Optimizing Online Permutation-Based AE Schemes for Lightweight Applications

Yu Sasaki(✉) and Kan Yasuda(✉)

NTT Secure Platform Laboratories, Tokyo, Japan
{sasaki.yu,yasuda.kan}@lab.ntt.co.jp

Abstract. We explore ways to optimize online, permutation-based authenticated-encryption (AE) schemes for lightweight applications. The lightweight applications demand that AE schemes operate in resource-constrained environments, which raise two issues: (1) implementation costs must be low, and (2) ensuring proper use of a nonce is difficult due to its small size and lack of randomness. Regarding the implementation costs, recently it has been recognized that permutation-based (rather than block-cipher-based) schemes frequently show advantages. However, regarding the security under nonce misuse, the standard permutation-based duplex construction cannot ensure confidentiality. There exists one permutation-based scheme named APE which offers certain robustness against nonce misuse. Unfortunately, the APE construction has several drawbacks such as ciphertext expansion and bidirectional permutation circuits. The ciphertext expansion would require more bandwidth, and the bidirectional circuits would require a larger hardware footprint. In this paper, we propose new constructions of online permutation-based AE that require less bandwidth, a smaller hardware footprint and lower computational costs. We provide security proofs for the new constructions, demonstrating that they are as secure as the APE construction.

Keywords: AEAD · Permutation-based · Sponge · APE · Bandwidth Hardware footprint · Inverse-free

1 Introduction

With the rise of Internet of Things (IoT), *lightweight* cryptography is drawing more and more attentions today [9,15,18]. This is because many of the IoT devices need to operate within tight resource constraints and hence may not be able to accommodate conventional cryptographic algorithms. The constraints include, for example, limited amount of storage, power and bandwidth.

The lightweight cryptography aims for essentially the same type of security goal as the conventional cryptography, with two most important security notions being confidentiality and integrity. The two notions can be simultaneously achieved by a symmetric-key primitive called *authenticated encryption* (AE) [4,5,11]. Hence it becomes one of the most fundamental problems in

J. K. Liu and P. Samarati (Eds.): ISPEC 2017, LNCS 10701, pp. 217–236, 2017.
https://doi.org/10.1007/978-3-319-72359-4_13

lightweight cryptography to come up with an AE scheme that can be efficiently run in resource-constrained environments.

Although the type of security goal is the same, appropriate design approaches may differ between lightweight cryptography and conventional one, due to the low-resource conditions in the former. Of the existing AE designs, some become more suitable for lightweight cryptography, while others remain less suitable. In recent years, it is recognized that *permutation-based* (rather than block-cipher-based) designs have comparative advantages in lightweight cryptography, owing to their small RAM footprint [3,8,13].

Unfortunately, naively building an AE scheme from permutations would not give us one that is workable in resource-constrained environments, because there is a major security issue inherent in lightweight AE: the initializing vector (IV) needs to be a *nonce* [20]. In general, the security of an AE scheme gets compromised if the same value of IV is used twice under the same key. However, in many resource-constrained scenarios it is difficult for devices to ensure their IV to be a nonce, as explained below.

Two typical methods to realize a nonce are *counter* and *randomization*. A counter IV would require a secure writable memory, which also needs to be non-volatile if the device is supplied with a weak battery or reboots frequently, because the IV may get reset due to loss of power or rebooting. Such memory tends to be costly to be securely implemented [16], and it is unlikely that devices with such a weak battery or unstable system would come with such rich memory. The other type, a randomized IV, is not easily realizable in lightweight environments, either. It is difficult for low-resource devices to ensure a source of randomness [10], which implies that a randomized IV may have insufficient entropy and produce collisions.

A related issue with nonce misuse is the fact that fully nonce-misuse-resistant AE schemes [14,21] require "two-pass" or "three-pass" operations on data, which result in a larger state size. This may make these schemes unsuitable for severely resource-constrained environments, even if the data size is relatively small. There is a "one-pass" permutation-based AE construction called the duplex construction [7], but it does not provide security under nonce misuse. Hence we aim at *online* permutation-based AE schemes [12], that is to say, when the same nonce is repeated, the only information leaked to adversaries is that the new message and associated data are the same as the previous ones up to the block where different data is processed for the first time. This onlineness provides us with a good tradeoff between performance (still "one-pass") and security.

There is previous work of online permutation-based AE called APE [2], but it requires relatively high bandwidth and large hardware footprint. The large footprint comes from the decryption process that uses both forward and inverse permutations, requiring independent circuits for the two permutation calls. Moreover, the computational costs of APE tend to be higher, because the technique called concurrent absorption [22] or full-state absorption [19], which reduce computational costs in the duplex construction, is not applicable to APE. A small note on APE is that it is equipped with backward decryption, which can be

problematic for high-end use involving streaming data. Fortunately this should not be problematic in our setting, because in most lightweight applications the data and state sizes remain small and latency is not an issue.

Contributions of This Paper. We provide online permutation-based AE schemes with minimal bandwidth, hardware footprint and computational cost. Our constructions improve those over APE by a few to several dozens of bytes, which make a big difference in the resource-constrained environments. We provide three different constructions, APE^{RI}, APE^{OW} and APE^{CA}, which can be chosen depending on the situations.

1. The APE^{RI} scheme optimizes the hardware footprint, so that developers need to implement only the forward permutation f for the encryption circuit and only the inverse f^{-1} for the decryption circuit. Recall that APE required implementation of both f and f^{-1} for decryption. The bandwidth, computational cost and security are exactly the same as APE, hence APE^{RI} simply improves APE in the hardware footprint. The core idea is replacing the standard nonce-based AE framework of APE with the protected-IV (PIV) framework formalized by Shrimpton and Terashima [23], which converts nonce N to other value called reconstruction information (RI) and sends RI to the receiver instead of N.
2. The APE^{OW} scheme further modifies APE^{RI} in order to improve bandwidth while it inherits improved hardware footprint of APE^{RI}. The most interesting feature of APE^{OW} is that it adopts the overwrite-mode of the sponge construction instead of the XOR-absorbing mode for processing N. Namely after absorbing N, we replace r bits (called rate) with the first block of associated data A. This allows the receiver to verify authenticity and privacy of received (A, C, T) without N. Hence, it saves bandwidth for sending N and computational cost for processing N in decryption.
3. The APE^{CA} scheme improves bandwidth in different approach from APE^{OW}. APE^{OW} improves bandwidth by not sending N. The advantage of APE^{OW} becomes bigger as the size of N increases. APE^{CA} improves bandwidth even if N is small or even users choose not to use N. The idea here is avoiding the expansion of ciphertext or tag. Namely, when the input data size to encryption is $|N| + |A| + |M|$, we aim to achieve the output size of $|N| + |A| + |M| + c/2$ where $c/2$ is proven security level, i.e. $|C| = |M|$ and tag size is $c/2$ bits. This is the optimal bandwidth in the nonce based AE framework because $|C|$ cannot be smaller than $|M|$ and using $c/2$-bit tag is inevitable to ensure $c/2$-bit security. In other words, even by being based on the permutation, APE^{CA} achieves the competitive bandwidth with standard AE schemes.

Paper Outline. Section 2 summarizes specification of APE and its disadvantages. Section 3 introduces AE framework by Shrimpton and Terashima [23] and defines security under this framework. Our three new constructions APE^{RI}, APE^{OW}, and APE^{CA} are proposed in Sects. 4, 5, and 6, respectively. Finally, we compare the performance of those schemes and APE in Sect. 7.

2 Previous Work: APE

In this section, we introduce the specification of APE in Sect. 2.1 and explains several drawbacks of APE in Sect. 2.2.

2.1 Specification of APE

The APE scheme is the only existing permutation based AE mode which satisfies onlineness and offers a certain level of robustness against nonce-misuse, which is often called "up to prefix security," i.e. even if nonce is repeated, the scheme only leaks the information that the new message and associated data are the same as the previous ones up to the block where different data is processed for the first time. The mode of operation was firstly proposed at FSE 2014 [2], then it was later submitted to CAESAR with a specific primitive [1]. In this paper, we only focus our attention on the mode of operation.

The APE scheme adopts a b-bit permutation f as its underlying primitive. The b-bit state is further divided into r bits called *rate* and c bits called *capacity* like the well-known sponge or duplex constructions [6,7].

Encryption of APE. The APE scheme uses a c-bit key K. It takes an associated data A, a nonce N, a message M as input and computes the corresponding ciphertext C and a tag T. If the user compromises security to be "up to prefix security," the nonce input is not necessary. In order to unify the description, it is assumed that the nonce is a part of associated data A, thus N is not explicitly written even if N is used. In this paper, N is an important factor to minimize the bandwidth, thus N is often explicitly written independently of A.

The APE scheme initializes the state to r bits of zeros and c bits of K. Then A and M are divided into r bits of $A_0, A_1, A_2, \cdots$ and $M_0, M_1, M_2, \cdots$. Here, the designers limit that A and M must be a multiple of r.

To process A, the scheme first xors A_0 to rate and updates the state by computing f. This is iterated until all the associated data blocks are processed. In the end, the scheme xors a single bit one to capacity, which makes a border between A and M. Then, the scheme xors M_0 to rate, updates the state by f, and outputs r bits of rate as the corresponding ciphertext block C_0. This is iterated until all the message blocks are processed. Finally, c bits of K is xored to capacity, and the resulted c bits are output as tag T.

A typical choice of the ratio of r and c is $r = c/2$, which comes from $c/2$-bit security of the construction. The encryption of APE for $r = c/2$ is illustrated in Fig. 1.

Decryption of APE. The decryption of APE is a bit tricky, which is often called *backward decryption.* By concatenating the last ciphertext block and $K \oplus T$, the receiver constructs the b-bit state. Then, the receiver updates the state by f^{-1}, outputs the XOR of the rate and the next ciphertext block as the last plaintext block and replaces the rate with the next ciphertext block. This

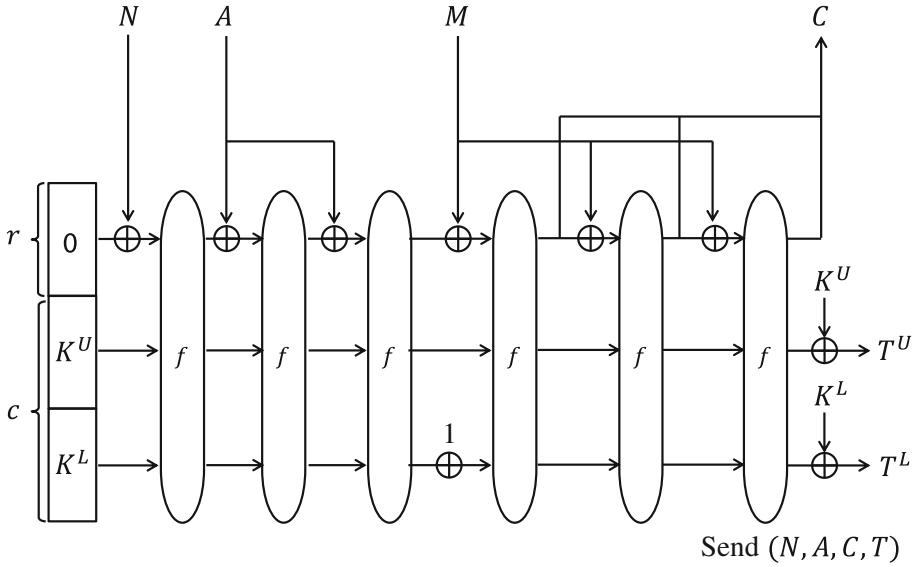

Fig. 1. Encryption of APE

is iterated until the second message block M_1 is recovered. The scheme then replaces rate by C_0 and updates the state by f^{-1}, but M_0 is not recovered at this stage. Let rate and capacity of the resulted state be S_r and S_c, respectively.

Procedures to recover the first message block M_0 and verification are very different. The receiver processes A as the encryption process (in the forward direction). Let rate and capacity of the resulted state be S'_r and S'_c. The receiver checks the match of S_c and S'_c for verification. If they match, the scheme computes $M_0 \leftarrow S_r \oplus S'_r$ and outputs the recovered M. If they do not match, the scheme returns the failure symbol $\perp$. The decryption process of APE for $r = c/2$ is illustrated in Fig. 2.

Security of APE. Intuitively, both of privacy and integrity of APE are proven to be secure up to $2^{c/2}$ queries in both of the nonce-respect and nonce-repeat settings.

2.2 Drawbacks of APE

Requiring High Bandwidth. Although integrity of APE is secure up to $2^{c/2}$ queries, owing to its computational structure, it is necessary to output a c-bit tag T, which increases communication cost compared to ordinary AE schemes that produce a $c/2$-bit tag for $c/2$-bit security.

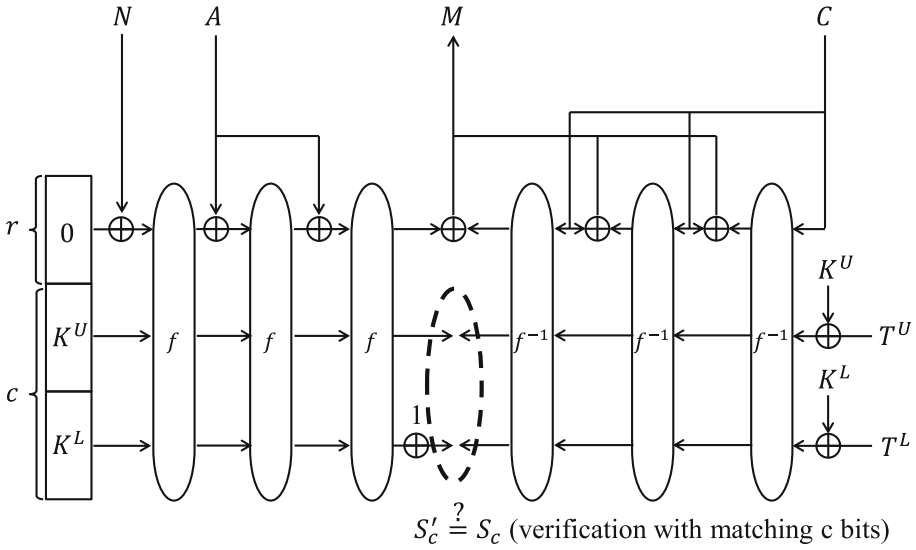

Fig. 2. Decryption of APE

Large Hardware Footprint. As illustrated in Figs. 1 and 2, the encryption of APE only requires to implement f, while the decryption of APE requires to implement both f and f^{-1}. This forces bigger hardware footprint for decryption devices.

High Computational Cost. The number of calls of f or f^{-1} is $(|N| + |A| + |M|)/r$, where $|X|$ represents the size of the variable X. At a glance this seems optimal. However, for the duplex construction, it is known that M and A can be processed simultaneously, e.g. *the concurrent absorption* [22], thus the number of permutation calls can be $(|N| + |A| + |M|)/b$. From a security reason, such an optimization cannot be applied to APE. (Intuitively, a tag reveals some information on the capacity value, which makes impossible to prove its security when the direct modification of any capacity value is allowed to the adversary.)

Remarks on Backward Decryption. The backward decryption of APE recovers the message from the last block to the first block. It is often said that this can be a drawback when the message length is big because it cannot be used for data streaming. In fact, APE was designed as a general-purpose AE scheme, hence the criticism makes sense. On the other hand, we point out that the backward decryption is not a problem at all in lightweight applications for IoT because the packet size is designed to be very short. For example, `LoRa` [17], a popular standard for Low Power Wide Area (LPWA), specifies that the maximum packet size is around 40 bytes, which is easy to store even for resource-restricted devices.

3 Security Definitions

In this section we first give a syntactical definition of authenticated encryption. Then we provide a security model for online authenticated encryption scheme.

3.1 Authenticated Encryption

Put $\mathbf{R} := \{0,1\}^r$ and $\mathbf{C} := \{0,1\}^c$, corresponding to the rate r and the capacity c. We use the notation $\mathbf{R}^* := \cup_{i=0}^{\ell} \mathbf{R}^i$ and $\mathbf{R}^+ := \cup_{i=1}^{\ell} \mathbf{R}^i$ where ℓ is the maximum length of queries that an adversary (an oracle machine) makes to its oracles. Here by the usual convention we regard $\mathbf{R}^0 = \varnothing$.

We adopt the generalized framework of authenticated encryption formalized by Shrimpton and Terashima [23]. An AE scheme is a triplet $(\mathcal{K}, \mathcal{E}, \mathcal{D})$. The key generation algorithm $\mathcal{K}$ simply draws a key $K \xleftarrow{\$} \mathbf{C}$ uniformly at random. Given a key $K \leftarrow \mathcal{K}(\cdot)$, the encryption algorithm $\mathcal{E}_K$ takes as its input a nonce $N \in \mathbf{R}^n$ for some fixed n, associate data $A \in \mathbf{R}^+$ and a message $M \in \mathbf{R}^+$ and outputs reconstruction information $RI \in \mathbf{R}^*$, ciphertext $C \in \mathbf{R}^*$ and a tag $T \in \mathbf{C}$ as $(RI, C, T) \leftarrow \mathcal{E}_K(N, A, M)$. Similarly, given a key K, the decryption algorithm $\mathcal{D}_K$ takes as its input reconstruction information $RI \in \mathbf{R}^*$, associated data $A \in \mathbf{R}^+$, ciphertext $C \in \mathbf{R}^*$ and a tag $T \in \mathbf{C}$, and outputs either the reject symbol $\perp$ or a message $M \in \mathbf{R}^+$ as $M \leftarrow \mathcal{D}_K(RI, A, C, T)$ where M may be equal to $\perp$. Optionally, an AE scheme may be equipped with a nonce recovery algorithm $\mathcal{R}_K$ which takes as its input reconstruction information $RI \in \mathbb{R}^*$, associated data $A \in \mathbf{R}^+$, ciphertext $C \in \mathbf{R}^*$ and a tag $T \in \mathbf{C}$, and outputs a (possibly partial) nonce $N[1] \in \mathbf{R}$ as $N[1] \leftarrow \mathcal{R}_K(RI, A, C, T)$, irrespective of the verification result. In this case the AE scheme is a quadruplet $(\mathcal{K}, \mathcal{E}, \mathcal{D}, \mathcal{R})$.

3.2 Security of Online AE Schemes

We prove the security of our schemes in the random-permutation model, regarding the underlying permutation $f : \mathbf{B} \to \mathbf{B}$ as an ideal. Here $\mathbf{B} := \{0,1\}^{r+c} = \mathbf{R} \times \mathbf{C}$. We consider the strongest adversaries possible, namely computationally unbounded ones. Hence we limit the power of adversaries only by query complexity. Let q, ℓ, σ denote the maximum number of queries, the maximum length of each query, and the total number of blocks of queries, respectively.

An adversary is given access to three oracles. Two of them are offline oracles $y \leftarrow f(x)$ and $x \leftarrow f^{-1}(y)$ where f is drawn uniformly at random from permutations on $\mathbf{B}$. They correspond to the underlying permutation. The remaining two are an encryption oracle $\mathcal{E}(\cdot,\cdot,\cdot)$ and a decryption oracle $\mathcal{D}(\cdot,\cdot,\cdot,\cdot)$. The goal of the adversary is to distinguish, by outputting a bit $b \in \{0,1\}$ after its interaction with oracles, between two worlds. In the real game, the encryption oracle is the real oracle $(RI, C, T) \leftarrow \mathcal{E}_K(N, A, M)$, and similarly the decryption oracle is the real oracle $M \leftarrow \mathcal{D}_K(RI, A, C, T)$. In the ideal game, the encryption oracle $\$(N, A, M)$ is defined as follows, and the decryption oracle is simply $\perp(RI, A, C, T)$ which always returns the reject symbol $\perp$.

The ideal encryption oracle $(RI, C, T) \leftarrow \$(N, A, M)$ is defined as follows. The value RI is computed in exactly the same way as the real world, i.e. $(RI, \cdot, \cdot, \cdot) \leftarrow \mathcal{E}_K(N, A, M)$. To describe how C and T are generated in the ideal world, write $M = M[1]M[2]\cdots M[w]$. We choose functions $g : \mathbf{R}^+ \times \mathbf{R}^* \rightarrow \mathbf{R}$ and $g' : \mathbf{R}^+ \times \mathbf{R}^+ \rightarrow \mathbf{C}$ uniformly at random, and define

$$\begin{aligned} C[i] &:= g(NA, M[1]M[2]\cdots M[i]) \qquad \text{for } i = 1, 2, \ldots, w \\ T &:= g'(NA, M). \end{aligned}$$

When there is a nonce recovery algorithm $N[1] \leftarrow \mathcal{R}_K(RI, A, C, T)$, in the ideal world this is replaced with a random oracle $\$'$ which chooses an independently random function $g'' : \mathbf{R} \times \mathbf{R}^+ \times \mathbf{R}^* \times \mathbf{C} \rightarrow \mathbf{R}$ and outputs

$$N[1] \leftarrow g''(RI, A, C, T).$$

Now formally we define the advantage of an adversary D as

$$\mathrm{Adv}(D) := \Pr\left[D^{f, f^{-1}, \mathcal{E}_K, \mathcal{D}_K, \mathcal{R}_K} = 1\right] - \Pr\left[D^{f, f^{-1}, \$, \perp, \$'} = 1\right],$$

where $D^{\cdots} = 1$ denotes the event that D outputs 1 after interacting with its oracles $\cdots$. The probabilities are defined over random coins used by the oracles, and those used by D if any.

We assume that adversary D does not repeat a query or make a trivial-win query. That is, if D makes a query $(RI, C, T) \leftarrow \mathcal{E}_K(N, A, M)$, then D makes neither a $\mathcal{D}$-query (RI, A, C, T) nor an $\mathcal{R}$-query (RI, A, C, T).

4 APE^{RI}: Minimizing Hardware Footprint

In this section we present our first scheme, APE^{RI}, which offers a smaller hardware footprint than the original APE by its encryption algorithm making calls only to the forward permutation f while its decryption algorithm only to the inverse f^{-1}. The construction follows the generalized AE framework that utilizes reconstruction information RI. See Figs. 3 and 4 for illustration of the scheme.

The encryption algorithm of APE^{RI} is very similar to that of APE. We assume $N \in \mathbf{R}$. The main difference is that it additionally outputs r bits of the internal state as RI. The user (who has performed the encryption algorithm) does not send N but sends RI instead, together with C, T. Note that $|RI| = |N|$, and hence the communication cost of APE^{RI} is exactly the same as that of APE. Also note that the encryption of APE only calls f and not f^{-1}, and APE^{RI} inherits this good property. A small remark here is that the position of xoring 1 in the capacity is moved 1-block earlier in the new scheme than in APE. This is because APE^{RI} starts outputting the rate value 1-block earlier than APE, and in this way we can "reuse" the known results of APE for proving the security of APE^{RI}.

A major difference between APE^{RI} and APE comes in the decryption process. To decrypt (RI, A, C, T), the process is exactly the same up to the recovery

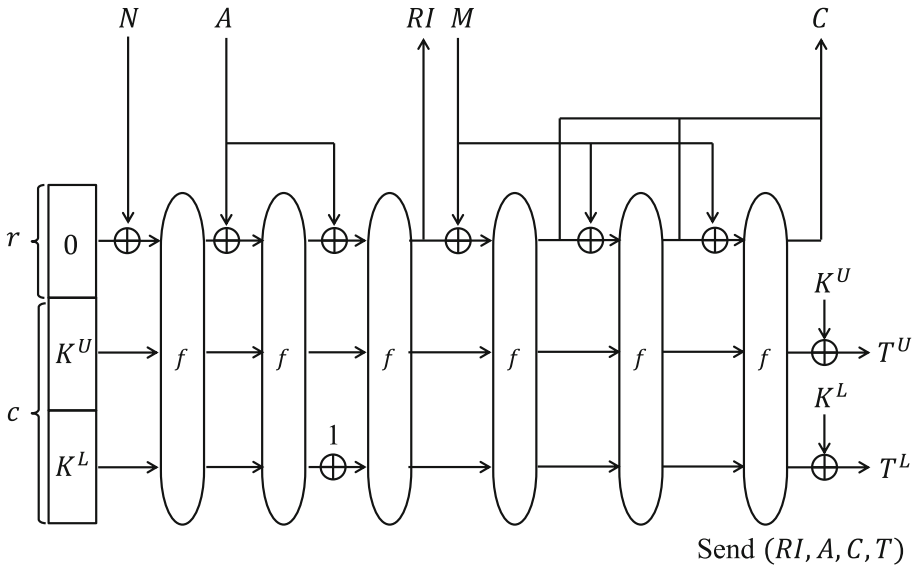

Fig. 3. Illustration of APE^{RI} encryption

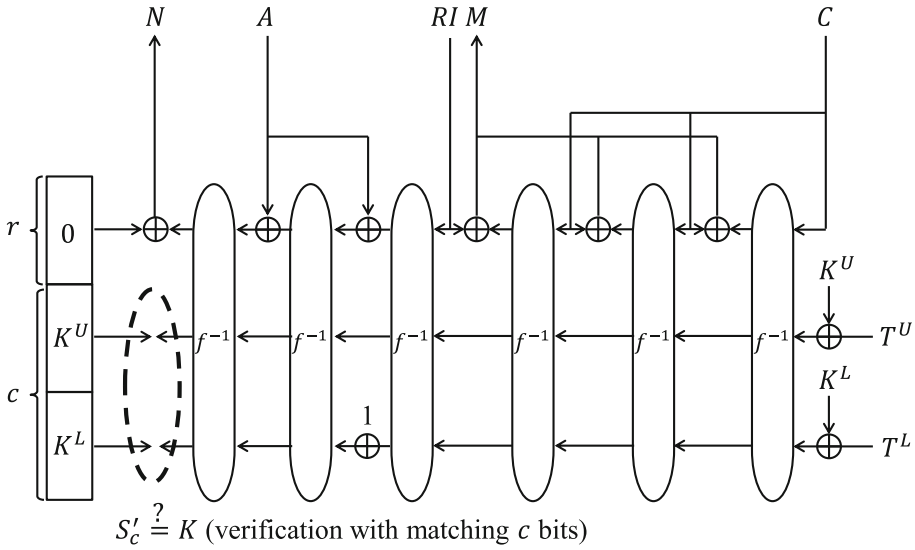

Fig. 4. Illustration of APE^{RI} decryption

of $M[2]$. In APE^{RI}, $M[1]$ can be recovered in a continuous way thanks to the presence of RI. After M is recovered, the decryption procedure continues to backtrack the computation by using A. After finishing absorbing all blocks of A, the capacity should match the value of K for verification. Formally, the encryption and decryption algorithms of APE^{RI} are defined in Fig. 5.

$M[1]M[2]\cdots M[w] \leftarrow M$
$A[1]A[2]\cdots A[u] \leftarrow A$
$V \leftarrow (0^r, K^U, K^L)$
$V \leftarrow f(N \oplus V^r, V^U, V^L)$
for $i = 1$ to $u-1$ **do**
    $V \leftarrow f(A[i] \oplus V^r, V^U, V^L)$
end for
$V \leftarrow f(A[u] \oplus V^r, V^U, V^L \oplus 1)$
$RI \leftarrow V^r$
for $i = 1$ to w **do**
    $V \leftarrow f(M[i] \oplus V^r, V^U, V^L)$
    $C[i] \leftarrow V^r$
end for
$T^U \leftarrow V^U \oplus K^U$
$T^L \leftarrow V^L \oplus K^L$
return (RI, C, T)

$C[1]C[2]\cdots C[w] \leftarrow C$
$A[1]A[2]\cdots A[u] \leftarrow A$
$V \leftarrow (0^r, T^U \oplus K^U, T^L \oplus K^L)$
for $i = w$ to 2 **do**
    $V \leftarrow f^{-1}(C[i], V^U, V^L)$
    $M[i] \leftarrow V^r \oplus C[i-1]$
end for
$V \leftarrow f^{-1}(C[1], V^U, V^L)$
$M[1] \leftarrow V^r \oplus RI$
$V \leftarrow f^{-1}(RI, V^U, V^L)$
$V \leftarrow f^{-1}(A[u] \oplus V^r, V^U, V^L \oplus 1)$
for $i = u-1$ to 2 **do**
    $V \leftarrow f^{-1}(A[i] \oplus V^r, V^U, V^L)$
end for
if $V^U \| V^T = K$ **then**
    return M
else
    return $\perp$
end if

Fig. 5. Encryption and decryption algorithms of APE^{RI}

4.1 Security of APE^{RI}

In this section we prove that APE^{RI} is as secure as the original APE as an authenticated encryption scheme. Recall that for APE^{RI} we assume $N \in \mathbf{R}$ (i.e. $n = 1$).

Theorem 1. *Let $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be APE^{RI}. Then Π is at least as secure as the original APE scheme $\Pi' = (\mathcal{K}, \mathcal{E}', \mathcal{D}')$ that uses the same underlying permutation f and the parameters r, c. Specifically, for any adversary D attacking Π, there exists an adversary D' that attacks Π' and satisfies*

$$\mathrm{Adv}_{\Pi}(D) \leq \mathrm{Adv}_{\Pi'}(D') + \frac{4\sigma^2}{2^{r+c}} + \frac{4\sigma(2\sigma+1)}{2^c},$$

where σ denotes the query complexity of D and D' makes at most twice many queries to its oracles as D.

Proof. Consider an intermediate scheme $\tilde{\Pi} := (\mathcal{K}, \mathcal{E}, \mathcal{D}')$. We first show that $\tilde{\Pi}$ is as secure as the original APE $\Pi' = (\mathcal{K}, \mathcal{E}', \mathcal{D}')$. Given an adversary $\tilde{D}$ that

attacks $\tilde{\Pi}$, we construct an adversary D' that attacks Π'. Simply, D' runs $\tilde{D}$. When $\tilde{D}$ makes queries to its $f/f^{-1}/\mathcal{D}'$ oracles, the adversary D' forwards the queries to its $f/f^{-1}/\mathcal{D}'$ oracles, respectively, and returns to $\tilde{D}$ whatever D' gets from its oracles. When $\tilde{D}$ makes an $\mathcal{E}$-query (N, A, M), the adversary D' makes an $\mathcal{E}'$-query $(N, A[1] \cdots A[u-1], A[u]M)$ and receives a reply (C, T) from its $\mathcal{E}'$-oracle. Then D' returns (RI, C', T) to $\tilde{D}$, where $RI := C[1]$ and $C' = C[2] \cdots C[w+1]$. Eventually, the adversary D' outputs the bit b that $\tilde{D}$ outputs. We see that

$$\mathrm{Adv}_{\tilde{\Pi}}(\tilde{D}) \leq \mathrm{Adv}_{\Pi'}(D'), \tag{1}$$

where the query complexity of D' is the same as that of $\tilde{D}$.

Next we consider another intermediate scheme $\Pi^+ := (\mathcal{K}, \mathcal{E}, \mathcal{D}', \mathcal{R})$ which is nothing but the above $\tilde{\Pi}$ now equipped with the recovery function $N \leftarrow \mathcal{R}_K(A, RI, C, T)$. Given D^+ that attacks Π^+, we can construct an adversary $\tilde{D}$ that attacks $\tilde{\Pi}$ by simulating the $\mathcal{R}$-oracle with a random function. The simulation fails only when the recovery function $\mathcal{R}_K$ does not behave random, and such a probability can be bounded by $\sigma^2/2^{r+c} + 2\sigma(\sigma+1)/2^c$ (Andreeva et al. [2, Theorem 2]) where σ denotes the query complexity of D^+. Therefore, we have

$$\mathrm{Adv}_{\Pi^+}(D^+) \leq \mathrm{Adv}_{\tilde{\Pi}}(\tilde{D}) + \frac{\sigma^2}{2^{r+c}} + \frac{2\sigma(\sigma+1)}{2^c}, \tag{2}$$

where the query complexity of $\tilde{D}$ is no more than that of D^+.

Finally, given an adversary D that attacks $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$, we construct an adversary D^+ that attacks Π^+ as follows. The adversary D^+ runs D as its subroutine and forwards all $f/f^{-1}/\mathcal{E}$ queries and replies. When D makes a $\mathcal{D}$-query (A, RI, C, T), the adversary D^+ first makes an $\mathcal{R}$-query (A, RI, C, T) and receives $N \leftarrow \mathcal{R}(A, RI, C, T)$. Then D^+ makes a $\mathcal{D}$-query (N, A, C, T). We see that D^+ perfectly simulates the real and ideal worlds for D and hence

$$\mathrm{Adv}_{\Pi}(\tilde{D}) \leq \mathrm{Adv}_{\Pi^+}(D^+), \tag{3}$$

where the query complexity of D^+ is at most twice that of D. Combining (1), (2) and (3) proves the theorem. □

5 APE^{OW}: Lower Bandwidth via Nonce-Less Decryption

APE^{RI} introduced in the previous section could improve the hardware footprint, while another strong drawback of APE, namely bandwidth, was untouched with APE^{RI}. The main purpose of this section is modifying APE^{RI} to improve the bandwidth by keeping the same hardware footprint of APE^{RI}.

The most interesting feature in this construction is using the *overwrite-mode* of the sponge hash construction for processing N. During encryption, we process N as the standard keyed sponge construction to make a b-bit state. We then replace r-bit rate with zeros. The remaining c bits of the state inherit the result of processing N. Intuitively, the r bits of zeros are the bit-string used for authentication. Hence, the sender does not need to communicate N to the

$M[1]M[2]\cdots M[w] \leftarrow M$
$A[1]A[2]\cdots A[u] \leftarrow A$
$N[1]N[2]\cdots N[v] \leftarrow N$
$V \leftarrow (0^r, K^U, K^L)$
for $i = 1$ to v **do**
  $V \leftarrow f(N[i] \oplus V^r, V^U, V^L)$
end for
$V \leftarrow (0^r, V^U, V^L)$
for $i = 1$ to $u - 1$ **do**
  $V \leftarrow f(A[i] \oplus V^r, V^U, V^L)$
end for
$V \leftarrow f(A[u] \oplus V^r, V^U, V^L \oplus 1)$
$RI \leftarrow V^r$
for $i = 1$ to w **do**
  $V \leftarrow f(M[i] \oplus V^r, V^U, V^L)$
  $C[i] \leftarrow V^r$
end for
$T^U \leftarrow V^U \oplus K^U$
$T^L \leftarrow V^L \oplus K^L$
return (RI, C, T)

$C[1]C[2]\cdots C[w] \leftarrow C$
$A[1]A[2]\cdots A[u] \leftarrow A$
$V \leftarrow (0^r, T^U \oplus K^U, T^L \oplus K^L)$
for $i = w$ to 2 **do**
  $V \leftarrow f^{-1}(C[i], V^U, V^L)$
  $M[i] \leftarrow V^r \oplus C[i-1]$
end for
$V \leftarrow f^{-1}(C[1], V^U, V^L)$
$M[1] \leftarrow V^r \oplus RI$
$V \leftarrow f^{-1}(RI, V^U, V^L)$
$V \leftarrow f^{-1}(A[u] \oplus V^r, V^U, V^L \oplus 1)$
for $i = u - 1$ to 2 **do**
  $V \leftarrow f^{-1}(A[i] \oplus V^r, V^U, V^L)$
end for
if $V^r \oplus A[1] = 0$ **then**
  return M
else
  return $\perp$
end if

Fig. 6. Encryption and decryption algorithms of APE^{OW}

receiver, which contributes to improve the bandwidth. In order to decrypt the first message block without implementing f^{-1}, we need r-bits of RI as introduced in APE^{RI}. The construction is named APE^{OW}, and the encryption and decryption procedures of APE^{OW} are defined in Fig. 6. Their illustrations for $r = c/2$ are given in Figs. 7 and 8.

Advantages of APE^{OW}. Advantages of APE^{OW} can be summarized as follows.

Requiring Low Bandwidth. The amount of communicated data is reduced by a factor of $|N|$ bits due to the omission of sending N, while it is increased by a factor of r bits due to RI. Thus, the bandwidth is improved from the original APE by a factor of $|N| - r$ bits. Obviously, if $|N|$ is so small that $|N| - r$ is negative, users should use APE^{RI} instead of APE^{OW}. If $|N| > r$, APE^{OW} simply outperforms APE^{RI}.

Small Hardware Footprint. APE^{OW} inherits the advantage of APE^{RI}, namely users need to implement only f for encryption devices and only f^{-1} for decryption devices.

Low Computational Cost. The encryption procedure of APE^{OW} is exactly the same as APE^{RI} but for overwriting the rate after processing N with 0^r instead of directly xoring A. Hence the computational cost of encryption of APE^{OW} is the same as one for APE^{RI} and even for the original APE. Computational cost of decryption is greatly improved from APE^{RI} and APE owing to the omission of processing N. This is another big advantage of APE^{OW}.

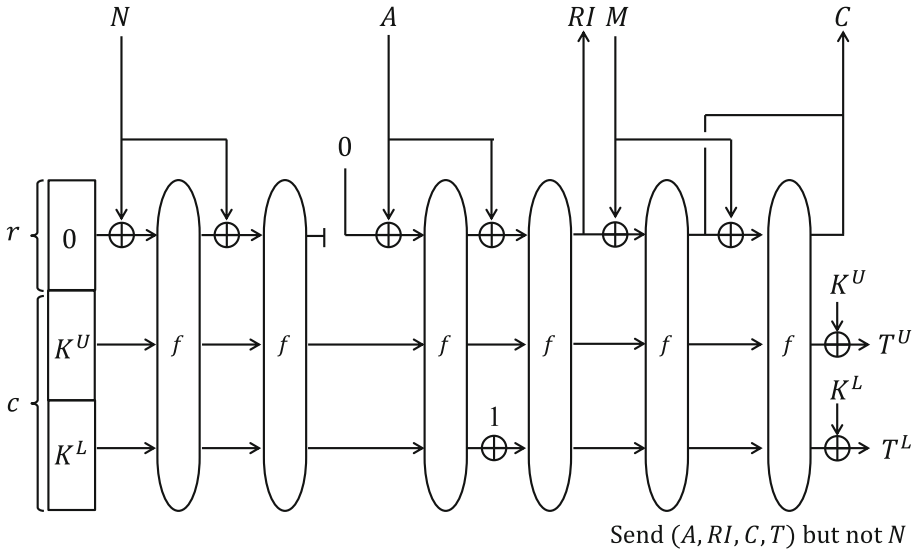

Fig. 7. Encryption of APE^{OW}

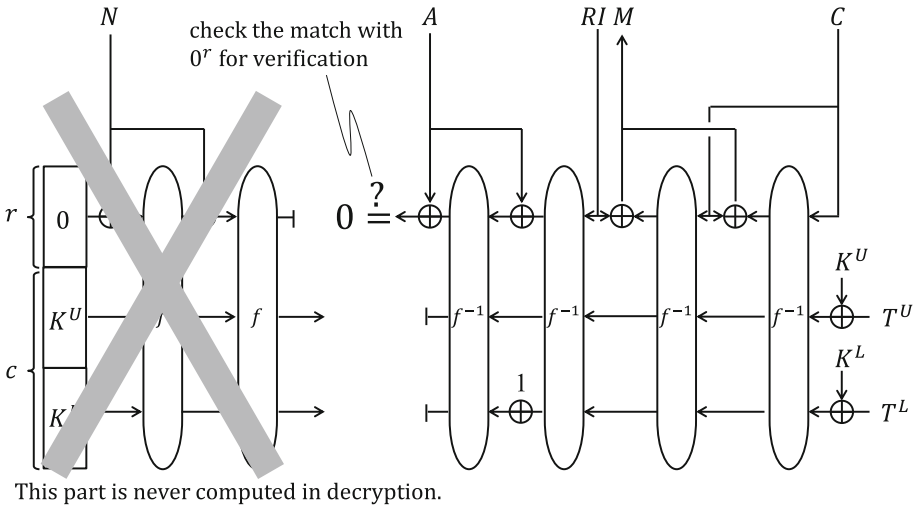

Fig. 8. Decryption of APE^{OW}

Recommended Parameters of APE^{OW}. As defined in Fig. 6, verification is performed by matching the r-bit information, thus security for tag guessing is up to r bits. When $r = c/2$, this matches the security of APE. When $r > c/2$, this part is not the bottleneck and thus the security is standard $c/2$ bits. When $r < c/2$, this part lowers the security of the entire construction. Hence, we do not recommend using APE^{OW} when $r < c/2$. Instead, we recommend another construction, which will be explained in Sect. 6.

5.1 Security of APE^{OW}

In this section we prove that APE^{OW} is secure as an AE scheme. The scheme is secure up to $\min\{2^r, 2^{c/2}\}$ queries, which becomes $2^{c/2}$ when $r \geq c/2$.

Theorem 2. *Let $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be* APE^{OW}*. Then Π is secure as an AE scheme. Specifically, let $\Pi' = (\mathcal{K}, \mathcal{E}', \mathcal{D}')$ be the original APE scheme that uses the same underlying permutation f and the parameters r, c. Then, for any adversary D attacking Π, there exists an adversary D' that attacks Π' and satisfies*

$$\mathrm{Adv}_{\Pi}(D) \leq \mathrm{Adv}_{\Pi'}(D') + \frac{2\sigma^2}{2^{r+c}} + \frac{3\sigma(2\sigma+1)}{2^c} + \frac{\sigma}{2^r},$$

where σ denotes the query complexity of D and D' makes at most twice many queries to its oracles as D.

Proof. We consider an intermediate scheme $\tilde{\Pi} = (\tilde{\mathcal{K}}, \tilde{\mathcal{E}}, \tilde{\mathcal{D}})$ which is a modification of APE^{RI}, as follows:

1. Two independent keys $K_1, K_2 \in \mathbf{C}$ are used for the initialization and the masking of tags, respectively. So we have $(K_1, K_2) \leftarrow \tilde{\mathcal{K}}(\cdot)$.
2. The encryption algorithm $\tilde{\mathcal{E}}$ generates RI just like APE^{RI}, as $(RI, C, T) \leftarrow \tilde{\mathcal{E}}_{K_1,K_2}(N, A, M)$.
3. The decryption algorithm $\tilde{\mathcal{D}}$ takes as its input both the nonce N and the reconstruction information RI, and the verification is done not by comparing the capacity state value with K_1 but by comparing the rate state value with the first block $N[1]$ of the nonce.

Now the security proof of the original APE by Andreeva et al. [2, Theorem 2] also applies to $\tilde{\Pi}$, and we obtain

$$\mathrm{Adv}(\tilde{D}) \leq \frac{\sigma^2}{2^{r+c}} + \frac{2\sigma(\sigma+1)}{2^c} + \frac{\sigma}{2^r} \tag{4}$$

for any adversary $\tilde{D}$ that attacks $\tilde{\Pi}$ and makes queries of complexity at most σ.

Now we consider intermediate scheme $\Pi^* = (\mathcal{K}^*, \mathcal{E}^*, \mathcal{D}^*)$ which operates as follows:

1. Choose a random function $g : \mathbf{R}^n \rightarrow \mathbf{C}$. This is used for generating the initialization key as $K_1 \leftarrow g(N)$.

2. An independent key $K_2 \xleftarrow{\$} \mathbf{C}$ is used for masking tags.
3. The encryption algorithm $\mathcal{E}^*$ and the decryption algorithm $\mathcal{D}^*$ are exactly the same as those of $\tilde{\Pi}$, except the key K_1 is generated as above.

Now let D^* be an adversary attacking Π^*, and then by a hybrid argument we get

$$\mathrm{Adv}(D^*) \leq \frac{\sigma^2}{2^{r+c}} + \frac{2\sigma(\sigma+1)}{2^c} + \frac{\sigma}{2^r}, \tag{5}$$

where again σ denotes the total query complexity of the adversary.

Lastly, we compare Π and Π^*. In the former the "keys" are generated from N and K_1 through the calls of permutation f, whereas in the latter the "keys" are generated as $g(N)$. Hence by the same argument as the privacy proof of APE by Andreeva et al. [2, Theorem 1], we get

$$\mathrm{Adv}(D) \leq \mathrm{Adv}(D^*) + \frac{\sigma^2}{2^{r+c}} + \frac{\sigma(\sigma+1)}{2^c}, \tag{6}$$

where σ denotes the total query complexity of D. From (4), (5) and (6) we see that the theorem is proved. □

6 APE^{CA}: Lower Bandwidth via Absorption in Capacity

The idea of improving bandwidth by APE^{OW} is omitting the communication of N between encryption and decryption players. In this section, we present another construction to improve bandwidth from a different point of view. Recall that one of the drawbacks of APE is that the tag size (c bits) is always bigger than the security parameter ($c/2$ bits) owing to its decryption procedure. In this section, our goal is minimizing the expansion of ciphertext or expansion of tag in order to make the bandwidth to be competitive as standard AE schemes, i.e. when the input data size to encryption is $|N| + |A| + |M|$, we aim to achieve the output size of $|N| + |A| + |M| + c/2$ by making $|C| = |M|$ and $|T| = c/2$.

The overall idea is as follows. In the original APE, verification is performed by checking the match of c bits as illustrated in Fig. 2. The same applies to the verification of APE^{RI}. Considering that the security of the entire construction is $c/2$ bits, using a c-bit string for verification can be regarded as the waste of the information. Hence, our idea is separating the c-bit string used for verification of APE^{RI} (K) into two $c/2$-bit strings (K^U and K^L), and use one of them for verification and use the other one for encrypting $c/2$ bits of M denoted by $M^{c/2}$. Differently from APE^{RI}, we now send N in clear, thus do not need to hide r bits of N at the very beginning by using 0^r in the initial state. Instead, we encrypt r bits of M denoted by M^r at this position. In the end, r bits of RI in APE^{RI} can be a ciphertext of M^r and $c/2$ bits of T^U in APE^{RI} can be a ciphertext of $M^{c/2}$, which achieves $|M| = |C|$. The remaining tag size is $c/2$ bits, thus $|T| = c/2$ is achieved.

Our idea of absorbing M both in rate and (a half of) capacity can be regarded as a variant of the concurrent absorption [22], which absorbs M in rate and A in capacity. We call this scheme APE^{CA}, and the encryption and decryption algorithms are defined in Fig. 9. They are also illustrated in Figs. 10 and 11.

$M^r M^{c/2} M[1] M[2] \cdots M[w] \leftarrow M$
$A[1] A[2] \cdots A[u] \leftarrow A$
$V \leftarrow (0^r, K^U, K^L)$
$V \leftarrow f(M^r \oplus V^r, M^{c/2} \oplus V^U, V^L)$
$V \leftarrow f(N \oplus V^r, V^U, V^L)$
for $i = 1$ to $u - 1$ **do**
    $V \leftarrow f(A[i] \oplus V^r, V^U, V^L)$
end for
$V \leftarrow (A[u] \oplus V^r, V^U, V^L \oplus 1)$
$C^r \leftarrow V^r$
for $i = 1$ to w **do**
    $V \leftarrow f(M[i] \oplus V^r, V^U, V^L)$
    $C[i] \leftarrow V^r$
end for
$C^{c/2} \leftarrow V^U \oplus K^U$
$T \leftarrow V^L \oplus K^L$
$C \leftarrow C^r C^{c/2} C[1] C[2] \cdots C[w]$
return (C, T)

$C^r C^{c/2} C[1] C[2] \cdots C[w] \leftarrow C$
$A[1] A[2] \cdots A[u] \leftarrow A$
$V \leftarrow (0^r, K^U \oplus C^{c/2}, K^L \oplus T)$
for $i = w$ to 2 **do**
    $V \leftarrow f^{-1}(C[i], V^U, V^L)$
    $M[i] \leftarrow V^r \oplus C[i-1]$
end for
$V \leftarrow f^{-1}(C[1], V^U, V^L)$
$M[1] \leftarrow V^r \oplus C^r$
$V \leftarrow f^{-1}(C^r, V^U, V^L)$
$V \leftarrow f^{-1}(A[u] \oplus V^r, V^U, V^L \oplus 1)$
for $i = u - 1$ to 1 **do**
    $V \leftarrow f^{-1}(A[i] \oplus V^r, V^U, V^L)$
end for
$V \leftarrow f^{-1}(N \oplus V^r, V^U, V^L)$
if $V^L = K^L$ **then**
    $M^r \leftarrow V^r$
    $M^{c/2} \leftarrow V^U \oplus K^U$
    $M \leftarrow M^r M^{c/2} M[1] M[2] \cdots M[w]$
    return M
else
    return $\perp$
end if

Fig. 9. Encryption and decryption algorithms of APE^{CA}

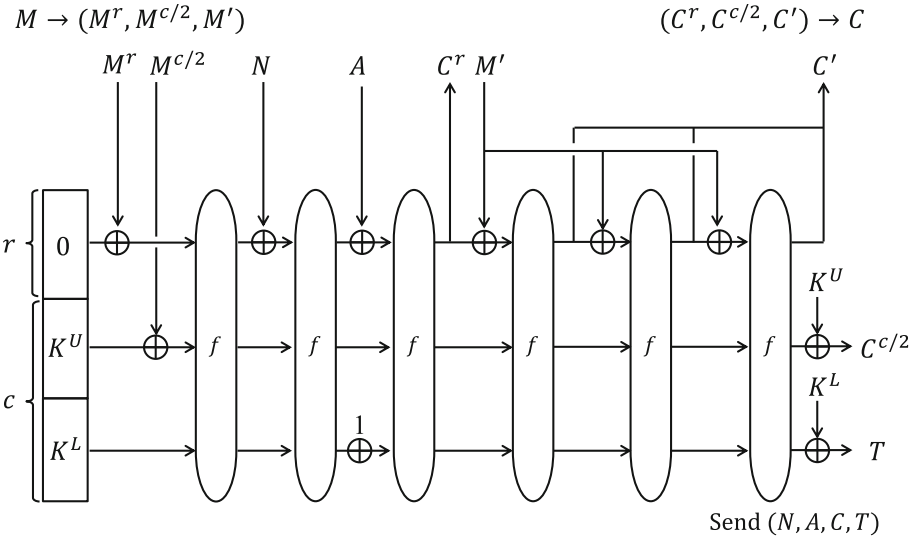

Fig. 10. Encryption of APE^{CA}

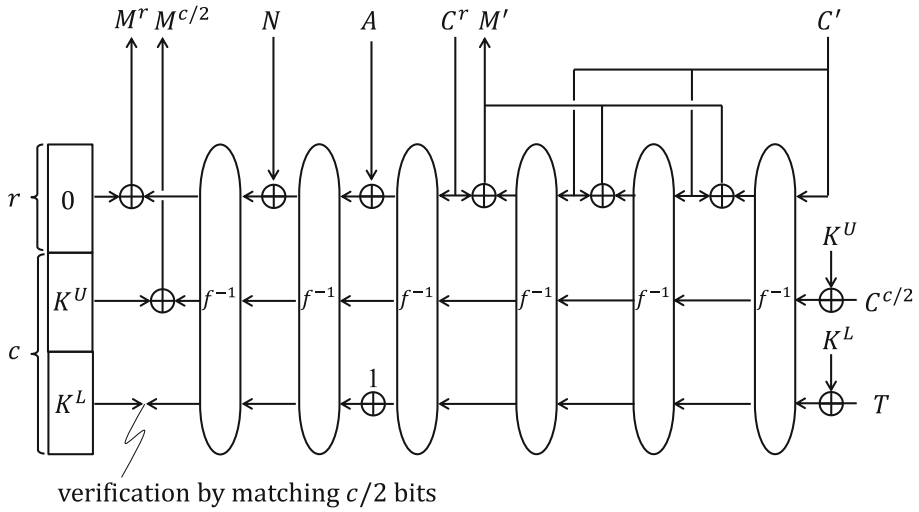

Fig. 11. Decryption of APE^{CA}

Advantages of APE^{CA}. Advantages of APE^{CA} can be summarized as follows.

Requiring Low Bandwidth. The amount of communicated data is reduced by a factor of $r + c/2$ bits due to the omission of sending M^r and $M^{c/2}$, while it is increased by a factor of r bits due to C^r. Thus, the bandwidth is improved from the original APE or APE^{RI} by a factor of $c/2$ bits.

Small Hardware Footprint. APE^{OW} inherits the advantage of APE^{RI}, namely users need to implement only f for encryption devices and only f^{-1} for decryption devices.

Low Computational Cost. At the very beginning, $r + c/2$ bits of M are absorbed accordingly to the line of concurrent absorption. When $r = c/2$, this corresponds to reducing the number of f or f^{-1} calls by 1. Differently from APE^{OW}, improvement of the computational cost can be exploited both in encryption and decryption algorithms.

7 Comparisons of Proposed Schemes

In this section, we compare the performance of APE, APE^{RI}, APE^{OW}, and APE^{CA}. Let $|N|$, $|A|$ and $|M|$ be nonce size, associated data size, and message size, respectively. We then compare the bandwidth and computational cost for encrypting this message and for decrypting its ciphertext. Hardware footprint is simply measured by the types of permutations to be implemented. The comparison is given in Table 1.

When the message length is $|M|$ and security level is $c/2$ bits, the bandwidth should ideally be $|N|+|A|+|M|+c/2$, while APE requires $|N|+|A|+|M|+c$ for

Table 1. Performance comparison of our AE schemes. We put $X := |N| + |A| + |M|$.

Scheme	Bandwidth	Hardware footprint		Computational cost		Security
		Enc	Dec	Enc	Dec	
APE	$X + c$	f	f, f^{-1}	X/r	X/r	$c/2$
APE^{RI}	$X + c$	f	f^{-1}	X/r	X/r	$c/2$
APE^{OW}	$X - \|N\| + r + c$	f	f^{-1}	X/r	$(X - \|N\|)/r$	$\min\{r, c/2\}$
APE^{CA}	$X + c/2$	f	f^{-1}	$(X - c)/r$	$(X - c)/r$	$c/2$

the expanded tag. APE requires both f and f^{-1} for decryption, and the computational cost is standard $(|N| + |A| + |M|)/r$ in both encryption and decryption.

APE^{RI} simply improves APE by removing the necessity of f in decryption. APE^{OW} omits sharing N between the sender and the receiver. It should be stressed that security of APE^{OW} also depends on b. The condition to ensure the standard $c/2$-bit security is $r \geq c/2$. In APE^{OW}, the bandwidth is reduced from APE when $|N| \geq r$. For example, when the permutation size is 256 bits and $r = 96$, $c = 160$ for 80-bit security, APE^{OW} has better bandwidth than APE to process the nonce which is longer than or equal to 96 bits. Another advantage of APE^{OW} is that N does not have to be processed during decryption. APE^{CA} always outperforms APE with respect to all of bandwidth, hardware footprint, computational cost.

The better choice between APE^{OW} and APE^{CA} depends on the nonce length and the choice of the rate and capacity sizes. Considering that communication speed is slower than computation speed, minimizing the bandwidth is likely to be the most important issue.

Condition 1: To ensure $c/2$-bit security, APE^{CA} should be chosen when $r < c/2$.
Condition 2a: If $r \geq c/2$, compare the size of $r + c/2$ and N. If $N < r + c/2$, APE^{CA} offers better bandwidth than APE^{OW}.
Condition 2b: Otherwise, APE^{OW} offers better bandwidth than APE^{CA}.

For example, when the ratio of a rate size to a capacity size is one to two, $|N| < c$ is the border to choose APE^{OW} or APE^{CA}. Considering the practical parameters, APE^{OW} should be chosen when $|N| = 64$ for a 80-bit permutation, or $|N| = 48$ for a 64-bit permutation.

References

1. Andreeva, E., Bilgin, B., Bogdanov, A., Luykx, A., Mendel, F., Mennink, B., Mouha, N., Wang, Q., Yasuda, K.: PRIMATEs v1. Submission to CAESAR (2014)
2. Andreeva, E., Bilgin, B., Bogdanov, A., Luykx, A., Mennink, B., Mouha, N., Yasuda, K.: APE: authenticated permutation-based encryption for lightweight cryptography. In: Cid, C., Rechberger, C. (eds.) FSE 2014. LNCS, vol. 8540, pp. 168–186. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46706-0_9

3. 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). https://doi.org/10.1007/978-3-642-15031-9_1
4. Bellare, M., Namprempre, C.: Authenticated encryption: relations among notions and analysis of the generic composition paradigm. In: Okamoto, T. (ed.) ASIACRYPT 2000. LNCS, vol. 1976, pp. 531–545. Springer, Heidelberg (2000). https://doi.org/10.1007/3-540-44448-3_41
5. Bernstein, D.: CAESAR Competition (2013). http://competitions.cr.yp.to/caesar.html
6. Bertoni, G., Daemen, J., Peeters, M., Van Assche, G.: On the indifferentiability of the sponge construction. In: Smart, N. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 181–197. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-78967-3_11
7. 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). https://doi.org/10.1007/978-3-642-28496-0_19
8. 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). https://doi.org/10.1007/978-3-642-23951-9_21
9. CRYPTREC Lightweight Cryptography Working Group: CRYPTREC cryptographic technology guideline (lightweight cryptography) (2017). https://www.cryptrec.go.jp/report/cryptrec-gl-0001-2016-e.pdf
10. Dinca, L.M., Hancke, G.: Behavioural sensor data as randomness source for IoT devices. In: ISIE 2017, pp. 2038–2043. IEEE (2017)
11. Dworkin, M.: Recommendation for block cipher modes of operation: Galois/Counter Mode (GCM) and GMAC. NIST Special Publication 800-38D (2007)
12. Fleischmann, E., Forler, C., Lucks, S.: McOE: a family of almost foolproof online authenticated encryption schemes. In: Canteaut, A. (ed.) FSE 2012. LNCS, vol. 7549, pp. 196–215. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-34047-5_12
13. 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). https://doi.org/10.1007/978-3-642-22792-9_13
14. Hoang, V.T., Krovetz, T., Rogaway, P.: AEZ v1: Authenticated-encryption by enciphering. Submission to CAESAR (2014)
15. JTC 1/SC 27: Information technology–Security techniques–Lightweight cryptography–Part 1: General. ISO/IEC 29192-1 (2012)
16. Kannan, S., Karimi, N., Sinanoglu, O., Karri, R.: Security vulnerabilities of emerging nonvolatile main memories and countermeasures. IEEE Trans. CAD Integr. Circ. Syst. **34**(1), 2–15 (2015)
17. LoRa Alliance: LoRa specification (2015). https://www.lora-alliance.org/
18. McKay, K.A., Bassham, L., Turan, M.S., Mouha, N.: Report on lightweight cryptography. NISTIR 8114 (2017). http://nvlpubs.nist.gov/nistpubs/ir/2017/NIST.IR.8114.pdf

19. Mennink, B., Reyhanitabar, R., Vizár, D.: Security of full-state keyed sponge and duplex: applications to authenticated encryption. In: Iwata, T., Cheon, J.H. (eds.) ASIACRYPT 2015. LNCS, vol. 9453, pp. 465–489. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-48800-3_19
20. Rogaway, P.: Nonce-based symmetric encryption. In: Roy, B., Meier, W. (eds.) FSE 2004. LNCS, vol. 3017, pp. 348–358. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-25937-4_22
21. 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). https://doi.org/10.1007/11761679_23
22. Sasaki, Y., Yasuda, K.: How to incorporate associated data in sponge-based authenticated encryption. In: Nyberg, K. (ed.) CT-RSA 2015. LNCS, vol. 9048, pp. 353–370. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-16715-2_19
23. Shrimpton, T., Terashima, R.S.: A modular framework for building variable-input-length tweakable ciphers. In: Sako, K., Sarkar, P. (eds.) ASIACRYPT 2013. LNCS, vol. 8269, pp. 405–423. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-42033-7_21