IACR Transactions on Symmetric Cryptology
ISSN 2519-173X, Vol. 2020, No. S1, pp. 5–30. DOI:10.13154/tosc.v2020.iS1.5-30

Dumbo, Jumbo, and Delirium: Parallel Authenticated Encryption for the Lightweight Circus

Tim Beyne[1], Yu Long Chen[1], Christoph Dobraunig[2] and Bart Mennink[2]

[1] KU Leuven and imec-COSIC, Leuven, Belgium
[2] Radboud University, Nijmegen, The Netherlands
elephant@cs.ru.nl

Abstract. With the trend to connect more and more devices to the Internet, authenticated encryption has become a major backbone in securing the communication, not only between these devices and servers, but also the direct communication among these devices. Most authenticated encryption algorithms used in practice are developed to perform well on modern high-end devices, but are not necessarily suited for usage on resource-constrained devices. We present a lightweight authenticated encryption scheme, called Elephant. Elephant retains the advantages of GCM such as parallelism, but is tailored to the needs of resource-constrained devices. The two smallest instances of Elephant, Dumbo and Jumbo, are based on the 160-bit and 176-bit Spongent permutation, respectively, and are particularly suited for hardware; the largest instance of Elephant, Delirium, is based on 200-bit Keccak and is developed towards software use. All three instances are parallelizable, have a small state size while achieving a high level of security, and are constant time by design.

Keywords: authenticated encryption · lightweight · parallel · minimalism · efficient

1 Introduction

Authenticated encryption has become an integral part of our modern communication infrastructure. Considering the rise of the Internet of Things, the usage will not only expand, but it will also be required that authenticated encryption algorithms run on resource-constrained devices. Many modern cryptographic protocols like TLS [Res18] or the Signal protocol [PM16, CCD+17] rely at their core on authenticated encryption. For instance, TLS 1.3 [Res18] relies on AES-GCM, or ChaCha20 with Poly1305, whereas in the Signal protocol [PM16, CCD+17], the task of authenticated encryption can be performed using AES in CBC mode for encryption paired with HMAC-SHA-2 for authentication. While the performance of these constructions may be sufficient on modern high-end systems, they have inadvertently some drawbacks for the usage in lightweight systems.

A first drawback is the use of components such as the AES [DR02], ChaCha [Ber08], and SHA-2 [FIP12], which were not designed with lightweight applications in mind. Moreover, ChaCha and SHA-2 make extensive use of modular additions, which is not the best choice for lightweight hardware implementations. A second problem is the need for the implementation of two different primitives (one for encryption and one for authentication) for performing the single task of authenticated encryption, which is a potential waste of resources in lightweight applications. This is still true if the primitives within these constructions are replaced with more lightweight counterparts. Furthermore, the usage of lightweight 64-bit block ciphers for the aforementioned mode implies stringent restrictions on the amount of data that can be safely encrypted [BL16,LS18]. The need for authenticated

Received: 2019-12-10, Accepted: 2020-02-15, Published: 2020-06-22

encryption schemes that perform well on resource-constrained devices has recently been addressed by NIST's call for lightweight authenticated encryption schemes [Nat18]. The call specifies a request for authenticated encryption schemes having at least 112-bit security provided that the online complexity is at most around 2^{50} bytes.

To provide an alternative for lightweight applications, we introduce the authenticated encryption scheme Elephant. The mode of Elephant is a nonce-based encrypt-then-MAC construction, where encryption is performed using counter mode and message authentication using a variant of the Wegman-Carter-Shoup MAC [WC81, Sho96, Ber05]. Both modes use a cryptographic permutation masked using LFSRs, akin to the masked Even-Mansour construction of Granger et al. [GJMN16].

The mode is permutation-based and only evaluates this permutation in the forward direction. As such, there is no need to implement multiple primitives or the inverse of the primitive, unlike in OCB-based [RBBK01, Rog04, KR11] authenticated encryption schemes. Furthermore, this allows us to rely and build on the extensive literature of permutations used for sponge-based lightweight hashing [AHMN10, GPP11, BKL+11]. That said, Elephant itself is not sponge-based: on the contrary, it departs from the conventional approach of serial permutation-based authenticated encryption. Elephant is parallelizable by design, easy to implement due to the use of LFSRs for masking (no need for finite field multiplication), and finally, it is efficient due to elegant decisions on how the masking should be performed exactly. A security analysis in the ideal permutation model demonstrates that the mode of Elephant is structurally sound.

Due to the parallelizability of Elephant, there is no need for instances with a large permutation: we can go as small as 160-bit permutations while still matching the security goals recommended by the NIST lightweight call [Nat18]. In detail, the Elephant scheme consists of three instances:

1. Dumbo: Elephant-Spongent-π[160]. This instance meets the minimum permutation size as dictated by the security analysis: it achieves 112-bit security provided that the online complexity is at most around 2^{46} blocks. This instance is particularly well-suited for hardware, as Spongent [BKL+11] itself is;

2. Jumbo: Elephant-Spongent-π[176]. This is a slightly more conservative instance of Elephant: it is based on the same permutation family, yet achieves 127-bit security under the same conditions on the online complexity. We note, in particular, that Spongent-π[176] is ISO/IEC standardized [BKL+11, ISO16];

3. Delirium: Elephant-Keccak-f[200]. This variant is developed more towards software use, although it still performs reasonably well in hardware. Elephant instantiated with Keccak-f[200] also achieves 127-bit security, with a higher bound of around 2^{70} blocks on the online complexity. The permutation is the smallest instance that is specified in the NIST SHA-3 standard [BDPV11b, FIP15] that fits our needs.

Dumbo and Jumbo are named after two famous elephants; Delirium is named after a Belgian beer, whose logo is a pink elephant. As each of the permutations is relatively small, all versions of Elephant have a small state size, despite its support for parallelism. The LFSRs used for masking are tailored to the specific instance, one for each, and are developed to operate well with the specific cryptographic permutation. For example, the LFSRs paired with the Spongent instances have been chosen to minimize the number of XOR operations that have to be performed for a state-update, while the Keccak-based instance has been selected to perform well on software platforms.

We note that the three cryptographic permutations in Elephant can also be used for cryptographic hashing – in fact, Spongent [BKL+11] and Keccak [BDPV11b] themselves are sponges – but due to our quest for small permutations, these cryptographic hash functions cannot meet the 112-, or 127-bit security level guaranteed by our authenticated

encryption schemes. In contrast, in order to perform sponge-based hashing with at least 112-bit security, a cryptographic permutation of size at least 225 bits must be used.

1.1 Related Work

Basing authenticated encryption on public permutations has become more and more popular with the standardization of the sponge-based [BDPV07] hash function Keccak [BDPV11b] as SHA-3 [FIP15], and the associated duplex construction [BDPV11a, MRV15, DMV17]. Besides these sequential approaches, several permutation-based authenticated encryption schemes have been proposed that allow for parallel processing of the input. Examples of such constructions are Minalpher [STA+15] and OPP [GJMN16] that require the inverse for decryption, MRO [GJMN16] that requires the processing of the whole associated data and message before encryption can start, and Farfalle [BDH+17] that shows how to instantiate a PRF by using permutations for parallel compressing and expanding, and that builds authenticated encryption on top of it. In some sense, Elephant can be seen as a complement to those existing parallel modes that puts its focus on lightweight authenticated encryption while still allowing for parallel computations.

1.2 Outline

Cryptographic preliminaries and the security model are discussed in Section 2. We describe the Simplified Masked Even-Mansour (SiM) tweakable block cipher in Section 3. This tweakable block cipher will be used in the security analysis of the Elephant authenticated encryption scheme. Elephant itself is discussed in Section 4, with its specification in Section 4.1 and its security analysis in Section 4.2. The three instances, and in particular the choice of the LFSRs, are described in Section 5.1 (for Dumbo), Section 5.2 (for Jumbo), and Section 5.3 (for Delirium), respectively. We give a detailed discussion of the design rationale, including implementation aspects, of Elephant in Section 6. Security proofs of SiM and Elephant are given in Sections 7 and 8, respectively. The work is concluded in Section 9.

2 Security Model

For $n \in \mathbb{N}$, we let $\{0,1\}^n$ denote the set of n-bit strings and $\{0,1\}^*$ the set of arbitrarily length strings. For $X \in \{0,1\}^*$, we define

$$X_1 \dots X_\ell \xleftarrow{n} X \tag{1}$$

to be the function that partitions X into $\ell = \lceil |X|/n \rceil$ blocks of size n bits, where the last block is appended with 0s. The expression "A ? B : C" equals B if A is true, and equals C if A is false. For $x \in \{0,1\}^n$ and $i \leq n$, we denote by $x \ll i$ (resp., $x \gg i$) a shift of x to the left (resp., right) over i positions. We likewise denote by $x \lll i$ (resp., $x \ggg i$) a rotation of x to the left (resp., right) over i positions. We denote by $\lfloor x \rfloor_i$ the i left-most bits of x.

For a finite set $\mathcal{T}$, we denote by $\mathrm{perm}(n)$ the set of all n-bit permutations and by $\mathrm{perm}(\mathcal{T}, n)$ the set of all families of permutations indexed by $T \in \mathcal{T}$. For a finite set $\mathcal{S}$, we denote by $s \xleftarrow{\$} \mathcal{S}$ the uniform random sampling of an element s from $\mathcal{S}$.

An adversary $\mathcal{A}$ is an algorithm that is given access to one or more oracles $\mathcal{O}$, and after interaction with $\mathcal{O}$ outputs a bit $b \in \{0,1\}$. This event is denoted as $\mathcal{A}^{\mathcal{O}} \to b$. In our work, we will be concerned with computationally unbounded adversaries $\mathcal{A}$; their complexities are only measured by the number of oracle queries. For two randomized oracles $\mathcal{O}, \mathcal{P}$, we denote the advantage of an adversary $\mathcal{A}$ in distinguishing both by

$$\Delta_{\mathcal{A}}\left(\mathcal{O}\,;\,\mathcal{P}\right) = \mathbf{Pr}\left(\mathcal{A}^{\mathcal{O}} \to 1\right) - \mathbf{Pr}\left(\mathcal{A}^{\mathcal{P}} \to 1\right). \tag{2}$$

Finally, let $k, m, n, t \in \mathbb{N}$ with $k, m, t \leq n$ throughout.

2.1 Authenticated Encryption

An authenticated encryption scheme $\_$ consists of two algorithms enc and dec. Encryption enc gets as input a key $K \in \{0,1\}^k$, a nonce $N \in \{0,1\}^m$, associated data $A \in \{0,1\}^*$, and a message $M \in \{0,1\}^*$, and it outputs a ciphertext $C \in \{0,1\}^{|M|}$ and a tag $T \in \{0,1\}^t$. Decryption dec gets as input a key $K \in \{0,1\}^k$, a nonce $N \in \{0,1\}^m$, associated data $A \in \{0,1\}^*$, a ciphertext $C \in \{0,1\}^*$, and a tag $T \in \{0,1\}^t$, and it outputs a message $M \in \{0,1\}^{|C|}$ if the tag is correct, or a dedicated $\perp$-sign otherwise. The two functions are required to satisfy

$$\mathsf{dec}(K, N, A, \mathsf{enc}(K, N, A, M)) = M\,.$$

In our work, the authenticated encryption scheme $\_$ is based on an n-bit permutation P, which is modeled as a random permutation: $\mathsf{P} \xleftarrow{\$} \mathrm{perm}(n)$. The security of $\_$ against an adversary $\mathcal{A}$ is defined as

$$\mathbf{Adv}^{\mathrm{ae}}_{\_}(\mathcal{A}) = \Delta_{\mathcal{A}}\left(\mathsf{enc}^{\mathsf{P}}_K, \mathsf{dec}^{\mathsf{P}}_K, \mathsf{P}^{\pm}\ ;\ \mathsf{rand}, \perp, \mathsf{P}^{\pm}\right)\,, \tag{3}$$

where the randomness of the oracles is taken over $K \xleftarrow{\$} \{0,1\}^k$, $\mathsf{P} \xleftarrow{\$} \mathrm{perm}(n)$, and the function rand that for each input (N, A, M) returns a random string of size $|M| + t$ bits. The superscript $^{\pm}$ indicates two-sided access by $\mathcal{A}$. The function $\perp$ returns the $\perp$-sign for each query.

We only consider *nonce-respecting* adversaries: $\mathcal{A}$ is not allowed to make two encryption queries for the same nonce. It is also not allowed to relay the output of the encryption oracle (enc_K in the real world and rand in the ideal world) to the decryption oracle (dec_K in the real world and $\perp$ in the ideal world).

2.2 Tweakable Block Ciphers

A tweakable block cipher $\widetilde{\mathsf{E}}$ is a function that gets as input a key $K \in \{0,1\}^k$, tweak $T \in \mathcal{T}$,[1] and message $M \in \{0,1\}^n$, and it outputs a ciphertext $C \in \{0,1\}^n$. The tweakable block cipher is required to be bijective for any fixed (K, T).

In our application, we will not make use of the inverse $\widetilde{\mathsf{E}}^{-1}$. More importantly, for our authenticated encryption scheme it suffices to use a tweakable block cipher that is secure against adversaries that only have access to $\widetilde{\mathsf{E}}$, and not to $\widetilde{\mathsf{E}}^{-1}$. The tweakable block cipher considered in this work is based on an n-bit permutation P, which is modeled as a random permutation: $\mathsf{P} \xleftarrow{\$} \mathrm{perm}(n)$. The security of $\widetilde{\mathsf{E}}$ against an adversary $\mathcal{A}$ is defined as

$$\mathbf{Adv}^{\mathrm{tprp}}_{\widetilde{\mathsf{E}}}(\mathcal{A}) = \Delta_{\mathcal{A}}\left(\widetilde{\mathsf{E}}^{\mathsf{P}}_K, \mathsf{P}^{\pm}\ ;\ \widetilde{\pi}, \mathsf{P}^{\pm}\right)\,, \tag{4}$$

where the randomness of the oracles is taken over $K \xleftarrow{\$} \{0,1\}^k$, $\mathsf{P} \xleftarrow{\$} \mathrm{perm}(n)$, and $\widetilde{\pi} \xleftarrow{\$} \mathrm{perm}(\mathcal{T}, n)$.

3 Simplified Masked Even-Mansour

The Elephant authenticated encryption family uses its underlying permutation in a "Masked Even-Mansour" (MEM) construction [GJMN16]: the input to and output of the permutation

[1] In our application, the tweak space is of a specific form and cannot be conveniently expressed as a set of binary strings.

P are masked using an LFSR evaluated on the secret key. However, the tweakable block cipher used in our proposal is simpler than the original construction in two ways: (i) the tweak only consists of the exponents of the LFSRs and not the nonce and (ii) in our application, the tweakable block cipher is only evaluated in the forward direction. The changes are not huge, but they do allow for a simpler description, security analysis, and bound. We will refer to this scheme as SiM (Simplified MEM). For generality, we will keep the formalization for an arbitrary amount of LFSRs, even though we will only use it for two LFSRs.

3.1 Specification

Let $k, n, z \in \mathbb{N}$. Let $\mathsf{P} \in \mathrm{perm}(n)$ be an n-bit permutation, and let $\varphi_1, \ldots, \varphi_z : \{0,1\}^n \to \{0,1\}^n$ be z LFSRs. Let $\mathcal{T} \subseteq \mathbb{N}^z$ be a finite tweak space. Define the function $\mathsf{mask} : \{0,1\}^k \times \mathcal{T} \to \{0,1\}^n$ as follows:

$$\mathsf{mask}_K^{a_1,\ldots,a_z} = \mathsf{mask}(K, a_1, \ldots, a_z) = \varphi_z^{a_z} \circ \cdots \circ \varphi_1^{a_1} \circ \mathsf{P}(K \| 0^{n-k}) \,. \tag{5}$$

We define the tweakable block cipher $\mathsf{SiM} : \{0,1\}^k \times \mathcal{T} \times \{0,1\}^n \to \{0,1\}^n$ as

$$\mathsf{SiM}(K, (a_1, \ldots, a_z), M) = \mathsf{P}(M \oplus \mathsf{mask}_K^{a_1,\ldots,a_z}) \oplus \mathsf{mask}_K^{a_1,\ldots,a_z} \,. \tag{6}$$

3.2 Security of SiM

We need a restriction on the tweak space $\mathcal{T}$ in order for SiM to be a secure tweakable block cipher. As Granger et al. [GJMN16], we say that $\mathcal{T}$ is $2^{-\alpha}$-proper with respect to $(\varphi_1, \ldots, \varphi_z)$ if the function $L \mapsto \varphi_z^{a_z} \circ \cdots \circ \varphi_1^{a_1}(L)$ is $2^{-\alpha}$-uniform and $2^{-\alpha}$-XOR-uniform.

Definition 1. Let $n, z \in \mathbb{N}$. Let $\varphi_1, \ldots, \varphi_z : \{0,1\}^n \to \{0,1\}^n$ be z LFSRs. The tweak space $\mathcal{T}$ is called $2^{-\alpha}$-*proper* with respect to $(\varphi_1, \ldots, \varphi_z)$ if the following two properties hold:

1. For any $Y \in \{0,1\}^n$ and $(a_1, \ldots, a_z) \in \mathcal{T} \cup \{(0, \ldots, 0)\}$,
$$\mathbf{Pr}\left(L \xleftarrow{\$} \{0,1\}^n \;:\; \varphi_z^{a_z} \circ \cdots \circ \varphi_1^{a_1}(L) = Y\right) \leq 2^{-\alpha} \,;$$
2. For any $Y \in \{0,1\}^n$ and distinct $(a_1, \ldots, a_z), (a'_1, \ldots, a'_z) \in \mathcal{T} \cup \{(0, \ldots, 0)\}$,
$$\mathbf{Pr}\left(L \xleftarrow{\$} \{0,1\}^n \;:\; \varphi_z^{a_z} \circ \cdots \circ \varphi_1^{a_1}(L) \oplus \varphi_z^{a'_z} \circ \cdots \circ \varphi_1^{a'_1}(L) = Y\right) \leq 2^{-\alpha} \,.$$

In Section 7, we will prove Theorem 1, which says that if the tweak space is $2^{-\alpha}$-proper for sufficiently small $2^{-\alpha}$ (note that $2^{-\alpha}$ cannot be smaller than 2^{-n}), then SiM is a secure tweakable block cipher. The proof is a direct simplification of Granger et al.'s analysis of MEM [GJMN16], due to the changes described in the introductory text of Section 3. These simplifications allow us to derive a slightly improved bound on the advantage, noting for comparison that Granger et al. [GJMN16] proved security up to $(4.5q^2 + 3qp)/2^\alpha + p/2^k$.

Theorem 1. *Let $k, n, z \in \mathbb{N}$. Let $\varphi_1, \ldots, \varphi_z : \{0,1\}^n \to \{0,1\}^n$ be z LFSRs, and let $\mathcal{T}$ be a $2^{-\alpha}$-proper tweak space with respect to $(\varphi_1, \ldots, \varphi_z)$. Consider SiM of (6) based on random permutation $\mathsf{P} \xleftarrow{\$} \mathrm{perm}(n)$. For any adversary $\mathcal{A}$ making at most $q \leq 2^{n-1}$ construction queries and p primitive queries,*

$$\mathbf{Adv}_{\mathsf{SiM}}^{\mathrm{tprp}}(\mathcal{A}) \leq \frac{q^2 + 2qp}{2^\alpha} + \frac{2q + p}{2^n} + \frac{p}{2^k} \,.$$

The proof is given in Section 7.

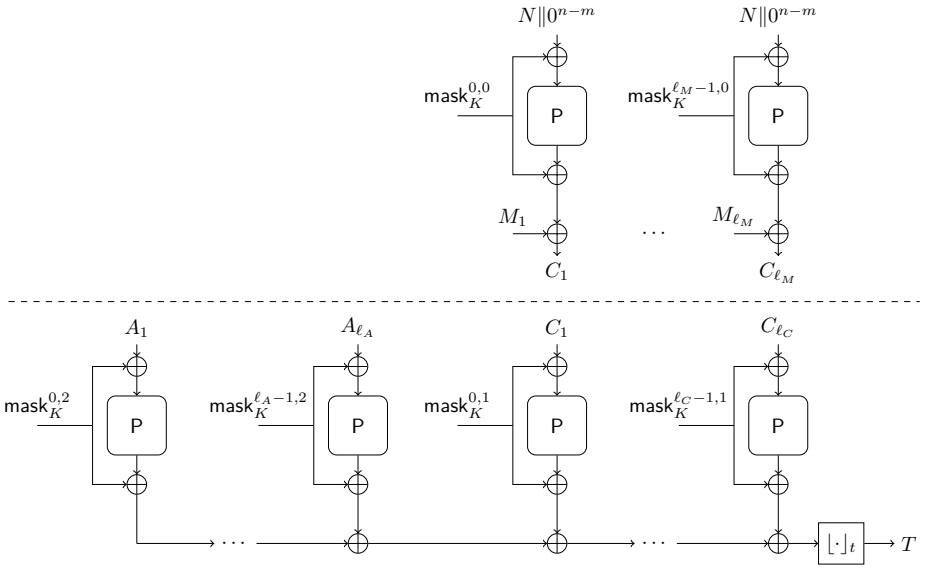

Figure 1: Depiction of Elephant. For the encryption part (top): message is padded as $M_1 \dots M_{\ell_M} \xleftarrow{n} M$, and ciphertext equals $C = \lfloor C_1 \dots C_{\ell_M} \rfloor_{|M|}$. For the authentication part (bottom): nonce and associated data are padded as $A_1 \dots A_{\ell_A} \xleftarrow{n} N\|A\|1$, and ciphertext is padded as $C_1 \dots C_{\ell_C} \xleftarrow{n} C\|1$.

4 Elephant Authenticated Encryption

The Elephant authenticated encryption mode is specified in Section 4.1, and it is proven to be secure relative to the tweakable block cipher security of SiM in Section 4.2.

4.1 Specification

Let $k, m, n, t \in \mathbb{N}$ with $k, m, t \leq n$. Let $\mathsf{P} : \{0,1\}^n \to \{0,1\}^n$ be an n-bit permutation, and $\varphi_1 : \{0,1\}^n \to \{0,1\}^n$ be an LFSR. Define $\varphi_2 = \varphi_1 \oplus \mathsf{id}$, where id is the identity function. Define the function $\mathsf{mask} : \{0,1\}^k \times \mathbb{N}^2 \to \{0,1\}^n$ as follows:

$$\mathsf{mask}_K^{a,b} = \mathsf{mask}(K, a, b) = \varphi_2^b \circ \varphi_1^a \circ \mathsf{P}(K\|0^{n-k})\,. \tag{7}$$

We will describe the generic authenticated encryption mode of Elephant. It consists of two algorithms: encryption enc and decryption dec.

4.1.1 Encryption

Encryption enc gets as input a key $K \in \{0,1\}^k$, a nonce $N \in \{0,1\}^m$, associated data $A \in \{0,1\}^*$, and a message $M \in \{0,1\}^*$, and it outputs a ciphertext $C \in \{0,1\}^{|M|}$ and a tag $T \in \{0,1\}^t$. The description of enc is given in Algorithm 1, and it is depicted in Figure 1.

4.1.2 Decryption

Decryption dec gets as input a key $K \in \{0,1\}^k$, a nonce $N \in \{0,1\}^m$, associated data $A \in \{0,1\}^*$, a ciphertext $C \in \{0,1\}^*$, and a tag $T \in \{0,1\}^t$, and it outputs a message $M \in \{0,1\}^{|M|}$ if the tag is correct, or a dedicated $\perp$-sign otherwise. The description of dec is given in Algorithm 2.

Algorithm 1 Elephant encryption algorithm enc

Input: $(K, N, A, M) \in \{0,1\}^k \times \{0,1\}^m \times \{0,1\}^* \times \{0,1\}^*$
Output: $(C, T) \in \{0,1\}^{|M|} \times \{0,1\}^t$
$M_1 \dots M_{\ell_M} \xleftarrow{n} M$
for $i = 1, \dots, \ell_M$ **do**
  $C_i \leftarrow M_i \oplus \mathsf{P}(N \| 0^{n-m} \oplus \mathsf{mask}_K^{i-1,0}) \oplus \mathsf{mask}_K^{i-1,0}$
$C \leftarrow \lfloor C_1 \dots C_{\ell_M} \rfloor_{|M|}$
$T = 0$
$A_1 \dots A_{\ell_A} \xleftarrow{n} N \| A \| 1$
$C_1 \dots C_{\ell_C} \xleftarrow{n} C \| 1$
for $i = 1, \dots, \ell_A$ **do**
  $T \leftarrow T \oplus \mathsf{P}(A_i \oplus \mathsf{mask}_K^{i-1,2}) \oplus \mathsf{mask}_K^{i-1,2}$
for $i = 1, \dots, \ell_C$ **do**
  $T \leftarrow T \oplus \mathsf{P}(C_i \oplus \mathsf{mask}_K^{i-1,1}) \oplus \mathsf{mask}_K^{i-1,1}$
return $(C, \lfloor T \rfloor_t)$

Algorithm 2 Elephant decryption algorithm dec

Input: $(K, N, A, C, T) \in \{0,1\}^k \times \{0,1\}^m \times \{0,1\}^* \times \{0,1\}^* \times \{0,1\}^t$
Output: $M \in \{0,1\}^{|C|}$ or $\perp$
$C_1 \dots C_{\ell_M} \xleftarrow{n} C$
for $i = 1, \dots, \ell_M$ **do**
  $M_i \leftarrow C_i \oplus \mathsf{P}(N \| 0^{n-m} \oplus \mathsf{mask}_K^{i-1,0}) \oplus \mathsf{mask}_K^{i-1,0}$
$M \leftarrow \lfloor M_1 \dots M_{\ell_M} \rfloor_{|C|}$
$\bar{T} = 0$
$A_1 \dots A_{\ell_A} \xleftarrow{n} N \| A \| 1$
$C_1 \dots C_{\ell_C} \xleftarrow{n} C \| 1$
for $i = 1, \dots, \ell_A$ **do**
  $\bar{T} \leftarrow \bar{T} \oplus \mathsf{P}(A_i \oplus \mathsf{mask}_K^{i-1,2}) \oplus \mathsf{mask}_K^{i-1,2}$
for $i = 1, \dots, \ell_C$ **do**
  $\bar{T} \leftarrow \bar{T} \oplus \mathsf{P}(C_i \oplus \mathsf{mask}_K^{i-1,1}) \oplus \mathsf{mask}_K^{i-1,1}$
return $\lfloor \bar{T} \rfloor_t = T \ ? \ M \ : \ \perp$

4.2 Security of Elephant

We will prove security of Elephant of Section 4.1 for any $2^{-\alpha}$-proper tweak space. The specific choice of tweak space will be discussed in Section 5.

Theorem 2. *Let $k, m, n, t \in \mathbb{N}$ with $k, m, t \leq n$. Let $\varphi_1, \varphi_2 : \{0,1\}^n \to \{0,1\}^n$ be LFSRs, and let $\mathcal{T}$ be a $2^{-\alpha}$-proper tweak space with respect to (φ_1, φ_2). Consider* Elephant $= (\mathsf{enc}, \mathsf{dec})$ *of Section 4.1 based on random permutation $\mathsf{P} \xleftarrow{\$} \mathrm{perm}(n)$. For any adversary $\mathcal{A}$ making at most $q_e \leq 2^{n-1}$ construction encryption queries, q_d construction decryption queries, each query at most ℓ padded nonce and associated data and message blocks, and in total at most σ padded nonce and associated data and message blocks, and p primitive queries,*

$$\mathbf{Adv}^{\mathrm{ae}}_{\mathsf{Elephant}}(\mathcal{A}) \leq \ell \binom{q_e}{2} / 2^n + \frac{2^{n-t} q_d}{2^n - 1} e^{(q_e+1)q_e/2^n} + \mathbf{Adv}^{\mathrm{tprp}}_{\mathsf{SiM}}(\mathcal{A}'),$$

for some $\mathcal{A}'$ that makes 2σ construction queries and p primitive queries.

The proof is given in Section 8.

5 Instantiation

While it is possible to instantiate our scheme with any permutation, we aimed for permutations that have a lightweight footprint in either hardware or software. Hence, for our three instances we rely on well-established permutations operating on as small as possible state sizes, in order to still fulfill the security goals recommended by the NIST lightweight call [Nat18] of having at least 112-bit security provided that the online complexity is at most around 2^{50} bytes. We propose the three instances given in Table 1. The instances are further elaborated on in Sections 5.1, 5.2, and 5.3, respectively.

Table 1: Instances of Elephant.

instance	k	m	n	t	P	φ_1
Dumbo	128	96	160	64	Spongent-π[160]	(8)
Jumbo	128	96	176	64	Spongent-π[176]	(9)
Delirium	128	96	200	128	Keccak-f[200]	(10)

5.1 Dumbo: 160-Bit Elephant

The 160-bit instance of Elephant is based on the Spongent-π[160] permutation [BKL+11]. The choice for Spongent is natural: it is particularly well-suited for hardware, and the existing third-party analysis [Abd12,ZL17,HKS18,ZBRL15] does not indicate any weakness of the Spongent family relevant for our use-case. We have used the 160-bit version of Spongent as this is the smallest possible permutation that can be used to efficiently[2] to meet the NIST call for proposals.

Bogdanov et al. [BKL+11] do not explicitly specify the number of rounds of the 160-bit version of the Spongent permutation; we opt for 80 rounds since this ensures that at least 160 S-boxes are differentially active. This is in accordance with the Spongent design strategy. Note further that this implies that the 7-bit LFSR specified in [BKL+11] should be used (with initial value `0x75`) to generate the round constants for the permutation.

For generating the masks of our scheme, we use the approach of Granger et al. [GJMN16]. We define φ_1 as the following $\mathbb{F}_2$-linear map, where the x_i's correspond to 8-bit words:

$$(x_0, \ldots, x_{19}) \mapsto (x_1, \ldots, x_{19}, x_0 \lll 3 \oplus x_3 \ll 7 \oplus x_{13} \gg 7)\,. \tag{8}$$

This LFSR aims to minimize the area required when implemented in hardware. In particular, in addition to the shift register, only two 2-bit XOR gates are needed. Hence, this choice of the LFSRs is in line with the strength of the Spongent permutations, making a perfect match for small area hardware implementations. Despite the particular suitability of both LFSRs for small area hardware implementations, it is still possible to implement them rather efficiently on 8-bit platforms.

We will prove that the 160-bit LFSR defined by (8) has maximal length, and that the tweak space used in Elephant with this LFSR is 2^{-n}-proper with respect to (φ_1, φ_2).

Proposition 1. *Let $n = 160$. Let $\varphi_1 : \{0,1\}^{160} \mapsto \{0,1\}^{160}$ be the LSFR given in (8), and $\varphi_2 = \varphi_1 \oplus \mathsf{id}$. The tweak space $\mathcal{T} = \mathcal{T}_1 \times \mathcal{T}_2 = \{0, 1, \ldots, 2^{154}\} \times \{0, 1, 2\}$ is 2^{-n}-proper with respect to (φ_1, φ_2).*

Proof. The proof is almost identical to [GJMN16, Lemma 4], with the main difference that a different discrete logarithm must be computed. Let V be the 160×160 matrix over $\mathbb{F}_2$

[2] Beyond birthday bound solutions may use even smaller permutations, but only at an efficiency penalty.

that represents φ_1 of (8). As shown in [GJMN16, Lemma 3], $\varphi_1^i(L) = V^i \cdot L$ is 2^{-n}-proper for $i \in \{0, \dots 2^n - 2\}$ if the minimal polynomial of V is primitive and of degree n. A quick computation using Sage [The17] shows that this polynomial

$$p(x) = x^{160} + x^{136} + x^{83} + x^{53} + 1$$

is irreducible and primitive.

Next, let $\ell = \log_x(x+1)$ in the field $\mathbb{F}_2[x]/p(x)$. We have to show that $\varphi_2^b \circ \varphi_1^a(L) = (V+I)^b \cdot V^a \cdot L = V^{\ell \cdot b} \cdot V^a \cdot L$ is unique for any distinct set of tweaks. A simple Sage computation gives the following values for ℓ and 2ℓ:

$$\begin{aligned} \ell &= 74280011654209447488264356271465075847453668488 9 \approx 2^{159.02}\,, \\ 2\ell &= \ \ 24098595753286031561602292713018497293140826803 \approx 2^{154.08}\,. \end{aligned}$$

If we consider that $b \in \{0,1,2\}$ divides the tweak space into three sets, the smallest difference is between the set with $b = 0$ and the set corresponding to $b = 2$, which is bigger than 2^{154}. Hence, by ensuring that $0 \leq a \leq 2^{154}$, we have that for any two distinct $(a,b),(a',b') \in \{0,1,\dots,2^{154}\} \times \{0,1,2\}$, $\varphi_2^b \circ \varphi_1^a \neq \varphi_2^{b'} \circ \varphi_1^{a'}$.

Finally, using both of the above observations, one can easily observe that $\mathcal{T}$ is 2^{-n}-proper in light of Definition 1. □

We directly obtain that Dumbo is secure in the random permutation model.

Corollary 1. *Let* $(k,m,n,t) = (128,96,160,64)$. *Let* $\mathcal{T} = \{0,1,\dots,2^{154}\} \times \{0,1,2\}$. *Consider* Dumbo*:* Elephant $=$ (enc, dec) *of Section 4.1 based on the permutation* Spongent-$\pi[160]$*, modeled as a random* 160*-bit permutation, and on* $\varphi_1 : \{0,1\}^{160} \to \{0,1\}^{160}$ *of (8). For any adversary* $\mathcal{A}$ *making at most* q_e *construction encryption queries,* q_d *construction decryption queries, each query at most* ℓ *padded nonce and associated data and message blocks, and in total at most* $\sigma \leq 2^{158}$ *padded nonce and associated data and message blocks, and* p *primitive queries,*

$$\mathbf{Adv}^{\mathrm{ae}}_{\mathsf{Dumbo}}(\mathcal{A}) \leq \ell \binom{q_e}{2} / 2^{160} + \frac{2^{96} q_d}{2^{160}-1} e^{(q_e+1)q_e/2^{160}} + \frac{4\sigma^2 + 4\sigma p + 4\sigma + p}{2^{160}} + \frac{p}{2^{128}}\,,$$

Recall that NIST's call for lightweight authenticated encryption schemes [Nat18] requested security up to an online complexity of around 2^{50} bytes. By limiting the total online complexity σ to $2^{50}/(n/8)$ blocks, the bound of Corollary 1 is at most 1 for $p \leq 2^{112}$.

5.2 Jumbo: 176-Bit Elephant

The 176-bit instance of Elephant is also based on a Spongent permutation, namely Spongent-$\pi[176]$ [BKL+11]. It has the same features as Spongent-$\pi[160]$ (see Section 5.1), but offers a slightly more comfortable 127-bit security margin. In addition, this particular Spongent permutation is part of the ISO/IEC standard on lightweight hash functions [ISO16].

For generating the masks of our scheme, we use the approach of Granger et al. [GJMN16]. The LFSR φ_1 is defined as the following $\mathbb{F}_2$-linear map, where the x_i's correspond to 8-bit words:

$$(x_0, \dots, x_{21}) \mapsto (x_1, \dots, x_{21}, x_0 \lll 1 \oplus x_3 \ll 7 \oplus x_{19} \gg 7)\,. \tag{9}$$

This LFSR has the same advantages and implementation features as the 160-bit LFSR of (8) in Section 5.1.

We will prove that the 176-bit LFSR defined by (9) has maximal length, and that the tweak space used in Elephant with this LFSR is 2^{-n}-proper with respect to (φ_1, φ_2).

Proposition 2. *Let $n = 176$. Let $\varphi_1 : \{0,1\}^{176} \mapsto \{0,1\}^{176}$ be the LSFR given in (9), and $\varphi_2 = \varphi_1 \oplus \mathsf{id}$. The tweak space $\mathcal{T} = \mathcal{T}_1 \times \mathcal{T}_2 = \{0,1,\ldots,2^{173}\} \times \{0,1,2\}$ is 2^{-n}-proper with respect to (φ_1, φ_2).*

Proof. The proof is identical to that of Proposition 1, with the difference that for the 176×176 matrix V that represents φ_1 of (9), the corresponding polynomial

$$p(x) = x^{176} + x^{154} + x^{135} + x^{19} + 1$$

is irreducible and primitive. The discrete logarithm $\ell = \log_x(x+1)$ in the field $\mathbb{F}_2[x]/p(x)$ and its related 2ℓ are computed as

$$\ell = 188813761514037867774814634320294502941004615622206 99 \approx 2^{173.66},$$
$$2\ell = 377627523028075735549629268640589005882009231244413 98 \approx 2^{174.66}.$$

Again, dividing the tweak space into 3 sets according to the value $b \in \{0,1,2\}$, the smallest difference is between set $b = 0$ and set $b = 1$, or $b = 1$ and $b = 2$, which is bigger than 2^{173}. Hence, by ensuring that $0 \leq a \leq 2^{173}$, we have that for any two distinct $(a,b),(a',b') \in \{0,1,\ldots,2^{173}\} \times \{0,1,2\}$, $\varphi_2^b \circ \varphi_1^a \neq \varphi_2^{b'} \circ \varphi_1^{a'}$. □

We directly obtain that Jumbo is secure in the random permutation model.

Corollary 2. *Let $(k,m,n,t) = (128,96,176,64)$. Let $\mathcal{T} = \{0,1,\ldots,2^{173}\} \times \{0,1,2\}$. Consider* Jumbo*: Elephant $= (\mathsf{enc},\mathsf{dec})$ of Section 4.1 based on the permutation* Spongent-$\pi[176]$*, modeled as a random 176-bit permutation, and on $\varphi_1 : \{0,1\}^{176} \to \{0,1\}^{176}$ of (9). For any adversary $\mathcal{A}$ making at most q_e construction encryption queries, q_d construction decryption queries, each query at most ℓ padded nonce and associated data and message blocks, and in total at most $\sigma \leq 2^{174}$ padded nonce and associated data and message blocks, and p primitive queries,*

$$\mathbf{Adv}^{\mathrm{ae}}_{\mathsf{Jumbo}}(\mathcal{A}) \leq \ell \binom{q_e}{2} / 2^{176} + \frac{2^{112} q_d}{2^{176}-1} e^{(q_e+1)q_e/2^{176}} + \frac{4\sigma^2 + 4\sigma p + 4\sigma + p}{2^{176}} + \frac{p}{2^{128}},$$

As before, limiting the total online complexity σ to $2^{50}/(n/8)$ blocks, the bound of Corollary 2 is at most 1 for $p \leq 2^{127}$.

5.3 Delirium: 200-Bit Elephant

The 200-bit instance of Elephant is based on the Keccak-$f[200]$ permutation [BDPV11b]. The 200-bit instance is the smallest of the instances that is specified in the NIST standard [FIP15] that fits our need; it is still reasonable in hardware, and particularly good in software on 8-bit platforms, considering that it is naturally defined using 8-bit lanes [BDP+12, KY10]. As such, it is complementary to the Spongent-based instantiation of Elephant.

For generating the masks of our scheme, we use the approach of Granger et al. [GJMN16]. The LFSR φ_1 is now defined as the following $\mathbb{F}_2$-linear map, where the x_i's correspond to 8-bit words:

$$(x_0,\ldots,x_{24}) \mapsto (x_1,\ldots,x_{24},x_0 \lll 1 \oplus x_2 \lll 1 \oplus x_{13} \ll 1). \tag{10}$$

This LFSR shows its full potential when implemented on 8-bit platforms. A state update within the LFSR just updates one byte, while the content of the other 24 bytes is not changed and basically just relabeled. The single updated byte is computed as the XOR sum of 3 bytes other state bytes that are just rotated or shifted by one bit position. Hence, the essential operations that have to be performed on 8-bit platforms are 3 XOR operations, two rotations by one bit to the left plus one shift by one bit to the left.

We will prove that the 200-bit LFSR defined by (10) has maximal length, and that the tweak space used in Elephant with this LFSR is 2^{-n}-proper with respect to (φ_1, φ_2).

Proposition 3. *Let $n = 200$. Let $\varphi_1 : \{0,1\}^{200} \mapsto \{0,1\}^{200}$ be the LSFR given in (10), and $\varphi_2 = \varphi_1 \oplus \mathsf{id}$. The tweak space $\mathcal{T} = \mathcal{T}_1 \times \mathcal{T}_2 = \{0,1,\ldots,2^{197}\} \times \{0,1,2\}$ is 2^{-n}-proper with respect to (φ_1, φ_2).*

Proof. The proof is identical to that of Proposition 1, with the difference that for the 200×200 matrix V that represents φ_1 of (10), the corresponding polynomial

$$\begin{aligned} p(x) = x^{200} &+ x^{93} + x^{91} + x^{82} + x^{78} + x^{71} + x^{69} + x^{67} + x^{65} \\ &+ x^{60} + x^{52} + x^{49} + x^{47} + x^{41} + x^{39} + x^{38} + x^{34} + x^{30} + x^{27} \\ &+ x^{26} + x^{25} + x^{23} + x^{21} + x^{19} + x^{17} + x^{16} + x^{15} + x^{13} + 1 \end{aligned}$$

is irreducible and primitive. The discrete log $\ell = \log_x(x+1)$ in the field $\mathbb{F}_2[x]/p(x)$ and its related 2ℓ are computed as

$$\begin{aligned} \ell &= \ 692180606625676931900534627786122994390018641930530681719698 \approx 2^{198.78}\,, \\ 2\ell &= 1384361213251353863801069255572245988780037283861061363439396 \approx 2^{199.78}\,. \end{aligned}$$

Again, dividing the tweak space into 3 sets according to the value $b \in \{0,1,2\}$, the smallest difference is between set $b = 2$ and set $b = 0$, which is bigger than 2^{197}. Hence, by ensuring that $0 \leq a \leq 2^{197}$, we have that for any two distinct $(a,b),(a',b') \in \{0,1,\ldots,2^{197}\} \times \{0,1,2\}$, $\varphi_2^b \circ \varphi_1^a \neq \varphi_2^{b'} \circ \varphi_1^{a'}$. □

We directly obtain that Delirium is secure in the random permutation model.

Corollary 3. *Let $(k,m,n,t) = (128,96,200,128)$. Let $\mathcal{T} = \{0,1,\ldots,2^{197}\} \times \{0,1,2\}$. Consider* Delirium*: Elephant $= (\mathsf{enc},\mathsf{dec})$ of Section 4.1 based on the permutation* Keccak-$f[200]$*, modeled as a random 200-bit permutation, and on $\varphi_1 : \{0,1\}^{200} \to \{0,1\}^{200}$ of (10). For any adversary $\mathcal{A}$ making at most q_e construction encryption queries, q_d construction decryption queries, each query at most ℓ padded nonce and associated data and message blocks, and in total at most $\sigma \leq 2^{198}$ padded nonce and associated data and message blocks, and p primitive queries,*

$$\mathbf{Adv}^{\mathrm{ae}}_{\mathsf{Delirium}}(\mathcal{A}) \leq \ell \binom{q_e}{2} / 2^{200} + \frac{2^{72} q_d}{2^{200}-1} e^{(q_e+1)q_e/2^{200}} + \frac{4\sigma^2 + 4\sigma p + 4\sigma + p}{2^{200}} + \frac{p}{2^{128}}\,,$$

As before, limiting the total online complexity σ to $2^{74}/(n/8)$ blocks, the bound of Corollary 3 is at most 1 for $p \leq 2^{127}$.

6 Design Rationale

The Elephant mode is an encrypt-then-MAC mode, where encryption is performed by counter mode and message authentication by a variant of Wegman-Carter-Shoup [WC81, Sho96], both implicitly instantiated using a simplification of the masked Even-Mansour (MEM) tweakable block cipher of Granger et al. [GJMN16]. We explain the design rationale of Elephant at the following two levels of granularity: the generic mode in Section 6.1, and how the mode uses the permutation, i.e., the masking scheme, in Section 6.2. Finally, Section 6.3 briefly discusses implementation aspects.

6.1 Mode

Generically, encrypt-then-MAC is the most secure approach [BN00, NRS14]: unlike its alternatives encrypt-and-MAC and MAC-then-encrypt, this approach yields integrity of ciphertexts. Stated differently, malformed ciphertexts yield failure upon MAC verification,

and for these no decryption is needed. This prevents unintended leakage from verification failures. The approach also makes it possible to easily prevent leakage due to release of unverified plaintext: simply do not start decrypting before the tag is verified. Note that for the generic alternatives encrypt-and-MAC and MAC-then-encrypt, such a simple countermeasure is impossible. This makes the encrypt-then-MAC mode of Elephant preferable over its alternatives, not only in the lightweight setting but also for general purpose.

The counter encryption mode and Wegman-Carter-Shoup MAC mode within Elephant, in turn, are both fully parallelizable and only evaluate the underlying permutation P in forward direction. The fact that Elephant evaluates its primitive in forward direction is important in the lightweight setting: it allows for smaller implementations, since there is no need to implement the inverse of P. Note, in particular, that due to the rise of the sponge, various cryptographic permutations, including Ascon [DEMS16], Gimli [BKL+17], Keccak [BDPV11b], and XOODOO [DHVV18], are developed to be particularly efficient in forward direction.

By being parallelizable, Elephant distinguishes itself from a wide range of authenticated encryption schemes that employ a serial permutation-based mode of operation, such as APE [ABB+14], Beetle [CDNY18], or the Duplex construction [BDPV11a,MRV15,DMV17]. To support parallelism, we need to store the internal state value, but on the upside, it turns out to give various elegant implementation advantages (see Section 6.2 and Section 6.3) and it means that there is no strict need to employ larger permutations.

We briefly elaborate on existing generic authenticated encryption schemes that are both parallel and permutation-based (but not necessarily inverse-free). Granger et al. [GJMN16] introduced OPP, a parallel and permutation-based scheme derived from ΘCB [KR11], but it is not inverse-free. Minalpher [STA+15], likewise, is parallel and permutation-based but not inverse-free. Finally, a permutation-based version of OTR [Min16] exists in the embodiment of Prøst-OTR [KLL+14]. This construction is parallel, permutation-based, and inverse free, just like Elephant. However, because it processes pairs of message blocks using a two-round Feistel structure, the encryption process differs depending on the parity of the number of message blocks. This stands in contrast to the conceptual simplicity of Elephant. In addition, for short messages, less parallelism is available in Prøst than for Elephant. If the implementation maximally exploits parallelism, Elephant would compare favorably for short messages in terms of latency.

The mode is nonce-based: each of the members of Elephant uses a 96-bit nonce. The nonce is prepended to the associated data, which is then padded and split into n-bit blocks $A_1 \dots A_{\ell_A}$ (see line 6 of Algorithm 1). This way, the scheme is optimized for the parameters specified in the NIST call [Nat18]: the nonce is 96 bits, and in order to avoid a waste of $n - 96$ bits due to padding (where $n \in \{160, 176, 200\}$), the nonce is appended with the first $n - 96$ bits of the associated data. Caution must be paid here, namely that the *nonce is always of fixed length of* 96 *bits*. If variable-length nonces were allowed, the scheme would be vulnerable to trivial padding attacks. We remark that it is theoretically possible to adjust the Elephant mode to allow longer nonces or flexible-length nonces, but we discourage this as it might lead to error-prone designs. Furthermore, we clarify that the nonce is used *both* for encryption and for authentication: the former is needed for confidentiality and the latter is needed in case of authenticated encryption of an empty message. Also, as the mode is nonce-based, security is guaranteed *only if* the adversary does not repeat nonces for encryption queries.

6.2 Masking

As specified in Section 4.1, the inputs to and outputs of the permutation P are masked using $\mathsf{mask}_K^{a,b}$ of (7). The masking function is defined using two LFSRs $\varphi_1, \varphi_2 : \{0,1\}^n \to \{0,1\}^n$ that satisfy $\varphi_2 = \varphi_1 \oplus \mathsf{id}$, and it is parameterized by (a, b) which are used in a manner so as to

assure that every occurrence of the masking in the Elephant mode gets different parameters. We have heuristically chosen our LFSRs to give a good match when used in combination with the particular permutations. For the LFSR's matching Spongent, we selected versions that have a small gate count in hardware. In the case of the 200-bit Keccak permutation, we chose an LFSR that can be implemented with a small number of instructions. Hence, we selected an LFSR that allows for implementations with shift/rotation by one. The number of gates needed for a hardware implementation was a secondary consideration in this case.

The LFSR-based masking technique is taken from Granger et al. [GJMN16], and so is the security analysis (although different state sizes, discrete logarithm computations, LFSRs, and tweak domains are considered). Granger et al. have argued in favor of this technique over its alternatives for various reasons: (i) the approach is simpler to implement, as the masking is purely linear and does not use finite field multiplication, (ii) it is more efficient (depending on the primitive used), and (iii) the masking is constant time.

The latter point is important in the lightweight setting where resistance against timing attacks comes at a cost. In this respect, the LFSR-based masking approach compares favorably with another, and very popular, masking technique, namely powering-up-based masking (simplified to allow for fair comparison with (7)):

$$3^b 2^a \mathsf{P}(K \| 0^{n-k}) \,,$$

where 2 and 3 are coordinates in the monomial basis in the finite field $\mathbb{F}_{2^n}$. The technique was introduced by Rogaway [Rog04] in the context of OCB2, and it has seen many applications, including CAESAR submissions AES-OTR [Min16], AEZ [HKR17], COLM [ABD+16], Minalpher [STA+15], POET [AFF+15], and SHELL [Wan15]. These multiplications can be implemented as an LFSR on one-bit words, but the masking functions φ_1 and φ_2 are constant time by design and allow for more flexibility in the word size.

A related masking approach is that of OCB3 [KR11] and OMD [CMN+15], which use masking based on Gray coding. In detail, Gray coding-based masks can be updated as $G(i) = G(i-1) \oplus 2^{\mathsf{ntz}(i)}$, were $\mathsf{ntz}(i)$ is the number of trailing zeros in the binary representation of i. The masking, unlike powering-up, does not need a conditional XOR, but it requires $\log_2(i)$ field doublings (which may be precomputed). As the LFSR-based masking used in Elephant does not incur such a cost, it also compares favorably with this technique.

The particular choice of masking, namely $(a, b) = (i, 0)$ in the encryption layer, $(a, b) = (i, 1)$ for ciphertext authentication, and $(a, b) = (i, 2)$ for associated data authentication, allows maskings to cancel out nicely in the implementation. To see this, consider the authentication of ciphertext C_i (for $i < \ell_M \leq \ell_C$), and more detailed the contribution T_i it makes to tag T. This value is computed as

$$T_i = \mathsf{P}\left(M_i \oplus \mathsf{P}(N \| 0^{n-m} \oplus \mathsf{mask}_K^{i-1,0}) \oplus \mathsf{mask}_K^{i-1,0} \oplus \mathsf{mask}_K^{i-1,1}\right) \oplus \mathsf{mask}_K^{i-1,1} \,.$$

By definition of $\mathsf{mask}_K^{a,b}$, and as $\varphi_2 = \varphi_1 \oplus \mathsf{id}$, we have

$$\begin{aligned} \mathsf{mask}_K^{i-1,0} \oplus \mathsf{mask}_K^{i-1,1} &= \varphi_1^{i-1} \circ \mathsf{P}(K \| 0^{n-k}) \oplus (\varphi_1 \oplus \mathsf{id}) \circ \varphi_1^{i-1} \circ \mathsf{P}(K \| 0^{n-k}) \\ &= \varphi_1^{i} \circ \mathsf{P}(K \| 0^{n-k}) \,. \end{aligned}$$

This, not surprisingly, is the mask used for the encryption of the next message block M_{i+1}. We note that exploiting this requires extra state.

Another optimization in mask management is in the masks that contribute to the tag, i.e., the sum of all masks that appear in the final tag T. The contribution coming from

the ciphertext authentication equals

$$\left(\bigoplus_{i=1}^{\ell_C} \mathsf{mask}_K^{i-1,1}\right) = \left(\bigoplus_{i=1}^{\ell_C} (\varphi_1 \oplus \mathsf{id}) \circ \varphi_1^{i-1} \circ \mathsf{P}(K\|0^{n-k})\right) = (\varphi_1^{\ell_C} \oplus \mathsf{id}) \circ \mathsf{P}(K\|0^{n-k})\,, \tag{11}$$

and that coming from the associated data likewise equals

$$\left(\bigoplus_{i=1}^{\ell_A} \mathsf{mask}_K^{i-1,2}\right) = (\varphi_1^{\ell_A+1} \oplus \varphi_1^{\ell_A} \oplus \varphi_1 \oplus \mathsf{id}) \circ \mathsf{P}(K\|0^{n-k})\,. \tag{12}$$

This feature of the masking may be useful if Elephant is used for fixed-length data, in which case the (11) and (12) could be precomputed.

6.3 Implementation

As discussed in Section 6.1, the Elephant mode allows for a high degree of parallelism. For the hardware-oriented variants of Elephant (Dumbo and Jumbo), this makes it easy to trade-off area for additional throughput. Hardware implementations of the 176-bit Spongent permutation are given by Bogdanov et al. [BKL+11], e.g., just needing 1329 GE to implement the Spongent-160 hash function, which is based on the 176-bit Spongent permutation. The 200-bit variant of Elephant primarily targets (embedded) software, but the same remarks concerning hardware implementations apply as, e.g., demonstrated by an implementation of a hash function based on the 200-bit Keccak permutation needing just 2520 GE by Kavun and Yalçin [KY10].

Software implementations of 200-bit Elephant (Delirium) can also exploit parallelism. If multiple cores are available, several blocks can be processed concurrently – but this is only useful for long messages. More importantly, on processors with a word size above 16 bits, the available parallelism makes it possible to increase the efficiency of the implementation by combining two or more calls to the Keccak permutation. For mid- and high-end processors with SIMD instructions, the same technique can be used to obtain even greater speed-ups.

An increasingly common requirement is the ability to protect implementations against side-channel attacks. As discussed in Section 6.2, the masking scheme is constant time by design. The same applies to the Spongent and Keccak permutations. In addition, all variants of Elephant are well-suited for Boolean masking techniques such as threshold implementations [NRR06].

Finally, it is worth mentioning that a few specific use-cases of Elephant allow for additional optimizations. As discussed in Section 6.2, the contribution of the mask values to the tag can be precomputed for fixed-length messages. In addition, if one or more blocks of associated data are static, it is possible to precompute their contribution to the tag – with the exception of the first block, which involves the nonce.

A reference implementation of Dumbo, Jumbo, and Delirium written in C99 can be found at https://github.com/TimBeyne/Elephant.

7 Proof of Theorem 1 (on SiM)

The proof closely follows Granger et al. [GJMN16] and is performed using the H-coefficient technique [Pat08, CS14].

Let $K \xleftarrow{\$} \{0,1\}^k$, $\mathsf{P} \xleftarrow{\$} \mathrm{perm}(n)$, and $\widetilde{\pi} \xleftarrow{\$} \mathrm{perm}(\mathcal{T}, n)$, where $\mathcal{T}$ is $2^{-\alpha}$-proper with respect to LFSRs $(\varphi_1, \ldots, \varphi_z)$. Consider a computationally unbounded adversary $\mathcal{A}$ that

tries to distinguish $\mathcal{O} := (\widetilde{\mathsf{E}}_K^{\mathsf{P}}, \mathsf{P}^{\pm})$ from $\mathcal{P} := (\widetilde{\pi}, \mathsf{P}^{\pm})$. Without loss of generality, we can consider it to be deterministic: for any probabilistic adversary there exists a deterministic one that has at least the same success probability. The interaction of $\mathcal{A}$ with its oracle ($\mathcal{O}$ or $\mathcal{P}$) is gathered in a view ν. Denote by $D_{\mathcal{O}}$ (resp., $D_{\mathcal{P}}$) the probability distribution of views in interaction with $\mathcal{O}$ (resp., $\mathcal{P}$). Denote by $\mathcal{V}$ the set of "attainable views", i.e., views ν such that $\mathbf{Pr}\left(D_{\mathcal{P}} = \nu\right) > 0$.

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

$$\Delta_{\mathcal{A}}\left(\mathcal{O}\ ;\ \mathcal{P}\right) \leq \varepsilon + \mathbf{Pr}\left(D_{\mathcal{P}} \in \mathcal{V}_{\text{bad}}\right). \tag{13}$$

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

The remainder of the proof is structured as follows. We specify the views of an adversary in Section 7.1 and define the bad views in Section 7.2. The probability of bad views is analyzed in Section 7.3 and the probability ratio for good views is considered in Section 7.4. Section 7.5 concludes the proof.

7.1 Views

The adversary can make q construction queries to $\widetilde{\mathsf{E}}_K^{\mathsf{P}}$ or $\widetilde{\pi}$, all *in forward direction only*. Each such query is made for some tweak $\bar{a}_i = (a_1, \ldots, a_z)_i$ and message input M_i, and results in an output C_i. The q queries are summarized in a view

$$\nu_c = \{(\bar{a}_1, M_1, C_1), \ldots, (\bar{a}_q, M_q, C_q)\}.$$

The adversary can make p primitive queries to $\mathsf{P}^{\pm}$, and these are likewise summarized in a view

$$\nu_p = \{(X_1, Y_1), \ldots, (X_p, Y_p)\}.$$

After the conversation of $\mathcal{A}$ with its oracle, but before it makes its final decision, we reveal the key material used in the interaction. This can be done without loss of generality; it only improves the adversarial success probability. The first value that is revealed is a value K. In the real world, this is the key $K \xleftarrow{\$} \{0,1\}^k$ that is actually used by the construction oracle; in the ideal world, it is a dummy key $K \xleftarrow{\$} \{0,1\}^k$. The second value that is revealed is a value $L \in \{0,1\}^n$. In the real world, it is the value $L = \mathsf{P}(K \| 0^{n-k})$; in the ideal world, it is a dummy key $L \xleftarrow{\$} \{0,1\}^n$.[3] The revealed data is summarized in a view

$$\nu_k = \{(K, L)\}.$$

The complete view is defined as $\nu = (\nu_c, \nu_p, \nu_k)$. We assume that the adversary never makes any duplicate query, hence ν_c and ν_p contain no duplicate elements.

7.2 Definition of Good and Bad Views

In the real world, all tuples in ν_p define exactly one input-output pair for P. Likewise, the sole tuple in ν_k is an input-output pair for P. Using this tuple, one can observe that any tuple $(\bar{a}_i, M_i, C_i) \in \nu_c$ also defines an input-output pair for P, namely

$$\left(M_i \oplus \mathsf{mask}_K^{\bar{a}_i}, C_i \oplus \mathsf{mask}_K^{\bar{a}_i}\right).$$

[3] In the original analysis of MEM [GJMN16], the mask involves a computation $\mathsf{P}(K \| N)$ for nonce N. This not only complicates the values that have to be revealed; it also results in a larger view and hence a higher collision probability among tuples in the view.

If among all these $q+p+1$ input-output pairs defined by ν, there are two that have colliding input or output values, we consider ν to be a bad view. Formally, ν is called "bad" if one of the following conditions is satisfied, where we recall that $\nu_k = \{(K, L)\}$ is a singleton:

$$\begin{aligned}
\mathrm{bad}_{c,c}: &\text{ for some distinct } (\bar{a}, M, C), (\bar{a}', M', C') \in \nu_c: \\
&\mathsf{mask}_K^{\bar{a}}(L) \oplus \mathsf{mask}_K^{\bar{a}'}(L) \in \{M \oplus M', C \oplus C'\}\,, \\
\mathrm{bad}_{c,p}: &\text{ for some } (\bar{a}, M, C) \in \nu_c \text{ and } (X, Y) \in \nu_p: \\
&\mathsf{mask}_K^{\bar{a}}(L) \in \{M \oplus X, C \oplus Y\}\,, \\
\mathrm{bad}_{c,k}: &\text{ for some } (\bar{a}, M, C) \in \nu_c: \\
&\mathsf{mask}_K^{\bar{a}}(L) \in \{M \oplus K\|0^{n-k}, C \oplus L\}\,, \\
\mathrm{bad}_{p,k}: &\text{ for some } (X, Y) \in \nu_p: \\
&X = K\|0^{n-k} \text{ or } Y = L\,.
\end{aligned}$$

We write $\mathrm{bad} = \mathrm{bad}_{c,c} \vee \mathrm{bad}_{c,p} \vee \mathrm{bad}_{c,k} \vee \mathrm{bad}_{p,k}$.

7.3 Probability of Bad View in Ideal World

Our goal is to bound $\mathbf{Pr}\left(D_{\mathcal{P}} \in \mathcal{V}_{\mathrm{bad}}\right)$, the probability of a bad view in the ideal world $\mathcal{P} = (\widetilde{\pi}, \mathsf{P}^{\pm})$. For brevity, denote by $D_{\mathcal{P}} \propto \mathrm{bad}$ the event that $D_{\mathcal{P}}$ satisfies bad. By the union bound,

$$\begin{aligned}
\mathbf{Pr}\left(D_{\mathcal{P}} \propto \mathrm{bad}\right) &= \mathbf{Pr}\left(D_{\mathcal{P}} \propto \mathrm{bad}_{c,c} \vee \mathrm{bad}_{c,p} \vee \mathrm{bad}_{c,k} \vee \mathrm{bad}_{p,k}\right) \\
&\le \mathbf{Pr}\left(D_{\mathcal{P}} \propto \mathrm{bad}_{c,c}\right) + \mathbf{Pr}\left(D_{\mathcal{P}} \propto \mathrm{bad}_{c,p}\right) \\
&+ \mathbf{Pr}\left(D_{\mathcal{P}} \propto \mathrm{bad}_{c,k}\right) + \mathbf{Pr}\left(D_{\mathcal{P}} \propto \mathrm{bad}_{p,k}\right)\,. \qquad (14)
\end{aligned}$$

We will analyze the four probabilities separately, thereby noticing that (i) $K \xleftarrow{\$} \{0,1\}^k$ and $L \xleftarrow{\$} \{0,1\}^n$ are random variables, and (ii) as the adversary only makes forward construction queries, each tuple $(\bar{a}, M, C) \in \nu_c$ satisfies that C is randomly drawn from a set of size at least $2^n - q$.

Event $\mathbf{bad_{c,c}}$. For $\mathrm{bad}_{c,c}$, let $(\bar{a}, M, C), (\bar{a}', M', C') \in \nu_c$ be any two distinct tuples. If $\bar{a} = \bar{a}'$, then necessarily $M \neq M'$ and $C \neq C'$, and $\mathrm{bad}_{c,c}$ holds with probability 0. Otherwise, if $\bar{a} \neq \bar{a}'$, we can deduce from $2^{-\alpha}$-properness of $\mathcal{T}$, namely property 2 of Definition 1, that event $\mathrm{bad}_{c,c}$ holds with probability at most $2/2^{\alpha}$. Thus, summing over all $\binom{q}{2}$ possible choices of queries,

$$\mathbf{Pr}\left(D_{\mathcal{P}} \propto \mathrm{bad}_{c,c}\right) \le \frac{q(q-1)}{2^{\alpha}}\,.$$

Event $\mathbf{bad_{c,p}}$. For $\mathrm{bad}_{c,p}$, let $(\bar{a}, M, C) \in \nu_c$ and $(X, Y) \in \nu_p$ be any two tuples. We can deduce from $2^{-\alpha}$-properness of $\mathcal{T}$, namely property 1 of Definition 1, that event $\mathrm{bad}_{c,p}$ holds with probability at most $2/2^{\alpha}$. Thus, summing over all qp possible choices of queries,

$$\mathbf{Pr}\left(D_{\mathcal{P}} \propto \mathrm{bad}_{c,p}\right) \le \frac{2qp}{2^{\alpha}}\,.$$

Event $\mathbf{bad_{c,k}}$. For $\mathrm{bad}_{c,k}$, let $(\bar{a}, M, C) \in \nu_c$ be any tuple. We consider the two equations of $\mathrm{bad}_{c,k}$ separately. For the first equation,

$$\mathsf{mask}_K^{\bar{a}}(L) = M \oplus K\|0^{n-k}\,,$$

we will use that $L \xleftarrow{\$} \{0,1\}^n$ is a randomly generated value independent of K. We can deduce from $2^{-\alpha}$-properness of $\mathcal{T}$, namely property 1 of Definition 1, that this equation holds with probability at most $1/2^\alpha$.

For the second equation,

$$\mathsf{mask}_K^{\bar{a}}(L) = C \oplus L\,,$$

we will use that all construction queries are made in forward direction, and that C is randomly drawn from a set of size at least $2^n - q$ elements. Above equation thus holds with probability at most $1/(2^n - q)$.

Thus, summing over all q possible choices of queries,

$$\mathbf{Pr}\left(D_{\mathcal{P}} \propto \mathrm{bad}_{c,k}\right) \leq \frac{q}{2^\alpha} + \frac{q}{2^n - q}\,.$$

Event $\mathbf{bad_{p,k}}$. For $\mathrm{bad}_{p,k}$, let $(X, Y) \in \nu_p$ be any tuple. As $K \xleftarrow{\$} \{0,1\}^k$ and $L \xleftarrow{\$} \{0,1\}^n$, the tuple sets $\mathrm{bad}_{p,k}$ with probability at most $1/2^k + 1/2^n$. Thus, summing over all p possible choices of queries,

$$\mathbf{Pr}\left(D_{\mathcal{P}} \propto \mathrm{bad}_{p,k}\right) \leq \frac{p}{2^k} + \frac{p}{2^n}\,.$$

Conclusion. Concluding, we obtain for (14):

$$\mathbf{Pr}\left(D_{\mathcal{P}} \propto \mathrm{bad}\right) \leq \frac{q^2 + 2qp}{2^\alpha} + \frac{2q + p}{2^n} + \frac{p}{2^k}\,. \tag{15}$$

using that $2^n - q \geq 2^{n-1}$.

7.4 Probability Ratio for Good Views

Consider any good view $\nu \in \mathcal{V}_{\mathrm{good}}$. We will prove the inequality $\mathbf{Pr}\left(D_{\mathcal{O}} = \nu\right) \geq \mathbf{Pr}\left(D_{\mathcal{P}} = \nu\right)$. The proof is a direct simplification of that of Granger et al. [GJMN16], noting that in our case, ν_k consists of just one element. The proof is included for completeness.

Real World. In the real world $\mathcal{O} = (\widetilde{\mathsf{E}}_K^{\mathsf{P}}, \mathsf{P}^{\pm})$, goodness of the view means that $\nu = (\nu_c, \nu_p, \nu_k)$ defines exactly $q + p + 1$ input-output pairs for P and ν_k consists of a random value $K \xleftarrow{\$} \{0,1\}^k$, and there are no two of them that collide on the input or output. Therefore, we obtain:

$$\begin{aligned}\mathbf{Pr}\left(D_{\mathcal{O}} = \nu\right) &= \mathbf{Pr}\left(K' \xleftarrow{\$} \{0,1\}^k \;:\; K' = K\right) \cdot \\ &\qquad \mathbf{Pr}\left(\mathsf{P} \xleftarrow{\$} \mathrm{perm}(n) \;:\; \widetilde{\mathsf{E}}_K^{\mathsf{P}} \vdash \nu_c \wedge \mathsf{P} \vdash \nu_p \wedge \mathsf{P} \vdash \nu_k\right) \\ &= \frac{1}{2^k} \cdot \frac{(2^n - (q+p+1))!}{2^n!}\,.\end{aligned} \tag{16}$$

Ideal World. In the ideal world $\mathcal{P} = (\widetilde{\pi}, \mathsf{P}^{\pm})$, the view $\nu = (\nu_c, \nu_p, \nu_k)$ consists of three lists of independent tuples: ν_c defines exactly q input-output pairs for $\widetilde{\pi}$, ν_p defines exactly p input-output pairs for P, and ν_k consists of two random values $(K, L) \xleftarrow{\$} \{0,1\}^k \times \{0,1\}^n$. For counting, it is convenient to group the tuples in ν_c depending on the tweak value $\bar{a}$. For $T \in \mathcal{T}$, define

$$q_T = \left|\{(\bar{a}, M, C) \in \nu_c \mid \bar{a} = T\}\right|\,,$$

where $\sum_{T\in\mathcal{T}} q_T = q$. We obtain:

$$\begin{aligned}\mathbf{Pr}\left(D_{\mathcal{P}}=\nu\right) &= \mathbf{Pr}\left((K',L') \xleftarrow{\$} \{0,1\}^k\times\{0,1\}^n \;:\; (K',L')=(K,L)\right)\cdot \\ &\quad \mathbf{Pr}\left(\widetilde{\pi}\xleftarrow{\$}\mathrm{perm}(\mathcal{T},n) \;:\; \widetilde{\pi}\vdash\nu_c\right)\cdot \mathbf{Pr}\left(\mathsf{P}\xleftarrow{\$}\mathrm{perm}(n) \;:\; \mathsf{P}\vdash\nu_p\right) \\ &= \frac{1}{2^{k+n}}\cdot\prod_{T\in\mathcal{T}}\frac{(2^n-q_T)!}{2^n!}\cdot\frac{(2^n-p)!}{2^n!} \\ &= \frac{1}{2^k}\cdot\frac{(2^n-1)!}{2^n!}\cdot\prod_{T\in\mathcal{T}}\frac{(2^n-q_T)!}{2^n!}\cdot\frac{(2^n-p)!}{2^n!} \\ &\le \frac{1}{2^k}\cdot\frac{(2^n-(q+p+1))!}{2^n!}\,, \end{aligned} \tag{17}$$

using that for any $\sigma+\tau\le 2^n$ we have $\frac{(2^n-\sigma)!}{2^n!}\cdot\frac{(2^n-\tau)!}{2^n!}\le\frac{(2^n-(\sigma+\tau))!}{2^n!}$.

Conclusion. Combining (16) and (17), we obtain that for any good view $\nu\in\mathcal{V}_{\mathrm{good}}$:

$$\frac{\mathbf{Pr}\left(D_{\mathcal{O}}=\nu\right)}{\mathbf{Pr}\left(D_{\mathcal{P}}=\nu\right)}\ge 1\,. \tag{18}$$

7.5 Conclusion

By the H-coefficient technique (Lemma 1), we directly obtain from (15) and (18):

$$\mathbf{Adv}^{\mathrm{tprp}}_{\widetilde{\mathsf{E}}}(\mathcal{A})\le 0+\frac{q^2+2qp}{2^\alpha}+\frac{2q+p}{2^n}+\frac{p}{2^k}\,.$$

8 Proof of Theorem 2 (on Elephant)

Let $K\xleftarrow{\$}\{0,1\}^k$, $\mathsf{P}\xleftarrow{\$}\mathrm{perm}(n)$, and rand be a function that for each input (N,A,M) returns a random string of size $|M|+t$ bits. Consider a deterministic computationally unbounded adversary $\mathcal{A}$ that tries to distinguish $\mathcal{O}:=(\mathsf{enc}^{\mathsf{P}}_K,\mathsf{dec}^{\mathsf{P}}_K,\mathsf{P}^{\pm})$ from $\mathcal{P}:=(\mathsf{rand},\bot,\mathsf{P}^{\pm})$:

$$\mathbf{Adv}^{\mathrm{ae}}_{\mathsf{Elephant}}(\mathcal{A})=\Delta_{\mathcal{A}}\left(\mathsf{enc}^{\mathsf{P}}_K,\mathsf{dec}^{\mathsf{P}}_K,\mathsf{P}^{\pm}\;;\;\mathsf{rand},\bot,\mathsf{P}^{\pm}\right)\,. \tag{19}$$

As a first step, we will describe an alternative authenticated encryption scheme $\_'$ based on a tweakable permutation $\widetilde{\pi}\xleftarrow{\$}\mathrm{perm}(\mathcal{T},n)$, where $\mathcal{T}$ is $2^{-\alpha}$-proper with respect to LFSRs (φ_1,φ_2). Its encryption function $\overline{\mathsf{enc}}$ and decryption function $\overline{\mathsf{dec}}$ are given in Algorithms 3 and 4, respectively. Unlike the original functions enc and dec of Algorithms 1 and 2, the functions $\overline{\mathsf{enc}}$ and $\overline{\mathsf{dec}}$ are not explicitly keyed, but are instead implicitly keyed by the use of random secret tweakable permutation $\widetilde{\pi}$.

Algorithm 3 encryption $\overline{\mathsf{enc}}$

Input: (N, A, M)
Output: (C, T)
1: $M_1 \dots M_{\ell_M} \xleftarrow{n} M$
2: **for** $i = 1, \dots, \ell_M$ **do**
3: $\quad C_i \leftarrow M_i \oplus \widetilde{\pi}((i-1, 0), N \| 0^{n-m})$
4: $C \leftarrow \lfloor C_1 \dots C_{\ell_M} \rfloor_{|M|}$
5: $T = 0$
6: $A_1 \dots A_{\ell_A} \xleftarrow{n} N \| A \| 1$
7: $C_1 \dots C_{\ell_C} \xleftarrow{n} C \| 1$
8: **for** $i = 1, \dots, \ell_A$ **do**
9: $\quad T \leftarrow T \oplus \widetilde{\pi}((i-1, 2), A_i)$
10: **for** $i = 1, \dots, \ell_C$ **do**
11: $\quad T \leftarrow T \oplus \widetilde{\pi}((i-1, 1), C_i)$
12: **return** $(C, \lfloor T \rfloor_t)$

Algorithm 4 decryption $\overline{\mathsf{dec}}$

Input: (N, A, C, T)
Output: M or $\perp$
1: $C_1 \dots C_{\ell_M} \xleftarrow{n} C$
2: **for** $i = 1, \dots, \ell_M$ **do**
3: $\quad M_i \leftarrow C_i \oplus \widetilde{\pi}((i-1, 0), N \| 0^{n-m})$
4: $M \leftarrow \lfloor M_1 \dots M_{\ell_M} \rfloor_{|C|}$
5: $\bar{T} = 0$
6: $A_1 \dots A_{\ell_A} \xleftarrow{n} N \| A \| 1$
7: $C_1 \dots C_{\ell_C} \xleftarrow{n} C \| 1$
8: **for** $i = 1, \dots, \ell_A$ **do**
9: $\quad \bar{T} \leftarrow \bar{T} \oplus \widetilde{\pi}((i-1, 2), A_i)$
10: **for** $i = 1, \dots, \ell_C$ **do**
11: $\quad \bar{T} \leftarrow \bar{T} \oplus \widetilde{\pi}((i-1, 1), C_i)$
12: **return** $\lfloor \bar{T} \rfloor_t = T \ ? \ M \ : \ \perp$

By a simple hybrid argument, we obtain for the distance of (19):

$$\begin{aligned}(19) \le{}& \Delta_{\mathcal{A}}\left(\mathsf{enc}_K^{\mathsf{P}}, \mathsf{dec}_K^{\mathsf{P}}, \mathsf{P}^{\pm} \ ; \ \overline{\mathsf{enc}}^{\mathsf{SiM}_K^{\mathsf{P}}}, \overline{\mathsf{dec}}^{\mathsf{SiM}_K^{\mathsf{P}}}, \mathsf{P}^{\pm}\right) \\ &+ \Delta_{\mathcal{A}}\left(\overline{\mathsf{enc}}^{\mathsf{SiM}_K^{\mathsf{P}}}, \overline{\mathsf{dec}}^{\mathsf{SiM}_K^{\mathsf{P}}}, \mathsf{P}^{\pm} \ ; \ \overline{\mathsf{enc}}^{\widetilde{\pi}}, \overline{\mathsf{dec}}^{\widetilde{\pi}}, \mathsf{P}^{\pm}\right) \\ &+ \Delta_{\mathcal{A}}\left(\overline{\mathsf{enc}}^{\widetilde{\pi}}, \overline{\mathsf{dec}}_K^{\widetilde{\pi}}, \mathsf{P}^{\pm} \ ; \ \mathsf{rand}, \perp, \mathsf{P}^{\pm}\right). \end{aligned} \tag{20}$$

The first distance of (20) equals 0 by design of $\_'$. The second distance of (20) is at most $\Delta_{\mathcal{A}'}\left(\mathsf{SiM}_K^{\mathsf{P}}, \mathsf{P}^{\pm} \ ; \ \widetilde{\pi}, \mathsf{P}^{\pm}\right) = \mathbf{Adv}_{\mathsf{SiM}}^{\mathrm{tprp}}(\mathcal{A}')$, where $\mathcal{A}'$ is an adversary that makes 2σ construction queries and p primitive queries in order to simulate $\mathcal{A}$'s oracles. For the third distance of (20), access to P does not help the adversary, and the oracle can be dropped. We obtain from (20):

$$\begin{aligned}(19) &\le \mathbf{Adv}_{\mathsf{SiM}}^{\mathrm{tprp}}(\mathcal{A}') + \Delta_{\mathcal{A}}\left(\overline{\mathsf{enc}}^{\widetilde{\pi}}, \overline{\mathsf{dec}}^{\widetilde{\pi}} \ ; \ \mathsf{rand}, \perp\right) \\ &\le \mathbf{Adv}_{\mathsf{SiM}}^{\mathrm{tprp}}(\mathcal{A}') + \Delta_{\mathcal{A}}\left(\overline{\mathsf{enc}}^{\widetilde{\pi}}, \overline{\mathsf{dec}}^{\widetilde{\pi}} \ ; \ \overline{\mathsf{enc}}^{\widetilde{\pi}}, \perp\right) + \Delta_{\mathcal{A}}\left(\overline{\mathsf{enc}}^{\widetilde{\pi}}, \perp \ ; \ \mathsf{rand}, \perp\right). \end{aligned} \tag{21}$$

In order to upper bound the two remaining distances of (21), we will introduce the following two functions. First, define $h : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^t$ as

$$h(X, Y) = \left\lfloor \left(\bigoplus_{i=1}^{\ell_X} \widetilde{\pi}((i, 2), X_i) \right) \oplus \left(\bigoplus_{i=1}^{\ell_Y} \widetilde{\pi}((i-1, 1), Y_i) \right) \right\rfloor_t,$$

where $X_1 \dots X_{\ell_X} \xleftarrow{n} X \| 1$ and $Y_1 \dots Y_{\ell_Y} \xleftarrow{n} Y \| 1$. For permutation $\pi \xleftarrow{\$} \mathrm{perm}(n)$, define the MAC function

$$\mathsf{mac}^{\pi,h}(Z, X, Y) = \lfloor \pi(Z) \rfloor_t \oplus h(X, Y), \tag{22}$$

and let $\mathsf{vfy}^{\pi,h}$ be the corresponding verification function. We will use a result of Bernstein [Ber05] on Wegman-Carter-Shoup [WC81, Sho96] authenticators, translated to our setting.

Lemma 2. *Let $\pi \xleftarrow{\$} \mathrm{perm}(n)$, and $h : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^t$ be $2^{-\alpha}$-XOR-uniform and independent of π. Consider the message authentication code $\mathsf{mac}^{\pi,h}$ and its corresponding*

verification function $\mathsf{vfy}^{\pi,h}$ *of (22). For any adversary* $\mathcal{A}$ *making at most* $q_e \leq 2^{n-1}$ *MAC queries and* q_d *forgery attempts,*

$$\Delta_{\mathcal{A}}\left(\mathsf{mac}^{\pi,h}, \mathsf{vfy}^{\pi,h} \;;\; \mathsf{mac}^{\pi,h}, \bot\right) \leq q_d \cdot 2^{-\alpha} \cdot e^{(q_e+1)q_e/2^n}\,.$$

The proof will be given in Section 8.1.

One can reduce a distinguishing attack for the first distance of (21) to a forgery on $\mathsf{mac}^{\pi,h}$ with $\pi := \widetilde{\pi}((0,2),\cdot)$. Hence, using Lemma 2 along with the fact that h is $2^{n-t}(2^n-1)^{-1}$-XOR-uniform, we obtain

$$\Delta_{\mathcal{A}}\left(\widetilde{\overline{\mathsf{enc}}^{\pi}}, \widetilde{\overline{\mathsf{dec}}^{\pi}_{K}} \;;\; \widetilde{\overline{\mathsf{enc}}^{\pi}}, \bot\right) \leq \Delta_{\mathcal{A}'}\left(\mathsf{mac}^{\pi,h}, \mathsf{vfy}^{\pi,h} \;;\; \mathsf{mac}^{\pi,h}, \bot\right) \leq \frac{2^{n-t}q_d}{2^n-1} e^{(q_e+1)q_e/2^n}\,, \tag{23}$$

where $\mathcal{A}'$ has the same resources as $\mathcal{A}$.

For the second distance of (21), we remark that every query is made for a unique nonce, and in more detail:

- The i-th block of ciphertext equals $\widetilde{\pi}((i-1,0),N) \oplus M_i$, where M_i is the i-th block of plaintext;
- The tag equals $\lfloor \widetilde{\pi}((0,2),N\|A')\rfloor_t \oplus h(A'',C)$, where A' equals the first $n-m$ bits of padded associated data and A'' equals the rest, and where h never evaluates $\widetilde{\pi}$ for tweak $(\cdot,0)$ or $(0,2)$.

The tweakable permutation $\widetilde{\pi}$ is independent for different tweaks, but two different inputs for the same tweak never collide. Therefore, this second distance of (21) satisfies

$$\Delta_{\mathcal{A}}\left(\widetilde{\overline{\mathsf{enc}}^{\pi}}, \bot \;;\; \mathsf{rand}, \bot\right) \leq \ell\binom{q_e}{2}/2^n\,. \tag{24}$$

We thus obtain from (21), (23), and (24):

$$(19) \leq \mathbf{Adv}^{\mathrm{tprp}}_{\mathsf{SiM}}(\mathcal{A}') + \frac{2^{n-t}q_d}{2^n-1} e^{(q_e+1)q_e/2^n} + \ell\binom{q_e}{2}/2^n\,,$$

and this completes the proof of Theorem 2.

8.1 Proof of Lemma 2 (On $\mathsf{mac}^{\pi,h}$)

We write $f_t(N) = \lfloor \pi(N)\rfloor_t$ for brevity. Define the maximum k-interpolation probability of f_t as the maximum of

$$\mathbf{Pr}\left(f_t(x_1) = y_1, \ldots, f_t(x_k) = y_k\right) \tag{25}$$

taken over any distinct $x_1,\ldots,x_k \in \{0,1\}^n$ and any $y_1,\ldots,y_k \in \{0,1\}^t$.

Bernstein [Ber05, Theorem 5.1] states that if f_t has maximum q_e-interpolation probability at most $\delta/2^{tq_e}$ and maximum (q_e+1)-interpolation probability at most $2^{-\alpha}\delta/2^{tq_e}$, then the message authentication code $\mathsf{mac}^{\pi,h}$ of (22) satisfies[4]

$$\Delta_{\mathcal{A}}\left(\mathsf{mac}^{\pi,h}, \mathsf{vfy}^{\pi,h} \;;\; \mathsf{mac}^{\pi,h}, \bot\right) \leq q_d \cdot 2^{-\alpha} \cdot \delta\,.$$

[4] A sharp eye may note that the size of the range of f_t is at most the size of its domain, therewith violating the condition "$\#N \leq \#G$" in [Ber05, Theorem 5.1]. However, close inspection of the proof reveals that the condition is not used.

The maximum k-interpolation probability of f_t, for $k \leq q_e + 1 \leq 2^{n-1} + 1$, satisfies:

$$\begin{aligned}
\mathbf{Pr}\left(f_t(x_1) = y_1, \ldots, f_t(x_k) = y_k\right) &\leq \prod_{i=1}^{k} \frac{2^{n-t}}{2^n - (i-1)} \\
&= 2^{-tk} \cdot \prod_{i=1}^{k} \left(1 + \frac{i-1}{2^n - (i-1)}\right) \\
&\leq 2^{-tk} \cdot \prod_{i=1}^{k} \left(1 + \frac{2(i-1)}{2^n}\right) \\
&\leq 2^{-tk} \cdot e^{2\sum_{i=1}^{k} \frac{i-1}{2^n}} \\
&= e^{k(k-1)/2^n} / 2^{tk},
\end{aligned}$$

where we used that $k - 1 \leq 2^{n-1}$. As $2^{-\alpha} \geq 2^{-t}$, the bound satisfies the constraints put forward by Bernstein for $\delta = e^{(q_e+1)q_e/2^n}$.

We remark that for $t = n$, i.e., for f_n an injective function, Bernstein computed the same maximum k-interpolation probability in [Ber05, Theorem 4.2] and derived a similar bound on the security of $\mathsf{mac}^{\pi,h}$ in [Ber05, Theorem 5.3].

9 Conclusion

In this paper, we presented the Elephant family of lightweight authenticated encryption schemes. Our construction combines a provably secure mode of operation with standardized lightweight permutations. As a result, we end up with a parallel authenticated encryption scheme that is suitable for dedicated hardware implementations on resource-constrained devices, but also for software implementations on small 8-bit microcontrollers. Hence, Elephant fulfills the increasing demand for secure lightweight authenticated encryption schemes.

Acknowledgments. This work was supported in part by the Research Council KU Leuven: GOA TENSE (C16/15/058). Tim Beyne and Yu Long Chen are supported by a Ph.D. Fellowship from the Research Foundation - Flanders (FWO). Christoph Dobraunig is supported by the Austrian Science Fund (FWF): J 4277-N38. Bart Mennink is supported by a postdoctoral fellowship from the Netherlands Organisation for Scientific Research (NWO) under Veni grant 016.Veni.173.017. The authors thank the reviewers of the ToSC special issue for their valuable comments and suggestions.

References

[ABB+14] Elena Andreeva, Begül Bilgin, Andrey Bogdanov, Atul Luykx, Bart Mennink, Nicky Mouha, and Kan Yasuda. APE: Authenticated Permutation-Based Encryption for Lightweight Cryptography. In Carlos Cid and Christian Rechberger, editors, *Fast Software Encryption - 21st International Workshop, FSE 2014, London, UK, March 3-5, 2014. Revised Selected Papers*, volume 8540 of *LNCS*, pages 168–186. Springer, 2014.

[Abd12] Mohamed Ahmed Abdelraheem. Estimating the Probabilities of Low-Weight Differential and Linear Approximations on PRESENT-Like Ciphers. In Taekyoung Kwon, Mun-Kyu Lee, and Daesung Kwon, editors, *Information Security and Cryptology - ICISC 2012 - 15th International Conference, Seoul, Korea, November 28-30, 2012, Revised Selected Papers*, volume 7839 of *LNCS*, pages 368–382. Springer, 2012.

[ABD+16] Elena Andreeva, Andrey Bogdanov, Nilanjan Datta, Atul Luykx, Bart Mennink, Mridul Nandi, Elmar Tischhauser, and Kan Yasuda. COLM v1. Submission to the CAESAR competition, 2016.

[AFF+15] Farzaneh Abed, Scott Fluhrer, John Foley, Christian Forler, Eik List, Stefan Lucks, David McGrew, and Jakob Wenzel. The POET Family of On-Line Authenticated Encryption Schemes v2.0. Submission to the CAESAR competition, 2015.

[AHMN10] Jean-Philippe Aumasson, Luca Henzen, Willi Meier, and María Naya-Plasencia. Quark: A Lightweight Hash. In Stefan Mangard and François-Xavier Standaert, editors, *Cryptographic Hardware and Embedded Systems, CHES 2010, 12th International Workshop, Santa Barbara, CA, USA, August 17-20, 2010. Proceedings*, volume 6225 of *LNCS*, pages 1–15. Springer, 2010.

[BDH+17] Guido Bertoni, Joan Daemen, Seth Hoffert, Michaël Peeters, Gilles Van Assche, and Ronny Van Keer. Farfalle: parallel permutation-based cryptography. *IACR Transactions on Symmetric Cryptology*, 2017(4):1–38, 2017.

[BDP+12] Guido Bertoni, Joan Daemen, Michaël Peeters, Gilles Van Assche, and Ronny Van Keer. Keccak implementation overview (Version 3.2), May 2012.

[BDPV07] Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. Sponge functions. Ecrypt Hash Workshop 2007, May 2007.

[BDPV11a] Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. Duplexing the Sponge: Single-Pass Authenticated Encryption and Other Applications. In Ali Miri and Serge Vaudenay, editors, *Selected Areas in Cryptography - 18th International Workshop, SAC 2011, Toronto, ON, Canada, August 11-12, 2011, Revised Selected Papers*, volume 7118 of *LNCS*, pages 320–337. Springer, 2011.

[BDPV11b] Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. The Keccak reference, January 2011.

[Ber05] Daniel J. Bernstein. Stronger Security Bounds for Wegman-Carter-Shoup Authenticators. In Ronald Cramer, editor, *Advances in Cryptology - EUROCRYPT 2005, 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, May 22-26, 2005, Proceedings*, volume 3494 of *LNCS*, pages 164–180. Springer, 2005.

[Ber08] Daniel J. Bernstein. Chacha, a variant of salsa20. Online Document: https://cr.yp.to/chacha/chacha-20080128.pdf, January 2008.

[BKL+11] Andrey Bogdanov, Miroslav Knezevic, Gregor Leander, Deniz Toz, Kerem Varici, and Ingrid Verbauwhede. Spongent: A Lightweight Hash Function. In Bart Preneel and Tsuyoshi Takagi, editors, *Cryptographic Hardware and Embedded Systems - CHES 2011 - 13th International Workshop, Nara, Japan, September 28 - October 1, 2011. Proceedings*, volume 6917 of *LNCS*, pages 312–325. Springer, 2011.

[BKL+17] Daniel J. Bernstein, Stefan Kölbl, Stefan Lucks, Pedro Maat Costa Massolino, Florian Mendel, Kashif Nawaz, Tobias Schneider, Peter Schwabe, François-Xavier Standaert, Yosuke Todo, and Benoît Viguier. Gimli : A Cross-Platform Permutation. In Wieland Fischer and Naofumi Homma, editors, *Cryptographic Hardware and Embedded Systems - CHES 2017 - 19th International Conference, Taipei, Taiwan, September 25-28, 2017, Proceedings*, volume 10529 of *LNCS*, pages 299–320. Springer, 2017.

[BL16] Karthikeyan Bhargavan and Gaëtan Leurent. On the Practical (In-)Security of 64-bit Block Ciphers: Collision Attacks on HTTP over TLS and OpenVPN. In Edgar R. Weippl, Stefan Katzenbeisser, Christopher Kruegel, Andrew C. Myers, and Shai Halevi, editors, *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, October 24-28, 2016*, pages 456–467. ACM, 2016.

[BN00] Mihir Bellare and Chanathip Namprempre. Authenticated Encryption: Relations among Notions and Analysis of the Generic Composition Paradigm. In Tatsuaki Okamoto, editor, *Advances in Cryptology - ASIACRYPT 2000, 6th International Conference on the Theory and Application of Cryptology and Information Security, Kyoto, Japan, December 3-7, 2000, Proceedings*, volume 1976 of *LNCS*, pages 531–545. Springer, 2000.

[CCD+17] Katriel Cohn-Gordon, Cas J. F. Cremers, Benjamin Dowling, Luke Garratt, and Douglas Stebila. A Formal Security Analysis of the Signal Messaging Protocol. In *2017 IEEE European Symposium on Security and Privacy, EuroS&P 2017, Paris, France, April 26-28, 2017*, pages 451–466. IEEE, 2017.

[CDNY18] Avik Chakraborti, Nilanjan Datta, Mridul Nandi, and Kan Yasuda. Beetle Family of Lightweight and Secure Authenticated Encryption Ciphers. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2018(2):218–241, 2018.

[CMN+15] Simon Cogliani, Diana-Stefania Maimut, David Naccache, Rodrigo Portella do Canto, Reza Reyhanitabar, Serge Vaudenay, and Damian Vizár. Offset Merkle-Damgård (OMD) version 2.0. Submission to the CAESAR competition, 2015.

[CS14] Shan Chen and John P. Steinberger. Tight Security Bounds for Key-Alternating Ciphers. In Nguyen and Oswald [NO14], pages 327–350.

[DEMS16] Christoph Dobraunig, Maria Eichlseder, Florian Mendel, and Martin Schläffer. Ascon v1.2. Submission to the CAESAR competition, 2016.

[DHVV18] Joan Daemen, Seth Hoffert, Gilles Van Assche, and Ronny Van Keer. The design of Xoodoo and Xoofff. *IACR Transactions on Symmetric Cryptology*, 2018(4):1–38, 2018.

[DMV17] Joan Daemen, Bart Mennink, and Gilles Van Assche. Full-State Keyed Duplex with Built-In Multi-user Support. In Tsuyoshi Takagi and Thomas Peyrin, editors, *Advances in Cryptology - ASIACRYPT 2017 - 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, December 3-7, 2017, Proceedings, Part II*, volume 10625 of *LNCS*, pages 606–637. Springer, 2017.

[DR02] Joan Daemen and Vincent Rijmen. *The Design of Rijndael: AES - The Advanced Encryption Standard*. Information Security and Cryptography. Springer, 2002.

[FIP12] FIPS 180-4: Secure Hash Standard, March 2012.

[FIP15] FIPS 202: SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions, August 2015.

[GJMN16] Robert Granger, Philipp Jovanovic, Bart Mennink, and Samuel Neves. Improved Masking for Tweakable Blockciphers with Applications to Authenticated Encryption. In Marc Fischlin and Jean-Sébastien Coron, editors, *Advances in Cryptology - EUROCRYPT 2016 - 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016, Proceedings, Part I*, volume 9665 of *LNCS*, pages 263–293. Springer, 2016.

[GPP11] Jian Guo, Thomas Peyrin, and Axel Poschmann. The PHOTON Family of Lightweight Hash Functions. In Phillip Rogaway, editor, *Advances in Cryptology - CRYPTO 2011 - 31st Annual Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2011. Proceedings*, volume 6841 of *LNCS*, pages 222–239. Springer, 2011.

[HKR17] Viet Tung Hoang, Ted Krovetz, and Phillip Rogaway. AEZ v5: Authenticated Encryption by Enciphering. Submission to the CAESAR competition, 2017.

[HKS18] Deukjo Hong, Bonwook Koo, and Changho Seo. Differential property of Present-like structure. *Discrete Applied Mathematics*, 241:13–24, 2018.

[ISO16] ISO/IEC 29192-5:2016. Information technology – Security techniques – Lightweight cryptography – Part 5: Hash-functions, 2016.

[KLL+14] Elif Bilge Kavun, Martin M. Lauridsen, Gregor Leander, Christian Rechberger, Peter Schwabe, and Tolga Yalçın. Prøst v1.1, 2014. Submission to CAESAR competition.

[KR11] Ted Krovetz and Phillip Rogaway. The Software Performance of Authenticated-Encryption Modes. In Antoine Joux, editor, *Fast Software Encryption - 18th International Workshop, FSE 2011, Lyngby, Denmark, February 13-16, 2011, Revised Selected Papers*, volume 6733 of *LNCS*, pages 306–327. Springer, 2011.

[KY10] Elif Bilge Kavun and Tolga Yalçin. A Lightweight Implementation of Keccak Hash Function for Radio-Frequency Identification Applications. In Siddika Berna Ors Yalcin, editor, *Radio Frequency Identification: Security and Privacy Issues - 6th International Workshop, RFIDSec 2010, Istanbul, Turkey, June 8-9, 2010, Revised Selected Papers*, volume 6370 of *LNCS*, pages 258–269. Springer, 2010.

[LS18] Gaëtan Leurent and Ferdinand Sibleyras. The Missing Difference Problem, and Its Applications to Counter Mode Encryption. In Jesper Buus Nielsen and Vincent Rijmen, editors, *Advances in Cryptology - EUROCRYPT 2018 - 37th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tel Aviv, Israel, April 29 - May 3, 2018 Proceedings, Part II*, volume 10821 of *LNCS*, pages 745–770. Springer, 2018.

[Min16] Kazuhiko Minematsu. AES-OTR v3.1. Submission to the CAESAR competition, 2016.

[MRV15] Bart Mennink, Reza Reyhanitabar, and Damian Vizár. Security of Full-State Keyed Sponge and Duplex: Applications to Authenticated Encryption. In Tetsu Iwata and Jung Hee Cheon, editors, *Advances in Cryptology - ASIACRYPT 2015 - 21st International Conference on the Theory and Application of Cryptology and Information Security, Auckland, New Zealand, November 29 - December 3, 2015, Proceedings, Part II*, volume 9453 of *LNCS*, pages 465–489. Springer, 2015.

[Nat18] National Institute of Standards and Technology (NIST). Submission requirements and evaluation criteria for the lightweight cryptography standardization process, August 2018.

[NO14] Phong Q. Nguyen and Elisabeth Oswald, editors. *Advances in Cryptology - EUROCRYPT 2014 - 33rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Copenhagen, Denmark, May 11-15, 2014. Proceedings*, volume 8441 of *LNCS*. Springer, 2014.

[NRR06] Svetla Nikova, Christian Rechberger, and Vincent Rijmen. Threshold Implementations Against Side-Channel Attacks and Glitches. In Peng Ning, Sihan Qing, and Ninghui Li, editors, *Information and Communications Security, 8th International Conference, ICICS 2006, Raleigh, NC, USA, December 4-7, 2006, Proceedings*, volume 4307 of *LNCS*, pages 529–545. Springer, 2006.

[NRS14] Chanathip Namprempre, Phillip Rogaway, and Thomas Shrimpton. Reconsidering Generic Composition. In Nguyen and Oswald [NO14], pages 257–274.

[Pat08] Jacques Patarin. The "Coefficients H" Technique. In Roberto Maria Avanzi, Liam Keliher, and Francesco Sica, editors, *Selected Areas in Cryptography, 15th International Workshop, SAC 2008, Sackville, New Brunswick, Canada, August 14-15, Revised Selected Papers*, volume 5381 of *LNCS*, pages 328–345. Springer, 2008.

[PM16] Trevor Perrin and Moxie Marlinspike. The double ratchet algorithm (revision 1). Online Document: https://signal.org/docs/specifications/doubleratchet/doubleratchet.pdf, November 2016.

[RBBK01] Phillip Rogaway, Mihir Bellare, John Black, and Ted Krovetz. OCB: a block-cipher mode of operation for efficient authenticated encryption. In Michael K. Reiter and Pierangela Samarati, editors, *CCS 2001, Proceedings of the 8th ACM Conference on Computer and Communications Security, Philadelphia, Pennsylvania, USA, November 6-8, 2001.*, pages 196–205. ACM, 2001.

[Res18] Eric Rescorla. The Transport Layer Security (TLS) Protocol Version 1.3. RFC 8446, August 2018.

[Rog04] Phillip Rogaway. Efficient Instantiations of Tweakable Blockciphers and Refinements to Modes OCB and PMAC. In Pil Joong Lee, editor, *Advances in Cryptology - ASIACRYPT 2004, 10th International Conference on the Theory and Application of Cryptology and Information Security, Jeju Island, Korea, December 5-9, 2004, Proceedings*, volume 3329 of *LNCS*, pages 16–31. Springer, 2004.

[Sho96] Victor Shoup. On Fast and Provably Secure Message Authentication Based on Universal Hashing. In Neal Koblitz, editor, *Advances in Cryptology - CRYPTO '96, 16th Annual International Cryptology Conference, Santa Barbara, California, USA, August 18-22, 1996, Proceedings*, volume 1109 of *LNCS*, pages 313–328. Springer, 1996.

[STA+15] Yu Sasaki, Yosuke Todo, Kazumaro Aoki, Yusuke Naito, Takeshi Sugawara, Yumiko Murakami, Mitsuru Matsui, and Shoichi Hirose. Minalpher v1.1. Submission to the CAESAR competition, 2015.

[The17] The Sage Developers. *SageMath, the Sage Mathematics Software System (Version 8.1)*, 2017. `https://www.sagemath.org`.

[Wan15] Lei Wang. SHELL v2.0. Submission to the CAESAR competition, 2015.

[WC81] Mark N. Wegman and Larry Carter. New Hash Functions and Their Use in Authentication and Set Equality. *J. Comput. Syst. Sci.*, 22(3):265–279, 1981.

[ZBRL15] Wentao Zhang, Zhenzhen Bao, Vincent Rijmen, and Meicheng Liu. A New Classification of 4-bit Optimal S-boxes and Its Application to PRESENT, RECTANGLE and SPONGENT. In Gregor Leander, editor, *Fast Software Encryption - 22nd International Workshop, FSE 2015, Istanbul, Turkey, March 8-11, 2015, Revised Selected Papers*, volume 9054 of *LNCS*, pages 494–515. Springer, 2015.

[ZL17] Guoyan Zhang and Meicheng Liu. A distinguisher on PRESENT-like permutations with application to SPONGENT. *SCIENCE CHINA Information Sciences*, 60(7):72101, 2017.