APE: Authenticated Permutation-Based Encryption for Lightweight Cryptography

Elena Andreeva[1], Begül Bilgin[1,2], Andrey Bogdanov[3], Atul Luykx[1], Bart Mennink[1(✉)], Nicky Mouha[1], and Kan Yasuda[1,4]

[1] Department of Electrical Engineering, ESAT/COSIC, KU Leuven and iMinds, Ghents, Belgium
{elena.andreeva,begul.bilgin,atul.luykx, bart.mennink,nicky.mouha}@esat.kuleuven.be
[2] Faculty of EEMCS, DIES, UTwente, Enschede, Netherlands
[3] Department of Mathematics, Technical University of Denmark, Lyngby, Denmark
anbog@dtu.dk
[4] NTT Secure Platform Laboratories, Tokyo, Japan
yasuda.kan@lab.ntt.co.jp

Abstract. The domain of lightweight cryptography focuses on cryptographic algorithms for extremely constrained devices. It is very costly to avoid nonce reuse in such environments, because this requires either a hardware source of randomness, or non-volatile memory to store a counter. At the same time, a lot of cryptographic schemes actually require the nonce assumption for their security. In this paper, we propose APE as the first permutation-based authenticated encryption scheme that is resistant against nonce misuse. We formally prove that APE is secure, based on the security of the underlying permutation. To decrypt, APE processes the ciphertext blocks in reverse order, and uses inverse permutation calls. APE therefore requires a permutation that is both efficient for forward and inverse calls. We instantiate APE with the permutations of three recent lightweight hash function designs: Quark, Photon, and Spongent. For any of these permutations, an implementation that supports both encryption and decryption requires less than 1.9 kGE and 2.8 kGE for 80-bit and 128-bit security levels, respectively.

Keywords: APE · Authenticated encryption · Sponge function · Online · Deterministic · Permutation-based · Misuse resistant

1 Introduction

In constrained environments, conventional solutions to cryptographic problems are prohibitively expensive to implement. Lightweight cryptography deals with cryptographic algorithms within the stringent requirements imposed by devices such as low-cost smart cards, sensor networks, and electronic body implants where energy, power, or hardware area consumption can be heavily restricted.

Although symmetric-key cryptography predominantly makes use of solutions based on block ciphers, recently permutation-based constructions [9] are gaining

C. Cid and C. Rechberger (Eds.): FSE 2014, LNCS 8540, pp. 168–186, 2015.
DOI: 10.1007/978-3-662-46706-0_9

traction for a wide range of platforms, and on lightweight devices in particular. Lightweight permutation-based hash functions include Gluon [6], Photon [19], Quark [2,3], and Spongent [11].

Lightweight applications in practice require not only hash functions but also secret-key cryptographic functions, such as authenticated encryption (AE). AE is a cryptographic primitive that guarantees two security goals: privacy and integrity. The prevalent solutions in this direction are block cipher based [24,28,31]. Permutation-based AE schemes were only recently proposed, such as the deterministic key-wrap scheme [21] of Khovratovich and SpongeWrap [7,10] of the Keccak team.

These two constructions unfortunately have their limitations. With the key-wrap scheme [21], the message length is restricted to one block by design. While sufficient for key wrapping [29], this construction cannot handle arbitrary-length data and is therefore not a full AE scheme. SpongeWrap [7,10] can encrypt messages of varying lengths but relies on the uniqueness of the nonce value: failure to ensure so makes it possible to reuse the keystream of the encryption. For example, if a pair of plaintexts share a common prefix, the XOR of the first pair of plaintext blocks after this common prefix is leaked.

In Rogaway's security formalism of nonce-based encryption [26,27], the nonce is considered to be unique for every evaluation. While this approach has theoretical merits, in practice it is challenging to ensure that a nonce is never reused. This is especially the case in lightweight cryptography, as a nonce is realized either by keeping a state (and correctly updating it) or by providing a hardware source of randomness. Indeed, nonce misuse is a security threat in plenty of practical applications, not necessarily limited to the lightweight setting. Examples include flawed implementations of nonces [13,15,22,23,32], bad management of nonces by the user, and backup resets or virtual machine clones when the nonce is stored as a counter.

Nonce *misuse resistance* has become an important criterion in the design of AE schemes. The CAESAR competition [14] considers misuse resistance in detail for their selection of a portfolio of AE algorithms. The problem of nonce misuse has also been addressed by the recent deterministic AE scheme SIV [29], by the online AE scheme McOE [18], and in part by the aforementioned deterministic key-wrap scheme [21]. However, there are currently no permutation-based AE schemes that are resistant to nonce misuse.

Our Contributions. In this work we introduce APE (Authenticated Permutation-based Encryption). APE is the first permutation-based *and* nonce misuse resistant authenticated encryption scheme. APE is inspired by SpongeWrap [7,10], but differs in several fundamental aspects in order to achieve misuse resistance. Most importantly, in APE the ciphertexts are extracted from the state, whereas SpongeWrap generates a keystream to perform the encryption. APE encryption processes data in an online manner, whereas decryption is done backwards using the inverse of the permutation. APE is formally introduced in Sect. 3. Here, we initially focus on associated data and messages of an integral

number of blocks. In Appendix A, we show how APE can be generalized at almost no extra cost to handle fractional associated data and message blocks.

We prove that APE achieves privacy and integrity up to about $2^{c/2}$ queries, where c is the capacity parameter of APE. This result is proven in two different models: first, in Sect. 4, we prove security of APE in the ideal permutation model, where the underlying permutation is assumed to behave perfectly random. Next, in Sect. 5, we consider APE *with block ciphers*, which is a generalization of APE where the permutation with surrounding key XORs is replaced with a block cipher call. We use the results from the ideal model to prove that APE with block ciphers is secure in the standard model up to roughly $2^{c/2}$ queries as well.

APE is designed to be well suited for lightweight applications. However, APE decrypts in inverse direction and requires an efficiently invertible permutation. In Sect. 6, we implement APE in less than 1.9 kGE and 2.8 kGE for 80-bit and 128-bit security respectively with the permutations of Quark, Photon, and Spongent. The results indicate that including the inverses of these permutations only leads to a marginal increase of the size of the implementation when compared to the cost of providing a hardware source of randomness to generate nonces.

2 Notation

Set $\mathbf{R} := \{0,1\}^r$ and $\mathbf{C} := \{0,1\}^c$. Given two strings A and B, we use $A\|B$ and AB interchangeably, so for example $AB = A\|B \in \mathbf{R} \times \mathbf{C} \cong \{0,1\}^{r+c}$. Given $X \in \mathbf{R} \times \mathbf{C}$, X_r denotes its projection onto $\mathbf{R}$, also known as its rate part, and X_c denotes its projection onto $\mathbf{C}$, or capacity part. We write $0 \in \mathbf{R}$ for a shorthand for $00\cdots0 \in \mathbf{R}$ and $1 \in \mathbf{C}$ for $00\cdots01 \in \mathbf{C}$. The symbol $\oplus$ denotes the bitwise XOR operation of two (or more) strings.

An element of $\mathbf{R}$ is called a block. Let $\mathbf{R}^*$ denote the set of strings whose length is a multiple of r, with at most $2^{c/2}$ blocks. This explicit bound of $2^{c/2}$ is needed in order to define a random function as being sampled over a finite set of functions. Note that the bounds we prove become trivial for queries of length $2^{c/2}$ blocks. Similarly, let $\mathbf{R}^+$ denote the set of strings whose length is a positive multiple of r, with at most $2^{c/2}$ blocks. Given $M \in \mathbf{R}^+$, we divide it into blocks and write $M[1]M[2]\cdots M[w] \leftarrow M$, where each $M[i]$ is a block and w the block length of the string M.

Let $\mathcal{A}$ be some class of adversaries. For convenience, we use the notation

$$\Delta_{\mathcal{A}}[f,g] := \sup_{A\in\mathcal{A}} \left|\Pr[A^f = 1] - \Pr[A^g = 1]\right|$$

to denote the supremum of the distinguishing advantages over all adversaries distinguishing f and g. Providing access to multiple algorithms is denoted with

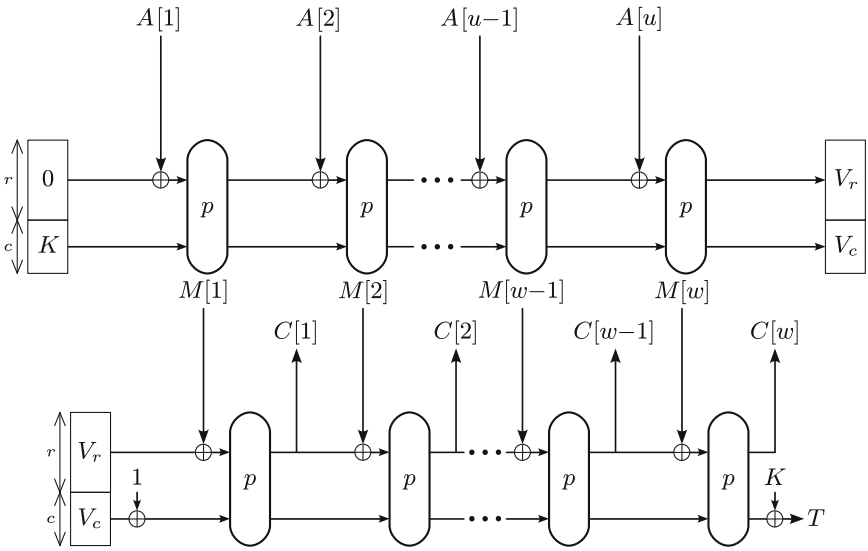

Fig. 1. The APE mode of operation (encryption). If there is no associated data ($A = \varnothing$), we have $V_r := 0$ and $V_c := K$.

a comma, e.g. $\Delta[f_1, f_2\,;\, g_1, g_2]$ denotes distinguishing the combination of f_1 and f_2 from the combination of g_1 and g_2.

3 APE Authenticated Encryption Mode

We now define our APE mode for the case of plaintexts and associated data of length a multiple of the block size. We refer to Sect. A for the generalization of APE to fractional data blocks. APE iterates a fixed permutation $p : \mathbf{R} \times \mathbf{C} \to \mathbf{R} \times \mathbf{C}$ in a way similar to the sponge construction. The permutation p is the only underlying cryptographic primitive used by APE. A diagram of APE is given in Fig. 1 and an algorithmic description in Fig. 2.

The encryption algorithm $\mathcal{E}$ takes as input a key $K \in \mathcal{K} = \mathbf{C}$, associated data $A \in \mathbf{R}^*$, and a message $M \in \mathbf{R}^+$, and returns a ciphertext $C \in \mathbf{R}^+$ and a tag $T \in \mathbf{C}$, as $(C, T) \leftarrow \mathcal{E}_K(A, M)$. On the other hand, $\mathcal{D}$ takes as input a key $K \in \mathbf{C}$, associated data $A \in \mathbf{R}^*$, a ciphertext $C \in \mathbf{R}^+$, and a tag $T \in \mathbf{C}$, and returns either a message $M \in \mathbf{R}^+$ or the reject symbol $\perp$, as $M/\perp \leftarrow \mathcal{D}_K(A, C, T)$. The two functionalities are sound, in the sense that whenever we encrypt a message as $(C, T) \leftarrow \mathcal{E}_K(A, M)$, we always get the message back, not $\perp$, via the decryption process $M \leftarrow \mathcal{D}_K(A, C, T)$.

In APE, the inclusion of a nonce is optional. If a nonce is required, it can be included as part of the associated data. Care must be taken when allowing nonces of varying lengths, as the nonce and the associated data should be clearly distinguishable.

Algorithm 1. $\mathcal{E}_K(A, M)$

Input: $K \in \mathbf{C}$, $A \in \mathbf{R}^*$, $M \in \mathbf{R}^+$
Output: $C \in \mathbf{R}^+$, $T \in \mathbf{C}$

$V \leftarrow (0, K) \in \mathbf{R} \times \mathbf{C}$
if $A \neq \varnothing$ **then**
  $A[1]A[2] \cdots A[u] \leftarrow A$
  for $i = 1$ **to** u **do**
    $V \leftarrow p\big(A[i] \oplus V_r, V_c\big)$
  end
end
$V \leftarrow V \oplus (0, 1)$
$M[1]M[2] \cdots M[w] \leftarrow M$
for $i = 1$ **to** w **do**
  $V \leftarrow p\big(M[i] \oplus V_r, V_c\big)$
  $C[i] \leftarrow V_r$
end
$C \leftarrow C[1]C[2] \cdots C[w]$
$T \leftarrow V_c \oplus K$
return (C, T)

Algorithm 2. $\mathcal{D}_K(A, C, T)$

Input: $K \in \mathbf{C}$, $A \in \mathbf{R}^*$, $C \in \mathbf{R}^+$, $T \in \mathbf{C}$
Output: $M \in \mathbf{R}^+$ or $\perp$

$V \leftarrow (0, K) \in \mathbf{R} \times \mathbf{C}$
if $A \neq \varnothing$ **then**
  $A[1]A[2] \cdots A[u] \leftarrow A$
  for $i = 1$ **to** u **do**
    $V \leftarrow p\big(A[i] \oplus V_r, V_c\big)$
  end
end
$C[1]C[2] \cdots C[w] \leftarrow C$
$C[0] \leftarrow V_r$
$V \leftarrow (C[w], K \oplus T)$
for $i = w$ **to** 1 **do**
  $V \leftarrow p^{-1}(V)$
  $M[i] \leftarrow C[i-1] \oplus V_r$
  $V \leftarrow C[i-1] \| V_c$
end
$M \leftarrow M[1]M[2] \cdots M[w]$
if $V_c = V_c \oplus 1$ **then**
  return M
else
  return $\perp$
end

Fig. 2. The encryption $\mathcal{E}_K(A, M)$ and decryption $\mathcal{D}_K(A, C, T)$ algorithms of APE.

4 Privacy and Integrity of APE

We prove that APE satisfies privacy under chosen plaintext attacks (CPA) and integrity security up to about $c/2$ bits. Before doing so, in Sect. 4.1 we present the security model, where we formalize the notion of an ideal online function, and where we introduce the CPA and integrity security definitions. Then, privacy is proven in Sect. 4.2 and integrity in Sect. 4.3. The security results in this section assume that the underlying permutation p is ideal. Later, in Sect. 5, we consider the security of APE with block ciphers in the standard model.

4.1 Security Model

Let $\mathrm{Perm}(n)$ be the set of all permutations on n bits. By $\perp$, we denote a function that returns $\perp$ on every input. When writing $x \xleftarrow{\$} \mathbf{X}$ for some finite set $\mathbf{X}$ we mean that x is sampled uniformly from $\mathbf{X}$. To avoid confusion, for $X \in \mathbf{R} \times \mathbf{C}$ we sometimes write $[X]_c := X_c$ to denote the projection of X onto $\mathbf{C}$.

Online functions were first introduced in [4]. We deviate slightly from their approach by explicitly defining our ideal online function in terms of random functions.

Definition 1 (Ideal Online Function). *Let* $g : \mathbf{R}^* \times \mathbf{R}^* \to \mathbf{R}$ *and* $g' : \mathbf{R}^* \times \mathbf{R}^* \to \mathbf{C}$ *be random functions. Then, on input of* (A, M) *with* $w = |M|/r$*, we define* $\$: \mathbf{R}^* \times \mathbf{R}^+ \to \mathbf{R}^+ \times \mathbf{C}$ *as*

$$\$(A, M[1]\|M[2]\|\cdots\|M[w]) = (C[1]\|C[2]\|\cdots\|C[w], T),$$

where

$$\begin{aligned} C[j] &= g(A, M[1]\|\cdots\|M[j]) \text{ for } j = 1, \ldots, w, \\ T &= g'(A, M). \end{aligned}$$

Notice that the above function is actually online: prefixes of outputs remain the same if prefixes of the inputs remain constant. Furthermore, if the associated data in the ideal online function is unique for each invocation of the function, then we achieve full privacy. This is because the inputs to g and g' will then be unique for each invocation, and since g and g' are random functions, we get outputs that are independent and uniformly distributed. By comparing our scheme to the above ideal online function we will automatically achieve the notions of CPA security and integrity from Rogaway and Zhang [30] and Fleischmann et al. [18].

Definition 2. *Let* $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ *denote an AE scheme. The CPA advantage of a distinguisher* D *is defined as*

$$\mathbf{Adv}^{\mathrm{cpa}}_{\Pi}(D) = \left| \begin{array}{r} \Pr\left[p \xleftarrow{\$} \mathrm{Perm}(r+c)\,,\, K \xleftarrow{\$} \mathcal{K} \;:\; D^{\mathcal{E}_K, \perp, p, p^{-1}} = 1\right] - \\ \Pr\left[p \xleftarrow{\$} \mathrm{Perm}(r+c) \;:\; D^{\$, \perp, p, p^{-1}} = 1\right] \end{array} \right|.$$

By $\mathbf{Adv}^{\mathrm{cpa}}_{\Pi}(q, m)$ *we denote the supremum taken over all distinguishers making* q *queries of total length* m *blocks.*

Definition 3. *Let* $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ *denote an AE scheme. The integrity advantage of a distinguisher* D *is defined as*

$$\mathbf{Adv}^{\mathrm{int}}_{\Pi}(D) = \left| \begin{array}{r} \Pr\left[p \xleftarrow{\$} \mathrm{Perm}(r+c)\,,\, K \xleftarrow{\$} \mathcal{K} \;:\; D^{\mathcal{E}_K, \mathcal{D}_K, p, p^{-1}} = 1\right] - \\ \Pr\left[p \xleftarrow{\$} \mathrm{Perm}(r+c)\,,\, K \xleftarrow{\$} \mathcal{K} \;:\; D^{\mathcal{E}_K, \perp, p, p^{-1}} = 1\right] \end{array} \right|.$$

We assume that the distinguisher does not make a decryption query (A, C, T) *if it ever obtained* $(C, T) \leftarrow \mathcal{E}_K(A, M)$ *for some* M*. By* $\mathbf{Adv}^{\mathrm{int}}_{\Pi}(q, m)$ *we denote the supremum taken over all distinguishers making* q *queries of total length* m *blocks.*

4.2 Privacy

In this section, we present a privacy security proof for APE.

Theorem 1. *Let $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be the APE construction. Then,*

$$\mathbf{Adv}^{\mathrm{cpa}}_{\Pi}(q, m) \leq \frac{m^2}{2^{r+c}} + \frac{m(m+1)}{2^c}.$$

Proof. We consider the strongest possible type of distinguishers: let D be any information-theoretic distinguisher which has unbounded computational power and whose complexity is measured solely by the number of queries it makes to its oracles. Without loss of generality, we restrict ourselves to distinguishers that do not ask "trivial" queries, queries to which it knows the answer in advance.

As a first step, we make a PRP-PRF switch [5]: we move from random permutation (p, p^{-1}) to a primitive (f, f^{-1}) defined as follows. This primitive maintains an initially empty list of responses, $\mathcal{F}$, and we denote its domain/range by $\mathsf{dom}(\mathcal{F})/\mathsf{rng}(\mathcal{F})$. Now, on a non-trivial forward query $f(x)$, the response y is randomly drawn from $\mathbf{R} \times \mathbf{C}$. The primitive aborts if y happens to be in $\mathsf{rng}(\mathcal{F})$ already; otherwise, the fresh tuple (x, y) is added to $\mathcal{F}$. Similarly for inverse queries to f^{-1}. Clearly, (p, p^{-1}) and (f, f^{-1}) behave identically as long as the latter does not abort. Given that the distinguisher makes at most q queries of total length m blocks (each block corresponds to a new (f, f^{-1})-query), such an abort happens with probability at most $\binom{m}{2}/2^{r+c} \leq m^2/2^{r+c+1}$. We apply this PRP-PRF switch to both the ideal and the real world, and hence we find

$$\Delta_D(\mathcal{E}_K, \perp, p, p^{-1}; \$, \perp, p, p^{-1}) \leq \frac{m^2}{2^{r+c}} + \Delta_D(\mathcal{E}_K, \perp, f, f^{-1}; \$, \perp, f, f^{-1}). \quad (1)$$

In the remainder, we consider D to have oracle access to one of the two worlds: (F, f, f^{-1}), where $F \in \{\mathcal{E}_K, \$\}$ (without loss of generality we can drop the $\perp$).

If f is called by D then we call this a direct f-query, and similar for direct f^{-1}-queries. A call of f by $\mathcal{E}_K$ (as a result of D calling $\mathcal{E}_K$) is called an indirect f-query. When we do not specify whether an f-query is indirect or direct, we mean that it could be either. Note that indirect queries do not occur in the random world $(\$, f, f^{-1})$. Every indirect f-query has a sequence of associated data blocks and message blocks leading up to it (from the $\mathcal{E}_K$-query calling it); we call this sequence the message chain associated to the indirect f-query.

Let $\mathcal{Q}_i$ denote the set of all prefixes of all queries made by D to its F-oracle before the ith (f, f^{-1})-query, where a query (A, M) results in prefixes $\{A[1], A[1]\|A[2], \ldots, A\|M\}$. Regarding all *direct* queries before the ith query, we denote by X_i^{dir} the set of all capacity values input to f-queries or output of f^{-1}-queries. For example, a direct forward query $y \leftarrow f(x)$ adds $[x]_c$ to X_i^{dir} and a direct inverse query $x \leftarrow f^{-1}(y)$ adds $[x]_c$ to X_i^{dir}. Similarly, by X_i^{ind} we denote the set of all capacity values input to *indirect* f-queries before the ith f-query. We write $X_i = X_i^{\mathrm{dir}} \cup X_i^{\mathrm{ind}}$, and initialize $X_0^{\mathrm{ind}} = \{K\}$.

We define event $\mathsf{E}_i = \mathsf{E}_i^{\text{dir-}X} \cup \mathsf{E}_i^{\text{ind-}X}$, where

$\mathsf{E}_i^{\text{dir-}X}$: direct query $y \leftarrow f(x)$ or $x \leftarrow f^{-1}(y)$ satisfies $[x]_c \in X_i^{\text{ind}} \cup X_i^{\text{ind}} \oplus 1$,

$\mathsf{E}_i^{\text{ind-}X}$: indirect query $f(x)$ with message chain $(A, M) \notin \mathcal{Q}_i$ satisfies $[f(x)]_c \in X_i \cup X_i \oplus 1$.

We furthermore define

$$\hat{\mathsf{E}}_i := \mathsf{E}_i \cap \bigcap_{j=1}^{i-1} \overline{\mathsf{E}_j}, \text{ and } \mathsf{E} := \bigcup_{i=1}^{m} \hat{\mathsf{E}}_i, \tag{2}$$

where $\overline{\mathsf{E}_j}$ is the complement of E_j.

Now, the remainder of the proof is divided as follows. In Lemma 1 we will prove that $(\mathcal{E}_K, f, f^{-1})$ and $(\$, f, f^{-1})$ are indistinguishable as long as E does not occur. From (1) and the fundamental lemma of game playing [5] we find

$$\mathbf{Adv}_{\Pi}^{\text{cpa}}(q, m) \leq \frac{m^2}{2^{r+c}} + \Pr[D^{\mathcal{E}_K, f, f^{-1}} \text{ sets } \mathsf{E}].$$

Then, in Lemma 2, we will prove that $\Pr[D^{\mathcal{E}_K, f, f^{-1}} \text{ sets } \mathsf{E}] \leq \frac{m(m+1)}{2^c}$, which completes the proof. □

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

Proof. Note that in the ideal world, each direct f-query is new, and is answered with a uniformly randomly drawn response. Now, consider a direct query $f(x)$ in the real world. As the distinguisher does not make trivial queries, it does not coincide with any previous direct query. Additionally, if $[x]_c \in X_i^{\text{ind}} \cup X_i^{\text{ind}} \oplus 1$, where $f(x)$ is the ith f-query, then this would trigger $\mathsf{E}_i^{\text{dir-}X}$, hence we can assume $[x]_c \notin X_i^{\text{ind}} \cup X_i^{\text{ind}} \oplus 1$. This means that the query $f(x)$ is truly new, and its value is independently and uniformly distributed. The same reasoning applies to f^{-1}-queries. Therefore, we only need to consider queries to the big oracle $F \in \{\mathcal{E}_K, \$\}$. Let (A, M) be a query made by the distinguisher. Denote by w the number of blocks of M. Denote the corresponding ciphertext and tag by (C, T).

First consider the case $(A, M[1] \| \cdots \| M[j]) \in \mathcal{Q}_i$ for some $j \in \{0, \ldots, w\}$ and assume j is maximal (we will come back to the case of $(A, *) \notin \mathcal{Q}_i$ later in the proof). Let (A', M') be the corresponding earlier query, so $M[1] \| \cdots \| M[j] = M'[1] \| \cdots \| M'[j]$, and denote its ciphertext and tag by (C', T') and block length by w'. Clearly, in the ideal world $(\$, f, f^{-1})$, we have $C[i] = C'[i]$ for $i = 1, \ldots, j$, but $C[i]$ for $i = j+1, \ldots, w$ and T are uniformly randomly drawn. We will consider how these values are distributed in the real world $(\mathcal{E}_K, f, f^{-1})$. We first consider the general case $j < w$, the case $j = w$ is discussed afterwards.

1. $C[1], \ldots, C[j]$. Also in the real world, these values equal $C'[1], \ldots, C'[j]$, which follows clearly from the specification of $\mathcal{E}_K$. Note that in particular, the state value V equals V' after the jth round.

2. $C[j+1]$. We make a distinction between $j > 0$ and $j = 0$, and start with the former case. Write the indirect query corresponding to the jth round as $f(x)$. The input of the $(j+1)$th query will be $f(x) \oplus (M[j+1], 0)$. Note that $(A, M' \| M[j+1]) \notin \mathcal{Q}$, as this would contradict the fact that j is maximal. Now, assume this $(j+1)$th query has already been made before, i.e. $[f(x)]_c \in X^{\text{dir}} \cup X^{\text{ind}}$. This may be the case (it may even date from before the evaluation of (A, M')), but at this particular time the capacity part $[f(x)]_c$ did not hit any element from $X^{\text{dir}} \cup X^{\text{ind}}$ (otherwise it would have triggered $\mathsf{E}^{\text{ind-}X}$). After this query has been made, there has not been any newer indirect query or any newer direct query whose capacity part hit $[f(x)]_c$ (both cases would have triggered $\mathsf{E}^{\text{dir-}X} \cup \mathsf{E}^{\text{ind-}X}$). Thus, the query corresponding to the $(j+1)$th round is generated independently and uniformly at random.

   Now, in the case $j = 0$, V denotes the state right after the hashing of A ($V = (0, K)$ if $A = \varnothing$). The same story as before applies with the difference that now the input to the $(j+1)$th query is $V \oplus (M[j+1], 1)$. Here we use that by $\overline{\mathsf{E}}$, no other query hit $X_j^{\text{ind}} \oplus 1$ (for direct queries) or $X_j \oplus 1$ (for indirect queries) in the meanwhile, and that X^{ind} is initialized with $\{K\}$.
3. $C[j+2], \ldots, C[w]$. By the above argument, the indirect query made in the $(j+1)$th round of (A, M), say $f(x)$ for the sake of presentation, is responded with a uniformly random answer. This query would have triggered $\mathsf{E}^{\text{ind-}X}$ if $[f(x)]_c \in X_i$. Therefore, we know that also the $(j+2)$th query is truly random and so is $C[j+2]$. The same reasoning applies up to $C[w]$.
4. T. The same reasoning applies: the previous query is responded with a truly random answer $f(x)$. Consequently $T = [f(x)]_c \oplus K$ is random too.

A special treatment is needed for $j = w$. In this case, $C[1], \ldots, C[w]$ equals $C'[1], \ldots, C'[w]$ by construction, but the query producing T is not new. Yet, the distinguisher never made that query itself by virtue of $\overline{\mathsf{E}^{\text{dir-}X}}$, so it never learnt $T \oplus K$. Besides, due to the absence of indirect capacity collisions, $\overline{\mathsf{E}^{\text{ind-}X}}$, every f-query will produce a tag at most once. This means that T will look uniformly random to the distinguisher, as it would look if it were produced by \$.

Finally, we consider the case $(A, *) \notin \mathcal{Q}_i$, hence this is the first time a query for this particular associated data A is made. Then, the above reasoning carries over for $j = 0$ with the simplification that if $A \neq \varnothing$, the value V_c right after the hashing of A can be considered new. □

Lemma 2. $\Pr[D^{\mathcal{E}_K, f, f^{-1}} \textit{ sets } \mathsf{E}] \leq \dfrac{m(m+1)}{2^c}$.

Proof. Inspired by (2), we start bounding $\Pr[\mathsf{E}_i \cap \bigcap_{j=1}^{i-1} \overline{\mathsf{E}_j}]$ for $i \in \{1, \ldots, m\}$. Clearly,

$$\Pr[\mathsf{E}_i \cap \textstyle\bigcap_{j=1}^{i-1} \overline{\mathsf{E}_j}] \leq \Pr[\mathsf{E}_i \mid \textstyle\bigcap_{j=1}^{i-1} \overline{\mathsf{E}_j}].$$

Therefore, we assume $\bigcap_{j=1}^{i-1} \overline{\mathsf{E}_j}$ and consider the probability the ith query triggers E_i.

If the ith query is a direct (forward or inverse) query, it triggers $\mathsf{E}_i^{\text{dir-}X}$ if the distinguisher guesses (in case of forward) or hits (in case of inverse) a capacity part in $X_i^{\text{ind}} \cup X_i^{\text{ind}} \oplus 1$, which happens with probability at most $2|X_i^{\text{ind}}|/2^c$. On the other hand, if the ith query is a new indirect query (i.e. for which $(A, M) \notin \mathcal{Q}_i$) it triggers $\mathsf{E}_i^{\text{ind-}X}$ if $[f(x)]_c \in X_i \cup X_i \oplus 1$. This occurs with probability at most $2|X_i|/2^c$.

As the query is either direct or indirect, we could take the maximum of both values. Given that $|X_i^{\text{ind}}| \leq |X_i| \leq i$, we find:

$$\Pr[\mathsf{E}_i \mid \bigcap_{j=1}^{i-1} \overline{\mathsf{E}_j}] \leq \frac{2i}{2^c}.$$

The result is now obtained by summing over $i = 1, \ldots, m$ (as in (2)). □

4.3 Integrity

In this section, we present an integrity security proof for APE.

Theorem 2. *Let $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be the APE construction. Then,*

$$\mathbf{Adv}_{\Pi}^{\text{int}}(q, m) \leq \frac{m^2}{2^{r+c}} + \frac{2m(m+1)}{2^c}.$$

Proof. The basic idea of the proof is the same as for Theorem 1. Again, let D be any information-theoretic distinguisher. By the PRP-PRF switch, we find

$$\Delta_D(\mathcal{E}_K, \mathcal{D}_K, p, p^{-1}; \mathcal{E}_K, \perp, p, p^{-1}) \leq \frac{m^2}{2^{r+c}} + \Delta_D(\mathcal{E}_K, \mathcal{D}_K, f, f^{-1}; \mathcal{E}_K, \perp, f, f^{-1}). \tag{3}$$

We consider D to have oracle access to one of the two worlds: $(\mathcal{E}_K, F, f, f^{-1})$, where $F \in \{\mathcal{D}_K, \perp\}$.

We use the same notation as in Theorem 1, but slightly more involved definitions are required and we re-introduce them. If f is called by D then we call this a direct f-query, and similar for direct f^{-1}-queries. A call of f by $\mathcal{E}_K$ or $\mathcal{D}_K$ (as a result of D calling them) is called an indirect f-query, and similar for indirect f^{-1}-queries (via $\mathcal{D}_K$). Every indirect f-query has a sequence of associated data blocks and/or message blocks leading up to it (from the $\mathcal{E}_K$- or $\mathcal{D}_K$-query calling it); we call this sequence the message chain associated to the indirect f-query. Every indirect f^{-1}-query has a tag and a sequence of ciphertext blocks leading up to it, and we call this sequence the associated ciphertext chain.

Let $\mathcal{Q}_i$ denote the set of all prefixes of all queries made by D to its $\mathcal{E}_K$-oracle before the ith (f, f^{-1})-query, where an $\mathcal{E}_K$-query (A, M) results in prefixes $\{A[1], A[1]\|A[2], \ldots, A\|M\}$. In this set, we also include $\{A[1], \ldots, A\}$ for an F-query (A, C, T). Let $\mathcal{Q}_i^{-1}$ denote the set of all suffixes of all queries made by D to its F-oracle before the ith query, where an F-query (A, C, T) results in suffixes $\{C[w]\|T, C[w-1]\|C[w]\|T, \ldots, C\|T\}$. (The tag value T is included here for technical reasons.) Regarding all *direct* queries before the ith query, we denote

by X_i^{dir} the set of all capacity values input to f-queries or output of f^{-1}-queries, and by Y_i^{dir} the set of all capacity values input to f^{-1}-queries or output of f-queries. For example, a direct forward query $y \leftarrow f(x)$ adds $[x]_c$ to X_i^{dir} and $[y]_c$ to Y_i^{dir}. The sets X_i^{ind} and Y_i^{ind} are defined similarly. We write $X_i = X_i^{\text{dir}} \cup X_i^{\text{ind}}$ and $Y_i = Y_i^{\text{dir}} \cup Y_i^{\text{ind}}$, and initialize $X_0^{\text{ind}} = Y_0^{\text{ind}} = \{K\}$.

We define event $\mathsf{E}_i = \mathsf{E}_i^{\text{dir-}X} \cup \mathsf{E}_i^{\text{ind-}X} \cup \mathsf{E}_i^{\text{dir-}Y} \cup \mathsf{E}_i^{\text{ind-}Y}$, where $\mathsf{E}_i^{\text{dir-}X}$ and $\mathsf{E}_i^{\text{ind-}X}$ are as in the proof of Theorem 1 with the renewed definitions of the sets, and where

$$\begin{aligned}
\mathsf{E}_i^{\text{dir-}Y} &: \text{ direct query } y \leftarrow f(x) \text{ or } x \leftarrow f^{-1}(y) \text{ satisfies } [y]_c \in Y_i^{\text{ind}} \cup Y_i^{\text{ind}} \oplus 1, \\
\mathsf{E}_i^{\text{ind-}Y} &: \text{ indirect query } f^{-1}(y) \text{ with ciphertext chain } (C,T) \notin \mathcal{Q}_i^{-1} \text{ satisfies} \\
&\qquad [f^{-1}(y)]_c \in Y_i \cup Y_i \oplus 1 \text{ or } [y]_c \in Y_i^{\text{dir}} \oplus K.
\end{aligned}$$

Definitions $\hat{\mathsf{E}}_i$ and E are as before. The latter condition of $\mathsf{E}_i^{\text{ind-}Y}$, $[y]_c \in Y_i^{\text{dir}} \oplus K$, covers the case the distinguisher obtains the key by making a direct inverse query and a $\mathcal{D}_K$-query.

Now, the remainder of the proof is divided as follows. In Lemma 3 we will prove that $(\mathcal{E}_K, \mathcal{D}_K, f, f^{-1})$ and $(\mathcal{E}_K, \perp, f, f^{-1})$ are indistinguishable as long as E does not occur. From (3) and the fundamental lemma of game playing [5] we find

$$\mathbf{Adv}_{\Pi}^{\text{cpa}}(q,m) \leq \frac{m^2}{2^{r+c}} + \Pr[D^{\mathcal{E}_K, \mathcal{D}_K, f, f^{-1}} \text{ sets } \mathsf{E}].$$

Then, in Lemma 4, we will prove that $\Pr[D^{\mathcal{E}_K, \mathcal{D}_K, f, f^{-1}} \text{ sets } \mathsf{E}] \leq \dfrac{2m(m+1)}{2^c}$, which completes the proof. □

Lemma 3. *Given that* E *does not occur,* $(\mathcal{E}_K, \mathcal{D}_K, f, f^{-1})$ *and* $(\mathcal{E}_K, \perp, f, f^{-1})$ *are indistinguishable.*

Proof. The proof is given in the full version [1]. It is similar to the proof of Lemma 1. □

Lemma 4. $\Pr[D^{\mathcal{E}_K, \mathcal{D}_K, f, f^{-1}} \text{ sets } \mathsf{E}] \leq \dfrac{2m(m+1)}{2^c}$.

Proof. The proof is given in the full version [1]. It is similar to the proof of Lemma 2. □

5 Standard Model Security of APE

As is conventionally done for existing permutation-based designs, our proof for APE assumes that the underlying permutation is ideal. By considering a generalized version of APE, we now provide a standard model security argument for our scheme. Inspired by [16], we note that APE can also be described as a block cipher based design: we drop the key additions at the beginning and end, and replace the permutations with a keyed block cipher E_K defined by $E_K := KpK := \oplus_{0\|K} \circ p \circ \oplus_{0\|K}$. (One can view E_K as the Even-Mansour [17]

block cipher with partial key addition.) We remark that this is, indeed, an equivalent description of APE if the block cipher is replaced by KpK. In our notation we denote APE as described and based on some block cipher E by $\Pi' = (\mathcal{K}, \mathcal{E}^E, \mathcal{D}^E)$. We first give the privacy and integrity definitions in the standard model and then show that our results of Theorems 1 and 2 easily translate to a standard model security of Π'.

Definition 4. *Let E be a block cipher, and let $\Pi' = (\mathcal{K}, \mathcal{E}^E, \mathcal{D}^E)$ denote an AE scheme. The CPA advantage of a distinguisher D is defined as*

$$\mathbf{Adv}^{\overline{\text{cpa}}}_{\Pi'}(D) = \left|\Pr\left[K \xleftarrow{\$} \mathcal{K} \ : \ D^{\mathcal{E}^E_K} = 1\right] - \Pr\left[D^{\$} = 1\right]\right| .$$

By $\mathbf{Adv}^{\overline{\text{cpa}}}_{\Pi'}(t, q, m)$ we denote the supremum taken over all distinguishers running in time t and making q queries of total length m blocks. Alternatively, we write $\mathbf{Adv}^{\overline{\text{cpa}}}_{\Pi'}(t, q, m) = \Delta^t_{q,m}(\mathcal{E}^E_K; \$)$ as in Definition 2 with the inclusion of t.

Definition 5. *Let E be a block cipher, and let $\Pi' = (\mathcal{K}, \mathcal{E}^E, \mathcal{D}^E)$ denote an AE scheme. The integrity advantage of a distinguisher D is defined as*

$$\mathbf{Adv}^{\overline{\text{int}}}_{\Pi'}(D) = \left|\Pr\left[K \xleftarrow{\$} \mathcal{K} \ : \ D^{\mathcal{E}^E_K, \mathcal{D}^E_K} = 1\right] - \Pr\left[K \xleftarrow{\$} \mathcal{K} \ : \ D^{\mathcal{E}^E_K, \perp} = 1\right]\right| .$$

By $\mathbf{Adv}^{\overline{\text{int}}}_{\Pi'}(t, q, m)$ we denote the supremum taken over all distinguishers running in time t and making q queries of total length m blocks. Alternatively, we write $\mathbf{Adv}^{\overline{\text{int}}}_{\Pi'}(t, q, m) = \Delta^t_{q,m}(\mathcal{E}^E_K, \mathcal{D}^E_K; \mathcal{E}^E_K, \perp)$ as in Definition 3 with the inclusion of t.

In both definitions we refer to the rate of Π', the number of block cipher calls per message block, as ρ. Furthermore, we need the notion of strong pseudorandom permutation, or prp$\pm$1, security of E.

Definition 6. *Let E be a block cipher. The* prp$\pm$1 *advantage of a distinguisher $\mathcal{D}$ is defined as*

$$\mathbf{Adv}^{\text{prp}\pm1}_{E}(D) = \left| \begin{array}{l} \Pr\left[K \xleftarrow{\$} \mathcal{K} \ : \ D^{E_K, E_K^{-1}} = 1\right] - \\ \qquad \Pr\left[\pi \xleftarrow{\$} \text{Perm}(r + c) \ : \ D^{\pi, \pi^{-1}} = 1\right] \end{array} \right| .$$

By $\mathbf{Adv}^{\text{prp}\pm1}_{E}(t, q)$ we denote the maximum advantage taken over all distinguishers that run in time t and make q queries.

We demonstrate that the standard model security of APE with block ciphers is implied by the results of Sect. 4. To this end we introduce two propositions, one with respect to the integrity and one with respect to the privacy of Π'.

Proposition 1. *Let E be a block cipher.*

$$\mathbf{Adv}^{\overline{\text{int}}}_{\Pi'}(t, q, m) \leq \frac{m^2}{2^{r+c}} + \frac{2m(m+1)}{2^c} + 2\mathbf{Adv}^{\text{prp}\pm1}_{E}(t', \rho m).$$

Proof. Let $K \xleftarrow{\$} \mathcal{K}$. Let E be a publicly available block cipher and $\pi, p \xleftarrow{\$} \mathrm{Perm}(r+c)$ be random permutations. We first switch from E to random π:

$$\begin{aligned}\Delta^t_{q,m}(\mathcal{E}^E_K, \mathcal{D}^E_K; \mathcal{E}^E_K, \perp) &\leq \Delta^t_{q,m}(\mathcal{E}^E_K, \mathcal{D}^E_K; \mathcal{E}^\pi_K, \mathcal{D}^\pi_K) + \Delta^t_{q,m}(\mathcal{E}^\pi_K, \mathcal{D}^\pi_K; \mathcal{E}^\pi_K, \perp) \\ &\quad + \Delta^t_{q,m}(\mathcal{E}^\pi_K, \perp; \mathcal{E}^E_K, \perp) \\ &\leq \Delta^t_{q,m}(\mathcal{E}^\pi_K, \mathcal{D}^\pi_K; \mathcal{E}^\pi_K, \perp) + 2\mathbf{Adv}^{\mathrm{prp}\pm 1}_E(t', \rho m),\end{aligned}$$

where $t' \approx t$. As π is a random permutation, we could give the distinguisher unlimited time (effectively considering information-theoretic distinguishers), and the bound simplifies to:

$$\Delta^t_{q,m}(\mathcal{E}^E_K, \mathcal{D}^E_K; \mathcal{E}^E_K, \perp) \leq \Delta_{q,m}(\mathcal{E}^\pi_K, \mathcal{D}^\pi_K; \mathcal{E}^\pi_K, \perp) + 2\mathbf{Adv}^{\mathrm{prp}\pm 1}_E(t', \rho m).$$

For the remaining Δ-term:

$$\begin{aligned}\Delta_{q,m}(\mathcal{E}^\pi_K, \mathcal{D}^\pi_K; \mathcal{E}^\pi_K, \perp) &\leq \Delta_{q,m}(\mathcal{E}^\pi_K, \mathcal{D}^\pi_K; \mathcal{E}^{KpK}_K, \mathcal{D}^{KpK}_K) \\ &\quad + \Delta_{q,m}(\mathcal{E}^{KpK}_K, \mathcal{D}^{KpK}_K; \mathcal{E}^{KpK}_K, \perp) + \Delta_{q,m}(\mathcal{E}^{KpK}_K, \perp; \mathcal{E}^\pi_K, \perp) \\ &\leq 0 + \Delta_{q,m}(\mathcal{E}^{KpK}_K, \mathcal{D}^{KpK}_K; \mathcal{E}^{KpK}_K, \perp) + 0,\end{aligned}$$

where we use that π and KpK are identically distributed as π and p are random permutations and K is random and unknown. The middle term equals $\mathbf{Adv}^{\mathrm{int}}_{\Pi}(q, m)$ with the difference that the distinguisher cannot access p. A distinguisher would only benefit from such additional access, thus:

$$\Delta_{q,m}(\mathcal{E}^{KpK}_K, \mathcal{D}^{KpK}_K; \mathcal{E}^{KpK}_K, \perp) \leq \mathbf{Adv}^{\mathrm{int}}_{\Pi}(q, m),$$

which is bounded in Theorem 2. This completes the proof. □

Proposition 2. *Let E be a block cipher.*

$$\mathbf{Adv}^{\overline{\mathrm{cpa}}}_{\Pi'}(t, q, m) \leq \frac{m^2}{2^{r+c}} + \frac{1m(m+1)}{2^c} + \mathbf{Adv}^{\mathrm{prp}\pm 1}_E(t', \rho m).$$

Proof. The proof is a straightforward simplification of the proof of Proposition 1, and therefore omitted. □

6 Hardware Implementation

We implement APE with the permutations of PHOTON [19], QUARK [3], and SPONGENT [11]. The results are given in Table 1. We use these permutations without any modifications to investigate the hardware performance of APE. As the designs of PHOTON, QUARK, and SPONGENT follow the hermetic sponge strategy [8], the underlying permutations are assumed to be indistinguishable from random permutations. This assumption is necessary in order to achieve the claimed privacy (Theorem 1) and integrity (Theorem 2) security bounds. Since APE is designed for constrained devices, we focus on a security level of 80 and 128 bits, which correspond to a capacity of 160 or 256 bits, respectively. One exception is APE based

on Quark: since Quark is not equipped with a version for 128 bits of security we resort to a permutation that offers 112 bits of security. The versions of Photon and Spongent with 80 bits of security are implemented with a 4-bit serialization, which means that we implement one 4-bit S-box. For the versions with higher security, we use an 8-bit serialization which requires two 4-bit S-boxes for Spongent and one 8-bit S-box for Photon. Unlike Photon and Spongent, the round permutation of Quark is based on Feedback Shift Registers (FSRs). Hence it is possible to update one bit per clock cycle, and in our implementation we choose to do so for area efficiency.

As APE decrypts in reverse order and requires the inverse permutation, for each of the algorithms (Photon, Quark, and Spongent) we have provided both an encryption-only implementation and an implementation with encryption and decryption. In brief, we have implemented APE as follows. The initial state is XORed with the first data inserted nibble by nibble (or byte by byte, or bit by bit). After each permutation evaluation, the resulting ciphertext is output as the new data is inserted in the same clock cycle. At the end of the iteration, the entire state is output and the capacity part is XORed with the key to generate the tag. Similarly, for decryption, the first state corresponds to the ciphertext concatenated with an XOR of the key and the tag, and at the end authenticity is verified.

For the hardware implementation results in Table 1, we used ModelSim to verify the functionality of the designs and Synopsys Design Vision D-2010.03-SP4 for synthesis. We used Faraday Standard Cell Library based on UMC 0.18μm and open-cell 45nm NANGATE [25] library. As main observations, we see that APE with an encryption and decryption mode can be implemented with less than 1.9 kGE and 2.8 kGE for 80-bit and 128-bit security respectively.

When implemented with the same permutation, encryption-only implementations of SpongeWrap and APE will have similar implementation figures. This is because in both constructions, the processing of every message block requires one XOR with the rate part and one permutation function call. We recall that the crucial difference between SpongeWrap and APE is that APE provides nonce misuse resistance. For the decryption operation, the cost of misuse resistance for APE is that the backwards permutation must be implemented as well.

As shown in Table 1, the overhead of implementing both p and p^{-1} is at most 283 GE on our 45 nm implementation. For devices without non-volatile memory, this overhead is very low compared to the cost of providing a hardware source of randomness to generate nonces.

Note that the permutation-based schemes are implemented on 180 nm and 45 nm CMOS, whereas for the block cipher based schemes, lightweight implementations on 65 nm CMOS are provided. Therefore, we cannot compare these implementations directly. Also note that the clock frequencies of the implementations differ, which lead to different throughput figures. However, it seems that APE and ALE have similar performance figures and APE is smaller than ASC-1 A, ASC-1 B and AES-CCM.

Table 1. APE is implemented using the PHOTON, QUARK, and SPONGENT permutations. For each algorithm, we provide an encryption-only implementation, as well as one that does both encryption and decryption (denoted as "e/d"). The area figures depend on the library that we have used: Area A refers to UMC 180 nm, Area B refers to NANGATE 45 nm. Our overview also includes lightweight implementations of the authenticated encryption schemes ALE [12], ASC-1 [20], and AES-CCM [31]. We remark that the clock frequency of the APE implementations is 100 kHz, compared to 20 MHz for the other ciphers.

APE on CMOS process @ 100 kHz, A: UMC 180 nm, B: NANGATE 45 nm						
Design	Security (bits)	Rate (bits)	Latency (cycles)	Throughput (kbps)	Area A (GE)	Area B (GE)
PHOTON-196	80	36	1248	2.9	1398	1309
PHOTON-196 e/d	80	36	1297	2.8	1634	1536
QUARK-176	80	16	880	1.81	1694	1606
QUARK-176 e/d	80	16	880	1.81	1871	1773
SPONGENT-176	80	16	4050	0.4	1423	1598
SPONGENT-176 e/d	80	16	4094	0.4	1868	1838
PHOTON-288	128	32	924	3.45	2154	2104
PHOTON-288 e/d	128	32	960	3.33	2449	2327
QUARK-256	112	32	1270	2.51	2286	2228
QUARK-256 e/d	112	32	1270	2.51	2470	2331
SPONGENT-272	128	16	4480	0.4	2105	2378
SPONGENT-272 e/d	128	16	4652	0.3	2781	2661

Other AE schemes on ST 65 nm CMOS LP-HVT process @ 20 MHz [12]				
Design	Security (bits)	Latency (cycles)	Throughput (kbps)	Area (GE)
ALE	128	105	121.9	2579
ALE e/d	128	105	121.9	2700
ASC-1 A	128	370	34.59	4793
ASC-1 A e/d	128	370	34.59	4964
ASC-1 B	128	235	54.47	5517
ASC-1 B e/d	128	235	54.47	5632
AES-CCM	128	452	28.32	3472
AES-CCM e/d	128	452	28.32	3765

7 Conclusions

In this paper, we introduced APE, the first misuse resistant permutation-based AE scheme. We proved that APE provides security and integrity up to the birthday bound of the capacity, in the ideal permutation model. We show that the security of APE in the ideal permutation model implies the security of APE with block ciphers in the standard model. This not only ensures security of APE when its underlying primitive is considered an ideal permutation, but also allows to employ it with any secure block cipher of specific form. To achieve misuse resistance, the decryption of APE as a permutation-based construction uses the inverse permutation to decrypt in a backwards manner. The advantage of having backwards decryption is that if the tag or last ciphertext block is missing, then decryption is impossible. Our hardware implementations of APE show that it is

well-suited for lightweight applications. In fact, using any of the permutations of Quark, Photon, and Spongent, less than 1.9 kGE (80-bit security) and less than 2.8 kGE (128-bit security) is required for an implementation of APE that supports both encryption and decryption. Due to its resistance against nonce reuse and its low area requirements in hardware, APE is suitable for environments where it is prohibitively expensive to require non-volatile memory or a hardware source of randomness.

Acknowledgments. We would like to thank the various anonymous reviewers for providing useful comments. Furthermore, we would like to thank Reza Reyhanitabar, Ivan Tjuawinata, Anthony Van Herrewege, Ingrid Verbauwhede, and Hongjun Wu for various suggestions to improve the quality of the text. This work was supported in part by the Research Council KU Leuven: GOA TENSE (GOA/11/007). In addition, this work was supported by the Research Fund KU Leuven, OT/13/071. Elena Andreeva and Nicky Mouha are supported by Postdoctoral Fellowships from the Flemish Research Foundation (FWO-Vlaanderen). Atul Luykx and Bart Mennink are supported by Ph.D. Fellowships from the Institute for the Promotion of Innovation through Science and Technology in Flanders (IWT-Vlaanderen). Begül Bilgin is partially supported by the FWO project G0B4213N.

References

1. Andreeva, E., Bilgin, B., Bogdanov, A., Luykx, A., Mennink, B., Mouha, N., Yasuda, K.: APE: Authenticated Permutation-Based Encryption for Lightweight Cryptography. Cryptology ePrint Archive, Report 2013/791 (2013) (full version of this paper)
2. Aumasson, J.-P., Henzen, L., Meier, W., Naya-Plasencia, M.: Quark: A Lightweight Hash. In: Mangard, S., Standaert, F.-X. (eds.) CHES 2010. LNCS, vol. 6225, pp. 1–15. Springer, Heidelberg (2010)
3. Aumasson, J.P., Henzen, L., Meier, W., Naya-Plasencia, M.: Quark: A Lightweight Hash. J. Cryptol. **26**(2), 313–339 (2013)
4. Bellare, M., Boldyreva, A., Knudsen, L.R., Namprempre, C.: On-Line Ciphers and the Hash-CBC Constructions. J. Cryptol. **25**(4), 640–679 (2012)
5. Bellare, M., Rogaway, P.: The Security of Triple Encryption and a Framework for Code-Based Game-Playing Proofs. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 409–426. Springer, Heidelberg (2006)
6. Berger, T.P., D'Hayer, J., Marquet, K., Minier, M., Thomas, G.: The GLUON Family: A Lightweight Hash Function Family Based on FCSRs. In: Mitrokotsa, A., Vaudenay, S. (eds.) AFRICACRYPT 2012. LNCS, vol. 7374, pp. 306–323. Springer, Heidelberg (2012)
7. Bertoni, G., Daemen, J., Peeters, M., Van Assche, G.: Permutation-Based Encryption, Authentication and Authenticated Encryption, Directions in Authenticated Ciphers, pp. 159–170 (July 2012). http://www.hyperelliptic.org/djb/diac/record.pdf
8. Bertoni, G., Daemen, J., Peeters, M., Van Assche, G.: Cryptographic Sponge Functions. http://sponge.noekeon.org/CSF-0.1.pdf
9. Bertoni, G., Daemen, J., Peeters, M., Van Assche, G.: Sponge Functions. In: ECRYPT Hash Function Workshop (May 2007)

10. 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.) SAC 2011. LNCS, vol. 7118, pp. 320–337. Springer, Heidelberg (2012)
11. Bogdanov, A., Knežević, M., Leander, G., Toz, D., Varıcı, K., Verbauwhede, I.: SPONGENT: A Lightweight Hash Function. In: Preneel, B., Takagi, T. (eds.) CHES 2011. LNCS, vol. 6917, pp. 312–325. Springer, Heidelberg (2011)
12. Bogdanov, A., Mendel, F., Regazzoni, F., Rijmen, V., Tischhauser, E.: ALE: AES-Based Lightweight Authenticated Encryption. In: Moriai, S. (ed.) FSE 2013. LNCS, vol. 8424, pp. 447–466. Springer, Heidelberg (2014)
13. Borisov, N., Goldberg, I., Wagner, D.: Intercepting Mobile Communications: The Insecurity of 802.11. In: Rose, C. (ed.) MOBICOM, pp. 180–189. ACM (2001)
14. CAESAR: Competition for Authenticated Encryption: Security, Applicability, and Robustness (April 2013)
15. Cantero, H.M., Peter, S., Bushing, S.: Console Hacking 2010 - PS3 Epic Fail. In: 27th Chaos Communication Congress, December 2010
16. Chang, D., Dworkin, M., Hong, S., Kelsey, J., Nandi, M.: A Keyed Sponge Construction with Pseudorandomness in the Standard Model. In: The Third SHA-3 Candidate Conference, March 2012
17. Even, S., Mansour, Y.: A Construction of a Cipher from a Single Pseudorandom Permutation. J. Cryptol. **10**(3), 151–162 (1997)
18. Fleischmann, E., Forler, C., Lucks, S.: McOE: A Family of Almost Foolproof On-Line Authenticated Encryption Schemes. In: Canteaut, A. (ed.) FSE 2012. LNCS, vol. 7549, pp. 196–215. Springer, Heidelberg (2012)
19. Guo, J., Peyrin, T., Poschmann, A.: The PHOTON Family of Lightweight Hash Functions. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 222–239. Springer, Heidelberg (2011)
20. Jakimoski, G., Khajuria, S.: ASC-1: An Authenticated Encryption Stream Cipher. In: Miri, A., Vaudenay, S. (eds.) SAC 2011. LNCS, vol. 7118, pp. 356–372. Springer, Heidelberg (2012)
21. Khovratovich, D.: Key Wrapping with a Fixed Permutation. Cryptology ePrint Archive, Report 2013/145 (2013)
22. Kohno, T.: Attacking and Repairing the WinZip Encryption Scheme. In: Atluri, V., Pfitzmann, B., McDaniel, P.D. (eds.) ACM Conference on Computer and Communications Security, pp. 72–81. ACM (2004)
23. Lenstra, A.K., Hughes, J.P., Augier, M., Bos, J.W., Kleinjung, T., Wachter, C.: Public Keys. In: Safavi-Naini, R., Canetti, R. (eds.) CRYPTO 2012. LNCS, vol. 7417, pp. 626–642. Springer, Heidelberg (2012)
24. McGrew, D.A., Viega, J.: The Security and Performance of the Galois/Counter Mode (GCM) of Operation. In: Canteaut, A., Viswanathan, K. (eds.) INDOCRYPT 2004. LNCS, vol. 3348, pp. 343–355. Springer, Heidelberg (2004)
25. NANGATE: The NanGate 45nm Open Cell Library. http://www.nangate.com
26. Rogaway, P.: Authenticated-Encryption with Associated-Data. In: Atluri, V. (ed.) ACM Conference on Computer and Communications Security 2002, pp. 98–107. ACM (2002)
27. Rogaway, P.: Nonce-Based Symmetric Encryption. In: Roy, B., Meier, W. (eds.) FSE 2004. LNCS, vol. 3017, pp. 348–359. Springer, Heidelberg (2004)
28. Rogaway, P., Bellare, M., Black, J.: OCB: A Block-Cipher Mode of Operation for Efficient Authenticated Encryption. ACM Trans. Inf. Syst. Secur. **6**(3), 365–403 (2003)

29. Rogaway, P., Shrimpton, T.: A Provable-Security Treatment of the Key-Wrap Problem. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 373–390. Springer, Heidelberg (2006)
30. Rogaway, P., Zhang, H.: Online Ciphers from Tweakable Blockciphers. In: Kiayias, A. (ed.) CT-RSA 2011. LNCS, vol. 6558, pp. 237–249. Springer, Heidelberg (2011)
31. Whiting, D., Housley, R., Ferguson, N.: Counter with CBC-MAC (CCM). Request For Comments 3610 (2003)
32. Wu, H.: The Misuse of RC4 in Microsoft Word and Excel. Cryptology ePrint Archive, Report 2005/007 (2005)

A APE for Fractional Data

The APE description of Sect. 3 should only be used when the application can guarantee that the length of the plaintext and the associated data is always a multiple of the block size r. In this section, we explain how to adjust APE to handle fractional plaintext and associated data. This is done by applying '10*'-padding to all plaintext and associate data (fractional or not).

The extension of APE to fractional associated data is given in Fig. 3, and to fractional messages in Fig. 4. We elaborate on the extension for fractional messages (the extension for fractional associated data being similar). Split a message M into r-bit blocks, where the last block $M[w]$ is possibly incomplete. We distinguish among three cases:

- $|M[w]| \leq r - 1$ and $w = 1$. The procedure can be seen in the top part of Fig. 4. Note that the corresponding ciphertext will be r bits. This is required for decryption to be possible;
- $|M[w]| \leq r - 1$ and $w \geq 2$. The procedure is depicted in the bottom part of Fig. 4. Note that the ciphertext $C[w-1]$ is of size equal to $M[w]$. The reason we opt for this design property is the following: despite $M[w]$ being smaller than r bits, we require its corresponding ciphertext to be r bits for decryption to be possible. As a toll, the extended APE generates ciphertext $C[w-1]$ to be of size equal to $M[w]$;
- $|M[w]| = r$. In this special case where M consists of integral message blocks, we nevertheless need a padding. However, instead of occupying an extra message block for this, the '10*'-padding spills over into the capacity. This can be seen as an XOR of $10 \cdots 00$ into the capacity part of the state. We recall the reader of the fact that the $\oplus 1$ in the beginning of the function is a shorthand notation for $\oplus 00 \cdots 01$, and hence, these values do not cancel each other out.

The adjustments have no influence on the decryption algorithm $\mathcal{D}$, except if $|M| \leq r$ for which a slightly more elaborate function is needed. Note that the spilling of the padding in case $|M[w]| = r$ causes security to degrade by half a bit: intuitively, APE is left with a capacity of $c' = c - 1$ bits. We have opted for this degrading over an efficiency loss due to an additional round.

The proofs of security of APE with fractional data can be found in the full version of this paper [1].

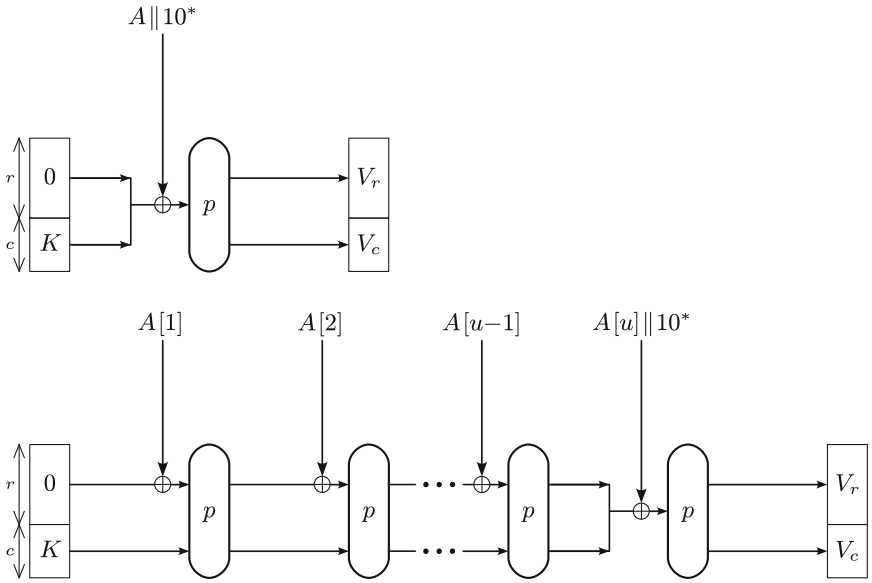

Fig. 3. A generalization of APE that can handle fractional associated data blocks.

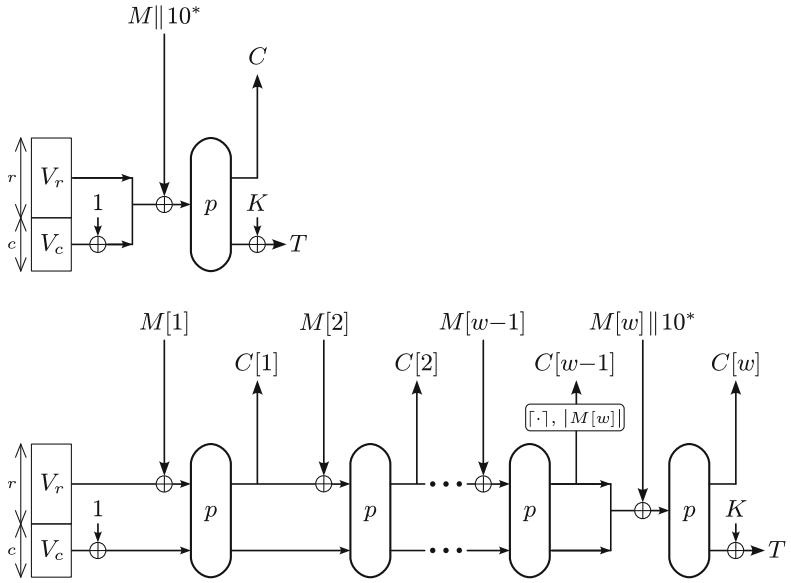

Fig. 4. A generalization of APE that can handle fractional message blocks.