.): CRYPTO 2015, Part I, LNCS, vol. 9215. Springer, (2015).
61. Giulietti M.: The geometry of covering codes: small complete caps and saturating sets in Galois spaces. In: S.R. Blackburn, S. Gerke, M. Wildon (eds.) Surveys in Combinatorics 2013, *London Mathematical Society Lecture Note Series*, vol. 409, pp. 51–90. Cambridge University Press (2013).
62. Granger R., Jovanovic P., Mennink B., Neves S.: Improved Masking for Tweakable Blockciphers with Applications to Authenticated Encryption. In: M. Fischlin, J. Coron (eds.) EUROCRYPT 2016, Part I, *LNCS*, vol. 9665, pp. 263–293. Springer (2016).
63. Gueron S., Mouha N.: Simpira v2: A Family of Efficient Permutations Using the AES Round Function. In: J.H. Cheon, T. Takagi (eds.) ASIACRYPT 2016, Part I, *LNCS*, vol. 10031, pp. 95–125 (2016).
64. Guo J., Jean J., Nikolic I., Qiao K., Sasaki Y., Sim S.M.: Invariant Subspace Attack Against Midori64 and The Resistance Criteria for S-box Designs. IACR Trans. Symmetric Cryptol. **2016**(1), 33–56 (2016).
65. Guo J., Karpman P., Nikolic I., Wang L., Wu S.: Analysis of BLAKE2. In: Benaloh [11], pp. 402–423.
66. Guo J., Peyrin T., Poschmann A.: The PHOTON Family of Lightweight Hash Functions. In: Rogaway [96], pp. 222–239
67. Hoang V.T., Krovetz T., Rogaway P.: Robust Authenticated-Encryption AEZ and the Problem That It Solves. In: Oswald and Fischlin [91], pp. 15–44.
68. Iwata T., Cheon J.H. (eds.): ASIACRYPT 2015, Part II, LNCS, vol. 9453. Springer, (2015).
69. Jean J.: Cryptanalysis of Haraka. IACR Trans. Symmetric Cryptol. **2016**(1), 1–12 (2016).
70. Jean J., Nikolic I., Peyrin T.: Tweaks and Keys for Block Ciphers: The TWEAKEY Framework. In: P. Sarkar, T. Iwata (eds.) ASIACRYPT 2014, Part II, *LNCS*, vol. 8874, pp. 274–288. Springer (2014).
71. Jean J., Nikolic I., Sasaki Y., Wang L.: Practical Cryptanalysis of PAES. In: Joux and Youssef [73], pp. 228–242.
72. Jean J., Nikolic I., Sasaki Y., Wang L.: Practical forgeries and distinguishers against PAES. IEICE Trans. **99–A**(1), 39–48 (2016).
73. Joux A., Youssef A.M. (eds.): SAC 2014, LNCS, vol. 8781. Springer, (2014).
74. Kavun E., Lauridsen M., Leander G., Rechberger C., Schwabe P., Yalçın T.: Prøst v1 (2014). Submission to CAESAR competition.
75. Knudsen L.R., Leander G., Poschmann A., Robshaw M.J.B.: PRINTcipher: A Block Cipher for IC-Printing. In: S. Mangard, F. Standaert (eds.) CHES 2010, *LNCS*, vol. 6225, pp. 16–32. Springer (2010).
76. Kölbl S., Lauridsen M.M., Mendel F., Rechberger C.: Haraka v2 - Efficient Short-Input Hashing for Post-Quantum Applications. IACR Trans. Symmetric Cryptol. **2016**(2), 1–29 (2016).
77. Krovetz T., Rogaway P.: The Software Performance of Authenticated-Encryption Modes. In: A. Joux (ed.) FSE 2011, *LNCS*, vol. 6733, pp. 306–327. Springer (2011).
78. Leander G. (ed.): FSE 2015, LNCS, vol. 9054. Springer, (2015).
79. Leander G., Abdelraheem M.A., AlKhzaimi H., Zenner E.: A Cryptanalysis of PRINTcipher: The Invariant Subspace Attack. In: Rogaway [96], pp. 206–221.
80. Leander G., Minaud B., Rønjom S.: A Generic Approach to Invariant Subspace Attacks: Cryptanalysis of Robin, iSCREAM and Zorro. In: Oswald and Fischlin [91], pp. 254–283.
81. Lee J., Luykx A., Mennink B., Minematsu K.: Connecting tweakable and multi-key blockcipher security. Des. Codes Cryptogr. **86**(3), 623–640 (2018).
82. Liskov M., Rivest R.L., Wagner D.A.: Tweakable Block Ciphers. In: M. Yung (ed.) CRYPTO 2002, *LNCS*, vol. 2442, pp. 31–46. Springer (2002).
83. McGrew D.A., Viega J.: The Security and Performance of the Galois/Counter Mode (GCM) of Operation. In: A. Canteaut, K. Viswanathan (eds.) INDOCRYPT 2004, *LNCS*, vol. 3348, pp. 343–355. Springer (2004).
84. Mennink B.: XPX: Generalized Tweakable Even-Mansour with Improved Security Guarantees. In: M. Robshaw, J. Katz (eds.) CRYPTO 2016, Part I, *LNCS*, vol. 9814, pp. 64–94. Springer (2016).
85. Mennink B., Reyhanitabar R., Vizár D.: Security of Full-State Keyed Sponge and Duplex: Applications to Authenticated Encryption. In: Iwata and Cheon [68], pp. 465–489.
86. Minaud B., Seurin Y.: The Iterated Random Permutation Problem with Applications to Cascade Encryption. In: Gennaro and Robshaw [60], pp. 351–367.
87. Mouha N.: Chaskey: a MAC Algorithm for Microcontrollers – Status Update and Proposal of Chaskey-12. Cryptology ePrint Archive, Report 2015/1182 (2015).
88. Mouha N., Luykx A.: Multi-key Security: The Even-Mansour Construction Revisited. In: Gennaro and Robshaw [60], pp. 209–223.

89. Mouha N., Mennink B., Herrewege A.V., Watanabe, D., Preneel, B., Verbauwhede, I.: Chaskey: An Efficient MAC Algorithm for 32-bit Microcontrollers. In: Joux and Youssef [73], pp. 306–323.
90. Naito Y., Yasuda K.: New Bounds for Keyed Sponges with Extendable Output: Independence Between Capacity and Message Length. In: T. Peyrin (ed.) FSE 2016, *LNCS*, vol. 9783, pp. 3–22. Springer (2016).
91. Oswald E., Fischlin M. (eds.): EUROCRYPT 2015, Part I, LNCS, vol. 9056. Springer, (2015).
92. Patarin J.: étude des générateurs de permutations basés sur le schéma du D.E.S. Ph.D. thesis, Université Paris 6, Paris, France (1991).
93. Patarin J.: The "Coefficients H" Technique. In: R.M. Avanzi, L. Keliher, F. Sica (eds.) SAC 2008, *LNCS*, vol. 5381, pp. 328–345. Springer (2008).
94. Peyrin T.: Improved Differential Attacks for ECHO and Grøstl. In: T. Rabin (ed.) CRYPTO 2010, *LNCS*, vol. 6223, pp. 370–392. Springer (2010). 10.1007/978-3-642-14623-7.
95. Pointcheval D., Johansson T. (eds.): EUROCRYPT 2012, LNCS, vol. 7237. Springer, (2012).
96. Rogaway P. (ed.): CRYPTO 2011, LNCS, vol. 6841. Springer, (2011).
97. Rønjom S.: Invariant subspaces in Simpira. Cryptology ePrint Archive, Report 2016/248 (2016).
98. Saarinen M.O.: CBEAM: Efficient Authenticated Encryption from Feebly One-Way $\phi$ Functions. In: Benaloh [11], pp. 251–269.
99. Stoffelen K., Daemen J.: Column Parity Mixers. IACR Trans. Symmetric Cryptol. **2018**(1), 126–159 (2018).
100. Todo Y., Leander G., Sasaki Y.: Nonlinear Invariant Attack - Practical Attack on Full SCREAM, iSCREAM, and Midori64. In: J.H. Cheon, T. Takagi (eds.) ASIACRYPT 2016, Part II, *LNCS*, vol. 10032, pp. 3–33 (2016).
101. Ughi E.: Saturated Configurations of Points in Projective Galois Spaces. Eur. J. Comb. **8**(3), 325–334 (1987).
102. Van Le T., Sparr R., Wernsdorf R., Desmedt Y.: Complementation-Like and Cyclic Properties of AES Round Functions. In: H. Dobbertin, V. Rijmen, A. Sowa (eds.) AES 2004, *LNCS*, vol. 3373, pp. 128–141. Springer (2004).
103. Wagner D.: Re: Re-rolled Salsa20 function. http://groups.google.com/group/sci.crypt/msg/0692e3aaf78687a3 (2005).
104. Whiting D., Housley R., Ferguson N.: AES Encryption and Authentication Using CTR Mode and CBC-MAC. IEEE 802.11-02/001r2 (2002).
105. Ye D., Wang P., Hu L., Wang L., Xie Y., Sun S., Wang P.: PAES v1: Parallelizable Authenticated Encryption Schemes based on AES Round Function (2014). Submission to CAESAR competition.