Beyond $2^{c/2}$ Security in Sponge-Based Authenticated Encryption Modes

Philipp Jovanovic[1], Atul Luykx[2], and Bart Mennink[2]

[1] Fakultät für Informatik und Mathematik, Universität Passau, Germany
jovanovic@fim.uni-passau.de
[2] Dept. Electrical Engineering, ESAT/COSIC, KU Leuven, and iMinds, Belgium
atul.luykx@esat.kuleuven.be, bart.mennink@esat.kuleuven.be

Abstract. The Sponge function is known to achieve $2^{c/2}$ security, where c is its capacity. This bound was carried over to keyed variants of the function, such as SpongeWrap, to achieve a $\min\{2^{c/2}, 2^{\kappa}\}$ security bound, with κ the key length. Similarly, many CAESAR competition submissions are designed to comply with the classical $2^{c/2}$ security bound. We show that Sponge-based constructions for authenticated encryption can achieve the significantly higher bound of $\min\{2^{b/2}, 2^{c}, 2^{\kappa}\}$ asymptotically, with $b > c$ the permutation size, by proving that the CAESAR submission NORX achieves this bound. Furthermore, we show how to apply the proof to five other Sponge-based CAESAR submissions: Ascon, CBEAM/STRIBOB, ICEPOLE, Keyak, and two out of the three PRIMATEs. A direct application of the result shows that the parameter choices of these submissions are overly conservative. Simple tweaks render the schemes considerably more efficient without sacrificing security. For instance, NORX64 can increase its rate and decrease its capacity by 128 bits and Ascon-128 can encrypt three times as fast, both without affecting the security level of their underlying modes in the ideal permutation model.

Keywords. Authenticated encryption, CAESAR, Ascon, CBEAM, ICEPOLE, Keyak, NORX, PRIMATEs, STRIBOB.

1 Introduction

Authenticated encryption schemes, cryptographic functions that aim to provide both privacy and integrity of data, have gained renewed attention in light of the recently commenced CAESAR competition [15]. A common approach to building such schemes is to design a mode of operation for a block cipher, as in CCM [30], OCB1-3 [20,23,24], and EAX [8]. Nevertheless a significant fraction of the CAESAR competition submissions use modes of operation for *permutations*.

Most of the permutation-based modes follow the basic design of the Sponge construction [13]: their output is computed from a state value, which in turn is repeatedly updated using key, nonce, associated data, and plaintext by calling a permutation. The state is divided into a rate part of r bits, through which the

user enters plaintext, and a capacity part of c bits, which is out of the user's control.

The security of the Sponge construction as a hash function follows from the fact that the user can only affect the rate, hence an adversary only succeeds with significant probability if it makes on the order of $2^{c/2}$ permutation queries, as this many are needed to produce a collision in the capacity [13]. Keyed versions of the Sponge construction, such as KeyedSponge [12] and SpongeWrap [11], are proven up to a similar bound of 2^{c-a}, assuming a limit of 2^a on online complexity, but are additionally restricted by the key size κ to 2^κ. The permutation-based CAESAR candidates are no exception and recommend parameters based on either the $2^{c/2}$ bound or 2^{c-a} bound, as shown in Table 1.

1.1 Our Results

Contrary to intuition, a wide range of permutation-based authenticated encryption schemes achieve a *significantly* higher mode security level: we prove that the bound is limited by approximately $\min\{2^{(r+c)/2}, 2^c, 2^\kappa\}$ as opposed to $\min\{2^{c/2}, 2^\kappa\}$. The main proof in this work concerns NORX mode [4], but we demonstrate its applicability to the CAESAR submissions Ascon [16], CBEAM[3] [27,28], ICEPOLE [22], Keyak [14], two out of three PRIMATEs [2], and STRIBOB[4] [25,29]. Additionally, we note that it directly applies to SpongeWrap [11] and DuplexWrap [14], upon which Keyak is built.

Our results imply that all of these CAESAR candidates have been overly conservative in choosing their parameters, since a smaller capacity would have lead to the same bound. For instance, Ascon-128 could take $(c, r) = (128, 192)$ instead of $(256, 64)$, NORX64 (the proposed mode with 256-bit security) could increase its rate by 128 bits, and GIBBON-120 and HANUMAN-120 could increase their rate by a factor of 4, all without affecting their mode security levels.

These observations only concern the *mode* security, where characteristics of the underlying permutation are set aside. Specifically, the concrete security of the underlying permutations plays a fundamental role in the choice of parameters. For instance, the authors of Ascon, NORX, and PRIMATEs [2,4,16] acknowledge that non-random properties of some of the underlying primitives exist. Although these properties are harmless, a non-hermetic design approach for the primitives affects the parameter choices.

1.2 Outline

We present our security model in Section 2. A security proof for NORX is derived in Section 3. In Section 4 we show that the proof of NORX generalizes to other CAESAR submissions, as well as to SpongeWrap and DuplexWrap. The work is concluded in Section 5, where we also discuss possible generalizations to Artemia [1] and π-Cipher [17].

[3] CBEAM was withdrawn after an attack by Minaud [21], but we focus on modes of operation.

[4] Both CBEAM and STRIBOB use the BLNK Sponge mode [26].

Table 1: Parameters and the achieved mode security levels of seven CAESAR submissions. We remark that ICEPOLE consists of three configurations (two with security level 128 and one with security level 256) and Keyak of four configurations (one with an 800-bit state and three with a 1600-bit state)

	b	c	r	κ	τ	**security**
Ascon [16]	320	192	128	96	96	**96**
	320	256	64	128	128	**128**
CBEAM [28]	256	190	66	128	64	**128**
ICEPOLE [22]	1280	254	1026	128	128	**128**
	1280	318	962	256	128	**256**
Keyak [14]	800	252	548	128..224	128	**128..224**
	1600	252	1348	128..224	128	**128..224**
NORX [4]	512	192	320	128	128	**128**
	1024	384	640	256	256	**256**
GIBBON/ HANUMAN [2]	200	159	41	80	80	**80**
	280	239	41	120	120	**120**
STRIBOB [29]	512	254	258	192	128	**192**

2 Security Model

For $n \in \mathbb{N}$, let $\mathsf{Perm}(n)$ denote the set of all permutations on n bits. When writing $x \xleftarrow{\$} \mathcal{X}$ for some finite set $\mathcal{X}$, we mean that x gets sampled uniformly at random from $\mathcal{X}$. For $x \in \{0,1\}^n$, and $a, b \leq n$, we denote by $[x]^a$ and $[x]_b$ the a leftmost and b rightmost bits of x, respectively. For tuples $(j,k), (j',k')$ we use lexicographical order: $(j,k) > (j',k')$ means that $j > j'$, or $j = j'$ and $k > k'$.

Let Π be an authenticated encryption scheme, which is specified by an encryption function $\mathcal{E}$ and a decryption function $\mathcal{D}$:

$$(C, A) \longleftarrow \mathcal{E}_K(N; H, M, T) \qquad \text{and} \qquad M/\perp \longleftarrow \mathcal{D}_K(N; H, C, T; A)\,.$$

Here N denotes a nonce value, H a header, M a message, C a ciphertext, T a trailer, and A an authentication tag. The values (H, T) will be referred to as associated data. If verification is correct, then the decryption function $\mathcal{D}_K$ outputs M, and $\perp$ otherwise. The scheme Π is also determined by a set of parameters such as the key size, state size, and block size, but these are left implicit. In addition, we define \$ to be an ideal version of $\mathcal{E}_K$, where \$ returns $(C, A) \xleftarrow{\$} \{0,1\}^{|M|+\tau}$ on input of a new $(N; H, M, T)$.

We follow the convention in analyzing modes of operation for permutations by modeling the underlying permutations as being drawn uniformly at random

from $\mathsf{Perm}(b)$, where b is a parameter determined by the scheme. We note that irregularities in the underlying permutation may invalidate the underlying assumption.

An adversary $\mathcal{A}$ is a probabilistic algorithm that has access to one or more oracles $\mathcal{O}$, denoted $\mathcal{A}^{\mathcal{O}}$. By $\mathcal{A}^{\mathcal{O}} = 1$ we denote the event that $\mathcal{A}$, after interacting with $\mathcal{O}$, outputs 1. We consider adversaries $\mathcal{A}$ that have unbounded computational power and whose complexities are solely measured by the number of queries made to their oracles. These adversaries have query access to the underlying idealized permutations, $\mathcal{E}_K$ or its counterpart \$, and possibly $\mathcal{D}_K$. The key K is randomly drawn from $\{0,1\}^\kappa$ at the beginning of the security experiment. The security definitions below follow [6, 18].

Privacy

Let $\mathbf{p}$ denote the list of underlying idealized permutations of Π. We define the advantage of an adversary $\mathcal{A}$ in breaking the privacy of Π as follows:

$$\mathbf{Adv}^{\mathrm{priv}}_{\Pi}(\mathcal{A}) = \left|\mathbf{Pr}_{\mathbf{p},K}\left(\mathcal{A}^{\mathbf{p}^{\pm},\mathcal{E}_K} = 1\right) - \mathbf{Pr}_{\mathbf{p},\$}\left(\mathcal{A}^{\mathbf{p}^{\pm},\$} = 1\right)\right| ,$$

where the probabilities are taken over the random choices of $\mathbf{p}, \$, K$, and the random choices of $\mathcal{A}$, if any. The fact that the adversary has access to both the forward and inverse permutations in $\mathbf{p}$ is denoted by $\mathbf{p}^{\pm}$. We assume that adversary $\mathcal{A}$ is nonce-respecting, which means that it never makes two queries to $\mathcal{E}_K$ or \$ with the same nonce. By $\mathbf{Adv}^{\mathrm{priv}}_{\Pi}(q_p, q_{\mathcal{E}}, \lambda_{\mathcal{E}})$ we denote the maximum advantage taken over all adversaries that query $\mathbf{p}^{\pm}$ at most q_p times, and that make at most $q_{\mathcal{E}}$ queries of total length at most $\lambda_{\mathcal{E}}$ blocks to $\mathcal{E}_K$ or \$. We remark that this privacy notion is also known as the CPA security of an (authenticated) encryption scheme.

Integrity

As above, let $\mathbf{p}$ denote the list of underlying idealized permutations of Π. We define the advantage of an adversary $\mathcal{A}$ in breaking the integrity of Π as follows:

$$\mathbf{Adv}^{\mathrm{auth}}_{\Pi}(\mathcal{A}) = \mathbf{Pr}_{\mathbf{p},K}\left(\mathcal{A}^{\mathbf{p}^{\pm},\mathcal{E}_K,\mathcal{D}_K} \text{ forges}\right) ,$$

where the probability is taken over the random choices of $\mathbf{p}, K$, and the random choices of $\mathcal{A}$, if any. Here, we say that "$\mathcal{A}$ forges" if $\mathcal{D}_K$ ever returns a valid message (other than $\perp$) on input of $(N; H, C, T; A)$ where (C, A) has never been output by $\mathcal{E}_K$ on input of a query $(N; H, M, T)$ for some M. We assume that adversary $\mathcal{A}$ is nonce-respecting, which means that it never makes two queries to $\mathcal{E}_K$ with the same nonce. Nevertheless, $\mathcal{A}$ is allowed to repeat nonces in decryption queries. By $\mathbf{Adv}^{\mathrm{auth}}_{\Pi}(q_p, q_{\mathcal{E}}, \lambda_{\mathcal{E}}, q_{\mathcal{D}}, \lambda_{\mathcal{D}})$ we denote the maximum advantage taken over all adversaries that query $\mathbf{p}^{\pm}$ at most q_p times, that make at most $q_{\mathcal{E}}$ queries of total length at most $\lambda_{\mathcal{E}}$ blocks to $\mathcal{E}_K$, and at most $q_{\mathcal{D}}$ queries of total length at most $\lambda_{\mathcal{D}}$ blocks to $\mathcal{D}_K/\perp$.

3 NORX

We introduce NORX at a level required for the understanding of the security proof, and refer to Aumasson et al. [4] for the formal specification. Let p be a permutation on b bits. All b-bit state values are split into a rate part of r bits and a capacity part of c bits. We denote the key size of NORX by κ bits, the nonce size by ν bits, and the tag size by τ bits. The header, message, and trailer can be of arbitrary length, and are padded using 10^*1-padding to a length of a multiple of r bits. Throughout, we denote the r-bit header blocks by $H_1, \ldots, H_u$, message blocks by $M_1, \ldots, M_v$, ciphertext blocks by $C_1, \ldots, C_v$, and trailer blocks by $T_1, \ldots, T_w$.

Unlike other permutation-based schemes, NORX allows for parallelism in the encryption part, which is described using a parameter $D \in \{0, \ldots, 255\}$ corresponding to the number of parallel chains. Specifically, if $D \in \{1, \ldots, 255\}$ NORX has D parallel chains, and if $D = 0$ it has v parallel chains, where v is the block length of M or C.

NORX consists of five proposed parameter configurations: NORXW-R-D for $(W, R, D) \in \{(64, 4, 1), (32, 4, 1), (64, 6, 1), (32, 6, 1), (64, 4, 4)\}$. The parameter R denotes the number of rounds of the underlying permutation p, and W denotes the word size which we use to set $r = 10W$ and $c = 6W$. The default key and tag size are $\kappa = \nu = 4W$. The corresponding parameters for the two different choices of W, 64 and 32, are given in Table 1.

Although NORX starts with an initialization function init which requires the parameters (D, R, τ) as input, as soon as our security experiment starts, we consider (D, R, τ) fixed and constant. Hence we can view init as a function that maps (K, N) to $(K \| N \| 0^{b-\kappa-\nu}) \oplus \mathsf{const}$, where const is irrelevant to the mode security analysis of NORX, and will be ignored in the remaining analysis.

After init is called, the header H is compressed into the rate, then the state is branched into D states (if necessary), the message blocks are encrypted in a streaming way, the D states are merged into one state (if necessary), the trailer is compressed, and finally the tag A is computed. All rounds are preceded with a domain separation constant XORed into the capacity: `01` for header compression, `02` for message encryption, `04` for trailer compression, and `08` for tag generation. If $D \neq 1$, domain separators `10` and `20` are used for branching and merging, along with pairwise distinct lane indices id_k for $k = 1, \ldots, D$ (if $D = 1$ we write $id_1 = \mathsf{0}$).

The privacy of NORX is proven in Section 3.1 and the integrity in Section 3.2. In both proofs we consider an adversary that makes q_p permutation queries and $q_{\mathcal{E}}$ encryption queries of total length $\lambda_{\mathcal{E}}$. In the proof of integrity, the adversary can additionally make $q_{\mathcal{D}}$ decryption queries of total length $\lambda_{\mathcal{D}}$. To aid the analysis, we compute the number of permutation calls made via the $q_{\mathcal{E}}$ encryption queries. The exact same computation holds for decryption queries with the parameters defined analogously.

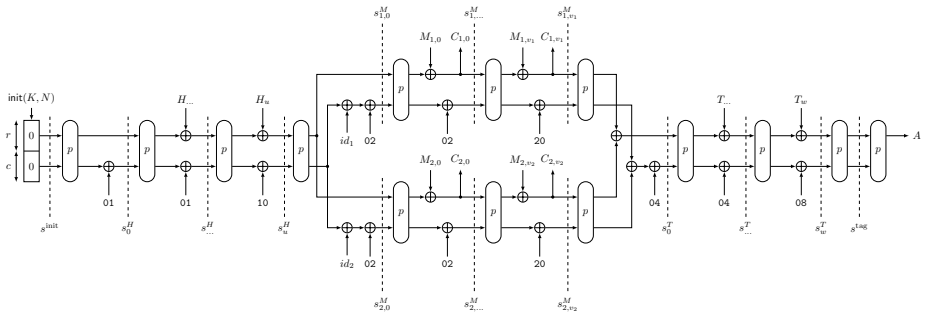


Fig. 1: NORX with $D = 2$

Consider a query to $\mathcal{E}_K$, consisting of u header blocks, v message blocks, and w trailer blocks. We denote its corresponding state values by

$$\left(s^{\text{init}};\ s_0^H, \ldots, s_u^H;\ \begin{bmatrix} s_{1,0}^M, \ldots, s_{1,v_1}^M \\ \vdots \qquad\qquad \vdots \\ s_{D,0}^M, \ldots, s_{D,v_D}^M \end{bmatrix};\ s_0^T, \ldots, s_w^T;\ s^{\text{tag}} \right), \tag{1}$$

as outlined in Figure 1. Here, $\sum_{k=1}^{D} v_k = v$. If there are no branching and merging phases, i.e. $D = 1$, then the state values corresponding to the branching and merging, $\{s_{1,0}^M, \ldots, s_{D,0}^M\}$ and s_0^T, are left out of the tuple. Note that the length of this tuple equals the number of primitive calls made in this encryption query, as every state value corresponds to the input of exactly one primitive call. A simple calculation shows that if the jth $\mathcal{E}_K$ query is of length $u+v+w$ blocks, it results in $u+v+w+3$ state values if $D = 1$, in $u+v+w+D+4$ state values if $D > 1$, and in $u+2v+w+4$ state values if $D = 0$.[5] We denote the number of state values by $\sigma_{\mathcal{E},j}$, where the dependence on D is suppressed as D does not change during the security game. In other words, $\sigma_{\mathcal{E},j}$ denotes the number of primitive calls in the jth query to $\mathcal{E}_K$. Furthermore, we define $\sigma_{\mathcal{E}}$ to be the total number of primitive evaluations via the encryption queries, and find that

$$\sigma_{\mathcal{E}} := \sum_{j=1}^{q_{\mathcal{E}}} \sigma_{\mathcal{E},j} \leq \begin{cases} 2\lambda_{\mathcal{E}} + 4q_{\mathcal{E}}, \text{ if } D = 0\,, \\ \lambda_{\mathcal{E}} + 3q_{\mathcal{E}}, \text{ if } D = 1\,, \\ \lambda_{\mathcal{E}} + (D+4)q_{\mathcal{E}}, \text{ if } D > 1\,. \end{cases} \tag{2}$$

This bound is rather tight. Particularly, for $D = 0$ an adversary can meet this bound by only making queries without header and trailer. For queries to $\mathcal{D}_K$ we define $\sigma_{\mathcal{D},j}$ and $\sigma_{\mathcal{D}}$ analogously.

[5] For $D = 0$, the original specification dictates an additional $10^{b-2}1$-padding for every complete message block. This means that lanes $1, \ldots, v-1$ consist of two rounds. We do not take this padding into account, noting that it is unnecessary for the security analysis.

3.1 Privacy of NORX

Theorem 1. *Let $\Pi = (\mathcal{E}, \mathcal{D})$ be NORX based on an ideal underlying primitive p. Then,*

$$\mathbf{Adv}^{\mathrm{priv}}_{\Pi}(q_p, q_{\mathcal{E}}, \lambda_{\mathcal{E}}) \leq \frac{3(q_p + \sigma_{\mathcal{E}})^2}{2^{b+1}} + \left(\frac{8eq_p\sigma_{\mathcal{E}}}{2^b}\right)^{1/2} + \frac{rq_p}{2^c} + \frac{q_p + \sigma_{\mathcal{E}}}{2^\kappa}\,,$$

where $\sigma_{\mathcal{E}}$ is defined in (2).

Theorem 1 can be interpreted as implying that NORX provides privacy security as long as the total complexity $q_p + \sigma_{\mathcal{E}}$ does not exceed $\min\{2^{b/2}, 2^\kappa\}$ and the total number of primitive queries q_p, also known as the offline complexity, does not exceed $2^c/r$. See Table 1 for the security level of the various parameter choices of NORX.

The proof is based on the observation that NORX is indistinguishable from a random scheme as long as there are no collisions among the (direct and indirect) evaluations of p. Due to uniqueness of the nonce, state values from evaluations of $\mathcal{E}_K$ collide with probability approximately $1/2^b$. Regarding collisions between direct calls to p and calls via $\mathcal{E}_K$: while these may happen with probability about $1/2^c$, they turn out not to significantly influence the bound. The latter is demonstrated in part using the principle of multiplicities [10]: roughly stated, the maximum number of state values with the same rate part. The formal security proof is more detailed. Furthermore, we remark that, at the cost of readability and simplicity of the proof, the bound could be improved by a constant factor.

Proof. We consider any adversary $\mathcal{A}$ that has access to either $(p^{\pm}, \mathcal{E}_K)$ or $(p^{\pm}, \$)$ and whose goal is to distinguish these two worlds. For brevity, we write

$$\mathbf{Adv}^{\mathrm{priv}}_{\Pi}(\mathcal{A}) = \Delta_{\mathcal{A}}(p^{\pm}, \mathcal{E}_K; p^{\pm}, \$)\,. \tag{3}$$

We start with replacing $p^{\pm}$ by a random function, as this simplifies the analysis. This is done with a PRP-PRF switch [3, 7], in which we make a transition from $p^{\pm}$ to a primitive $f^{\pm}$ defined as follows. This primitive $f^{\pm}$ maintains an initially empty list $\mathcal{F}$ of query/response tuples (x, y). For $\mathcal{F}$, we denote its set of domain and range values by $\mathrm{dom}(\mathcal{F})$ and $\mathrm{rng}(\mathcal{F})$, respectively. For a forward query $f(x)$ with $x \in \mathrm{dom}(\mathcal{F})$, the corresponding value $y = \mathcal{F}(x)$ is returned. For a new forward query $f(x)$, the response y is randomly drawn from $\{0,1\}^b$, then if y is in $\mathrm{rng}(\mathcal{F})$ the primitive aborts, otherwise the tuple (x, y) is added to $\mathcal{F}$. The description for f^{-1} is similar. The usage of $\mathcal{F}$ will remain implicit in the remaining usage of $f^{\pm}$. Now, $p^{\pm}$ and $f^{\pm}$ behave identically as long as the latter does not abort. Given that the adversary triggers at most $q_p + \sigma_{\mathcal{E}}$ evaluations of f, such an abort happens with probability at most $\binom{q_p+\sigma_{\mathcal{E}}}{2}/2^b \leq (q_p+\sigma_{\mathcal{E}})^2/2^{b+1}$. This PRP-PRF switch needs to be applied to both the real and ideal world, to get

$$\Delta_{\mathcal{A}}(p^{\pm}, \mathcal{E}_K; p^{\pm}, \$) \leq \Delta_{\mathcal{A}}(f^{\pm}, \mathcal{E}_K; f^{\pm}, \$) + \frac{(q_p + \sigma_{\mathcal{E}})^2}{2^b}\,. \tag{4}$$

We restrict our attention to $\mathcal{A}$ with oracle access to $(f^{\pm}, F)$, where $F \in \{\mathcal{E}_K, \$\}$. Without loss of generality, we can assume that the adversary only queries full blocks and that no padding rules are involved. We can do this because the padding rules are injective, allowing the proof to carry over to the case of fractional blocks with 10^*1-padding.

We introduce some terminology. Queries to $f^{\pm}$ are denoted (x_i, y_i) for $i = 1, \ldots, q_p$, while queries to F are written as elements $(N_j; H_j, M_j, T_j; C_j, A_j)$ for $j = 1, \ldots, q_{\mathcal{E}}$. If $F = \mathcal{E}_K$, the state values are denoted as in (1), subscripted with a j:

$$\left(s_j^{\text{init}};\ s_{j,0}^H, \ldots, s_{j,u}^H;\ \begin{bmatrix} s_{j,1,0}^M, \ldots, s_{j,1,v_1}^M \\ \vdots \qquad\qquad \vdots \\ s_{j,D,0}^M, \ldots, s_{j,D,v_D}^M \end{bmatrix};\ s_{j,0}^T, \ldots, s_{j,w}^T;\ s_j^{\text{tag}} \right). \tag{5}$$

If the structure of (5) is irrelevant we refer to the tuple as $(s_{j,1}, \ldots, s_{j,\sigma_{\mathcal{E},j}})$, where we use the convention to list the elements of the matrix column-wise. In this case, we write $\text{parent}(s_{j,k})$ to denote the state value that lead to $s_{j,k}$, with $\text{parent}(s_{j,1}) := \varnothing$ and $\text{parent}(s_{j,0}^T) := (s_{j,1,v_1}^M, \ldots, s_{j,D,v_D}^M)$. We remark that the characteristic structure of NORX, with the D parallel states, only becomes relevant in the two technical lemmas that will be used at the end of the proof. We point out that $s_{j,1}$ corresponds to the initial state value of the evaluation, which requires special attention throughout the remainder of the proof.

We define two collision events, guess and hit. Let $i \in \{1, \ldots, q_p\}$, $j, j' \in \{1, \ldots, q_{\mathcal{E}}\}$, $k \in \{1, \ldots, \sigma_{\mathcal{E},j}\}$, and $k' \in \{1, \ldots, \sigma_{\mathcal{E},j'}\}$:

$$\begin{aligned} \mathsf{guess}(i; j, k) &\equiv x_i = s_{j,k}\,, \\ \mathsf{hit}(j, k; j', k') &\equiv \text{parent}(s_{j,k}) \neq \text{parent}(s_{j',k'}) \wedge s_{j,k} = s_{j',k'}\,. \end{aligned}$$

Event $\mathsf{guess}(i; j, k)$ corresponds to a primitive call in an encryption query hitting a direct primitive query, or vice versa, while $\mathsf{hit}(j, k; j', k')$ corresponds to non-trivial primitive calls colliding in encryption queries. We write $\mathsf{guess} = \vee_{i;j,k}\, \mathsf{guess}(i; j, k)$, $\mathsf{hit} = \vee_{j,k;j',k'}\, \mathsf{hit}(j, k; j', k')$, and set $\mathsf{event} = \mathsf{guess} \vee \mathsf{hit}$.

The remainder of the proof is divided as follows. In Lemma 1 we prove that $(f^{\pm}, \mathcal{E}_K)$ and $(f^{\pm}, \$)$ are indistinguishable as long as $\neg\mathsf{event}$ holds. In other words,

$$\Delta_{\mathcal{A}}(f^{\pm}, \mathcal{E}_K; f^{\pm}, \$) \leq \mathbf{Pr}\left(\mathcal{A}^{f^{\pm}, \mathcal{E}_K} \text{ sets } \mathsf{event}\right). \tag{6}$$

Then, in Lemma 2 we bound this term by $\dfrac{q_p\sigma_{\mathcal{E}} + \sigma_{\mathcal{E}}^2/2}{2^b} + \left(\dfrac{8eq_p\sigma_{\mathcal{E}}}{2^b}\right)^{1/2} + \dfrac{rq_p}{2^c} + \dfrac{q_p + \sigma_{\mathcal{E}}}{2^\kappa}$. Noting that $\dfrac{q_p\sigma_{\mathcal{E}} + \sigma_{\mathcal{E}}^2/2}{2^b} \leq \dfrac{(q_p + \sigma_{\mathcal{E}})^2}{2^{b+1}}$, this completes the proof via equations (3,4,6). □

Lemma 1. *Given that* event *does not occur,* $(f^{\pm}, \mathcal{E}_K)$ *and* $(f^{\pm}, \$)$ *are indistinguishable.*

Proof. The outputs of $f^{\pm}$ are sampled uniformly at random in both $(f^{\pm}, \mathcal{E}_K)$ and $(f^{\pm}, \$)$, except when such an output collides with a state of an $\mathcal{E}_K$ evaluation in the real world. However, this event is excluded by assuming $\neg\mathsf{guess}$, hence it suffices to only consider queries to the big oracle $F \in \{\mathcal{E}_K, \$\}$.

Let N_j be a new nonce used in the F-query $(N_j; H_j, M_j, T_j)$, with corresponding ciphertext and authentication tag (C_j, A_j). Denote the query's state values as in (5). Let u, v, and w denote the number of padded header blocks, padded message blocks, and padded trailer blocks, respectively.

By the definition of \$, in the ideal world we have $(C_j, A_j) \xleftarrow{\$} \{0,1\}^{|M_j|+\tau}$. We will prove that (C_j, A_j) is identically distributed in the real world, under the assumption that $\mathsf{guess} \vee \mathsf{hit}$ does not occur. Denote the message blocks of M_j by $M_{j,k,\ell}$ for $k = 1, \ldots, D$ and $\ell = 1, \ldots, v_k$.

We know that $s^H_{j,u}$ is new and that $f(s^H_{j,u})$ does not collide with any other f-query because otherwise $\neg\mathsf{event}$ would have been violated. Since $s^M_{j,k,0} = f(s^H_{j,u}) \oplus id_k$ we conclude that $s^M_{j,k,0}$ is new for $k = 1, \ldots, D$, as otherwise event would be set. Similarly, $s^M_{j,k,\ell}$ is new for all $\ell > 0$. The ciphertext blocks $C_{j,k,\ell}$ are computed as

$$C_{j,k,\ell} = M_{j,k,\ell} \oplus [f(s^M_{j,k,\ell-1})]^r \,.$$

As the state value $s^M_{j,k,\ell-1}$ has not been evaluated by f before (neither directly nor indirectly via an encryption query), $f(s^M_{j,k,\ell-1})$ outputs a uniformly random value from $\{0,1\}^b$, hence $C_{j,k,\ell} \xleftarrow{\$} \{0,1\}^r$. We remark that similar reasoning shows that a ciphertext block corresponding to a truncated message block is uniformly randomly drawn as well, yet from a smaller set. The fact that $A_j \xleftarrow{\$} \{0,1\}^\tau$ follows the same reasoning, using that s^{tag}_j is a new input to f. Thus, $A_j = [f(s^{\mathrm{tag}}_j)]^\tau \xleftarrow{\$} \{0,1\}^\tau$. □

Lemma 2. $\mathbf{Pr}\left(\mathcal{A}^{f^{\pm},\mathcal{E}_K} \text{ sets } \mathsf{event}\right) \leq \frac{q_p\sigma_{\mathcal{E}} + \sigma^2_{\mathcal{E}}/2}{2^b} + \left(\frac{8eq_p\sigma_{\mathcal{E}}}{2^b}\right)^{1/2} + \frac{rq_p}{2^c} + \frac{q_p + \sigma_{\mathcal{E}}}{2^\kappa}$.

Proof. Consider the adversary interacting with $(f^{\pm}, \mathcal{E}_K)$, and let $\mathbf{Pr}\,(\mathsf{guess} \vee \mathsf{hit})$ denote the probability we aim to bound. For $i \in \{1, \ldots, q_p\}$, define

$$\mathsf{key}(i) \quad \equiv \quad [x_i]^\kappa = K \,,$$

and $\mathsf{key} = \vee_i \mathsf{key}(i)$. Event $\mathsf{key}(i)$ corresponds to a primitive query hitting the key. Let $j \in \{1, \ldots, q_{\mathcal{E}}\}$ and $k \in \{1, \ldots, \sigma_{\mathcal{E},j}\}$, and consider any threshold $\rho \geq 1$, then define

$$\mathsf{multi}(j,k) \quad \equiv \quad \left[\max_{\alpha \in \{0,1\}^r} \left|\left\{j' \leq j, 1 < k' \leq k \,:\, \alpha \in \{[s_{j',k'}]^r, [f(s_{j',k'})]^r\}\right\}\right| \right] > \rho \,.$$

Event $\mathsf{multi}(j,k)$ is used to bound the number of states that collide in the rate part. Note that state values $s_{j',1}$ are not considered here as they will be covered by key. We define $\mathsf{multi} = \mathsf{multi}(q_{\mathcal{E}}, \sigma_{\mathcal{E},q_{\mathcal{E}}})$, which is a monotone event. By basic probability theory,

$$\mathbf{Pr}\left(\mathsf{guess} \vee \mathsf{hit}\right) \leq \mathbf{Pr}\left(\mathsf{guess} \vee \mathsf{hit} \mid \neg(\mathsf{key} \vee \mathsf{multi})\right) + \mathbf{Pr}\left(\mathsf{key} \vee \mathsf{multi}\right) . \quad (7)$$

In the remainder of the proof, we bound these probabilities as follows (a formal explanation of the proof technique is given in Appendix A): we consider the ith forward or inverse primitive query (for $i \in \{1, \ldots, q_p\}$) or the kth state of the jth construction query (for $j \in \{1, \ldots, q_{\mathcal{E}}\}$ and $k \in \{1, \ldots, \sigma_{\mathcal{E},j}\}$), and bound the probability that this evaluation makes $\mathsf{guess} \vee \mathsf{hit}$ satisfied, under the assumption that this query does not set $\mathsf{key} \vee \mathsf{multi}$ and also that $\mathsf{guess} \vee \mathsf{hit} \vee \mathsf{key} \vee \mathsf{multi}$ has not been set before. For the analysis of $\mathbf{Pr}\left(\mathsf{key} \vee \mathsf{multi}\right)$ a similar technique is employed.

Event guess. This event can be set in the ith primitive query (for $i = 1, \ldots, q_p$) or in any state evaluation of the jth construction query (for $j = 1, \ldots, q_{\mathcal{E}}$). Denote the state values of the jth construction query as in (5). Consider any evaluation, assume this query does not set $\mathsf{key} \vee \mathsf{multi}$ and assume that $\mathsf{guess} \vee \mathsf{hit} \vee \mathsf{key} \vee \mathsf{multi}$ has not been set before. Firstly, note that $x_i = s_j^{\mathrm{init}}$ for some i, j would imply $\mathsf{key}(i)$ and hence invalidate our assumption. Therefore, we can exclude s_j^{init} from further analysis on guess. For $i = 1, \ldots, q_p$, let $j_i \in \{1, \ldots, q_{\mathcal{E}}\}$ be the number of encryption queries made before the ith primitive query. Similarly, for $j = 1, \ldots, q_{\mathcal{E}}$, denote by $i_j \in \{1, \ldots, q_p\}$ the number of primitive queries made before the jth encryption query.

- Consider a primitive query (x_i, y_i) for $i \in \{1, \ldots, q_p\}$, which may be a forward or an inverse query, and assume it has not been queried to $f^{\pm}$ before. If it is a forward query x_i, by $\neg\mathsf{multi}$ there are at most ρ state values s with $[x_i]^r = [s]^r$, and thus $x_i = s$ with probability at most $\rho/2^c$. Here, we remark that the capacity part of s is unknown to the adversary and it guesses it with probability at most $1/2^c$. A slightly more complicated reasoning applies for inverse queries. Denote the query by y_i. By $\neg\mathsf{multi}$ there are at most ρ state values s with $[y_i]^r = [f(s)]^r$, hence $y_i = f(s)$ with probability at most $\rho/2^c$. If y_i equals $f(s)$ for any of these states, then $x_i = s$, otherwise $x_i = s$ with probability at most $\sum_{j=1}^{j_i} \sigma_{\mathcal{E},j}/2^b$. Therefore the probability that guess is set via a direct query is at most $\frac{q_p \rho}{2^c} + \sum_{i=1}^{q_p} \sum_{j=1}^{j_i} \frac{\sigma_{\mathcal{E},j}}{2^b}$;
- Next, consider the probability that the jth construction query sets guess, for $j \in \{1, \ldots, q_{\mathcal{E}}\}$. For simplicity, first consider $D = 1$, hence the message is processed in one lane and we can use state labeling $(s_{j,1}, \ldots, s_{j,\sigma_{\mathcal{E},j}})$. We range from $s_{j,2}$ to $s_{j,\sigma_{\mathcal{E},j}}$ (recall that $s_{j,1} = s_j^{\mathrm{init}}$ can be excluded) and consider the probability that this state sets guess assuming it has not been set before. Let $k \in \{2, \ldots, \sigma_{\mathcal{E},j}\}$. The state value $s_{j,k}$ equals $f(s_{j,k-1}) \oplus v$, where v is some value determined by the adversarial input prior to the evaluation of $f(s_{j,k-1})$, including input from (H_j, M_j, T_j) and constants serving as domain separators. By assumption, $\mathsf{guess} \vee \mathsf{hit}$ has not been set before, and $f(s_{j,k-1})$ is thus

randomly drawn from $\{0,1\}^b$. It hits any x_i ($i \in \{1,\ldots,i_j\}$) with probability at most $i_j/2^b$. Next, consider the general case $D > 1$. We return to the labeling of (5). A complication occurs for the branching states $s^M_{j,1,0},\ldots,s^M_{j,D,0}$ and the merging state $s^T_{j,0}$. Starting with the branching states, these are computed from $s^H_{j,u}$ as

$$\begin{pmatrix} s^M_{j,1,0} \\ \vdots \\ s^M_{j,D,0} \end{pmatrix} = f(s^H_{j,u}) \oplus \begin{pmatrix} v_1 \\ \vdots \\ v_D \end{pmatrix},$$

where $v_1,\ldots,v_D$ are some distinct values determined by the adversarial input prior to the evaluation of the jth construction query. These are distinct by the XOR of the lane numbers $id_1,\ldots,id_D$. Any of these nodes equals x_i for $i \in \{1,\ldots,q_p\}$ with probability at most $i_jD/2^b$. Finally, for the merging node $s^T_{j,0}$ we can apply the same analysis, noting that it is derived from a sum of D new f-evaluations. Concluding, the jth construction query sets guess with probability at most $i_j\sigma_{\mathcal{E},j}/2^b$ (we always have in total at most $\sigma_{\mathcal{E},j}$ new state values). Summing over all $q_\mathcal{E}$ construction queries, we get $\sum_{j=1}^{q_\mathcal{E}} i_j\sigma_{\mathcal{E},j}/2^b$.

Concluding,

$$\mathbf{Pr}\left(\mathsf{guess} \mid \neg(\mathsf{key} \vee \mathsf{multi})\right) \leq \frac{q_p\rho}{2^c} + \sum_{i=1}^{q_p}\sum_{j=1}^{j_i}\frac{\sigma_{\mathcal{E},j}}{2^b} + \sum_{j=1}^{q_\mathcal{E}}\frac{i_j\sigma_{\mathcal{E},j}}{2^b} = \frac{q_p\rho}{2^c} + \frac{q_p\sigma_\mathcal{E}}{2^b}\,.$$

Here we use that $\sum_{i=1}^{q_p}\sum_{j=1}^{j_i}\sigma_{\mathcal{E},j} + \sum_{j=1}^{q_\mathcal{E}}\sum_{k=1}^{\sigma_{\mathcal{E},j}} i_j = q_p\sigma_\mathcal{E}$, which follows from a simple counting argument.

Event hit. We again employ ideas of guess, and particularly that as long as $\mathsf{guess} \vee \mathsf{hit}$ is not set, we can consider all new state values (except for the initial states) to be randomly drawn from a set of size 2^b. Particularly, we can refrain from explicitly discussing the branching and merging nodes (the detailed analysis of guess applies) and label the states as $(s_{j,1},\ldots,s_{j,\sigma_{\mathcal{E},j}})$. Clearly, $s_{j,1} \neq s_{j',1}$ for all j,j' by uniqueness of the nonce. Any state value $s_{j,k}$ for $k > 1$ (at most $\sigma_\mathcal{E} - q_\mathcal{E}$ in total) hits an initial state value $s_{j',1}$ only if $[s_{j,k}]^\kappa = K$, which happens with probability at most $\sigma_\mathcal{E}/2^\kappa$, assuming $s_{j,k}$ is generated randomly. Finally, any two other states $s_{j,k}, s_{j',k'}$ for $k,k' > 1$ collide with probability at most $\binom{\sigma_\mathcal{E}-q_\mathcal{E}}{2}/2^b$. Concluding, $\mathbf{Pr}\left(\mathsf{hit} \mid \neg(\mathsf{key} \vee \mathsf{multi})\right) \leq \binom{\sigma_\mathcal{E}}{2}/2^b + \sigma_\mathcal{E}/2^\kappa$.

Event key. For $i \in \{1,\ldots,q_p\}$, the query sets $\mathsf{key}(i)$ if $[x_i]^\kappa = K$, which happens with probability $1/2^\kappa$ (assuming it did not happen in queries $1,\ldots,i-1$). The adversary makes q_p attempts, and hence $\mathbf{Pr}\left(\mathsf{key}\right) \leq q_p/2^\kappa$.

Event multi. We again use the principles from the analysis for guess of construction queries (note that this part does not rely on multi itself). Particularly,

consider a new state value $s_{j,k-1}$; then for a fixed state value $x \in \{0,1\}^b$ it satisfies $f(s_{j,k-1}) = x$ or $s_{j,k} = f(s_{j,k-1}) \oplus v = x$ for some predetermined v with probability at most $2/2^b$. Now, let $\alpha \in \{0,1\}^r$. More than ρ state values hit α with probability at most $\binom{\sigma_\mathcal{E}}{\rho}(2/2^r)^\rho \leq \left(\frac{2e\sigma_\mathcal{E}}{\rho 2^r}\right)^\rho$, using Stirling's approximation ($x! \geq (x/e)^x$ for any x). Considering any possible choice of α, we obtain $\mathbf{Pr}\left(\mathsf{multi}\right) \leq 2^r \left(\frac{2e\sigma_\mathcal{E}}{\rho 2^r}\right)^\rho$.

Addition of the four bounds via (7) gives

$$\mathbf{Pr}\left(\mathsf{guess} \vee \mathsf{hit}\right) \leq \frac{q_p\sigma_\mathcal{E} + \sigma_\mathcal{E}^2/2}{2^b} + \frac{q_p\rho}{2^c} + \frac{q_p + \sigma_\mathcal{E}}{2^\kappa} + 2^r\left(\frac{e\sigma_\mathcal{E}}{\rho 2^r}\right)^\rho.$$

Putting $\rho = \max\left\{r, \left(\frac{2e\sigma_\mathcal{E}2^c}{q_p 2^r}\right)^{1/2}\right\}$ gives

$$\mathbf{Pr}\left(\mathsf{guess} \vee \mathsf{hit}\right) \leq \frac{q_p\sigma_\mathcal{E} + \sigma_\mathcal{E}^2/2}{2^b} + 2\left(\frac{2eq_p\sigma_\mathcal{E}}{2^b}\right)^{1/2} + \frac{rq_p}{2^c} + \frac{q_p + \sigma_\mathcal{E}}{2^\kappa},$$

assuming $2eq_p\sigma_\mathcal{E}/2^b < 1$ (which we can do, as the bound would otherwise be void anyway). This completes the proof. □

3.2 Authenticity of NORX

Theorem 2. *Let $\Pi = (\mathcal{E}, \mathcal{D})$ be NORX based on an ideal underlying primitive p. Then,*

$$\mathbf{Adv}^{\mathrm{auth}}_\Pi(q_p, q_\mathcal{E}, \lambda_\mathcal{E}, q_\mathcal{D}, \lambda_\mathcal{D}) \leq \frac{(q_p + \sigma_\mathcal{E} + \sigma_\mathcal{D})^2}{2^b} + \left(\frac{8eq_p\sigma_\mathcal{E}}{2^b}\right)^{1/2} + \frac{rq_p}{2^c} + \frac{q_p + \sigma_\mathcal{E} + \sigma_\mathcal{D}}{2^\kappa} + \frac{(q_p + \sigma_\mathcal{E} + \sigma_\mathcal{D})\sigma_\mathcal{D}}{2^c} + \frac{q_\mathcal{D}}{2^\tau},$$

where $\sigma_\mathcal{E}, \sigma_\mathcal{D}$ are defined in (2).

The bound is more complex than the one of Theorem 1, but intuitively implies that NORX offers integrity as long as it offers privacy and the number of forgery attempts $\sigma_\mathcal{D}$ is limited, where the total complexity $q_p + \sigma_\mathcal{E} + \sigma_\mathcal{D}$ should not exceed $2^c/\sigma_\mathcal{D}$. See Table 1 for the security level for the various parameter choices of NORX. Needless to say, the exact bound is more fine-grained.

Proof. We consider any adversary $\mathcal{A}$ that has access to $(p^\pm, \mathcal{E}_K, \mathcal{D}_K)$ and attempts to make $\mathcal{D}_K$ output a non-$\perp$ value. As in the proof of Theorem 1, we apply a PRP-PRF switch to find

$$\begin{aligned}\mathbf{Adv}^{\mathrm{auth}}_\Pi(\mathcal{A}) &= \mathbf{Pr}\left(\mathcal{A}^{p^\pm, \mathcal{E}_K, \mathcal{D}_K} \text{ forges}\right) \\ &\leq \mathbf{Pr}\left(\mathcal{A}^{f^\pm, \mathcal{E}_K, \mathcal{D}_K} \text{ forges}\right) + \frac{(q_p + \sigma_\mathcal{E} + \sigma_\mathcal{D})^2}{2^{b+1}}.\end{aligned} \tag{8}$$

Then we focus on $\mathcal{A}$ having oracle access to $(f^{\pm}, \mathcal{E}_K, \mathcal{D}_K)$. As before, we assume without loss of generality that the adversary only makes full-block queries.

We inherit terminology from Theorem 1. The state values corresponding to encryption and decryption queries will both be labeled (j,k), where j indicates the query and k the state value within the jth query. If needed we will add another parameter $\delta \in \{\mathcal{D}, \mathcal{E}\}$ to indicate that a state value $s_{\delta,j,k}$ is in the jth query to oracle δ, for $\delta \in \{\mathcal{D}, \mathcal{E}\}$ and $j \in \{1, \ldots, q_\delta\}$. Particularly, this means we will either label the state values as in (5) with a δ appended to the subscript, or simply as $(s_{\delta,j,1}, \ldots, s_{\delta,j,\sigma_{\delta,j}})$.

As before, we employ the collision events guess and hit, but expanded to the new notation with $\delta = \mathcal{E}$. Next, we define two $\mathcal{D}$-related collision events $\mathcal{D}\mathsf{guess}$ and $\mathcal{D}\mathsf{hit}$. Let $i \in \{1, \ldots, q_p\}$, $(\mathcal{D}, j, k)$ be a decryption query index, and (δ', j', k') be an encryption or decryption query index:

$$\begin{aligned} \mathcal{D}\mathsf{guess}(i; j, k) &\equiv x_i = s_{\mathcal{D},j,k}\,, \\ \mathcal{D}\mathsf{hit}(j, k; \delta', j', k') &\equiv \mathrm{parent}(s_{\mathcal{D},j,k}) \neq \mathrm{parent}(s_{\delta',j',k'}) \wedge s_{\mathcal{D},j,k} = s_{\delta',j',k'}\,, \end{aligned}$$

We write $\mathcal{D}\mathsf{guess} = \vee_{i;j,k}\, \mathcal{D}\mathsf{guess}(i; j, k)$ and $\mathsf{hit} = \vee_{j,k;\delta',j',k'}\, \mathcal{D}\mathsf{hit}(j, k; \delta', j', k')$, and define $\mathsf{event} = \mathsf{guess} \vee \mathsf{hit} \vee \mathcal{D}\mathsf{guess} \vee \mathcal{D}\mathsf{hit}$.

Observe that from (8) we get

$$\begin{aligned} \mathbf{Pr}\left(\mathcal{A}^{f^{\pm},\mathcal{E}_K,\mathcal{D}_K} \text{ forges}\right) \leq \mathbf{Pr}\left(\mathcal{A}^{f^{\pm},\mathcal{E}_K,\mathcal{D}_K} \text{ forges} \mid \neg\mathsf{event}\right) + \\ \mathbf{Pr}\left(\mathcal{A}^{f^{\pm},\mathcal{E}_K,\mathcal{D}_K} \text{ sets } \mathsf{event}\right)\,. \end{aligned} \tag{9}$$

A bound on the probability that $\mathcal{A}$ sets event is derived in Lemma 3.

The remainder of this proof centers on the probability that $\mathcal{A}$ forges given that event does not happen. Such a forgery requires that $[f(s^{\mathrm{tag}}_{\mathcal{D},j})]^\tau = A_j$ for some decryption query j. By $\neg\mathsf{event}$, we know that $s^{\mathrm{tag}}_{\mathcal{D},j}$ is a new state value for all $j \in \{1, \ldots, q_\mathcal{D}\}$, hence f's output under $s^{\mathrm{tag}}_{\mathcal{D},j}$ is independent of all other values and uniformly distributed for all j. As a result, we know that the jth forgery attempt is successful with probability at most $1/2^\tau$. Summing over all $q_\mathcal{D}$ queries, we get

$$\mathbf{Pr}\left(\mathcal{A}^{f^{\pm},\mathcal{E}_K,\mathcal{D}_K} \text{ forges} \mid \neg\mathsf{event}\right) \leq \frac{q_\mathcal{D}}{2^\tau}\,,$$

and the proof is completed via (8,9) and the bound of Lemma 3, where we again use that $\dfrac{q_p\sigma_\mathcal{E} + \sigma_\mathcal{E}^2/2}{2^b} \leq \dfrac{(q_p + \sigma_\mathcal{E} + \sigma_\mathcal{D})^2}{2^{b+1}}$. □

Lemma 3. $\mathbf{Pr}\left(\mathcal{A}^{f^{\pm},\mathcal{E}_K,\mathcal{D}_K} \textit{ sets } \mathsf{event}\right) \leq \dfrac{q_p\sigma_\mathcal{E} + \sigma_\mathcal{E}^2/2}{2^b} + \left(\dfrac{8eq_p\sigma_\mathcal{E}}{2^b}\right)^{1/2} + \dfrac{rq_p}{2^c} + \dfrac{q_p + \sigma_\mathcal{E} + \sigma_\mathcal{D}}{2^\kappa} + \dfrac{(q_p + \sigma_\mathcal{E})\sigma_\mathcal{D} + \sigma_\mathcal{D}^2/2}{2^c}$.

The proof of Lemma 3 is given in the full version of this paper [19].

4 Other CAESAR Submissions

In this section we discuss how the mode security proof of NORX generalizes to the CAESAR submissions Ascon, the BLNK mode underlying CBEAM/ STRIBOB, ICEPOLE, Keyak, and two out of the three PRIMATEs. Before doing so, we make a number of observations and note how the proof can accommodate small design differences.

- NORX uses domain separation constants at all rounds, but this is not strictly necessary and other solutions exist. In the privacy and integrity proofs of NORX, and more specifically at the analysis of state collisions caused by a decryption query in Lemma 3, the domain separations are only needed at the transitions between variable-length inputs, such as header to message data or message to trailer data. This means that the proofs would equally hold if there were simpler transitions at these positions, such as in Ascon. Alternatively, the domain separation can be done by using a different primitive, as in GIBBON and HANUMAN, or a slightly more elaborated padding, as in BLNK, ICEPOLE, and Keyak;
- The extra permutation evaluations at the initialization and finalization of NORX are not strictly necessary: in the proof we consider the monotone event that no state collides assuming no earlier state collision occurred. For instance, in the analysis of $\mathcal{D}\mathsf{hit}$ in the proof of Lemma 3, we necessarily have a new input to p at some point, and *consequently* all next inputs to p are new (except with some probability);
- NORX starts by initializing the state with $\mathsf{init}(K, N) = (K\|N\|0^{b-\kappa-\nu}) \oplus \mathsf{const}$ for some constant const and then permuting this value. Placing the key and nonce at different positions of the state does not influence the security analysis. The proof would also work if, for instance, the header is preceded with $K\|N$ or a properly padded version thereof and the starting state is 0^b;
- In a similar fashion, there is no problem in defining the tag to be a different τ bits of the final state; for instance, the rightmost τ bits;
- Key additions into the capacity part *after* the first permutation are harmless for the mode security proof. Particularly, as long as these are done at fixed positions, these have the same effect as XORing a domain separation constant.

These five modifications allow one to generalize the proof of NORX to Ascon, CBEAM and STRIBOB, ICEPOLE, Keyak, and two PRIMATEs, GIBBON and HANUMAN. The only major difference lies in the fact none of these designs accommodates a trailer, hence all are functions of the form

$$(C, A) \longleftarrow \mathcal{E}_K(N; H, M) \qquad \text{and} \qquad M/\perp \longleftarrow \mathcal{D}_K(N; H, C; A)\,,$$

except for one instance of ICEPOLE which accommodates a secret message number. Additionally, these designs have $\sigma_\delta \leq \lambda_\delta + q_\delta$ for $\delta \in \{\mathcal{D}, \mathcal{E}\}$ (or $\sigma_\delta \leq \lambda_\delta + 2q_\delta$ for CBEAM/STRIBOB). We always write $H = (H_1, \ldots, H_u)$ and $M = (M_1, \ldots, M_v)$ whenever notation permits. In below sections we elaborate on these

designs separately, where we slightly deviate from the alphabetical order to suit the presentation. Diagrams of all modes are given in Figure 2. The parameters and achieved provable security levels of the schemes are given in Table 1.

4.1 Ascon

Ascon is a submission by Dobraunig et al. [16] and is depicted in Figure 2a. It is originally defined based on two permutations p_1, p_2 that differ in the number of underlying rounds. We discard this difference, considering Ascon with one permutation p.

Ascon initializes its state using init that maps (K, N) to $(0^{b-\kappa-\nu} \| K \| N) \oplus \mathsf{const}$, where const is determined by some design-specific parameters set prior to the security experiment. The header and message can be of arbitrary length, and are padded to length a multiple of r bits using 10^*-padding. An XOR with 1 separates header processing from message processing. From the above observations, it is clear that the proofs of NORX directly carry over to Ascon.

4.2 ICEPOLE

ICEPOLE is a submission by Morawiecki et al. [22] and is depicted in Figure 2c. It is originally defined based on two permutations, p_1 and p_2, that differ in the number of underlying rounds. We discard this difference, considering ICEPOLE with one permutation p.

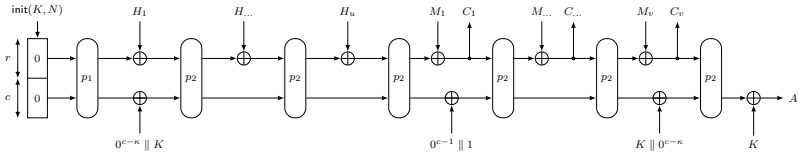
ICEPOLE initializes its state as NORX does, be it with a different constant. The header and message can be of arbitrary length, and are padded as follows. Every block is first appended with a frame bit: $\mathtt{0}$ for header blocks $H_1, \ldots, H_{u-1}$ and message block M_v, and $\mathtt{1}$ for header block H_u and message blocks $M_1, \ldots, M_{v-1}$. Then, the blocks are padded to length a multiple of r bits using 10^*-padding. In other words, every padded block of r bits contains at most $r - 2$ data bits. This form of domain separation using frame bits suffices for the proof to go through. One variant of ICEPOLE also allows for a secret message number M_{secret}, which consists of one block and is encrypted prior to the processing of the header, similar to the message. As this secret message number is of fixed length, no domain separation is required and the proof can easily be adapted. From above observations, it is clear that the proofs of NORX directly carry over to ICEPOLE. Without going into detail, we note that the same analysis can be generalized to the parallelized mode of ICEPOLE [22].

4.3 Keyak

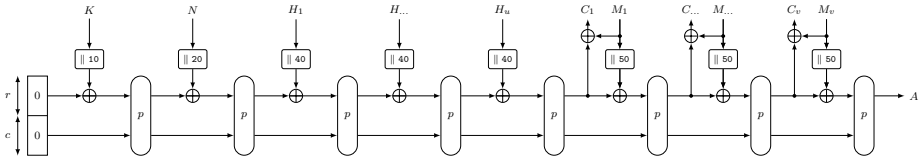
Keyak is a submission by Bertoni et al. [14]. The basic mode for the serial case is depicted in Figure 2d, yet due to its hybrid character it is slightly more general in nature. It is built on top of SpongeWrap [11].

Keyak initializes its state by 0^b, and concatenates K, N, and H using a special padding rule:

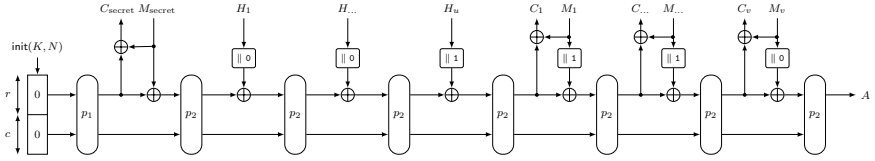
$$\mathsf{H}_{\text{pad}}(K, N, H) = \mathsf{keypack}(K, 240) \,\|\, \mathsf{enc}_8(1) \,\|\, \mathsf{enc}_8(0) \,\|\, N \,\|\, H\,,$$



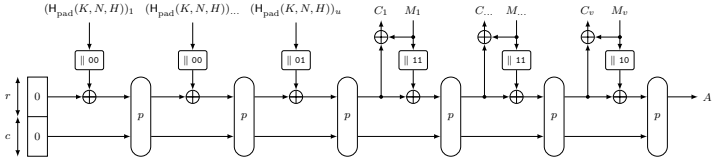
(a) Ascon



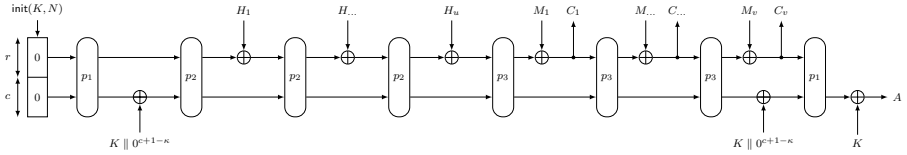
(b) BLNK (used in CBEAM and STRIBOB)



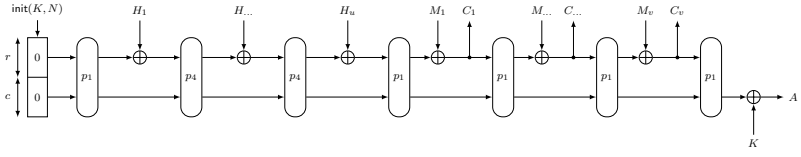
(c) ICEPOLE



(d) Keyak



(e) GIBBON (PRIMATEs)



(f) HANUMAN (PRIMATEs)

Fig. 2: CAESAR submission modes discussed in Section 4

where $\mathsf{enc}_8(x)$ is an encoding of x as a byte and $\mathsf{keypack}(K, \ell) = \mathsf{enc}_8(\ell/8)\|K\|$ $10^{-\kappa-1 \bmod (\ell-8)}$. The key-nonce-header combination $\mathsf{H}_{\mathrm{pad}}(K, N, H)$ and message M can be of arbitrary length, and are padded as follows: first, every block is appended with two frame bits, being 00 for header blocks $(\mathsf{H}_{\mathrm{pad}}(K, N, H))_1, \ldots,$ $(\mathsf{H}_{\mathrm{pad}}(K, N, H))_{u-1}$ and 01 for $(\mathsf{H}_{\mathrm{pad}}(K, N, H))_u$, and 11 for message blocks $M_1, \ldots, M_{v-1}$ and 10 for M_v. Then, the blocks are padded to length a multiple of r bits using 10^*1-padding. In other words, every padded block of r bits contains at most $r - 2$ data bits. This form of domain separation using frame bits suffices for the proof to go through. Due to above observations, our proof readily generalizes to SpongeWrap [11] and DuplexWrap [14], and thus to Keyak. Without going into detail, we note that the same analysis can be generalized to the parallelized mode of Keyak [14]. Additionally, Keyak also supports sessions, where the state is re-used for a next evaluation. Our proof generalizes to this case, simply with a more extended description of (1).

4.4 BLNK (CBEAM and STRIBOB)

CBEAM and STRIBOB are submissions by Saarinen [25, 27–29]. Minaud identified an attack on CBEAM [21], but we focus on the modes of operation. Both modes are based on the BLNK Sponge mode [26], which is depicted in Figure 2b.

The BLNK mode initializes its state by 0^b, compresses K into the state (using one or two permutation calls, depending on κ), and does the same with N. Then, the mode is similar to SpongeWrap [11], though using a slightly more involved domain separation system similar to the one of NORX. Due to above observations, our proof readily generalizes to BLNK [26], and thus to CBEAM and STRIBOB.

4.5 PRIMATEs: GIBBON and HANUMAN

PRIMATEs is a submission by Andreeva et al. [2], and consists of three algorithms: APE, GIBBON, and HANUMAN. The APE mode is the more robust one, and significantly differs from the other two, and from the other CAESAR submissions discussed in this work, in the way that ciphertexts are derived and because the mode is secure against nonce misusing adversaries up to common prefix [3]. We now focus on GIBBON and HANUMAN, which are depicted in Figures 2e and 2f. GIBBON is based on three related permutations $\mathbf{p} = (p_1, p_2, p_3)$, where the difference in p_2, p_3 is used as domain separation of the header compression and message encryption phases (the difference of p_1 from (p_2, p_3) is irrelevant for the mode security analysis). Similarly, HANUMAN uses two related permutations $\mathbf{p} = (p_1, p_2)$ for domain separation.

GIBBON and HANUMAN initialize their state using init that maps (K, N) to $0^{b-\kappa-\nu}\|K\|N$. The header and message can be of arbitrary length, and are padded to length a multiple of r bits using 10^*-padding. In case the true header (or message) happens to be a multiple of r bits long, the 10^*-padding is considered to spill over into the capacity. From above observations, it is clear that the proofs of NORX directly carry over to GIBBON and HANUMAN. A small

difference appears due to the usage of two different permutations: we need to make two PRP-PRF switches for each world. Concretely this means that the first term in Theorem 1 becomes $\frac{5(q_p+\sigma_\mathcal{E})^2}{2^{b+1}}$ and the first term in Theorem 2 becomes $\frac{3(q_p+\sigma_\mathcal{E}+\sigma_\mathcal{D})^2}{2^{b+1}}$.

5 Conclusions

In this work we analyzed one of the Sponge-based authenticated encryption designs in detail, NORX, and proved that it achieves security of approximately $\min\{2^{b/2}, 2^c, 2^\kappa\}$, significantly improving upon the traditional bound of $\min\{2^{c/2}, 2^\kappa\}$. Additionally, we showed that this proof straightforwardly generalizes to five other CAESAR modes, Ascon, BLNK (of CBEAM/STRIBOB), ICEPOLE, Keyak, and PRIMATEs. Our findings indicate an overly conservative parameter choice made by the designers, implying that some designs can improve speed by a factor of 4 at barely any security loss.

It is expected that the security proofs also generalize to the modes of Artemia [1] and π-Cipher [17]. However, they deviate slightly more from the other designs. Artemia is based on the JH hash function [31] and XORs data blocks in both the rate and capacity part. It does not use domain separations, rather it encodes the lengths of the inputs into the padding at the end [5]. Therefore, a generalization of the proof of NORX to Artemia is not entirely straightforward. π-Cipher, on the other hand, is structurally different in the way it maintains state. A so-called "common internal state" is used throughout the evaluation. For the processing of the header (or similarly the message) the state is forked into u chains to process $H_1, \ldots, H_u$ in parallel, resulting in u tag values, which are added into the common internal state. Due to this design property, the deviation of π-Cipher from NORX is too large to simply claim that the proof carries over.

The results in this work are derived in the ideal permutation model, where the underlying primitive is assumed to be ideal. We acknowledge that this model does not perfectly reflect the properties of the primitives. For instance, it is stated by the designers of Ascon, NORX, and PRIMATEs that non-random (but harmless) properties of the underlying permutation exist. Furthermore, it is important to realize that the proofs of security for the modes of operation in the ideal model do not have a direct connection with security analysis performed on the permutations, as is the case with block ciphers modes of operation. Nevertheless, we can use these proofs as heuristics to guide cryptanalysts to focus on the underlying permutations, rather than the modes themselves.

Acknowledgments. The authors would like to thank their co-designers of NORX and PRIMATEs and the designers of Ascon and Keyak for the discussions. In particular, we thank Samuel Neves for his useful comments. This work was supported in part by the Research Fund KU Leuven, OT/13/071, and in part by the Research Council KU Leuven: GOA TENSE (GOA/11/007). Atul Luykx is supported by a Ph.D. Fellowship from the Institute for the Promotion of Innovation through Science and Technology in Flanders (IWT-Vlaanderen).

Bart Mennink is a Postdoctoral Fellow of the Research Foundation – Flanders (FWO).

References

1. Alizadeh, J., Aref, M., Bagheri, N.: Artemia v1 (2014), submission to CAESAR competition
2. Andreeva, E., Bilgin, B., Bogdanov, A., Luykx, A., Mendel, F., Mennink, B., Mouha, N., Wang, Q., Yasuda, K.: PRIMATEs v1 (2014), submission to CAESAR competition
3. Andreeva, E., Bilgin, B., Bogdanov, A., Luykx, A., Mennink, B., Mouha, N., Yasuda, K.: APE: Authenticated permutation-based encryption for lightweight cryptography. In: Cid, C., Rechberger, C. (eds.) FSE. Lecture Notes in Computer Science, Springer (2014)
4. Aumasson, J., Jovanovic, P., Neves, S.: NORX v1 (2014), submission to CAESAR competition
5. Bagheri, N.: Padding of Artemia (2014), CAESAR mailing list
6. Bellare, M., Namprempre, C.: Authenticated encryption: Relations among notions and analysis of the generic composition paradigm. J. Cryptology 21(4), 469–491 (2008)
7. Bellare, M., Rogaway, P.: The security of triple encryption and a framework for code-based game-playing proofs. In: Vaudenay, S. (ed.) EUROCRYPT. Lecture Notes in Computer Science, vol. 4004, pp. 409–426. Springer (2006)
8. Bellare, M., Rogaway, P., Wagner, D.: The EAX mode of operation. In: Roy, B.K., Meier, W. (eds.) FSE. Lecture Notes in Computer Science, vol. 3017, pp. 389–407. Springer (2004)
9. Benaloh, J. (ed.): Topics in Cryptology - CT-RSA 2014 - The Cryptographer's Track at the RSA Conference 2014, San Francisco, CA, USA, February 25-28, 2014. Proceedings, Lecture Notes in Computer Science, vol. 8366. Springer (2014)
10. Bertoni, G., Daemen, J., Peeters, M., Van Assche, G.: Sponge-based pseudo-random number generators. In: Mangard, S., Standaert, F.X. (eds.) CHES. Lecture Notes in Computer Science, vol. 6225, pp. 33–47. Springer (2010)
11. Bertoni, G., Daemen, J., Peeters, M., Van Assche, G.: Duplexing the sponge: Single-pass authenticated encryption and other applications. In: Miri, A., Vaudenay, S. (eds.) Selected Areas in Cryptography. Lecture Notes in Computer Science, vol. 7118, pp. 320–337. Springer (2011)
12. Bertoni, G., Daemen, J., Peeters, M., Van Assche, G.: On the security of the keyed sponge construction. Symmetric Key Encryption Workshop (SKEW 2011) (2011)
13. Bertoni, G., Daemen, J., Peeters, M., Van Assche, G.: Sponge functions (ECRYPT Hash Function Workshop 2007)
14. Bertoni, G., Daemen, J., Peeters, M., Van Assche, G., Van Keer, R.: Keyak v1 (2014), submission to CAESAR competition
15. CAESAR: Competition for Authenticated Encryption: Security, Applicability, and Robustness (May 2014), `http://competitions.cr.yp.to/caesar.html`
16. Dobraunig, C., Eichlseder, M., Mendel, F., Schläffer, M.: Ascon v1 (2014), submission to CAESAR competition
17. Gligoroski, D., Mihajloska, H., Samardjiska, S., Jacobsen, H., El-Hadedy, M., Jensen, R.: π-Cipher v1 (2014), submission to CAESAR competition

18. Iwata, T., Ohashi, K., Minematsu, K.: Breaking and repairing GCM security proofs. In: Safavi-Naini, R., Canetti, R. (eds.) CRYPTO. Lecture Notes in Computer Science, vol. 7417, pp. 31–49. Springer (2012)
19. Jovanovic, P., Luykx, A., Mennink, B.: Beyond $2^{c/2}$ security in sponge-based authenticated encryption modes. Cryptology ePrint Archive, Report 2014/373 (2014), full version of this paper
20. Krovetz, T., Rogaway, P.: The software performance of authenticated-encryption modes. In: Joux, A. (ed.) FSE. Lecture Notes in Computer Science, vol. 6733, pp. 306–327. Springer (2011)
21. Minaud, B.: Re: CBEAM Withdrawn as of today! (2014), CAESAR mailing list
22. Morawiecki, P., Gaj, K., Homsirikamol, E., Matusiewicz, K., Pieprzyk, J., Rogawski, M., Srebrny, M., Wójcik, M.: ICEPOLE v1 (2014), submission to CAESAR competition
23. Rogaway, P.: Efficient instantiations of tweakable blockciphers and refinements to modes OCB and PMAC. In: Lee, P.J. (ed.) ASIACRYPT. Lecture Notes in Computer Science, vol. 3329, pp. 16–31. Springer (2004)
24. Rogaway, P., Bellare, M., Black, J., Krovetz, T.: OCB: a block-cipher mode of operation for efficient authenticated encryption. In: Reiter, M.K., Samarati, P. (eds.) ACM Conference on Computer and Communications Security. pp. 196–205. ACM (2001)
25. Saarinen, M.: Authenticated encryption from GOST R 34.11-2012 LPS permutation. In: CTCrypt 2014 (2014)
26. Saarinen, M.: Beyond modes: Building a secure record protocol from a cryptographic sponge permutation. In: Benaloh [9], pp. 270–285
27. Saarinen, M.: CBEAM: Efficient authenticated encryption from feebly one-way ϕ functions. In: Benaloh [9], pp. 251–269
28. Saarinen, M.: CBEAM r1 (2014), submission to CAESAR competition
29. Saarinen, M.: STRIBOB r1 (2014), submission to CAESAR competition
30. Whiting, D., Housley, R., Ferguson, N.: AES Encryption and Authentication Using CTR Mode and CBC-MAC. IEEE 802.11-02/001r2 (2002)
31. Wu, H.: The Hash Function JH (2011), submission to NIST's SHA-3 competition

A Proof Technique Used in Lemma 2

Formally, the proof technique used in Lemma 2 relies on the following paradigm. Note that there is an ordering of the $q_p + \sigma_{\mathcal{E}}$ primitive queries, and we can reformulate $\mathsf{guess}(\ell)$, $\mathsf{hit}(\ell)$, $\mathsf{key}(\ell)$, and $\mathsf{multi}(\ell)$ for $\ell = 1, \ldots, q_p + \sigma_{\mathcal{E}}$ analogously. Defining $\mathsf{event}(\ell) = \mathsf{guess}(\ell) \vee \mathsf{hit}(\ell)$ and $\mathsf{help}(\ell) = \mathsf{key}(\ell) \vee \mathsf{multi}(\ell)$, then

$$\begin{aligned}\mathbf{Pr}\left(\mathsf{event}\right) \leq\ & \mathbf{Pr}\left(\mathsf{event}(q_p{+}\sigma_{\mathcal{E}}) \mid \neg\mathsf{event}(1 \cdots q_p{+}\sigma_{\mathcal{E}}{-}1) \wedge \neg\mathsf{help}(1 \cdots q_p{+}\sigma_{\mathcal{E}})\right) + \\ & \mathbf{Pr}\left(\mathsf{event}(1 \cdots q_p{+}\sigma_{\mathcal{E}}{-}1) \vee \mathsf{help}(1 \cdots q_p{+}\sigma_{\mathcal{E}})\right) ,\end{aligned}$$

and inductively $\mathbf{Pr}\left(\mathsf{event}\right) \leq \sum_{\ell=1}^{q_p+\sigma_{\mathcal{E}}} \mathbf{Pr}\left(\mathsf{event}(\ell) \mid \neg\mathsf{event}(1 \cdots \ell - 1) \wedge \neg\mathsf{help}(1 \cdots \ell)\right) + \mathbf{Pr}\left(\mathsf{help}(\ell) \mid \neg\mathsf{help}(1 \cdots \ell - 1)\right)$. This formulation would however merely reduce the readability of the proof.