APE: Authenticated Permutation-Based Encryption for Lightweight Cryptography

Elena Andreeva[1,2], Begül Bilgin[1,2,3], Andrey Bogdanov[4], Atul Luykx[1,2], Bart Mennink[1,2], Nicky Mouha[1,2], and Kan Yasuda[1,5]

[1] Department of Electrical Engineering, ESAT/COSIC, KU Leuven, Belgium.
[2] iMinds, Belgium.
[3] Faculty of EEMCS, DIES, UTwente, Netherlands.
[4] Department of Mathematics, Technical University of Denmark, Denmark.
[5] NTT Secure Platform Laboratories, Japan.

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 secure pseudorandom number generator (PRNG), 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 [5] are gaining traction for a wide range of platforms, and on lightweight devices in particular. Lightweight permutation-based hash functions include Quark [1, 2], Photon [15], and Spongent [7].

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 [23, 26, 19]. Permutation-based AE schemes were only recently proposed, such as the deterministic key-wrap scheme [16] of Khovratovich and SpongeWrap [6, 4] of the Keccak team.

These two constructions unfortunately have their limitations. With the key-wrap scheme [16], the message length is restricted to one block by design. While sufficient for key wrapping [24], this construction cannot handle arbitrary-length data and is therefore not a full AE scheme. SpongeWrap [6, 4] 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 [22, 21], 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 a secure pseudorandom number generator (PRNG). 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 [9, 17, 27, 11, 18], 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 upcoming CAESAR competition [10] 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 [24], by the online AE scheme McOE [14], and in part by the aforementioned deterministic key-wrap scheme [16]. However, there are currently no permutation-based AE schemes that are resistant to nonce misuse.

Contributions. In Sect. 3, we introduce APE (Authenticated Permutation-based Encryption). APE is the first permutation-based *and* nonce misuse resistant authenticated encryption scheme. Here, we initially focus on associated data and messages of an integral number of blocks. In App. A, we show how APE can be generalized at almost no extra cost to handle fractional associated data and message blocks.

The security of APE is analyzed in Sects. 4-5. First, in Sect. 4 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 derived in the ideal permutation model, where the underlying permutation is assumed to behave perfectly random. Next, in Sect. 5, we step away from this ideality assumption, and prove that these security results also imply the security of APE in the standard model, with the same bound.

APE encryption processes data in an online manner, whereas decryption is done backwards using the inverse of the permutation. The backwards decryption allows us construct a permutation-based scheme that is secure against nonce misuse. We show that APE is very well suited for lightweight applications. 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.

We conclude the work in Sect. 7.

2 Notation

Set $\mathbf{R} := \{0,1\}^r$ and $\mathbf{C} := \{0,1\}^c$. Given two strings A and B, we use (A, B), $A\|B$ and AB interchangeably, so for example $(A, B) \in \mathbf{R} \times \mathbf{C} \cong \{0,1\}^{r+c} \ni A\|B = AB$. Given $X \in \mathbf{R} \times \mathbf{C}$, X_r denotes its rate part and X_c its 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, at most $2^{c/2}$ blocks. This explicit bound of $2^{c/2}$ is needed for a proper definition of random functions later on. Similarly, let $\mathbf{R}^+$ denote the set of strings whose length is a positive multiple of r, 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 computationally bounded 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 a "$\wedge$," e.g. $\Delta[f_1 \wedge f_2, g_1 \wedge g_2]$ denotes distinguishing the combination of f_1 and f_2 from the combination of g_1 and g_2.

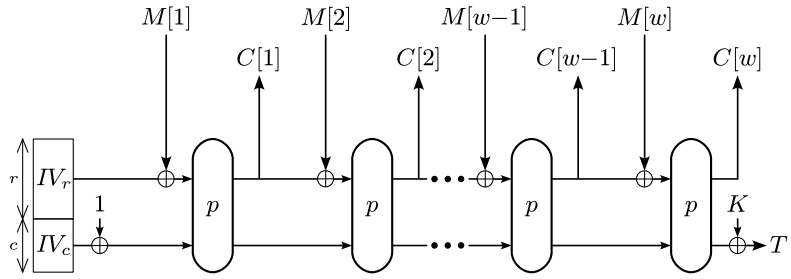

Fig. 1. The APE mode of operation (encryption): IV is computed from associated data (see Fig. 2). If there is no associated data ($A = \varnothing$), set $IV_r := 0$ and $IV_c := K$.

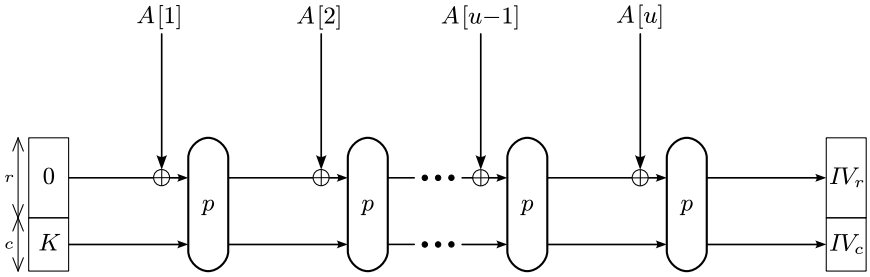

Fig. 2. Hashing associated data A in APE: IV used in Fig. 1 is computed as in this figure.

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 App. 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 Figs. 1 and 2.

More formally, APE consists of two functionalities: encryption $\mathcal{E}$ and decryption $\mathcal{D}$, which are defined in Fig. 3. These functions employ several subroutines which are given in Fig. 4.

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 pair of 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 $M \leftarrow \mathcal{D}_K(A, C, T)$.

APE can be used with any permutation that is indistinguishable from a random permutation. However, APE decrypts in inverse direction and requires an efficiently invertible permutation. For lightweight applications, where hardware area size is critical, we provide implementation results for APE based on the permutations of PHOTON, QUARK, and SPONGENT. In Sect. 6, we show that including the inverses of these permutations only leads to a marginal increase of the size of the implementation.

4 Privacy and Integrity of APE

In this section, we prove that APE satisfies privacy (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.

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

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

if $A = \varnothing$ **then**
  $IV \leftarrow (0, K) \in \mathbf{R} \times \mathbf{C}$
else
  $IV \leftarrow \mathsf{hash\text{-}data}_{(0,K)}(A)$
end
$\big(C, \widehat{V}_c\big) \leftarrow \mathsf{enc\text{-}message}_{IV}(M)$
$T \leftarrow \widehat{V}_c \oplus K$
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 $\perp$

if $A = \varnothing$ **then**
  $IV \leftarrow (0, K) \in \mathbf{R} \times \mathbf{C}$
else
  $IV \leftarrow \mathsf{hash\text{-}data}_{(0,K)}(A)$
end
$(M, V_c) \leftarrow \mathsf{dec\text{-}ctxt}_{IV_r}(C, T \oplus K)$
if $IV_c = V_c$ **then**
  return M
else
  return $\perp$
end

Fig. 3. The encryption $\mathcal{E}_K(A, M)$ and decryption $\mathcal{D}_K(A, C, T)$ algorithms of APE. In Fig. 4, the subroutines $\mathsf{enc\text{-}message}_{IV}(M)$, $\mathsf{dec\text{-}ctxt}_{IV_r}(C, \widehat{V}_c)$ and $\mathsf{hash\text{-}data}_V(A)$ are defined.

Algorithm 3: $\mathsf{enc\text{-}message}_{IV}(M)$

Input: $IV \in \mathbf{R} \times \mathbf{C}$, $M \in \mathbf{R}^+$
Output: $C \in \mathbf{R}^+$, $\widehat{V}_c \in \mathbf{C}$

$V \leftarrow IV \oplus (0, 1)$
$M[1]M[2] \cdots M[w] \leftarrow M$
for $i = 1$ **to** w **do**
  $\widehat{V} \leftarrow p\big(M[i] \oplus V_r, V_c\big)$
  $C[i] \leftarrow \widehat{V}_r$
  $V \leftarrow \widehat{V}$
end
return $\big(C[1]C[2] \cdots C[w], \widehat{V}_c\big)$

Algorithm 4: $\mathsf{dec\text{-}ctxt}_{IV_r}(C, \widehat{V}_c)$

Input: $IV_r \in \mathbf{R}$, $C \in \mathbf{R}^+$, $\widehat{V}_c \in \mathbf{C}$
Output: $M \in \mathbf{R}^+$, $V_c \in \mathbf{C}$

$C[1]C[2] \cdots C[w] \leftarrow C$
$C[0] \leftarrow IV_r$
for $i = w$ **to** 1 **do**
  $V \leftarrow p^{-1}\big(C[i], \widehat{V}_c\big)$
  $M[i] \leftarrow C[i-1] \oplus V_r$
  $\widehat{V}_c \leftarrow V_c$
end
return $\big(M[1]M[2] \cdots M[w], V_c \oplus 1\big)$

Algorithm 5: $\mathsf{hash\text{-}data}_V(A)$

Input: $V \in \mathbf{R} \times \mathbf{C}$, $A \in \mathbf{R}^+$
Output: $\widehat{V} \in \mathbf{R} \times \mathbf{C}$

$A[1]A[2] \cdots A[u] \leftarrow A$
for $i = 1$ **to** u **do**
  $\widehat{V} \leftarrow p\big(A[i] \oplus V_r, V_c\big)$
  $V \leftarrow \widehat{V}$
end
return $\widehat{V}$

Fig. 4. The subroutines $\mathsf{enc\text{-}message}_{IV}(M)$, $\mathsf{dec\text{-}ctxt}_{IV_r}(C, \widehat{V}_c)$, and $\mathsf{hash\text{-}data}_V(A)$ which are used in Fig. 3 of the APE algorithms.

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 $x \xleftarrow{\$} X$ for some finite set X we mean that x is sampled uniformly from 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 $\mathbf{C}$.

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

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

where

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

We use the notions of CPA security and integrity of authenticated encryption schemes from Rogaway and Zhang [25] and Fleischmann et al. [14].

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

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

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 authenticated encryption scheme. Then,*

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

Proof. In Def. 2, we consider the strongest possible type of distinguishers: D is an 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 may restrict ourselves to distinguishers which do not ask "trivial" queries. Trivial queries are either repeated queries, or inverse

queries for which the forward query has previously been asked. For example, a query $p(x)$ is never followed by a query $p^{-1}(y) = x$, and a query $(C, T) \leftarrow \mathcal{E}_K(A, M)$ is never followed by a query $\mathcal{D}_K(A, C, T)$. This allows us to perform a PRP-PRF switch [3] as a first step: both the CPA and integrity advantages of D are defined with access to a permutation (p, p^{-1}), which we can switch to access to random functions (f, f^{-1}). Here, f and f^{-1} are defined to provide a random answer to every new query (unless the query is defined before).

Definition 2 can be viewed as a distinguisher comparing a real world and an ideal world. If (R, p) denotes a real world, (I, p) an ideal world then

$$\Delta_{q,m}(R, p; I, p) \leq \Delta_{q,m}(R, p; R, f) + \Delta_{q,m}(R, f; I, p),$$

where $\Delta_{t,q,m}(R, p; R, f)$ is a PRP-PRF switch, and is bounded by at most $m^2/2^{r+c+1}$. We can perform a similar switch with (I, p) to get

$$\Delta_{q,m}(R, p; I, p) \leq \frac{m^2}{2^{r+c}} + \Delta_{q,m}(R, f; I, f), \tag{1}$$

where we recall that the distinguisher makes at most q queries of total length m blocks (each block corresponds to a new (f, f^{-1})-query). Now, in order to analyze the security of APE, it suffices to consider the distance $\Delta_{q,m}(R, f; I, f)$, and our result is obtained via (1).

We consider a CPA distinguisher which has oracle access to one of the two worlds, either $(\mathcal{E}_K, f, f^{-1})$ or $(\$, f, f^{-1})$ (for simplicity and without loss of generality we can drop out the $\perp$). Here, in the real world R represents $\mathcal{E}_K$ and in the ideal world I represents $\$$. Note that by construction f^{-1} is independent of $\mathcal{E}_K$ and f, hence no distinguisher will gain any advantage from querying f^{-1}. In our analysis we will not consider f^{-1}-queries and will represent both worlds by (F, f), for $F \in \{\mathcal{E}_K, \$\}$.

If f is called by D then we call this a direct f-query; a call of f by $\mathcal{E}_K$ (as a result of D calling $\mathcal{E}_K$) is called an indirect f-query. 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. When we do not specify whether an f-query is indirect or direct, we mean that it could be either.

Let $\mathcal{Q}_i$ denote the set of all prefixes of all queries made by D to its F-oracle before the ith f-query, where a query (A, M) results in prefixes $\{A[1], A[1]\|A[2], \ldots, A\|M[1], \ldots, A\|M\}$. Furthermore, we denote by X_i^{d} the set of all capacity values input to *direct* f-queries before the ith f-query, and by X_i^{i} the set of all capacity values input to *indirect* f-queries before the ith f-query. We write $X_i = X_i^{\mathrm{d}} \cup X_i^{\mathrm{i}}$, and initialize $X_0^{\mathrm{i}} = \{K\}$.

For the analysis of the real world $(\mathcal{E}_K, f)$, we define event $\mathsf{E}_i = \mathsf{E}_i^{\mathrm{d}} \cup \mathsf{E}_i^{\mathrm{i}}$, where

$$\begin{aligned}
\mathsf{E}_i^{\mathrm{d}} :&\ \text{direct query } f(x) \text{ satisfies } [x]_c \in X_i^{\mathrm{i}} \cup X_i^{\mathrm{i}} \oplus 1,\\
\mathsf{E}_i^{\mathrm{i}} :&\ \text{indirect query } f(x) \text{ with message chain } (A, M) \notin \mathcal{Q}_i \text{ satisfies}\\
&\quad [f(x)]_c \in X_i \cup X_i \oplus 1.
\end{aligned}$$

We furthermore define

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

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

Now, the remainder of the proof is divided as follows. Firstly, in Lem. 1 we will prove that, as long as E does not occur, $(\mathcal{E}_K, f)$ and $(\$, f)$ are indistinguishable. By (1) and the fundamental lemma of game playing this implies

$$\mathbf{Adv}_{\Pi}^{\mathrm{cpa}}(q, m) = \Delta_D[R \wedge p, I \wedge p] \leq \frac{m^2}{2^{r+c}} + \Pr[\mathsf{E}].$$

Then, in Lem. 2, we will prove that $\Pr[\mathsf{E}] \leq \dfrac{2m(m+1)}{2^c}$, which completes the proof. □

Lemma 1. *Given that* E *does not occur,* $(\mathcal{E}_K, f)$ *and* $(\$, f)$ *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 $[x]_c \in X_i^{\mathrm{i}} \cup X_i^{\mathrm{i}} \oplus 1$, where $f(x)$ is the ith f-query, then this would trigger $\mathsf{E}_i^{\mathrm{d}}$, hence we can assume $[x]_c \notin X_i^{\mathrm{i}} \cup X_i^{\mathrm{i}} \oplus 1$. This means that the query $f(x)$ is truly new, and its value is independently and uniformly distributed.

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 $(\$, f)$, 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 $(\mathcal{E}_K, f)$. 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 $f(x) \oplus (M[j+1], 0)$. Suppose this query has already been made before, then either $[f(x)]_c \in X^{\mathrm{d}} \cup X^{\mathrm{i}}$, or $(A, M'\|M[j+1]) \in \mathcal{Q}$. Since $[f(x)]_c \notin X^{\mathrm{d}} \cup X^{\mathrm{i}}$, it must be that $(A, M'\|M[j+1]) \in \mathcal{Q}$, but this contradicts the fact that j is maximal. This query has been made at an earlier point in time (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 $X^{\mathrm{d}} \cup X^{\mathrm{i}}$ (otherwise it would have triggered E^{i}). 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 $\mathsf{E}^{\mathrm{d}} \cup \mathsf{E}^{\mathrm{i}}$). Thus, the query corresponding to the $(j+1)$th round is generated independently and uniformly at random.
   Now, in the case $j = 0$, denote the state coming from $\mathsf{hash\text{-}data}_{0,K}(A)$ by IV (if $A = \varnothing$, $IV = (0, K)$). The same story as before applies with the difference that now the input to the $(j+1)$th query is $IV \oplus (M[j+1], 1)$. Here we use that by $\overline{\mathsf{E}}$, no other query hit $X_j^{\mathrm{i}} \oplus 1$ (for direct queries) or $X_j \oplus 1$ (for indirect queries) in the meanwhile, and that X^{i} 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 E^{i} 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 $\overline{\mathsf{E}^{\mathrm{d}}}$, so it never learnt $T \oplus K$. Besides, due to the absence of indirect capacity collisions, $\overline{\mathsf{E}^{\mathrm{i}}}$, 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 IV_c can be considered new. □

Lemma 2. $\Pr[\mathsf{E}] \leq \dfrac{2m(m+1)}{2^c}$.

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 query, it triggers $\mathsf{E}_i^{\mathrm{d}}$ if the distinguisher "guesses" a capacity part in $X_i^{\mathrm{i}} \cup X_i^{\mathrm{i}} \oplus 1$, which happens with probability at most $2|X_i^{\mathrm{i}}|/2^c$. 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 $\mathsf{E}_i^{\mathrm{i}}$ if $[f(x)]_c \in X_i \cup X_i \oplus 1$. This occurs with probability at most $2|X_i|/2^c$.

By construction, $X_i^{\mathrm{i}} \leq X_i \leq i$, henceforth we find:

$$\Pr[\mathsf{E}_i \mid \textstyle\bigcap_{j=1}^{i-1} \overline{\mathsf{E}_j}] \leq \frac{4i}{2^c}.$$

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 authenticated encryption scheme. Then,*

$$\mathbf{Adv}^{\mathrm{int}}_{\Pi}(q, m) \leq \frac{m^2}{2^{r+c}} + \frac{3m(m+1)}{2^c}.$$

Proof. The basic idea of the proof is the same as for Thm. 1, an in particular, we apply a PRP-PRF switch resulting in $\frac{m^2}{2^{r+c}}$. For integrity, however, we need to take into account inverse queries too.

We consider an integrity distinguisher which has query access to one of the two worlds, either $(\mathcal{E}_K, \mathcal{D}_K, f, f^{-1})$ or $(\mathcal{E}_K, \bot, f, f^{-1})$. Here, in the real world R represents $(\mathcal{E}_K, \mathcal{D}_K)$ and in the ideal world I represents $(\mathcal{E}_K, \bot)$.

If f (resp. f^{-1}) is called by D then we call this a direct f-query (resp. f^{-1}-query); 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 f^{-1}. 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.

We use similar notation as in the proof of Thm. 1, but as the proof now also involves inverse queries, slightly more involved definitions are needed, and we re-introduce them. 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 an F-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^{-1}-query (A, C, T). Let $\mathcal{Q}_i^{-1}$ denote the set of all suffixes of all queries made by D to its F^{-1}-oracle before the ith query, where an F^{-1}-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.) Furthermore, regarding all *direct* queries before the ith query, we denote by X_i^{d} the set of all capacity values input to f-queries or output of f^{-1}-queries, and by Y_i^{d} the set of all capacity values input to f^{-1}-queries or output of f-queries. For example, a direct forward query $f(x) \to y$ adds x to X_i^{d} and y to Y_i^{d}, and a direct inverse query $f^{-1}(y) \to x$ adds x to X_i^{d} and y to Y_i^{d}. The sets X_i^{i} and Y_i^{i} are defined similarly. We write $X_i = X_i^{\mathrm{d}} \cup X_i^{\mathrm{i}}$ and $Y_i = Y_i^{\mathrm{d}} \cup Y_i^{\mathrm{i}}$, and initialize $X_0^{\mathrm{i}} = Y_0^{\mathrm{i}} = \{K\}$.

We define event $\mathsf{E}_i = \mathsf{E}_i^{\mathrm{d}} \cup \mathsf{E}_i^{\mathrm{i}} \cup \mathsf{E}_i^{\mathrm{d\text{-}}} \cup \mathsf{E}_i^{\mathrm{i\text{-}}}$, where $\mathsf{E}_i^{\mathrm{d}}$ and $\mathsf{E}_i^{\mathrm{i}}$ are as in the proof of Thm. 1 with the renewed definitions of the sets, and where

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

Definitions $\hat{\mathsf{E}}_i$ and E are defined as before. The latter condition of $\mathsf{E}_i^{\text{i-}}$, $[y]_c \in Y_i^{\text{d}} \oplus K$ 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. Firstly, in Lem. 3 we will prove that, as long as E does not occur, $(\mathcal{E}_K, \mathcal{D}_K, f, f^{-1})$ and $(\mathcal{E}_K, \perp, f, f^{-1})$ are indistinguishable. By (1) and the fundamental lemma of game playing this implies

$$\mathbf{Adv}^{\text{int}}_{\Pi}(q, m) = \Delta_D[R \wedge p, I \wedge p] \leq \frac{m^2}{2^{r+c}} + \Pr[\mathsf{E}].$$

Then, in Lem. 4, we will prove that $\Pr[\mathsf{E}] \leq \dfrac{3m(m+1)}{2^c}$, 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. For direct f-queries, the analysis of Lem. 1 carries over. Similarly, for inverse direct f^{-1}-queries, the same reasoning applies with the difference that $[y]_c \in Y_i^{\text{i}} \cup Y_i^{\text{i}} \oplus 1$ would trigger event $\mathsf{E}^{\text{d-}}$. Also, queries made to $\mathcal{E}_K$ in the real and ideal world are handled the same. Here, we use that in the real world, indirect f-queries coming from $\mathcal{D}_K$ (corresponding to the evaluation of $\mathsf{hash\text{-}data}_{0,K}$), do not ruin the distribution of the responses from $\mathcal{E}_K$ by $\overline{\mathsf{E}^{\text{i}}}$, where we now deal with a larger set X_i^{i}.

Therefore, we consider queries to the big oracle $F \in \{\mathcal{D}_K, \perp\}$, and let (A, C, T) be a query made by the distinguisher. Denote by w the number of blocks of C. Denote the state coming from $\mathsf{hash\text{-}data}_{0,K}(A)$ by IV, and the corresponding message and capacity value by (M, V_c). Here, $V_c \oplus 1$ equals the capacity part of the call to f^{-1} corresponding to $C[1]$.

Clearly, in the ideal world $(\mathcal{E}_K, \perp, f, f^{-1})$, the query is responded with $\perp$. Therefore, the distinguisher has no advantage in the real world unless $IV_c = V_c$. We distinguish two cases:

- $(C, T) \notin \mathcal{Q}_i^{-1}$. A similar reasoning as for the value T in Lem. 1 subcase "$i < w$" results in the observation that the indirect f^{-1}-query corresponding to the ciphertext block $C[1]$ is new and the response is uniformly randomly drawn. The only fundamental difference lies in the fact that we now use events $\mathsf{E}^{\text{d-}}$ and $\mathsf{E}^{\text{i-}}$, and rely on the fact the first indirect query never matches a direct query $\oplus K$ (by $\overline{\mathsf{E}_i^{\text{i-}}}$) or vice versa (by $\overline{\mathsf{E}_i^{\text{d-}}}$). We skip the details. Now, D succeeds if the capacity part of this value, say $[f^{-1}(y)]_c$, equals $IV_c \oplus 1$, but then this indirect query would have triggered $\mathsf{E}^{\text{i-}}$. We note that this value IV_c may be an older value, e.g., if $A \in \mathcal{Q}_i$, but this does not invalidate the analysis.
- $(C, T) \in \mathcal{Q}_i^{-1}$. The further reasoning depends on whether $(A, M') \in \mathcal{Q}_i$ for some M'.
  - $(A, *) \notin \mathcal{Q}_i$. If $A = \varnothing$, the distinguisher succeeding would mean that $V_c = K \oplus 1$. But this means that at some earlier point in time an indirect f^{-1}-query has hit $K \oplus 1$ and thus invalidated $\mathsf{E}^{\text{d-}} \cup \mathsf{E}^{\text{i-}}$. Now, if $A \neq \varnothing$, the analysis of Lem. 1 for the same case "$(A, *) \notin \mathcal{Q}_i$" carries over: the value IV_c can be considered new and if it hits $V_c \oplus 1$, the query would trigger E^{i}.
  - $(A, M') \in \mathcal{Q}_i$ for some M'. By assumption, (A, M') and (C, T) cannot correspond to one and the same query. Without loss of generality, (A, M') corresponds to an earlier query than (C, T). Now, the distinguisher's query is valid if $IV_c = V_c \oplus 1$. If this is the case, the (C, T) query must have triggered $\mathsf{E}^{\text{i-}}$. Also, in case $A = \perp$, this equation could not hold as $V_c = K \oplus 1$ would have triggered $\mathsf{E}^{\text{i-}}$.

This completes the proof of Lem. 3. □

Lemma 4. $\Pr[\mathsf{E}] \leq \dfrac{3m(m+1)}{2^c}$.

Proof. The analysis is fairly similar to the proof of Lem. 2, with the difference that inverse queries have to be considered too.

Again, if the ith query is forward, E_i is triggered with probability at most $\frac{4i}{2^c}$. In a similar way, an inverse query sets E_i with probability at most $\frac{5i}{2^c}$ (note that $\mathsf{E}_i^{\text{i-}}$ contains an additional condition, compared with E_i^{i}). As the query is either forward or inverse, we could take the maximum of both values, and our bound is obtained by summing over $i = 1, \ldots, m$. □

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. Here we provide a standard model security argument for APE. Inspired by [12], 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 [13] block cipher with partial key addition.) 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 Thm. 1 and Thm. 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 authenticated encryption 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 Def. 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 authenticated encryption 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 Def. 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 is implied by the results of Sect. 4. We therefore prove 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.*

$$\mathbf{Adv}^{\overline{\text{int}}}_{\Pi'}(t, q, m) \leq \frac{m^2}{2^{r+c}} + \frac{3m(m+1)}{2^c} + 2\mathbf{Adv}^{\text{prp}\pm1}_{E}(t', \rho m).$$

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

$$\begin{aligned} \Delta^t_{q,m}(\mathcal{E}^E_K, \mathcal{D}^E_K; \mathcal{E}^E_K, \perp) &\leq \Delta^t_{q,m}(\mathcal{E}^E_K, \mathcal{D}^E_K; \mathcal{E}^\pi_K, \mathcal{D}^\pi_K) + \Delta^t_{q,m}(\mathcal{E}^\pi_K, \mathcal{D}^\pi_K; \mathcal{E}^\pi_K, \perp) + \\ &\quad \Delta^t_{q,m}(\mathcal{E}^\pi_K, \perp; \mathcal{E}^E_K, \perp) \\ &\leq \Delta^t_{q,m}(\mathcal{E}^\pi_K, \mathcal{D}^\pi_K; \mathcal{E}^\pi_K, \perp) + 2\mathbf{Adv}^{\text{prp}\pm1}_{E}(t', \rho m), \end{aligned}$$

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:

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

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 $\mathbf{Adv}^{\text{int}}_{APE}(q, m)$ with the difference that the distinguisher cannot access p. Finally:

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

This completes the proof. □

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

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

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

6 Hardware Implementation

We implement APE with the permutations of PHOTON [15], QUARK [2], and SPONGENT [7] (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, the underlying permutations are assumed to be indistinguishable from random permutations. This assumption is necessary in order to achieve the claimed privacy (Thm. 1) and integrity (Thm. 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 [20] 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.

The overhead of decryption ranges from 4 to 17 percent, which corresponds to at most 0.7 kGE. This cost is very low compared to the cost of securely generating nonces. Note that the permutation-based schemes are implemented on 180 nm and 45 nm CMOS, whereas the block cipher based schemes are implemented on 65 nm CMOS. 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.

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 *as well as* 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 secure PRNGs or non-volatile memory are prohibitively expensive to implement.

Acknowledgments. This work has been funded in part by the IAP Program P6/26 BCRYPT of the Belgian State (Belgian Science Policy), in part by the European Commission through the ICT program under contract ICT-2007-216676 ECRYPT II, in part by the Research Council KU Leuven: GOA TENSE, and in part by the Research Fund KU Leuven, OT/08/027. Elena Andreeva and Nicky Mouha are supported by Postdoctoral Fellowships from the Flemish Research Foundation (FWO-Vlaanderen). Bart Mennink is supported by a Ph.D. Fellowship from the Institute for the Promotion of Innovation through Science and Technology in Flanders (IWT-Vlaanderen).

References

1. Aumasson, J.P., Henzen, L., Meier, W., Naya-Plasencia, M.: Quark: A Lightweight Hash. In: Mangard, S., Standaert, F.X. (eds.) CHES. Lecture Notes in Computer Science, vol. 6225, pp. 1–15. Springer (2010)
2. Aumasson, J.P., Henzen, L., Meier, W., Naya-Plasencia, M.: Quark: A Lightweight Hash. J. Cryptology 26(2), 313–339 (2013)
3. 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)
4. Bertoni, G., Daemen, J., Peeters, M., Van Assche, G.: Permutation-Based Encryption, Authentication and Authenticated Encryption. Directions in Authenticated Ciphers (July 2012)
5. Bertoni, G., Daemen, J., Peeters, M., Van Assche, G.: Sponge Functions. ECRYPT Hash Function Workshop (May 2007)
6. 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 2011. Lecture Notes in Computer Science, vol. 7118, pp. 320–337. Springer (2012)

7. Bogdanov, A., Knezevic, M., Leander, G., Toz, D., Varıcı, K., Verbauwhede, I.: SPONGENT: A Lightweight Hash Function. In: Preneel, B., Takagi, T. (eds.) CHES. Lecture Notes in Computer Science, vol. 6917, pp. 312–325. Springer (2011)
8. Bogdanov, A., Mendel, F., Regazzoni, F., Rijmen, V.: Lightweight AES-Based Authenticated Encryption. Directions in Authenticated Ciphers (July 2012)
9. Borisov, N., Goldberg, I., Wagner, D.: Intercepting mobile communications: the insecurity of 802.11. In: Rose, C. (ed.) MOBICOM. pp. 180–189. ACM (2001)
10. CAESAR: Competition for Authenticated Encryption: Security, Applicability, and Robustness (April 2013)
11. Cantero, H.M., Peter, S., Bushing, Segher: Console Hacking 2010 – PS3 Epic Fail. 27th Chaos Communication Congress (December 2010)
12. Chang, D., Dworkin, M., Hong, S., Kelsey, J., Nandi, M.: A Keyed Sponge Construction with Pseudorandomness in the Standard Model. The Third SHA-3 Candidate Conference (March 2012)
13. Even, S., Mansour, Y.: A Construction of a Cipher from a Single Pseudorandom Permutation. J. Cryptology 10(3), 151–162 (1997)
14. Fleischmann, E., Forler, C., Lucks, S.: McOE: A Family of Almost Foolproof On-Line Authenticated Encryption Schemes. In: Canteaut, A. (ed.) FSE. Lecture Notes in Computer Science, vol. 7549, pp. 196–215. Springer (2012)
15. Guo, J., Peyrin, T., Poschmann, A.: The PHOTON Family of Lightweight Hash Functions. In: Rogaway, P. (ed.) CRYPTO. Lecture Notes in Computer Science, vol. 6841, pp. 222–239. Springer (2011)
16. Khovratovich, D.: Key Wrapping with a Fixed Permutation. Cryptology ePrint Archive, Report 2013/145 (2013)
17. 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)
18. 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. Lecture Notes in Computer Science, vol. 7417, pp. 626–642. Springer (2012)
19. McGrew, D.A., Viega, J.: The Security and Performance of the Galois/Counter Mode (GCM) of Operation. In: Canteaut, A., Viswanathan, K. (eds.) INDOCRYPT. Lecture Notes in Computer Science, vol. 3348, pp. 343–355. Springer (2004)
20. NANGATE: The NanGate 45nm Open Cell Library, available at `http://www.nangate.com`
21. Rogaway, P.: Authenticated-Encryption with Associated-Data. In: Atluri, V. (ed.) ACM Conference on Computer and Communications Security 2002. pp. 98–107. ACM (2002)
22. Rogaway, P.: Nonce-Based Symmetric Encryption. In: Roy, B.K., Meier, W. (eds.) FSE 2004. Lecture Notes in Computer Science, vol. 3017, pp. 348–359. Springer (2004)
23. 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)
24. Rogaway, P., Shrimpton, T.: A Provable-Security Treatment of the Key-Wrap Problem. In: Vaudenay, S. (ed.) EUROCRYPT 2006. Lecture Notes in Computer Science, vol. 4004, pp. 373–390. Springer (2006)
25. Rogaway, P., Zhang, H.: Online Ciphers from Tweakable Blockciphers. In: Kiayias, A. (ed.) CT-RSA. Lecture Notes in Computer Science, vol. 6558, pp. 237–249. Springer (2011)
26. Whiting, D., Housley, R., Ferguson, N.: Counter with CBC-MAC (CCM). Request For Comments 3610 (2003)
27. Wu, H.: The Misuse of RC4 in Microsoft Word and Excel. Cryptology ePrint Archive, Report 2005/007 (2005)

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"). We compare these implementations to ALE, ASC-1, and AES-CCM. We remark that the clock frequency of the APE implementations is 100 kHz, compared to 20 MHz for the other ciphers.

APE on UMC 180 nm CMOS process @ 100 kHz					
Design	Security (bits)	Rate (bits)	Latency (cycles)	Throughput (kbps)	Area (GE)
PHOTON-196	80	36	1248	2.9	1398
PHOTON-196 e/d	80	36	1297	2.8	1634
QUARK-176	80	16	880	1.81	1694
QUARK-176 e/d	80	16	880	1.81	1871
SPONGENT-176	80	16	4050	0.4	1423
SPONGENT-176 e/d	80	16	4094	0.4	1868
PHOTON-288	128	32	924	3.45	2154
PHOTON-288 e/d	128	32	960	3.33	2449
QUARK-256	112	32	1270	2.51	2286
QUARK-256 e/d	112	32	1270	2.51	2470
SPONGENT-272	128	16	4480	0.4	2105
SPONGENT-272 e/d	128	16	4652	0.3	2781

APE on NANGATE 45 nm CMOS process @ 100 kHz					
Design	Security (bits)	Rate (bits)	Latency (cycles)	Throughput (kbps)	Area (GE)
PHOTON-196	80	36	1248	2.9	1309
PHOTON-196 e/d	80	36	1297	2.8	1536
QUARK-176	80	16	880	1.81	1606
QUARK-176 e/d	80	16	880	1.81	1773
SPONGENT-176	80	16	4050	0.4	1598
SPONGENT-176 e/d	80	16	4094	0.4	1838
PHOTON-288	128	32	924	3.45	2104
PHOTON-288 e/d	128	32	960	3.33	2327
QUARK-256	112	32	1270	2.51	2228
QUARK-256 e/d	112	32	1270	2.51	2331
SPONGENT-272	128	16	4480	0.4	2378
SPONGENT-272 e/d	128	16	4652	0.3	2661

Other AE schemes on ST 65 nm CMOS LP-HVT process @ 20 MHz [8]					
Design	Security (bits)	Rate (bits)	Latency (cycles)	Throughput (kbps)	Area (GE)
ALE	128	N/A	105	121.9	2579
ALE e/d	128	N/A	105	121.9	2700
ASC-1 A	128	N/A	370	34.59	4793
ASC-1 A e/d	128	N/A	370	34.59	4964
ASC-1 B	128	N/A	235	54.47	5517
ASC-1 B e/d	128	N/A	235	54.47	5632
AES-CCM	128	N/A	452	28.32	3472
AES-CCM e/d	128	N/A	452	28.32	3765

A APE for Fractional Data

The APE description of Sect. 3 applies only to plaintexts and associated data whose length is a multiple of the block size r. In this section, we demonstrate how to adjust APE to handle fractional messages and associated data. The extension of $\mathsf{hash\text{-}data}_{0,K}$ to fractional associated data is given in Fig. 5, and for fractional messages the extension of $\mathsf{enc\text{-}message}_{IV}$ is given in Fig. 6. For both the processing of authenticated data and message data, we require the last block ($A[u]$ or $M[w]$) to be of length at most $r-1$ bits. Note that in Fig. 6, 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 $M[w]$. Clearly, this has no influence on the decryption algorithm $\mathcal{D}$.

Without going into detail, we note that the security results of Sect. 4 directly carry over to APE with fractional data. Here, we rely on the fact that $A[u]$ and $M[w]$ are of size at most $r-1$ bits.

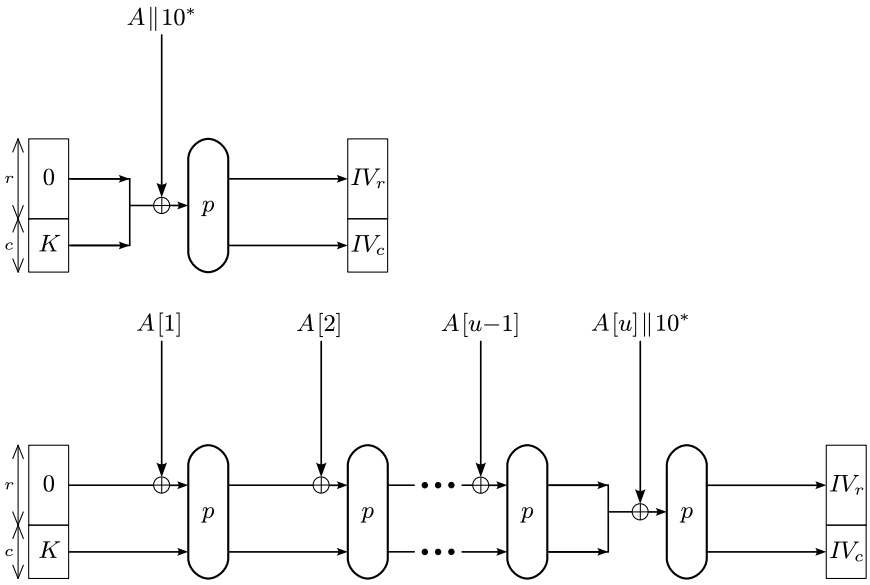

Fig. 5. A generalization of APE that can handle fractional associated data blocks.

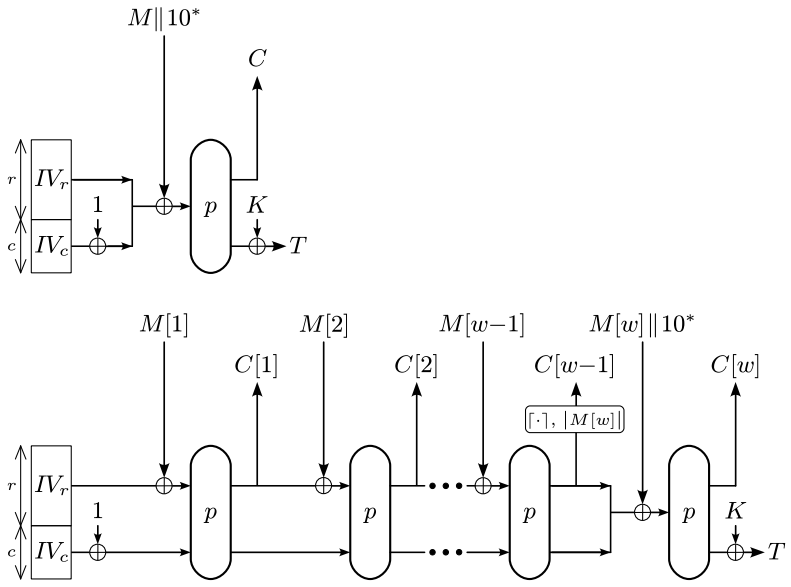

Fig. 6. A generalization of APE that can handle fractional message blocks.