The INT-RUP Security of OCB with Intermediate (Parity) Checksum

Ping Zhang[1], Peng Wang[2], and Honggang Hu[1]

[1] Key Laboratory of Electromagnetic Space information, CAS
University of Science and Technology of China, Hefei, China, 230027
zgp@mail.ustc.edu.cn,hghu2005@ustc.edu.cn
[2] Institute of Information Engineering, CAS, Beijing, China, 100049, wp@is.ac.cn

Abstract. OCB is neither integrity under releasing unvierified plaintext (INT-RUP) nor nonce-misuse resistant. The tag of OCB is generated by encrypting plaintext checksum, which is vulnerable in the INT-RUP security model. This paper focuses on the weakness of the checksum processing in OCB. We describe a new notion, called plaintext or ciphertext checksum (PCC), which is a generalization of plaintext checksum, and prove that all authenticated encryption schemes with PCC are insecure in the INT-RUP security model. Then we fix the weakness of PCC, and describe a new approach called intermediate (parity) checksum (I(P)C for short). Based on the I(P)C approach, we provide two modified schemes OCB-IC and OCB-IPC to settle the INT-RUP of OCB in the nonce-misuse setting. OCB-IC and OCB-IPC are proven INT-RUP up to the birthday bound in the nonce-misuse setting if the underlying tweakable blockcipher is a secure mixed tweakable pseudorandom permutation (MTPRP). The security bound of OCB-IPC is tighter than OCB-IC. To improve their speed, we utilize a "prove-then-prune" approach: prove security and instantiate with a scaled-down primitive (e.g., reducing rounds for the underlying primitive invocations).

Keywords: OCB, INT-RUP, nonce-misuse, checksum, MTPRP, prove-then-prune

1 Introduction

Background. Authenticated encryption (AE) is a cryptographic scheme, which provides privacy and authenticity concurrently. In classical security models, a conventional AE scheme consists of an encryption algorithm and a decryption algorithm. The decryption algorithm includes two phases: plaintext computing and integrity verification. The plaintext corresponding to the ciphertext is released only if the tag is successfully verified. However, in certain settings, it is desirable to release unverified plaintext before verification. This case occurs when lightweight environments and low-end devices, such as smart cards, have no enough memory to store the entire plaintext, or when the decrypted plaintext needs to real-time processing in some special settings. What's more, releasing

unverified plaintext improves the efficiency of certain applications. For example, the decryption algorithm of Encrypt-then-MAC composition [5] has two passes: the first pass to verify the MAC, and the second pass to obtain the plaintext. If this AE scheme is secure against the release of unverified plaintext, then a single pass would be sufficient. Moreover, even if the attacker cannot observe the unverified plaintext directly, it could find some certain properties of the plaintext through side channel attacks. This occurs, for example, in the padding oracle attacks introduced by Vaudenay [35], where an error message or the lack of an acknowledgment indicates whether the unverified plaintext was correctly padded. Canvel et al. [7] showed how to mount a padding oracle attack on a version of OpenSSL by exploiting timing differences in the decryption processing of TLS. As shown by Paterson and AlFardan [2, 29] for TLS and DTLS, it is very difficult to prevent an attacker from learning the cause of decryption failures.

The issue of releasing unverified plaintext has lead to in-depth discussions in the CAESAR competition. For several AE schemes, such as IACBC [20], IAPM [20], OCB1 [33], OCB2 [31], OCB3 [21], TAE [23, 24], COPA [3], AEGIS [36], and ALE [6], their designers warned that unverified plaintext cannot be released. Note that releasing unverified plaintext does not imply omitting verification. Verification is essential to prevent incorrect plaintexts from being accepted. In this paper, we consider the security in this scenario where the attacker can observe the unverified plaintext, or any information relating to it, before verification is complete. Andreeva et al. addressed the issue of releasing unverified plaintext, and formalized it as the RUP setting. In their paper [4], they provided two new notions called PA (Plaintext Awareness) and INT-RUP (Integrity under Releasing Unverified Plaintext). In the RUP setting, an adversary can obtain the unverified plaintexts resulting from decryption queries. On the one hand, for privacy, they proposed using both IND-CPA and PA. At the heart of PA notion is the plaintext extractor. We say that an encryption scheme is PA if there exists an efficient plaintext extractor for every adversary. The plaintext extractor is a stateful algorithm with the goal of mimicking the decryption oracle in order to fool the adversary. It cannot make encryption nor decryption queries, and does not know the secret key. An authenticated encryption scheme achieves PA if it is infeasible to distinguish the decryption oracle from the plaintext extractor. They defined two notations of plaintext awareness: PA1 and PA2. On the other hand, an AE scheme is INT-RUP if an adversary can not generate a fresh valid ciphertext-tag pair given the additional power of access to a unverified decryption oracle, after the encryption oracle. INT-RUP is stronger than INT-CTXT. INT-RUP clearly implies INT-CTXT, but the opposite is not necessarily true.

Problem Statement. Andreeva et al. showed that nonce-respecting AE schemes (OCB [33, 31, 21], GCM [14], CCM [15], etc) and parts of nonce-misuse AE schemes (COPA [3], McOE-G [16], etc) are not PA1. They proposed two techniques — nonce-decoy and PRF-to-IV — to restore PA1 for nonce-respecting and nonce-misuse AE schemes, and discussed the INT-RUP in these two cases. The nonce-respecting AE schemes require that all nonces used in the encryption queries are distinct, while the nonce-misuse AE schemes do not. The privacy and

integrity of OCB [33, 31, 21] are insecure in the nonce-misuse and RUP settings. For OCB [33, 31, 21], Andreeva et al. also showed how to construct a forgery in the INT-RUP security model. The tag of OCB is generated by encrypting plaintext checksum, which results in an attack in the RUP setting. As the plaintext and ciphertext blocks can be obtained by the adversary in the RUP setting, the adversary can forge the same checksum by changing some plaintext or ciphertext blocks. In their paper [4], they left fixing OCB [33, 31, 21] to be INT-RUP in an efficient way as an open problem.

Our Contributions. This paper mainly considers the INT-RUP security of OCB [33, 31, 21] in the nonce-misuse setting. We focus on the weakness of the checksum processing in OCB [33, 31, 21]. We first set up a concrete INT-RUP security model, which allows an adversary to make any queries. Our INT-RUP security model is a stronger notion, which allows an adversary to make any disorder queries, such as query the encryption oracle before/after the decryption/verification oracle or interleaved query between the encryption oracle and the decryption oracle. While, the previous INT-RUP security model only allows encryption-decryption-verification order queries. The checksum of OCB [33, 31, 21] is generated by XOR-sum of the plaintext blocks, which is vulnerable in the INT-RUP security model. Then we describe a new notion, called plaintext or ciphertext checksum (PCC), which is a generalization of plaintext checksum. It is very easy for an adversary to forge the same checksum by changing some plaintext or ciphertext blocks. Therefore, all authenticated encryption schemes, if their tag is generated by encrypting the XOR-sum of the plaintext or ciphertext blocks, are insecure in the INT-RUP security model. The detail INT-RUP attack is presented in Supporting Material A.

To fix the weakness of PCC, we provide a new approach called intermediate (parity) checksum (I(P)C) to generate the checksum. In the I(P)C approach, the internal states in the encryption algorithm are hidden from adversaries, and intermediate checksum obtained by the XOR-sum of internal states is again encrypted once or many times before being output, which guarantees no information leakage, except the collision before the last block encryptions for authentication. Based on the I(P)C approach, we propose two modified schemes called OCB-IC and OCB-IPC to settle the INT-RUP security of OCB [33, 31, 21] in the nonce-misuse setting. They inherit the advantages of OCB [33, 31, 21]. We prove that OCB-IC and OCB-IPC are INT-RUP in the nonce-misuse setting if the underlying tweakable blockcipher (TBC) is a secure mixed tweakable pseudorandom permutation (MTPRP). They are proven INT-RUP up to the birthday bound of $n/2$-bit security, where n is the block-size of the underlying TBC. In Supporting Materials B and C, we utilize a blockcipher-based TBC to instantiate OCB-IC and OCB-IPC, respectively, and illustrate that their INT-RUP bounds are up to the birthday bound too. In this paper, we do not settle the problem of privacy in the RUP setting. OCB-IC and OCB-IPC are neither PA1 security nor PA2 security.

Compared with OCB [33, 31, 21], the number of invoking the underlying primitive in OCB-I(P)C is about twice of it. In other words, the efficiency of

OCB-I(P)C is about half of OCB. OCB-I(P)C compromises the efficiency of the software and hardware implementations to achieve INT-RUP security, and the security bound of OCB-IPC is tighter than OCB-IC. OCB-IC and OCB-IPC are TBC-based authenticated encryption schemes. If this TBC can be instantiated by an AES and the XEX* construction, its cost is larger than the cost of AES-CTR. Therefore, we utilize a "prove-then-prune" approach [18] — prove security and instantiate with a scaled-down primitive — to improve their speed. We can utilize a round-reduced AES instead of full-round AES to instantiate it, e.g., using AES6 for the block cipher invocations.

Chakraborti et al. [9] considered "rate-1" blockcipher-based affine AE mode, showed an INT-RUP attack on this mode, and presented a mCPFB (rate 3/4) which achieves INT-RUP security. In their paper, they left it as an interesting open problem to find a property that makes "rate-1/2" AE schemes INT-RUP secure. Our works present a concrete evidence about their works and find a new approach I(P)C. The I(P)C approach will provide a new direction for settling the security of "rate < 1" block cipher-based AE schemes in the RUP setting. We believe that one can further extend it for any "rate < 1".

Organization of This Paper. Notations and some preliminaries are presented in Section 2. In Section 3, we describe an INT-RUP security model. In Section 4, we provide a new approach called intermediate checksum, describe two modified schemes called OCB-IC and OCB-IPC, and derive their security proofs. We utilize "prove-then-prune" approach to instantiate with a scaled-down primitive in Section 5. Finally, this paper ends up with a conclusion in Section 6.

2 Preliminaries

Notations. Let ϵ denote the empty string, and $\{0,1\}^*$ denote the set containing all finite bit strings (including ϵ). Let n be an integer, and $(\{0,1\}^n)^+$ be the set of all strings whose lengths are positive multiples of n bits. For a finite string x, $|x|$ stands for its length. For two finite strings x and y, let $x\|y$ or xy denote the concatenation of them. If X is a set, then $x \xleftarrow{\$} X$ is a value randomly chosen from X, and $|X|$ stands for the number of elements in X. Let $\varnothing$ be the empty set whose cardinality is 0. Let $\lceil\cdot\rceil$ be the operation that rounds up to an integer. Denote $Pr[\mathbf{A}|\mathbf{B}]$ as the conditional probability of event $\mathbf{A}$ given event $\mathbf{B}$.

Finite Field. Given a basis, the finite field $GF(2^n)$ can be viewed as the set $\{0,1\}^n$. For an n-bit string $a = a_{n-1}\cdots a_1 a_0 \in \{0,1\}^n$, we can define a polynomial $a(x) \in \mathbb{Z}[x]$ by $a(x) = a_{n-1}x^{n-1} + \cdots + a_1 x + a_0$ over the field $GF(2)$. Hence, any integer between 0 and $2^n - 1$ can also be viewed as a polynomial with binary coefficients of degree at most $n-1$. For example, 2 corresponds to x, 3 corresponds to $x+1$, and 7 corresponds to x^2+x+1. The addition in the field $GF(2^n)$ is the addition of polynomials over $GF(2)$. We denote this operation by bitwise XOR, such as $a \oplus b$, where $a, b \in GF(2^n)$. To define multiplication in the field $GF(2^n)$, we need an irreducible polynomial $f(x)$ of degree n over $GF(2)$. The multiplication of two elements $A(x), B(x) \in GF(2^n)$ is defined as the polynomial

multiplication over $GF(2)$ reduced modulo $f(x)$, that is $A(x)B(x) \bmod f(x)$. We use point doubling (multiply $a \in \{0,1\}^n$ by 2) and XOR operations to compute the multiplication in actual operation. Such as $3a = 2a \oplus a$, $5a = 2(2a) \oplus a$, $7a = 2(2a) \oplus 2a \oplus a$, and so forth.

Block Ciphers and Tweakable Blockciphers. A block cipher $E : \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$ is a function that takes as input a key $K \in \mathcal{K}$ and a plaintext $P \in \{0,1\}^n$, and produces a ciphertext $C = E(K,P)$, where $\mathcal{K}$ is a finite nonempty set and $n \geq 1$ is a number. For any $K \in \mathcal{K}$, $E_K(\cdot) = E(K,\cdot)$ is a permutation over $\{0,1\}^n$. A tweakable blockcipher $\widetilde{E} : \mathcal{K} \times \mathcal{T} \times \{0,1\}^n \to \{0,1\}^n$ is a function that takes as input a key $K \in \mathcal{K}$, a tweak $T \in \mathcal{T}$, and a plaintext $P \in \{0,1\}^n$, and produces a ciphertext $C = E(K,T,P)$, where $\mathcal{K}$ and n are defined as above, and $\mathcal{T}$ is a finite nonempty set. For any $K \in \mathcal{K}$, $T \in \mathcal{T}$, $\widetilde{E}_K^T(\cdot) = \widetilde{E}(K,T,\cdot)$ is a permutation over $\{0,1\}^n$. Here n is called the blocksize, $\mathcal{K}$ is called the key space, $\mathcal{T}$ is called the tweak space.

Let $Perm(n)$ be the set of all permutations on n bits. Let $Perm(\mathcal{T},n)$ be the set of all mappings from $\mathcal{T}$ to permutations on n bits. Then $\pi \xleftarrow{\$} Perm(n)$ stands for the choice of a random permutation $\pi(\cdot)$ on $\{0,1\}^n$, and $\widetilde{\pi} \xleftarrow{\$} Perm(\mathcal{T},n)$ stands for the choice of a random permutation $\widetilde{\pi}(T,\cdot) = \widetilde{\pi}_T(\cdot)$ on $\{0,1\}^n$ for each $T \in \mathcal{T}$. For a block cipher $E : \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$, its inverse is $D_K = E_K^{-1}$ for any $K \in \mathcal{K}$, where $D : \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$ is defined by $D_K(Y)$ being the unique point X such that $E_K(X) = Y$. For a tweakable blockcipher $\widetilde{E} : \mathcal{K} \times \mathcal{T} \times \{0,1\}^n \to \{0,1\}^n$, its inverse is $\widetilde{D}_K = \widetilde{E}_K^{-1}$ for any $K \in \mathcal{K}$, where $\widetilde{D} : \mathcal{K} \times \mathcal{T} \times \{0,1\}^n \to \{0,1\}^n$ is defined by $\widetilde{D}_K^T(Y)$ being the unique point X such that $\widetilde{E}_K^T(X) = Y$.

An adversary is a probabilistic algorithm with access to certain oracles. Let $\mathcal{A}^O \Rightarrow 1$ be the event that an adversary $\mathcal{A}$ outputs 1 after interacting with the oracle O. Suppose that $E : \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$ is a block cipher, and $\widetilde{E} : \mathcal{K} \times \mathcal{T} \times \{0,1\}^n \to \{0,1\}^n$ is a tweakable blockcipher.

1) Let $\mathcal{A}$ be an adversary with access to an encryption oracle, $K \xleftarrow{\$} \mathcal{K}, \pi \xleftarrow{\$} Perm(n)$, and $\widetilde{\pi} \xleftarrow{\$} Perm(\mathcal{T},n)$, then the advantages of $\mathcal{A}$ against E and $\widetilde{E}$ are respectively defined as

$$Adv_E^{prp}(\mathcal{A}) = Pr[\mathcal{A}^{E_K(\cdot)} \Rightarrow 1] - Pr[\mathcal{A}^{\pi(\cdot)} \Rightarrow 1],$$
$$Adv_{\widetilde{E}}^{\widetilde{prp}}(\mathcal{A}) = Pr[\mathcal{A}^{\widetilde{E}_K(\cdot,\cdot)} \Rightarrow 1] - Pr[\mathcal{A}^{\widetilde{\pi}(\cdot,\cdot)} \Rightarrow 1],$$

where the probabilities are taken over the random coins used by the oracles and also over internal coins of $\mathcal{A}$, if any. If the advantage $Adv_E^{prp}(\mathcal{A})$ is negligible, the underlying block cipher E_K is a secure pseudorandom permutation (PRP). If the advantage $Adv_{\widetilde{E}}^{\widetilde{prp}}(\mathcal{A})$ is negligible, the underlying tweakable blockcipher $\widetilde{E}_K$ is a secure tweakable pseudorandom permutation (TPRP).

2) Let $\mathcal{A}$ be an adversary with access to both encryption and decryption oracles, $K \xleftarrow{\$} \mathcal{K}, \pi \xleftarrow{\$} Perm(n)$, and $\widetilde{\pi} \xleftarrow{\$} Perm(\mathcal{T},n)$, then the advantages of

$\mathcal{A}$ against E and $\widetilde{E}$ are respectively defined as

$$Adv_{E}^{sprp}(\mathcal{A}) = Pr[\mathcal{A}^{E_K(\cdot),E_K^{-1}(\cdot)} \Rightarrow 1] - Pr[\mathcal{A}^{\pi(\cdot),\pi^{-1}(\cdot)} \Rightarrow 1],$$
$$Adv_{\widetilde{E}}^{\widetilde{sprp}}(\mathcal{A}) = Pr[\mathcal{A}^{\widetilde{E}_K(\cdot,\cdot),\widetilde{E}_K^{-1}(\cdot,\cdot)} \Rightarrow 1] - Pr[\mathcal{A}^{\widetilde{\pi}(\cdot,\cdot),\widetilde{\pi}^{-1}(\cdot,\cdot)} \Rightarrow 1],$$

where the probabilities are taken over the random coins used by the oracles and also over internal coins of $\mathcal{A}$, if any. If the advantage $Adv_{E}^{sprp}(\mathcal{A})$ is negligible, the underlying block cipher E_K is a secure strong pseudorandom permutation (SPRP). If the advantage $Adv_{\widetilde{E}}^{\widetilde{sprp}}(\mathcal{A})$ is negligible, the underlying tweakable blockcipher $\widetilde{E}_K$ is a secure strong tweakable pseudorandom permutation (STPRP).

If the resources used by adversaries are at most R, we define the maximum advantage as

$$Adv(R) = max_{\mathcal{A}} Adv(\mathcal{A}),$$

where the resources of interest to us include the running time t, the total of oracle queries q, the maximum block-length l, and the totally number of blocks in all queries (queries complexity) σ.

Constructions of Blockcipher-based TBC. One of the most important methods in blockcipher-based TBC is the XE and XEX constructions. The most prominent scheme is OCB2 [31], which uses a very simple masking technique: powering up. Every associated data or message block is transformed using a different tweak whose incrementation can be achieved efficiently by powering up technique. This technique had been used in second-round CAESAR candidates AEZ [18], COPA [3], ELmD [13], OTR [26], and POET [1]. Other techniques to masking include Gray code ordering (used in OCB1 [33], OCB3 [21], and OMD [10]), LFSR [8], cellular automata map [8], the word-oriented LFSR [8], and universal hashing [23, 24]. By the XE and XEX constructions, we can translate a block cipher $E : \mathcal{K} \times \{0,1\}^n \rightarrow \{0,1\}^n$ into a tweakable blockcipher $\widetilde{E} : \mathcal{K} \times \mathcal{T} \times \{0,1\}^n \rightarrow \{0,1\}^n$, where $\mathcal{T} = \{0,1\}^n \times \mathcal{I} \times \mathcal{J}$ is a tweak space, $\mathcal{I}$ is a set of tuples of larger integers, and $\mathcal{J}$ is a set of tuples of small integers. Let $N \in \{0,1\}^n$, i be an integer from a larger set $\mathcal{I}$, and j be an integer from a small set $\mathcal{J}$. When we say that a tweak $T = (N, i, j)$ is an increment of another tweak, we mean that one of i, j got incremented and another stayed the same. We require tweaks to increase monotonically, and make the "special" operation from the penultimate block to the final block. The methods of linear separation and interleaved separation are provided by Chakraborty and Sarkar in [8]. Linear separation is based on Rogaway's powering up technique [31], which requires the computation of a discrete logarithm, however, interleaved separation does not. They analyzed the efficiency of linear separation and interleaved separation. The efficiency of interleaved separation is slightly lower than linear separation. The blockcipher-based TBC constructed by the XEX* (combining XE and XEX) construction is described as follows.

Given a block cipher $E : \mathcal{K} \times \{0,1\}^n \rightarrow \{0,1\}^n$ and a secret mask Δ, let $\mathcal{T} = \{0,1\}^n \times \mathcal{I} \times \mathcal{J}$ be a tweak space, $\mathcal{I}$ be a set of tuples of lager integers,

and $\mathcal{J}$ be a set of tuples of small integers, we obtain a tweakable blockcipher $\widetilde{E} : \mathcal{K} \times \mathcal{T} \times \{0,1\}^n \to \{0,1\}^n$ by the XEX* construction:

$$\widetilde{E}_K^{N,i,j}(x) = E_K(x \oplus \Delta) \text{ and } \widetilde{E}_K^{N,i',j'}(x) = E_K(x \oplus \Delta') \oplus \Delta'$$

where $(N,i,j) \in \mathcal{T}_0$, $(N,i',j') \in \mathcal{T}_1$, $\mathcal{T}_0 \cap \mathcal{T}_1 = \varnothing$, $\mathcal{T}_0 \cup \mathcal{T}_1 = \mathcal{T}$, and $\Delta = 2^i 3^j L, \Delta' = 2^{i'} 3^{j'} L, L = E_K(N)$.

Let $\mathcal{A}$ be an adversary which makes an encryption query $\widetilde{E}_K$ for tweaks from $\mathcal{T}_0$ and makes encryption and decryption queries $\widetilde{E}_K^{\pm 1}$ for tweaks from $\mathcal{T}_1$. Let $K \xleftarrow{\$} \mathcal{K}$, $\widetilde{\pi} \xleftarrow{\$} Perm(\mathcal{T}_0, n)$, and $\widetilde{\pi}^{\pm 1} \xleftarrow{\$} Perm(\mathcal{T}_1, n)$. Then the advantage of $\mathcal{A}$ against $\widetilde{E} = XEX^*[E, 2^{\mathcal{I}} 3^{\mathcal{J}}]$ is defined as

$$Adv_{\widetilde{E}}^{\widetilde{mprp}}(\mathcal{A}) = Pr[\mathcal{A}^{\widetilde{E}_K, \widetilde{E}_K^{\pm 1}(\cdot,\cdot)} \Rightarrow 1] - Pr[\mathcal{A}^{\widetilde{\pi}, \widetilde{\pi}^{\pm 1}(\cdot,\cdot)} \Rightarrow 1],$$

where the probabilities are taken over the random coins used by the oracles and also over internal coins of $\mathcal{A}$, if any. If the advantage $Adv_{\widetilde{E}}^{\widetilde{mprp}}(\mathcal{A})$ is negligible, the underlying tweakable blockcipher $\widetilde{E}_K$ is a secure mixed tweakable pseudorandom permutation (MTPRP). Denote $Adv_{\widetilde{E}}^{\widetilde{mprp}}(q)$ as the maximum advantage over all adversaries that make at most q queries. The definition of MTPRP matches TPRP if $(\mathcal{T}_0, \mathcal{T}_1) = (\mathcal{T}, \varnothing)$ and STPRP if $(\mathcal{T}_0, \mathcal{T}_1) = (\varnothing, \mathcal{T})$.

Lemma 1 (XEX*, [31]). *Fix a block cipher $E : \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$ and a tweakable blockcipher $\widetilde{E} : \mathcal{K} \times \mathcal{T} \times \{0,1\}^n \to \{0,1\}^n$, where $\mathcal{T} = \{0,1\}^n \times \mathcal{I} \times \mathcal{J}$ is a tweak space, $\mathcal{I}$ is a set of tuples of lager integers, and $\mathcal{J}$ is a set of tuples of small integers. Assume $2^i 3^j \neq 1$ for all $(i,j) \in \mathcal{I} \times \mathcal{J}$. Let $\widetilde{E} = XEX^*[E, 2^{\mathcal{I}} 3^{\mathcal{J}}]$, one has*

$$Adv_{\widetilde{E}}^{\widetilde{mprp}}(t,q) \leq Adv_E^{sprp}(t', 2q) + 9.5q^2/2^n,$$

where $t' = t + 2cn(q+1)$ for some absolute constant c.

Authenticated Encryption (AE). A conventional authenticated encryption with associated data scheme Π consists of an encryption algorithm and a decryption algorithm [32]. In order to consider the security of Π in the RUP setting, we must separate the decryption algorithm from the verification algorithm so that the decryption algorithm always releases plaintext [4]. A separated AE scheme is a triplet $\Pi = (\mathcal{E}, \mathcal{D}, \mathcal{V})$ — an encryption algorithm $\mathcal{E} : \mathcal{K} \times \mathcal{N} \times \mathcal{H} \times \mathcal{P} \to \mathcal{C} \times \mathcal{T}$, a decryption algorithm $\mathcal{D}$: $\mathcal{K} \times \mathcal{N} \times \mathcal{H} \times \mathcal{C} \times \mathcal{T} \to \mathcal{P}$, and a verification algorithm $\mathcal{V} : \mathcal{K} \times \mathcal{N} \times \mathcal{H} \times \mathcal{C} \times \mathcal{T} \to \top/\bot$, where we write

$$\begin{aligned} &(C,T) \leftarrow \mathcal{E}_K(N,A,P), \\ &P \leftarrow \mathcal{D}_K(N,A,C,T), \\ &\top/\bot \leftarrow \mathcal{V}_K(N,A,C,T), \end{aligned}$$

where $K \in \mathcal{K}$ is a key, $\mathcal{K} = \{0,1\}^k$, $k \geq 1$, $N \in \mathcal{N}$ is a nonce, $\mathcal{N} \subseteq \{0,1\}^n$, $n \geq 1$, $A \in \mathcal{H}$ is an associate data, $\mathcal{H} \subseteq \{0,1\}^*$, $P \in \mathcal{P}$ is a plaintext, $\mathcal{P} \subseteq \{0,1\}^*$,

$C \in \mathcal{C}$ is a ciphertext, $\mathcal{C} \subseteq \{0,1\}^*$, and $T \in \mathcal{T}$ is a tag, $\mathcal{T} \subseteq \{0,1\}^*$. The symbols $\top$ and $\bot$ indicate the success and failure of the verification oracle, respectively. $\mathcal{E}_K(N,A,P) = (C,T)$ if and only if (iff) $\mathcal{D}_K(N,A,C,T) = P$ and $\mathcal{V}_K(N,A,C,T) = \top$. A secure AE scheme returns $\bot$ if it receives an error (C,T) pair. If there is no associated data, A can be omitted. Without loss of generality, we assume that the adversary doesn't make redundant queries, that is, i) it doesn't repeat prior queries for each oracle, ii) the adversary does not ask the decryption oracle $\mathcal{D}_K(Y)$ or the verification oracle $\mathcal{V}_K(Y)$ after receiving Y in response to an encryption query $\mathcal{E}_K(X)$, and iii) the adversary does not ask the encryption oracle $\mathcal{E}_K(X)$ after receiving X in response to a decryption query $\mathcal{D}_K(Y)$.

3 INT-RUP Security Model

In this section, we set up a concrete INT-RUP security model, which allows an adversary to make any queries. For an AE scheme $\Pi = (\mathcal{E}_K, \mathcal{D}_K, \mathcal{V}_K)$, we assume that $\mathcal{A}$ is an adversary which makes at most q_e queries to the encryption oracle $\mathcal{E}_K(\cdot)$, at most q_d queries to the decryption oracle $\mathcal{D}_K(\cdot)$, and at most q_v queries to the verification oracle $\mathcal{V}_K(\cdot)$ (See Fig. 1). $\mathcal{A}$ can perform any queries, such as query the encryption oracle $\mathcal{E}_K(\cdot)$ before the decryption oracle $\mathcal{D}_K(\cdot)$, or query the decryption oracle $\mathcal{D}_K(\cdot)$ before the encryption oracle $\mathcal{E}_K(\cdot)$, or make the interleaved queries to the encryption oracle $\mathcal{E}_K(\cdot)$ and the decryption oracle $\mathcal{D}_K(\cdot)$. $\mathcal{A}$ queries (N^i, A^i, P^i) and receives $(C^i, T^i) = \mathcal{E}_K(N^i, A^i, P^i), 1 \leq i \leq q_e$. $\mathcal{A}$ has access to the decryption oracle $\mathcal{D}_K$ and obtains unverified plaintext $P^{*j} = \mathcal{D}_K(N^{*j}, A^{*j}, C^{*j}, T^{*j}), 1 \leq j \leq q_d$. Without loss of generality, we assume the adversary doesn't make redundant oracle. Note that $(N^{*j}, A^{*j}, C^{*j}, T^{*j}) \neq (N^i, A^i, C^i, T^i), 1 \leq i \leq q_e, 1 \leq j \leq q_d$.

Finally, $\mathcal{A}$ forges a challenge ciphertext $(N', A', C', T') \neq (N^i, A^i, C^i, T^i), 1 \leq i \leq q_e$ to the verification oracle $\mathcal{V}_K(\cdot)$.

The forgery is success if $\mathcal{V}_K(N', A', C', T') = \top$, failure otherwise. The INT-advantage of $\mathcal{A}$ against $\Pi = (\mathcal{E}_K, D_K, \mathcal{V}_K)$ is defined as

$$\begin{aligned} Adv_{\Pi}^{int-rup}(\mathcal{A}) &= Pr[\mathcal{A}^{\mathcal{E}_K, \mathcal{D}_K, \mathcal{V}_K} \Rightarrow 1] - Pr[\mathcal{A}^{\mathcal{E}_K, \mathcal{D}_K, \bot} \Rightarrow 1] \\ &= Pr[\mathcal{A}^{\mathcal{E}_K, \mathcal{D}_K, \mathcal{V}_K} \ forges]. \end{aligned}$$

Let $Adv_{\Pi}^{int-rup}(t,q,l,\sigma) = max_{\mathcal{A}} Adv_{\Pi}^{int-rup}(\mathcal{A})$ be the maximum integrity advantage for all adversaries $\mathcal{A}$ whose time complexity is at most t, the number of queries is at most q, each of block-length is at most l, and queries complexity is at most σ.

4 TBC-based Authenticated Encryption Modes with Intermediate Checksum

OCB [33, 31, 21] is insecure in the INT-RUP security model. The tag of OCB is generated by encrypting plaintext checksum, which prompts an attack in the

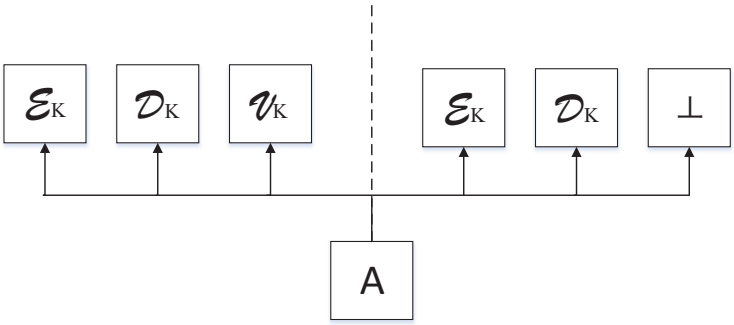


Fig. 1. INT-RUP security model. $\mathcal{A}$ is an adversary which makes any queries in the INT-RUP security model. **Left of dashed line**: Real world, with encryption oracle $\mathcal{E}_K(\cdot)$, decryption oracle $\mathcal{D}_K(\cdot)$, and verification oracle $\mathcal{V}_K(\cdot)$. **Right of dashed line**: Ideal world, with encryption oracle $\mathcal{E}_K(\cdot)$, decryption oracle $\mathcal{D}_K(\cdot)$, and failure oracle $\perp$ that always outputs $\perp$ for all queries. The goal of $\mathcal{A}$ is to distinguish real world from idea world. If the distinguishable advantage of $\mathcal{A}$ is negligible, the scheme is INT-RUP.

RUP setting. Andreeva et al. [4] presented how to violate the INT-RUP security by using the unverified plaintext to construct forgeries for OCB [33, 31, 21]. We describe a new notation PCC, which is a generalization of plaintext checksum, and prove that all authenticated encryption schemes with PCC are insecure in the RUP setting. INT-RUP is a stronger security notation than INT-CTXT, which gives the adversary the ability to perform decryption queries and observe the unverified plaintexts. PCC is the XOR-sum of the plaintext or ciphertext blocks, which leads to easily forge the same checksum by changing parts of the plaintext or ciphertext blocks for an adversary. The strategy of the attack is come from [4]. Our attack is an improved version. If an adversary $\mathcal{A}$ makes one encryption query, $2 \leq p \leq 2^d$ decryption queries and one forgery attempt, each consisting of l blocks of n bits, then the adversary makes a successful forgery with high probability (at least $1 - 2^{n-ld}$) by solving a system of linear equations in $GF(2)$ with n equations and ld unknowns. The details of proof are presented in Supporting Material A.

In this section, we fix the weakness of PCC and provide a new approach, called I(P)C, to generate the checksum. The internal states in the encryption algorithm are hidden from adversaries and intermediate checksum obtained by the XOR-sum of internal states is again encrypted once or many times, which guarantees no information leakage, except the collision before the last block encryptions for authentication in the same nonce. The tag is generated by the PMAC1 algorithm [31] of either the plaintext or the ciphertext. Moreover, the decryption algorithm and the verification algorithm share parts of computing resources. Therefore, the I(P)C approach greatly improves the efficiency in the software and hardware implementation. Based on the I(P)C approach, we propose two modified schemes called OCB-IC and OCB-IPC to settle the INT-RUP security of OCB [33, 31, 21] in the nonce-misuse setting. OCB-IC and OCB-IPC

are TBC-based authenticated encryption schemes, whose structures and analyses are simpler than blockcipher-based authenticated encryption schemes. They inherit the advantages of OCB [33, 31, 21]. We prove that OCB-IC and OCB-IPC are INT-RUP in the nonce-misuse setting if the underlying tweakable blockcipher is a secure MTPRP. Their INT-RUP is proven up to the birthday bound and the bound of OCB-IPC is tighter than OCB-IC.

4.1 OCB with Intermediate Checksum: OCB-IC

OCB-IC makes two invocations to the underlying TBC per the plaintext block. Moreover, the algorithm of generating the tag shares parts of computing resources with encryption/decryption algorithms and the cost of the generating tag is minimal (only invoking the underlying tweakable blockcipher twice). OCB-IC[$\widetilde{E}$] is parameterized by a tweakable blockcipher $\widetilde{E} : \mathcal{K} \times \mathcal{T} \times \{0,1\}^n \to \{0,1\}^n$, where $\mathcal{T} = \{0,1\}^n \times \mathcal{I} \times \mathcal{J}$ is a tweak space, $\mathcal{I}$ is a set of tuples of lager integers, and $\mathcal{J}$ is a set of tuples of small integers. We require tweaks to increase monotonically, and perform the "special" operation from the penultimate block to the final block, such as linear separation (powering up technique) and interleaved separation [8, 33, 31], which makes tweaks' update highly efficient. We assume that the plaintext length is a positive multiple of blocksize n. The length of associated data is arbitrary.

The overview of OCB-IC[$\widetilde{E}$] is depicted in Fig. 2. OCB-IC[$\widetilde{E}$] is made up of three algorithms — an encryption algorithm $\mathcal{E}_K$, a decryption algorithm $\mathcal{D}_K$, and a verification algorithm $\mathcal{V}_K$. The detailed description of OCB-IC[$\widetilde{E}$] is shown in Fig. 3.

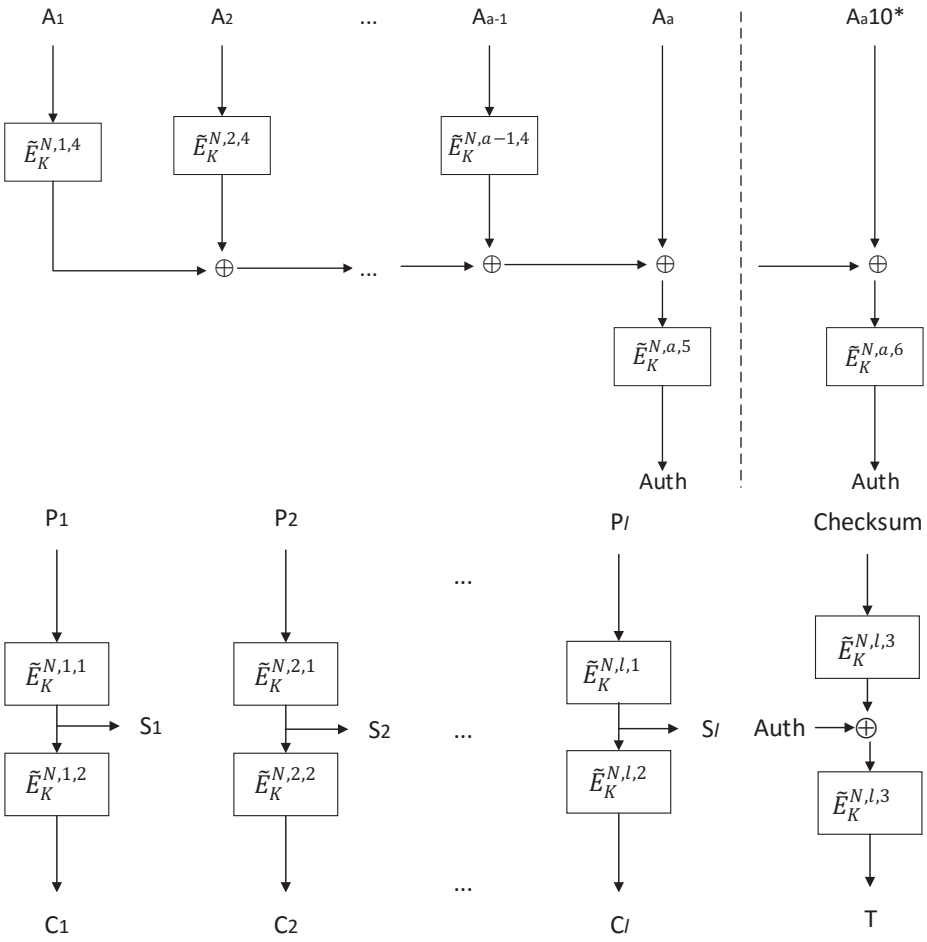
We analyze and obtain the following theorems for the information theoretic security of OCB-IC[$\widetilde{E}$]. If the underlying tweakable blockcipher $\widetilde{E}$ is a secure MTPRP, OCB-IC[$\widetilde{E}$] is INT-RUP in the nonce-misuse setting.

Theorem 1 (INT-RUP of OCB-IC with an Ideal TBC). *For OCB-IC[$\widetilde{E}$], we replace tweakable blockciphers $\widetilde{E}_K$ with tweakable random permutations $\widetilde{\pi} \xleftarrow{\$} Perm(\mathcal{T}, n)$ to obtain OCB-IC[$\widetilde{\pi}$], where $\mathcal{T} = \{0,1\}^n \times \mathcal{I} \times \mathcal{J}$ is a tweak space, $\mathcal{I}$ is a set of tuples of lager integers, and $\mathcal{J}$ is a set of tuples of small integers. Let $\mathcal{A}$ be a nonce-misusing adversary. Let q_v be the number of forgery queries. Then we have*

$$Adv^{int-rup}_{OCB-IC[\widetilde{\pi}]}(\mathcal{A}) \leq 1.5q^2/2^n + q_v q/2^n.$$

Proof. We assume that $\mathcal{A}$ is an adversary with access to the encryption oracle $\mathcal{E}(\cdot)$, the decryption oracle $\mathcal{D}(\cdot)$, and the verification oracle $\mathcal{V}(\cdot)$. The adversary $\mathcal{A}$ makes encryption queries (N^i, A^i, P^i) and receives $(C^i, T^i) = \mathcal{E}(N^i, A^i, P^i), 1 \leq i \leq q$. The adversary $\mathcal{A}$ has access to the decryption oracle $\mathcal{D}(\cdot)$ and obtains the unverified plaintext $P^{*j} = \mathcal{D}(N^{*j}, A^{*j}, C^{*j}, T^{*j}), 1 \leq j \leq q_d$. Note that $(N^{*j}, A^{*j}, C^{*j}, T^{*j}) \neq (N^i, A^i, C^i, T^i), 1 \leq i \leq q, 1 \leq j \leq q_d$.

Finally, $\mathcal{A}$ forges a challenge ciphertext $(N', A', C', T') \neq (N^i, A^i, C^i, T^i), 1 \leq i \leq q$ to the verification oracle $\mathcal{V}(\cdot)$, where $C' = C'_1 C'_2 \cdots C'_{l'}, C^i = C^i_1 C^i_2 \cdots C^i_{l^i}$.



$$\text{Checksum} = \sum_{i=1}^{l} S_i \,, \text{Auth} = PMAC1(A)$$

Fig. 2. Illustrating OCB-IC$[\widetilde{E}]$ with a tweakable blockcipher $\widetilde{E} : \mathcal{K} \times \mathcal{T} \times \{0,1\}^n \to \{0,1\}^n$, where $\mathcal{T} = \{0,1\}^n \times \mathcal{I} \times \mathcal{J}$ is a tweak space, $\mathcal{I}$ is a set of tuples of lager integers, and $\mathcal{J}$ is a set of tuples of small integers, e.g., $\mathcal{I} = \{0,1,2,\cdots,2^n - 1\}, \mathcal{J} = \{0,1,2,\cdots,10\}$. **Top row**: the authentication of associated data A: $Auth = PMAC1(A)$. If the length of associated data $|A|$ is not a positive multiple of n bits, padding 10^* to A such that $|A10^*|$ is a positive multiple of n bits. The authentication of associated data is achieved by a PMAC1 algorithm [31]. If there is no associated data, then we set $Auth = 0$. **Bottom row**: the encryption and authentication of the plaintext P. The length of the plaintext P is a positive multiple of n bits. The plaintext P is encrypted twice to produce the ciphertext C and the value of intermediate states is used to generate the checksum. If $Auth$ is simply xored with the encrypted Checksum to obtain the tag, we can get the difference of $Auth$ which can be easily used to obtain forging attack. Therefore the tag is generated by applying one more encryption for Checksum and $Auth$.

/*AD Processing*/
Algorithm $Auth(A)$ $//PMAC1_K^N(A)$
Partition A into $A_1\|\cdots\|A_a$
$|A_i| = n, 1 \le i \le a-1, 0 < |A_l| \le n$
for $i = 1$ to $a-1$
    $S_i \leftarrow \widetilde{E}_K^{N,i,4}(A_i)$
if $|A_a| = n$
    $\Sigma \leftarrow S_1 \oplus S_2 \oplus \cdots \oplus S_{a-1} \oplus A_a$
    $Auth = \widetilde{E}_K^{N,a,5}(\Sigma)$
else
    $\Sigma \leftarrow S_1 \oplus S_2 \oplus \cdots \oplus S_{a-1} \oplus A_a 10^*$
    $Auth = \widetilde{E}_K^{N,a,6}(\Sigma)$
return Auth

/*Encryption Algorithm*/
Algorithm $OCB-IC.\mathcal{E}_K^N(A,P)$:
Partition P into $P_1\|\cdots\|P_l$,
$|P_i| = n, 1 \le i \le l$
for $i = 1$ to l
    $S_i \leftarrow \widetilde{E}_K^{N,i,1}(P_i)$
    $C_i \leftarrow \widetilde{E}_K^{N,i,2}(S_i)$
$C \leftarrow C_1 C_2 \cdots C_l$
$\Sigma \leftarrow S_1 \oplus S_2 \oplus \cdots \oplus S_l$
$T = \widetilde{E}_K^{N,l,3}(\widetilde{E}_K^{N,l,3}(\Sigma) \oplus Auth(A))$
return $C||T$

/*Decryption Algorithm*/
Algorithm $OCB-IC.\mathcal{D}_K^N(A,C||T)$:
Partition C into $C_1\|\cdots\|C_l$,
$|C_i| = n, 1 \le i \le l$
for $i = 1$ to l
    $S_i \leftarrow (\widetilde{E}_K^{N,i,2})^{-1}(C_i)$
    $P_i \leftarrow (\widetilde{E}_K^{N,i,1})^{-1}(S_i)$
$P \leftarrow P_1 P_2 \cdots P_l$
return P

/*Verification Algorithm*/
Algorithm $OCB-IC.\mathcal{V}_K^N(A,C||T)$:
Partition C into $C_1\|\cdots\|C_l$,
$|C_i| = n, 1 \le i \le l$
for $i = 1$ to l
    $S_i \leftarrow (\widetilde{E}_K^{N,i,2})^{-1}(C_i)$
$\Sigma \leftarrow S_1 \oplus S_2 \oplus \cdots \oplus S_l$
$T' = \widetilde{E}_K^{N,l,3}(\widetilde{E}_K^{N,l,3}(\Sigma) \oplus Auth(A))$
if $T' = T$, return $\top$,
else return $\perp$

Fig. 3. OCB-IC$[\widetilde{E}]$ with a tweakable blockcipher $\widetilde{E} : \mathcal{K} \times \mathcal{T} \times \{0,1\}^n \to \{0,1\}^n$, where $\mathcal{T} = \{0,1\}^n \times \mathcal{I} \times \mathcal{J}$ is a tweak space, $\mathcal{I}$ is a set of tuples of lager integers, and $\mathcal{J}$ is a set of tuples of small integers, e.g., $\mathcal{I} = \{0,1,2,\cdots,2^n-1\}, \mathcal{J} = \{0,1,2,\cdots,10\}$. The encryption algorithm $\mathcal{E}_K$ includes the encryption of the plaintext blocks, the authentications of associated data and the plaintext. The authentications of associated data and the plaintext are achieved by PMAC1 algorithm. If there is no associated data, then we set $Auth = 0$. The decryption algorithm $\mathcal{D}_K$ is straightforward similar to the encryption algorithm except no authentication of the tag at the end of the decryption process. The verification algorithm $\mathcal{V}_K$ outputs $\top$ if the new tag generated by the associated data-ciphertext pair is equal to the original tag, $\perp$ otherwise.

Let $\Gamma = \{(N^i, A^i, P^i, C^i, T^i)\}_{i=1}^{q} \cup \{(N^{*j}, A^{*j}, P^{*j}, C^{*j}, T^{*j})\}_{j=1}^{q_d}$ be the transcript (input-output pairs of OCB-IC) obtained by the encryption queries and decryption queries. Γ can be seen as a random variable, then the INT-advantage of $\mathcal{A}$ is

$$\begin{aligned} Adv^{int-rup}_{OCB-IC[\widetilde{\pi}]}(\mathcal{A}) &= Pr[\mathcal{A}^{\mathcal{E}_K, \mathcal{D}_K, \mathcal{V}_K} \Rightarrow 1] - Pr[\mathcal{A}^{\mathcal{E}_K, \mathcal{D}_K, \perp} \Rightarrow 1] \\ &= Pr[\mathcal{A}^{\mathcal{E}_K, \mathcal{D}_K, \mathcal{V}_K} \ forges] \\ &\leq max_{\gamma \in \Gamma} Pr_{\mathcal{A}}[(N', A', C', T') \ is \ valid | \Gamma = \gamma]. \end{aligned}$$

For associated data A, we handle it by PMAC1 algorithm. Let $\mathbf{A}$ be the event that a collision of $Auth$ occurs for two different associated data. Let $\mathbf{T}$ be the event that a collision of the tag occurs for two different plaintexts in the encryption oracle. Denote an event $\mathbf{E}$ as a union of events $\mathbf{A}$ and $\mathbf{T}$, and $\mathbf{E} = \mathbf{A} \vee \mathbf{T}$. Let $\mathbf{F}$ be the event that the verification oracle $V(\cdot)$ returns $\top$ other than $\perp$. Then $Pr_{\mathcal{A}}[(N', A', C', T') \ is \ valid | \Gamma = \gamma] = Pr[\mathcal{A}^{OCB-IC[\widetilde{\pi}]} \ sets \ \mathbf{F} | \Gamma = \gamma]$ ($Pr[\mathbf{F}]$ for short). By the total probability formula, we can obtain

$$\begin{aligned} Pr[\mathbf{F}] &= Pr[\mathbf{F}|\neg\mathbf{E}]Pr[\neg\mathbf{E}] + Pr[\mathbf{F}|\mathbf{E}]Pr[\mathbf{E}] \\ &\leq Pr[\mathbf{F}|\neg\mathbf{E}] + Pr[\mathbf{E}] \\ &\leq Pr[\mathbf{F}|\neg\mathbf{E}] + Pr[\mathbf{A}] + Pr[\mathbf{T}] \end{aligned}$$

Claim (1). $Pr[\mathbf{A}] \leq q^2/2^n$, and $Pr[\mathbf{T}] \leq q^2/2^{n+1}$.

Proof. The authentications of associated data and the plaintext are achieved by PMAC1 algorithm. The probability of the event $\mathbf{A}$ is just equal to a collision probability of a random function $PMAC1[\widetilde{\pi}]$ plus the probability of $PMAC1[\widetilde{\pi}]$ hitting 0, which is at most $q(q-1)/2^{n+1} + q/2^n \leq q^2/2^n$. The probability of the event $\mathbf{T}$ is just equal to a collision probability of a random function $PMAC1[\widetilde{\pi}]$, which is at most $q(q-1)/2^{n+1} \leq q^2/2^{n+1}$.

Claim (2). $Pr[\mathbf{F}|\neg\mathbf{E}] \leq q_v q/2^n$.

Proof. To derive the probability that $\mathcal{A}^{OCB-IC[\widetilde{\pi}]}$ sets $\mathbf{F}$ under the condition $\neg\mathbf{E}$: $Pr[\mathbf{F}|\neg\mathbf{E}]$, we analyze some cases as follows.

Case 1: T' is new. In this case, $\mathcal{A}$ already knows all the tags after the encryption oracle and with this knowledge it is trying to guess the image of another point. The probability of guessing this correctly is at most $1/(2^n - q)$, which is its success probability of $\mathcal{A}$.

Case 2: T' is old. As all nonces can be repeated in all queries, we divide the set of nonce $\mathcal{N}$ used in the encryption oracle into two sets $\mathcal{N}_1$ and $\mathcal{N}_2$. The set $\mathcal{N}_1$ only contains an element N_0. i.e. $\mathcal{N} = \mathcal{N}_1 \bigcup \mathcal{N}_2$, $\mathcal{N}_1 = \{N_0\}$, $\mathcal{N}_2 = \mathcal{N} \backslash \mathcal{N}_1$. Similarity, we divide the set of block-length $\mathcal{L}$ used in the encryption oracle into two sets $\mathcal{L}_1$ and $\mathcal{L}_2$. The set $\mathcal{L}_1$ only contains an element l_0. i.e. $\mathcal{L} = \mathcal{L}_1 \bigcup \mathcal{L}_2$, $\mathcal{L}_1 = \{l_0\}$, $\mathcal{L}_2 = \mathcal{L} \backslash \mathcal{L}_1$. We divide the set of associated data $\mathcal{H}$ used in the encryption oracle into two sets $\mathcal{H}_1$ and $\mathcal{H}_2$. The set $\mathcal{H}_1$ only contains an element A_0. i.e. $\mathcal{H} = \mathcal{H}_1 \bigcup \mathcal{H}_2$, $\mathcal{H}_1 = \{A_0\}$, $\mathcal{H}_2 = \mathcal{H} \backslash \mathcal{H}_1$. We do a further case analysis as follows.

Case 2-1: If $N' \notin \mathcal{N}$, the finalization tweak $(N', l', 3)$ is new. The adversary tries to forge using a new nonce. The image of a single point under a random permutation is uniform, so the generated tag is an independent and uniform random value. Thus, the probability that the adversary can guess the correct value is $1/2^n$.

Case 2-2: If $N' \in \mathcal{N}_1$, $l' \notin \mathcal{L}$, the finalization tweak $(N', l', 3)$ is new. The adversary tries to forge using a new block-size. The image of a single point under a random permutation is uniform, so the generated tag is an independent and uniform random value. Thus, the probability that the adversary can guess the correct value is $1/2^n$.

Case 2-3: If $N' \in \mathcal{N}_1$, $l' \in \mathcal{L}_1$, the finalization tweak $(N', l', 3)$ in this case is the same with the forgery attempt.

1. If $A' \notin \mathcal{H}$, then it means that it yields a fresh random value by $PMAC1(A')$. The last tag generated from this value is a fresh random value. Therefore, the probability that the adversary can guess the correct value is $1/2^n$.
2. If $A' \in \mathcal{H}_1$, the authentication of associated data A' is the same with A_0 from many nonce-associated data-ciphertext-tag pairs in the encryption oracle, where only the ciphertexts are distinct. The rest is similar to the PMAC1 processing of the ciphertext blocks. The probability of successful forgery is equal to a collision probability of a random function PMAC1 plus the probability of PMAC1 hitting T^i, where $i = 1, 2, \cdots, q$, which is at most $0 + q/2^n \leq q/2^n$.

Summarizing all cases, we have

$$Pr[\mathbf{F}|\neg\mathbf{E}] \leq max\{1/(2^n - q), 1/2^n, q/2^n\} \leq q/2^n$$

for a single forgery query, where $q \geq 2$.

If the adversary $\mathcal{A}$ makes q_v forgery queries, then it is easy to obtain the probability $Pr[\mathbf{F}|\neg\mathbf{E}] \leq q_v \cdot q/2^n$.

By Claims (1) and (2), the INT-advantage of $\mathcal{A}$, after $q \geq 2$ encryption queries, q_d decryption queries, and q_v forgery queries, is

$$Adv^{int-rup}_{OCB-IC[\widetilde{\pi}]}(\mathcal{A}) \leq 1.5q^2/2^n + q_v q/2^n.$$

Theorem 2 (INT-RUP of OCB-IC with a Tweakable Blockcipher). *Let $\widetilde{E} : \mathcal{K} \times \mathcal{T} \times \{0,1\}^n \to \{0,1\}^n$ be a tweakable blockcipher. Fix $n \geq 1$, $\mathcal{T} = \{0,1\}^n \times \mathcal{I} \times \mathcal{J}$ is a tweak space, $\mathcal{I}$ is a set of tuples of lager integers, and $\mathcal{J}$ is a set of tuples of small integers, let $\mathcal{A}$ be a nonce-misusing adversary. Then we have*

$$Adv^{int-rup}_{OCB-IC[\widetilde{E}]}(t, \sigma) \leq Adv^{\widetilde{mprp}}_{\widetilde{E}}(t', 2\sigma) + 1.5q^2/2^n + q_v q/2^n,$$

where $t' = t + cn\sigma$ for some absolute constant c.

The INT-RUP of OCB-IC is proven up to the birthday bound in the nonce-misuse setting if the underlying tweakable blockcipher is a secure MTPRP. OCB-IC just settles the problems of integrity in the RUP and nonce-misuse settings,

while the problems of privacy in the RUP and nonce-misuse settings still exist. OCB-IC is neither PA1 security nor PA2 security. OCB-IC can be seen as a special instantiation of the generic B1 scheme of [28] in which one applies a PRF to the message as well as encrypting it with a secure nonce-based encryption scheme. It is a secure "rate-$\frac{1}{2}$" parallelizable authenticated encryption scheme. We utilize a blockcipher-based TBC to instantiate OCB-IC in Supporting Material B.

4.2 OCB with Intermediate Parity-Checksum: OCB-IPC

If we use a parity-checksum to replace the above checksum, we can obtain a new scheme OCB-IPC. Compared with OCB-IC, OCB-IPC has a tighter bound. The overview of OCB-IPC[$\widetilde{E}$] is depicted in Fig. 4. OCB-IPC[$\widetilde{E}$] is made up of three algorithms — an encryption algorithm $\mathcal{E}_K$, a decryption algorithm $\mathcal{D}_K$, and a verification algorithm $\mathcal{V}_K$. The detailed description of OCB-IPC[$\widetilde{E}$] is shown in Fig. 5.

We obtain the following theorems for the information-theoretic security of OCB-IPC[$\widetilde{E}$]. If the underlying tweakable blockcipher $\widetilde{E}$ is a secure MTPRP, OCB-IPC[$\widetilde{E}$] is INT-RUP in the nonce-misuse setting.

Theorem 3 (INT-RUP of OCB-IPC with an Ideal TBC). *For OCB-IPC with a tweakable blockcipher $\widetilde{E}_K$, we replace tweakable blockciphers with tweakable random permutations $\widetilde{\pi} \xleftarrow{\$} Perm(\mathcal{T}, n)$ to obtain OCB-IPC[$\widetilde{\pi}$], where $\mathcal{T} = \{0,1\}^n \times \mathcal{I} \times \mathcal{J}$ is a tweak space, $\mathcal{I}$ is a set of tuples of lager integers, and $\mathcal{J}$ is a set of tuples of small integers. Let $\mathcal{A}$ be a nonce-misusing adversary. Then we have*

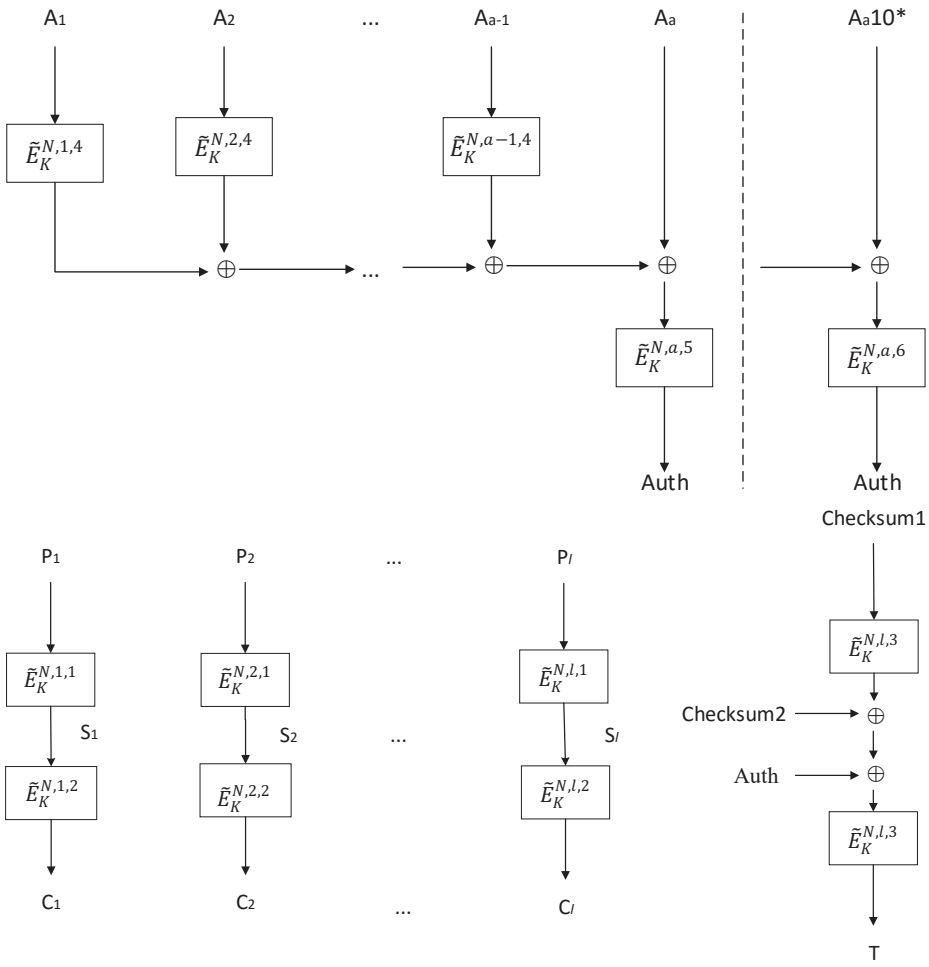
$$Adv^{int-rup}_{OCB-IPC[\widetilde{\pi}]}(\mathcal{A}) \leq \frac{q^2}{2^n} + \frac{(q-1)^2}{2^{n+1}} + \frac{q_v q}{2^{n+1}}.$$

Proof. We assume that $\mathcal{A}$ is a nonce-misusing adversary with access to an encryption oracle $\mathcal{E}(\cdot)$, a decryption oracle $\mathcal{D}(\cdot)$, and a verification oracle $\mathcal{V}(\cdot)$. The adversary $\mathcal{A}$ makes encryption queries (N^i, A^i, P^i) and receives $(C^i, T^i) = \mathcal{E}(N^i, A^i, P^i), 1 \leq i \leq q$. The adversary $\mathcal{A}$ has access to the decryption oracle $\mathcal{D}(\cdot)$ and obtains the unverified plaintext $P^{*j} = \mathcal{D}(N^{*j}, A^{*j}, C^{*j}, T^{*j}), 1 \leq j \leq q_d$. Note that $(N^{*j}, A^{*j}, C^{*j}, T^{*j}) \neq (N^i, A^i, C^i, T^i), 1 \leq i \leq q, 1 \leq j \leq q_d$.

Finally, $\mathcal{A}$ forges a challenge ciphertext $(N', A', C', T') \neq (N^i, A^i, C^i, T^i), 1 \leq i \leq q$ to the verification oracle $\mathcal{V}(\cdot)$, where $C' = C'_1 C'_2 \cdots C'_{l'}, C^i = C^i_1 C^i_2 \cdots C^i_{l^i}$.

Let $\Gamma = \{(N^i, A^i, P^i, C^i, T^i)\}_{i=1}^{q} \cup \{(N^{*j}, A^{*j}, P^{*j}, C^{*j}, T^{*j})\}_{j=1}^{q_d}$ be the transcript (input-output pairs of OCB-IPC) obtained by the encryption queries and decryption queries. Γ can be seen as a random variable, then the INT-advantage of $\mathcal{A}$ is

$$\begin{aligned} Adv^{int-rup}_{OCB-IPC[\widetilde{\pi}]}(\mathcal{A}) &= Pr[\mathcal{A}^{\mathcal{E}_K, \mathcal{D}_K, \mathcal{V}_K} \Rightarrow 1] - Pr[\mathcal{A}^{\mathcal{E}_K, \mathcal{D}_K, \perp} \Rightarrow 1] \\ &= Pr[\mathcal{A}^{\mathcal{E}_K, \mathcal{D}_K, \mathcal{V}_K} \ forges] \\ &\leq max_{\gamma \in \Gamma} Pr_{\mathcal{A}}[(N', A', C', T') \ is \ valid | \Gamma = \gamma]. \end{aligned}$$



$$\text{Checksum1} = \sum_i S_{2i}\,, \text{Checksum2} = \sum_i S_{2i-1}\,, \text{Auth} = PMAC1(A)$$

Fig. 4. OCB-IPC$[\widetilde{E}]$ with a tweakable blockcipher $\widetilde{E} : \mathcal{K} \times \mathcal{T} \times \{0,1\}^n \to \{0,1\}^n$, where $\mathcal{T} = \{0,1\}^n \times \mathcal{I} \times \mathcal{J}$ is a tweak space, $\mathcal{I}$ is a set of tuples of lager integers, and $\mathcal{J}$ is a set of tuples of small integers, e.g., $\mathcal{I} = \{0, 1, 2, \cdots, 2^n - 1\}$, and $\mathcal{J} = \{0, 1, \cdots, 10\}$. **Top row**: the authentication of associated data A: $Auth = PMAC1(A)$. If the length of associated data $|A|$ is not a positive multiple of n bits, padding 10^* to A so as to $|A10^*|$ is a positive multiple of n bits. The authentication of associated data is achieved by PMAC1 algorithm. If there is no associated data, then we set $Auth = 0$. **Bottom row**: the encryption and authentication of the plaintext P. The encryption procedure of the plaintext of OCB-IPC is the same with OCB-IC. The odd and even parts of intermediate states are used to produce separately Checksum1 and Checksum2, which is encrypted to generate the final message authentication code.

/*AD Processing*/
Algorithm $Auth(A)$ $//PMAC1_K^N(A)$
Partition A into $A_1\|\cdots\|A_a$
$|A_i| = n, 1 \le i \le a-1, 0 < |A_l| \le n$
for $i = 1$ to $a-1$
    $S_i \leftarrow \widetilde{E}_K^{N,i,4}(A_i)$
if $|A_a| = n$
    $\Sigma \leftarrow S_1 \oplus S_2 \oplus \cdots \oplus S_{a-1} \oplus A_a$
    $Auth = \widetilde{E}_K^{N,a,5}(\Sigma)$
else
    $\Sigma \leftarrow S_1 \oplus S_2 \oplus \cdots \oplus S_{a-1} \oplus A_a 10^*$
    $Auth = \widetilde{E}_K^{N,a,6}(\Sigma)$
return Auth

/*Encryption Algorithm*/
Algorithm $OCB - IPC.\mathcal{E}_K^N(A, P)$:
Partition P into $P_1\|\cdots\|P_l$,
$|P_i| = n, 1 \le i \le l$
for $i = 1$ to l
    $S_i \leftarrow \widetilde{E}_K^{N,i,1}(P_i)$
    $C_i \leftarrow \widetilde{E}_K^{N,i,2}(S_i)$
$C \leftarrow C_1 C_2 \cdots C_l$
$\Sigma_1 \leftarrow S_2 \oplus S_4 \oplus \cdots$
$\Sigma_2 \leftarrow S_1 \oplus S_3 \oplus \cdots$
$\Sigma = \widetilde{E}_K^{N,l,3}(\Sigma_1) \oplus \Sigma_2$
$T = \widetilde{E}_K^{N,l,3}(\Sigma)$
return $C||T$

/*Decryption Algorithm*/
Algorithm $OCB - IPC.\mathcal{D}_K^N(A, C||T)$:
Partition C into $C_1\|\cdots\|C_l$,
$|C_i| = n, 1 \le i \le l$
for $i = 1$ to l
    $S_i \leftarrow (\widetilde{E}_K^{N,i,2})^{-1}(C_i)$
    $P_i \leftarrow (\widetilde{E}_K^{N,i,1})^{-1}(S_i)$
$P \leftarrow P_1 P_2 \cdots P_l$
return P

/*Verification Algorithm*/
Algorithm $OCB - IPC.\mathcal{V}_K^N(A, C||T)$:
Partition C into $C_1\|\cdots\|C_l$,
$|C_i| = n, 1 \le i \le l$
for $i = 1$ to l
    $S_i \leftarrow (\widetilde{E}_K^{N,i,2})^{-1}(C_i)$
$\Sigma_1 \leftarrow S_2 \oplus S_4 \oplus \cdots$
$\Sigma_2 \leftarrow S_1 \oplus S_3 \oplus \cdots$
$\Sigma = \widetilde{E}_K^{N,l,3}(\Sigma_1) \oplus \Sigma_2$
$T' = \widetilde{E}_K^{N,l,3}(\Sigma)$
if $T' = T$, return $\top$
else return $\bot$

Fig. 5. OCB-IPC[$\widetilde{E}$] with a tweakable blockcipher $\widetilde{E} : \mathcal{K} \times \mathcal{T} \times \{0,1\}^n \to \{0,1\}^n$, where $\mathcal{T} = \{0,1\}^n \times \mathcal{I} \times \mathcal{J}$ is a tweak space, $\mathcal{I}$ is a set of tuples of lager integers, and $\mathcal{J}$ is a set of tuples of small integers, e.g., $\mathcal{I} = \{0, 1, 2, \cdots, 2^n - 1\}$, and $\mathcal{J} = \{0, 1, \cdots, 10\}$. The encryption algorithm $\mathcal{E}_K$ includes the encryption of the plaintext blocks, the authentications of associated data and the plaintext. If there is no associated data, then we set $Auth = 0$. The decryption algorithm $\mathcal{D}_K$ is straightforward similar to the encryption algorithm except no authentication of the tag at the end of the decryption process. The verification algorithm $\mathcal{V}_K$ outputs $\top$ if the new tag generated by the associated data-ciphertext pair is equal to the original tag, $\bot$ otherwise.

Let $\mathbf{F}$ be the event that the verification oracle $V(\cdot)$ returns $\top$ other than $\perp$. Then $Pr_{\mathcal{A}}[(N', A', C', T') \ is \ valid | \Gamma = \gamma] = Pr[\mathcal{A}^{OCB-IPC[\widetilde{\pi}]} \ sets \ \mathbf{F} | \Gamma = \gamma]$ ($Pr[\mathbf{F}]$ for short). Next, we need to bound the probability $Pr[\mathbf{F}]$. We define two events $\mathbf{A}, \mathbf{T}$, which are similar with OCB-IC. Let $\mathbf{E} = \mathbf{A} \vee \mathbf{T}$.

First, the probability $Pr[\mathbf{A}]$ is at most $q^2/2^n$ as shown in Claim (1). Then, we need to evaluate the probability $Pr[\mathbf{T}]$.

If l is an even, then

$$\sum_i S_{2i} = PMAC1(P_2 P_4 \cdots P_l),$$

and

$$\sum_i S_{2i-1} = PMAC1(P_1 P_3 \cdots P_{l-1}).$$

If l is an odd, then

$$\sum_i S_{2i} = PMAC1(P_2 P_4 \cdots P_{l-1}),$$

and

$$\sum_i S_{2i-1} = PMAC1(P_1 P_3 \cdots P_l).$$

Denote an event $\mathbf{T}_1$ as a collision of $\sum_i S_{2i}$ occurs for two different even plaintext blocks in the encryption oracle, an event $\mathbf{T}_2$ as a collision of $\sum_i S_{2i-1}$ occurs for two different odd plaintext blocks in the encryption oracle. Because $\sum_i S_{2i}$ is only related with the even parts of plaintexts and $\sum_i S_{2i-1}$ is only related with the odd parts of plaintexts, $\mathbf{T}_1$ and $\mathbf{T}_2$ are two independent events, and $Pr[\mathbf{T}_1] = Pr[\mathbf{T}_2] = 1/2^n$. The event $\mathbf{T}$ happens iff events $\mathbf{T}_1$ and $\mathbf{T}_2$ occur or don't occur at the same time.

By the formula of total probability, we can obtain

$$\begin{aligned}
Pr[\mathbf{T}] &= Pr[\mathbf{T} \wedge \mathbf{T}_1 \wedge \mathbf{T}_2] + Pr[\mathbf{T} \wedge \overline{\mathbf{T}_1} \vee \overline{\mathbf{T}_2}] \\
&= Pr[\mathbf{T} | \mathbf{T}_1 \wedge \mathbf{T}_2] Pr[\mathbf{T}_1 \wedge \mathbf{T}_2] + Pr[\mathbf{T} | \overline{\mathbf{T}_1} \vee \overline{\mathbf{T}_2}] Pr[\overline{\mathbf{T}_1} \vee \overline{\mathbf{T}_2}] \\
&= Pr[\mathbf{T} | \mathbf{T}_1 \wedge \mathbf{T}_2] Pr[\mathbf{T}_1 \wedge \mathbf{T}_2] + Pr[\mathbf{T} | \overline{\mathbf{T}_1} \wedge \overline{\mathbf{T}_2}] Pr[\overline{\mathbf{T}_1} \wedge \overline{\mathbf{T}_2}] \\
&\leq \sum_{1 \leq j < i \leq q} 1/2^n \times 1/2^n \times 1 + (1 - 1/2^n)^2 \times 1/2^n \\
&\leq q(q-1)/2^{n+1} - q(q-1)/2^{2n+1} + q(q-1)/2^{3n+1} \\
&\leq q(q-1)/2^{n+1} - q(q-1)/2^{2n+2} \\
&\leq (q-1)^2/2^{n+1}.
\end{aligned}$$

By the conditional probability formula, we can obtain

$$Pr[\mathbf{F}] \leq Pr[\mathbf{F} | \neg \mathbf{E}] + Pr[\mathbf{E}],$$

where $Pr[\mathbf{E}] \leq q^2/2^n + (q-1)^2/2^{n+1}$ is easily bounded in terms of OCB-IC. It follows that, we need to evaluate $Pr[\mathbf{F}|\neg\mathbf{E}']$. We analyze some cases as follows.

Case 1: T' is new. In this case, $\mathcal{A}$ already knows the value of T^i, where $1 \leq i \leq q$, and with this knowledge it is trying to guess the image of another point. The probability of guessing this correctly is at most $1/(2^n - q)$, which is its success probability of $\mathcal{A}$.

Case 2: T' is old. As all nonces can be repeated in the whole queries, we divide the set of nonce $\mathcal{N}$ used in the encryption oracle into two sets $\mathcal{N}_1$ and $\mathcal{N}_2$. The set $\mathcal{N}_1$ only contains an element N_0. i.e. $\mathcal{N} = \mathcal{N}_1 \bigcup \mathcal{N}_2$, $\mathcal{N}_1 = \{N_0\}$, $\mathcal{N}_2 = \mathcal{N} \backslash \mathcal{N}_1$. Similarity, we divide the set of block-length $\mathcal{L}$ used in the encryption oracle into two sets $\mathcal{L}_1$ and $\mathcal{L}_2$. The set $\mathcal{L}_1$ only contains an element l_0. i.e. $\mathcal{L} = \mathcal{L}_1 \bigcup \mathcal{L}_2$, $\mathcal{L}_1 = \{l_0\}$, $\mathcal{L}_2 = \mathcal{L} \backslash \mathcal{L}_1$. We divide the set of associated data $\mathcal{H}$ used in the encryption oracle into two sets $\mathcal{H}_1$ and $\mathcal{H}_2$. The set $\mathcal{H}_1$ only contains an element A_0. i.e. $\mathcal{H} = \mathcal{H}_1 \bigcup \mathcal{H}_2$, $\mathcal{H}_1 = \{A_0\}$, $\mathcal{H}_2 = \mathcal{H} \backslash \mathcal{H}_1$. We do a further case analysis as follows.

Case 2-1: If $N' \notin \mathcal{N}$, the finalization tweak $(N', l', 3)$ is new. The adversary tries to forge using a new nonce. The image of a single point under a random permutation is uniform, so the generated tag is an independent and uniform random value. Thus, the probability that the adversary can guess the correct value is $1/2^n$.

Case 2-2: If $N' \in \mathcal{N}_1$, $l' \notin \mathcal{L}$, the finalization tweak $(N', l', 3)$ is new. The adversary tries to forge using a new block-size. The image of a single point under a random permutation is uniform, so the generated tag is an independent and uniform random value. Thus, the probability that the adversary can guess the correct value is $1/2^n$.

Case 2-3: If $N' \in \mathcal{N}_1$, $l' \in \mathcal{L}_1$, the finalization tweak $(N', l', 3)$ in this case is the same with the forgery attempt.

1. If $A' \notin \mathcal{H}$, then it means that it yields a fresh random value by $PMAC1(A')$. The last tag generated from this value is a fresh random value. Therefore, the probability that the adversary can guess the correct value is $1/2^n$.
2. If $A' \in \mathcal{H}_1$, the authentication of associated data A' is the same with A_0 from many nonce-associated data-ciphertext-tag pairs in the encryption oracle. The rest is similar to the PMAC1 processing of the ciphertext blocks. In this case, we need to analyze the parity of l'. If l' is an even, then

$$\sum_i S_{2i} = PMAC1(C'_2 \cdots C'_{l'}),$$

and

$$\sum_i S_{2i-1} = PMAC1(C'_1 \cdots C'_{l'-1}).$$

If l' is an odd, then

$$\sum_i S_{2i} = PMAC1(C'_2 \cdots C'_{l'-1}),$$

and

$$\sum_i S_{2i-1} = PMAC1(C'_1 \cdots C'_{l'}).$$

Take even l' as example. Next we need to compute the probability that (N', A', C', T') is valid in this case. Let $\mathbf{T}$ be the event that a collision of the tag T occurs for the forgery and encryption oracles. Let $\mathbf{T}_1$ be the event that a collision of $\sum_i S_{2i}$ occurs for the forgery and encryption oracles, $\mathbf{T}_2$ be the event that a collision of $\sum_i S_{2i-1}$ occurs for the forgery and encryption oracles. $\mathbf{T}_1$ and $\mathbf{T}_2$ are independent events, and $Pr[\mathbf{T}_1] = Pr[\mathbf{T}_2] = 1/2^n$. By the formula of total probability, we can obtain

$$\begin{aligned} Pr[\mathbf{F}|\neg\mathbf{E}'] &= Pr[\mathbf{T}] \\ &= Pr[\mathbf{T} \wedge \mathbf{T}_1 \wedge \mathbf{T}_2] + Pr[\mathbf{T} \wedge (\overline{\mathbf{T}_1} \vee \overline{\mathbf{T}_2})] \\ &= Pr[\mathbf{T}|\mathbf{T}_1 \wedge \mathbf{T}_2]Pr[\mathbf{T}_1 \wedge \mathbf{T}_2] + Pr[\mathbf{T}|\overline{\mathbf{T}_1} \vee \overline{\mathbf{T}_2}]Pr[\overline{\mathbf{T}_1} \vee \overline{\mathbf{T}_2}] \\ &\leq q/2^n - q/2^{2n+1} \\ &\leq q/2^{n+1}. \end{aligned}$$

For odd l', we also have $Pr[\mathbf{F}|\neg\mathbf{E}'] \leq q/2^{n+1}$.

Summarizing all above cases, we have

$$Pr[\mathbf{F}|\neg\mathbf{E}'] \leq max\{1/(2^n - q), 1/2^n, q/2^{n+1}\} \leq q/2^{n+1}$$

for a single forgery query, where $q \geq 4$.

If the adversary $\mathcal{A}$ makes q_v forgery queries, it is easy to obtain the probability $Pr[\mathbf{F}|\neg\mathbf{E}'] \leq q_v q/2^{n+1}$.

The INT-advantage of $\mathcal{A}$, after $q \geq 4$ encryption queries, q_d decryption queries, and q_v forgery queries, is

$$Adv^{int-rup}_{OCB-IPC[\tilde{\pi}]}(\mathcal{A}) \leq \frac{q^2}{2^n} + \frac{(q-1)^2}{2^{n+1}} + \frac{q_v q}{2^{n+1}}.$$

Theorem 4 (INT-RUP of OCB-IPC with a Tweakable Blockcipher). *Let $\widetilde{E} : \mathcal{K} \times \mathcal{T} \times \{0,1\}^n \to \{0,1\}^n$ be a tweakable blockcipher, where $\mathcal{T} = \{0,1\}^n \times \mathcal{I} \times \mathcal{J}$ is a tweak space, $\mathcal{I}$ is a set of tuples of lager integers, and $\mathcal{J}$ is a set of tuples of small integers. Fix $n \geq 1$, let $\mathcal{A}$ be a nonce-misusing adversary. Then we have*

$$Adv^{int-rup}_{OCB-IPC[\widetilde{E}]}(t, \sigma) \leq Adv^{\widetilde{mprp}}_{\widetilde{E}}(t', 2\sigma) + \frac{1.5q^2}{2^n} + \frac{q_v q}{2^{n+1}},$$

where $t' = t + cn\sigma$ for some absolute constant c, and l is the maximum block-length.

Compared with OCB-IC, OCB-IPC has a tighter bound. The INT-RUP of OCB-IPC is proven up to the birthday bound in the nonce-misuse setting if the underlying tweakable blockcipher is a secure MTPRP. We utilize a blockcipher-based TBC to instantiate OCB-IPC in Supporting Material C.

4.3 The Processing for Arbitrary Length Messages

When the length of the message is not a positive multiple of the blocksize n, our schemes require to be extended. We extend our schemes to handle with the arbitrary length message M. We can utilize a CTR-like encryption for the last incomplete block, which retain the mainly structure of OCB [33, 31, 21]. The concrete description is showed in Figs. 6. For simplicity, here we utilize a padding function *pad* such that the length of the message is a positive multiple of the blocksize n. Given an arbitrary length message $M \in \{0,1\}^*$, it needs to be padded to the plaintext $P = pad(M) = M10^{n-1-(|M| \ mod \ n)}$ before the encryption algorithm. Meanwhile, after the decryption algorithm, we use a corresponding un-padding function *unpad*, which removes the 10^* in P, to obtain the original message $M = unpad(P)$.

5 Discussion and Future Work

There are four approaches to realize a tweakable blockcipher. The first approach is direct design method, such as TWEAKEY [19] framework. The second approach is based on block ciphers, such as OCB2 [31], OCB3 [21], AEZ [18], COPA [3], ELmD [13], OTR [26], and POET [1]. The third approach is based on permutations, such as TEM [11], OPP [17], MRO [17]. The fourth approach is based on keyed-functions (hash functions), such as OMD [10], p-OMD [30]. We present blockcipher-based instances with the birthday bound in Supporting Materials B and C. They are INT-RUP against up to around $2^{n/2}$ queries, where n is the blocksize. There exists many TBC constructions with beyond the birthday bound (BBB), such as [11, 22, 27]. For example, CLRW2 [22] presented by Landecker, Shrimpton, and Terashima is a BBB secure construction. It is secure up to around $2^{2n/3}$ queries, where n is the blocksize. It remains an open problem to design a TBC-based, BBB-secure AE scheme in the nonce-misuse and RUP settings.

For OCB-IC and OCB-IPC, the number of the underlying primitive invocations is about twice than that of OCB [33, 31, 21]. If we use full-round AES to instantiate the underlying tweakable blockcipher, their efficiency are about half of it. In other words, OCB-IC and OCB-IPC compromise the efficiency of the software and hardware implementations to achieve INT-RUP security. However they make two invocations to the underlying TBC per the plaintext block, which greatly reduces their efficiency. In this paper, we employ an approach — prove-then-prune — to improve the speed. As our schemes are proven security, we utilize a scaled-down primitive instead of the original primitive (e.g., using round-reduced AES instead of full-round AES for block cipher invocations) to instantiate our schemes. This approach firstly presented by Hoang [18] works like this:

"*To achieve some complex cryptographic goal, design a scheme in the provable security tradition, choosing an underlying primitive and demonstrably achieving the goal when its instantiated by an object achieving some standard assumption. Then, to improve speed, selectively instantiate some of the applications of the*

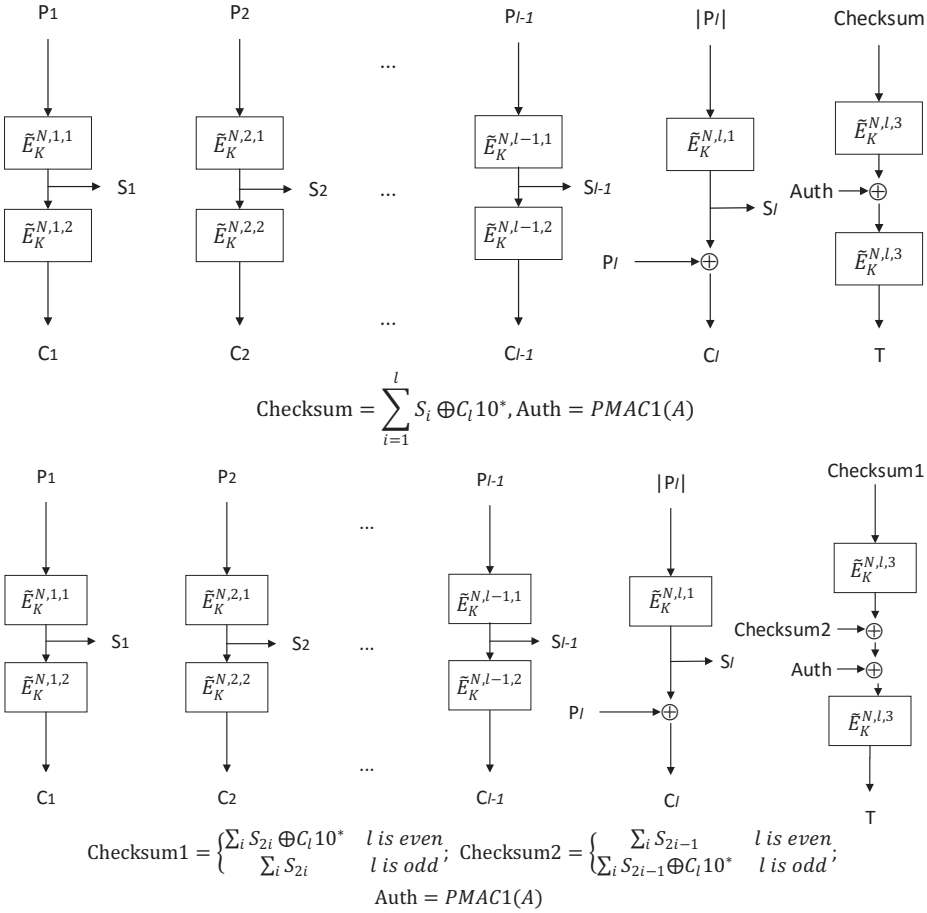


Fig. 6. The Constructions of OCB-I(P)C for Arbitrary Length Messages. We can utilize a CTR-like encryption for the last incomplete block, which makes that OCB-IC and OCB-IPC retain the mainly structure of OCB [33, 31, 21]. **Top row:** The construction of OCB-IC for arbitrary length messages. Auth is the authentication of associated data. If Auth is simply Xored with the encrypted Checksum to obtain the tag, we can get the difference of Auth which can be easily used to obtain forging attack. Therefore the tag is generated by applying one more encryption for Checksum and Auth. **Bottom row:** The construction of OCB-IPC for arbitrary length messages.

primitive using a scaled-down (e.g., reduced-round) construction. Use heuristic or cryptanalytic reasons to support the expectation that, despite scaling down, the scheme remains secure. The thesis underlying prove-then-prune approach is that it can be instrumental for devising highly efficient schemes for complex aims. If the instantiation is done judiciously, then the scaled-down scheme retains some assurance benefit. Still, it is important to emphasize the limitations of prove-then-prune. Naming an approach is not license to abuse it. The method is dangerous in the same sort of way that designing a confusion/diffusion primitive is: one has no guarantees for the object that will actually be used. Additionally, the set of people with provable-security competence is nearly disjoint from those with cryptanalytic competence."

Specifically, OCB-IC and OCB-IPC are designed in terms of a tweakable blockcipher. If this tweakable blockcipher can be instantiated by using AES and the XEX* construction, we can obtain an provable security instance, which is presented in Supporting Material B. The cost is larger than the cost of AES-CTR. To speed things up, we instantiate the underlying tweakable blockcipher invocations with a round-reduced AES construction, such as AES6. Heuristics reasons to suggest that security nonetheless remains. The goal of our design is to make an instance with a scaled-down primitive secure. The method of prove-then-prune is useful for designing highly efficient schemes with complex aims. In some way, prove-then-prune implies in prior work: schemes like ALRED [12] typify a trend that reduced-round AES instead of full-round AES. We leave it as another interesting open problem to deeper analyses for OCB-IC and OCB-IPC with the scaled-down primitive.

6 Conclusion

OCB [33, 31, 21] is insecure in the nonce-misuse and RUP settings. This paper mainly considers the INT-RUP security of OCB [33, 31, 21] in the nonce-misuse setting. We focus on the weakness of the checksum processing in OCB [33, 31, 21]. We first set up a concrete INT-RUP security model, which allows an adversary to make any queries. The tag of OCB [33, 31, 21] is generated by encrypting plaintext checksum, which is vulnerable against integrity security in the RUP setting. We describe a new notion PCC, which is a generalization of plaintext checksum, and prove that all authenticated encryption schemes with PCC are insecure in the INT-RUP security model. PCC is the XOR-sum of the plaintext or ciphertext blocks, which leads to easily forge the same checksum by changing some plaintext or ciphertext blocks for an adversary.

To fix the weakness of PCC, we provide a new approach I(P)C to generate the checksum. In the I(P)C approach, the internal states in the encryption algorithm are hidden from adversaries, and intermediate checksum obtained by the XOR-sum of internal states is again encrypted once or many times before being output, which guarantees no information leakage, except the collision before the last block encryptions for authentication. Based on the I(P)C approach, we propose two modified schemes called OCB-IC and OCB-IPC to settle the INT-RUP

security of OCB [33, 31, 21] in the nonce-misuse setting. OCB-IC and OCB-IPC are TBC-based authenticated encryption schemes. They retain the mainly structure of OCB [33, 31, 21] and inherit its advantages. We prove that OCB-IC and OCB-IPC are INT-RUP in the nonce-misuse setting if the underlying tweakable blockcipher is a secure MTPRP. Their INT-RUP is proven up to the birthday bound and OCB-IPC has a tighter bound. In this paper, we do not settle the problem of privacy in the RUP setting. OCB-IC and OCB-IPC are neither PA1 security nor PA2 security. We leave it as an open problem to settle the privacy of OCB-IC and OCB-IPC in an efficient way.

References

1. Abed, F., Fluhrer, S., Forler, C., List, E., Lucks, S., McGrew, D., Wenzel, J.: Pipelineable on-line encryption. In: Cid, C., Rechberger, C. (eds.) FSE 2014. LNCS, vol. 8540, pp. 205–223. Springer, Heidelberg (2015)
2. AlFardan, N.J., Paterson, K.G.: Lucky Thirteen: Breaking the TLS and DTLS Record Protocols. In: IEEE Symposium on Security and Privacy, pp. 526–540. IEEE Computer Society (2013)
3. Andreeva, E., Bogdanov, A., Luykx, A., Mennink, B., Tischhauser, E., Yasuda, K.: Parallelizable and authenticated online ciphers. In: Sako, K., Sarkar, P. (eds.) ASIACRYPT 2013. LNCS, vol. 8269, pp. 424–443. Springer, Heidelberg (2013)
4. Andreeva, E., Bogdanov, A., Luykx, A., Mennink, B., Mouha, N., Yasuda, K.: How to securely release unverified plaintext in authenticated encryption. In: Sarkar, P., Iwata, T. (eds.) ASIACRYPT 2014. LNCS, vol. 8873, pp. 105–125. Springer, Heidelberg (2014)
5. 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)
6. Bogdanov, A., Mendel, F., Regazzoni, F., Rijmen, V., Tischhauser, E.: ALE: AES-based lightweight authenticated encryption. In: Moriai, S. (ed.) FSE 2013. LNCS, vol. 8424, pp. 447–466. Springer, Heidelberg (2014)
7. Canvel, B., Hiltgen, A.P., Vaudenay, S., Vuagnoux, M.: Password Interception in a SSL/TLS Channel. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 583–599. Springer, Heidelberg (2003)
8. Chakraborty, D., Sarkar, P.: A general construction of tweakable block ciphers and different modes of operations. IEEE Transactions on Information Theory. 2008: 54 (5), pp. 1991–2006 (2008)
9. Chakraborti, A., Datta, N., Nandi, M.: INT-RUP analysis of block-cipher based authenticated encryption schemes. In: Sako, K. (ed.) CT-RSA 2016. LNCS, vol. 9610, pp. 39–54. Springer, Heidelberg (2016)
10. Cogliani, S., Maimuţ, D.-Ş., Naccache, D.: OMD: a compression function mode of operation for authenticated encryption. In: Joux, A., Youssef, A. (eds.) SAC 2014. LNCS, vol. 8781, pp. 112–128. Springer, Heidelberg (2014)
11. Cogliati, B., Lampe, R., Seurin, Y.: Tweaking Even-Mansour ciphers. In: Gennaro, R., Robshaw, M.J.B. (eds.) CRYPTO 2015. LNCS, vol. 9215, pp. 189–208. Springer, Heidelberg (2015)
12. Daemen, J., Rijmen, V.: A new mac construction ALRED and a specific instance ALPHA-MAC. In: Gilbert, H., Handschuh, H. (eds.) FSE 2005. LNCS, vol. 3557, pp. 1–17. Springer, Heidelberg (2005)

13. Datta, N., Nandi, M.: ELmD v2.0. Submission to the CAESAR submissions, 2015, https://competitions.cr.yp.to/round2/elmdv20.pdf
14. Dworkin, M. J.: Recommendation for block cipher modes of operation: Galois/Counter mode (GCM) and GMAC. NIST SP 800-38D (2007)
15. Dworkin, M. J.: Recommendation for block cipher modes of operation: The CCM mode for authentication and confidentiality. NIST SP 800-38C (2004)
16. Fleischmann, E., Forler, C., Lucks, S.: McOE: a family of almost fool proof on-line authenticated encryption schemes. In: Canteaut, A. (ed.) FSE 2012. LNCS, vol. 7549, pp. 196–215. Springer, Heidelberg (2012)
17. Granger, R., Jovanovic, P., Mennink, B., Neves, S..: Improved masking for tweakable blockciphers with applications to authenticated encryption. In: Fischlin, M., Coron, J. S. (eds.) EUROCRYPT 2016. LNCS, vol. 9665, pp. 263–293. Springer, Heidelberg (2016)
18. Hoang, V., Krovetz, T., Rogaway, P.: Robust authenticated-encryption AEZ and the problem that it solves. In: Oswald, E., Fischlin, M. (eds.) EUROCRYPT 2015. LNCS, vol. 9056, pp. 15–44. Springer, Heidelberg (2015)
19. Jean, J., Nikolić, I., Peyrin, I.: Tweaks and keys for block ciphers: the tweakey framework. In: Sarkar, P., Iwata, T. (eds.) ASIACRYPT 2014. LNCS, vol. 8874, pp. 274–288. Springer, Heidelberg (2014)
20. Jutla, C.: Encryption mode with almost free message integrity. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 529–544. Springer, Heidelberg (2001)
21. Krovetz, T., Rogaway, P.: The software performance of authenticated-encryption modes. In: Joux, A. (ed.) FSE 2011. LNCS, vol. 6733, pp. 306–327. Springer, Heidelberg (2011)
22. Landecker, W., Shrimpton, T., Terashima, R.S.: Tweakable blockciphers with beyond birthday-bound security. In: Safavi-Naini, R., Canetti, R. (eds.) CRYPTO 2012. LNCS, vol. 7417, pp. 14–30. Springer, Heidelberg (2012)
23. Liskov, M., Rivest, R., Wagner, D.: Tweakable block ciphers. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 31–46. Springer, Heidelberg (2002)
24. Liskov, M., Rivest, R., Wagner, D.: Tweakable block ciphers. Journal of Cryptology. 2011: 24(3), pp. 588–613 (2011)
25. Mennink, B., Reyhanitabar, R., Vizár, D.: Security of full-state keyed Sponge and Duplex: Applications to authenticated encryption. In: Cheon, J.H., Iwata, T. (eds.) ASIACRYPT 2015. LNCS, vol. 9453, pp. 465–489. Springer, Heidelberg (2015)
26. Minematsu, K.: Parallelizable rate-1 authenticated encryption from pseudorandom functions. In: Nguyen, P. Q., Oswald, E. (eds.) EUROCRYPT 2014. LCNS, vol. 8441, pp. 275–292. Springer, Heidelberg (2014)
27. Minematsu, K.: Beyond-birthday-bound security based on tweakable block cipher. In: Dunkelman, O. (ed.) FSE 2009. LNCS, vol. 5665, pp. 308–326. Springer, Heidelberg (2009)
28. Namprempre, C., Rogaway, P., Shrimpton, T.: Reconsidering generic composition. In: Nguyen, P.Q., Oswald, E. (eds.) EUROCRYPT 2014. LNCS, vol. 8441, pp. 257–274. Springer, Heidelberg (2014)
29. Paterson, K. G., AlFardan, N.J.: Plaintext-Recovery Attacks Against Datagram TLS. In: NDSS 2012. The Internet Society (2012)
30. Reyhanitabar, R., Vaudenay, S., Vizr, D.: Boosting OMD for Almost Free Authentication of Associated Data. In: Leander, G. (ed.) FSE 2015. LNCS, vol. 9054, pp. 411–427. Springer, Heidelberg (2015)

31. Rogaway, P.: Efficient instantiations of tweakable blockciphers and refinements to modes OCB and PMAC. In: Lee, P.J. (ed.) ASIACRYPT 2004. LNCS, vol. 3329, pp. 16–31. Springer, Heidelberg (2004)
32. Rogaway, P.: Authenticated-encryption with associated-data. In: Atluri, V. (ed.) ACM-CCS 2002. pp. 98–107. ACM (2002)
33. Rogaway, P., Bellare, M., Black, J., Krovetz, T.: OCB: A block-cipher mode of operation for efficient authenticated encryption. In: Reiter, M. K., Samarati, P. (eds.) ACM-CCS 2001. pp. 196–205. ACM (2001)
34. Sui, H., Wu, W., Zhang, L., Wang, P.: Attacking and fixing the CS mode. In: Qing, S., et al. (eds.) ICICS 2013. LNCS, vol. 8233, pp. 318–330. Springer, Heidelberg (2013)
35. Vaudenay, S.: Security Flaws Induced by CBC Padding – Applications to SSL, IPSEC, WTLS. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 534–546. Springer, Heidelberg (2002)
36. Wu, H., Preneel, B.: AEGIS: a fast authenticated encryption algorithm. In: Lange, T., Lauter, K., Lisoněk, P. (eds.) SAC 2013. LNCS, vol. 8282, pp. 185–201. Springer, Heidelberg (2014)
37. Wu, S., Wu, H., Huang, T., Wang, M., Wu, W.: Leaked-state-forgery attack against the authenticated encryption algorithm ALE. In: Sako, K., Sarkar, P. (eds.) ASIACRYPT 2013. LNCS, vol. 8269, pp. 377–404. Springer, Heidelberg (2013)

Supporting Material

A: INT-RUP Analysis of AE Modes with PCC

For all authenticated encryption schemes, if their checksum is generated by the XOR-sum of the plaintext blocks, they are insecure in the RUP setting. Such as, IAPM [20], IACBC [20], OCB1[33], OCB2 [31], OCB3 [21], TAE [23, 24], COPA [3], OPP [17], and so on.

We describe a new notion PCC, which is a generalization of plaintext checksum, and prove that all authenticated encryption schemes with PCC are not secure in the INT-RUP security model. PCC is the XOR-sum of the plaintext or ciphertext blocks, including the XOR-sum of the whole plaintext or ciphertext blocks, the XOR-sum of the whole plaintext and ciphertext blocks, and the XOR-sum of the parts of plaintext and ciphertext blocks. INT-RUP is a stronger security notation than INT-CTXT, which gives the adversary the ability to make decryption queries and observe the unverified plaintexts. The adversary can forge the same checksum by changing some plaintext or ciphertext blocks. Therefore the tag generated by plaintext or ciphertext checksum is vulnerable against integrity security in the RUP setting. The strategy of the attack is come from [4]. We present an improved version in Theorem 5.

Let $\Pi = (\mathcal{E}_K, \mathcal{D}_K, \mathcal{V}_K)$ be an authenticated encryption mode. Let $(f_i^{N,A}, h_i^{N,A}, g^{N,A})$ be independent random functions, where $1 \le i \le l$, then

$$
\begin{aligned}
&(N, A, C_1C_2\cdots C_l, T) \leftarrow \mathcal{E}_K(N, A, P_1P_2\cdots P_l)\\
&P = P_1P_2\cdots P_l \leftarrow \mathcal{D}_K(N, A, C_1C_2\cdots C_l, T),\\
&\top/\bot \leftarrow \mathcal{V}_K(N, A, C_1C_2\cdots C_l, T),
\end{aligned}
$$

where

$$
\begin{aligned}
&C_1C_2\cdots C_i \leftarrow f_i^{N,A}(P_1P_2\cdots P_i), 1 \le i \le l,\\
&P_1P_2\cdots P_i \leftarrow h_i^{N,A}(C_1C_2\cdots C_i), 1 \le i \le l,\\
&T \leftarrow g^{N,A}(\sum_{i=1}^{l}(P_iz_i^1 \oplus C_iz_i^2)), z_i^1, z_i^2 \in \{0,1\}.
\end{aligned}
$$

Remark 1. There are three cases for PCC as follows.

1) If $z_i^1 = 0, z_i^2 = 1$ for all $1 \le i \le l$, then the tag is generated from the XOR-sum of the whole ciphertext blocks;

2) if $z_i^1 = 1, z_i^2 = 0$ for all $1 \le i \le l$, then the tag is generated from the XOR-sum of the whole plaintext blocks;

3) if $z_i^1 = 1$ for $i \in I$ and $z_j^2 = 1$ for $j \in J$, where the sets I and J are a subset of a set $\{1, \cdots, l\}$, and $I \cup J \neq \varnothing$, then the tag is generated from the XOR-sum of the plaintext and ciphertext blocks. For example, I and J may be a partial of a set $\{1, \cdots, l\}$, such as odd/even partial. If I or J is an empty set, this case is reduced to case 1) or case 2).

Theorem 5. *For the above scheme $\Pi = (\mathcal{E}_K, \mathcal{D}_K, \mathcal{V}_K)$, for all $ld \geq n$ there exists an adversary $\mathcal{A}$ such that*

$$Adv_{\Pi}^{int-rup}(\mathcal{A}) \geq 1 - 2^{n-ld},$$

where $\mathcal{A}$ makes one encryption query, $2 \leq p \leq 2^d$ decryption queries and one forgery attempt, each consisting of l blocks of n bits. Then, the adversary solves a system of linear equations in $GF(2)$ with n equations and ld unknowns.

Proof. In the INT-RUP security model, we assume that the adversary $\mathcal{A}$ firstly makes one encryption query and receives $(C, T) = \mathcal{E}_K(N, A, P)$, where $P = P_1 P_2 \cdots P_l$. Then $\mathcal{A}$ makes at most p decryption queries with the same nonce-associated data pair and obtains the corresponding unverified plaintexts $P^j = D_K(N, A, C^j, T^j)$ where $0 \leq j \leq p-1, C^j = C_1^j C_2^j \cdots C_l^j$. Finally, $\mathcal{A}$ forges a new ciphertext $C' = C_1^{x_1} C_2^{x_2} \cdots C_l^{x_l}$ such that the new tag is equal to T, where $x_1, x_2, \cdots, x_l \in GF(p)$. If the forgery succeeds, the adversary needs to find $x_1, x_2, \cdots, x_l \in GF(p)$ such that

$$checksum = \sum_{i=1}^{l} (P_i z_i^1 \oplus C_i z_i^2) = \sum_{i=1}^{l} (P_i^{x_i} z_i^1 \oplus C_i^{x_i} z_i^2), \tag{1}$$

where $z_i^1, z_i^2 \in \{0, 1\}$ for all i.

Eq. (1) can be converted into a system of linear equations in $GF(p)$ with n equations and l unknowns, one for every bit j:

$$checksum[j] = \sum_{i=1}^{l} (P_i^{x_i}[j] z_i^1 \oplus C_i^{x_i}[j] z_i^2), \qquad 0 \leq j \leq n-1, \tag{2}$$

where $X[j]$ selects j-th bit of X, with $j = 0$ corresponding to the least significant bit, and $z_i^1, z_i^2 \in \{0, 1\}$ for all i.

The system of linear equations in $GF(p)$ with n equations and l unknowns is equivalent to a system of linear equations in $GF(2)$ with n equations and ld unknowns, where $d = \lceil \log_2 p \rceil$. The operation of this process is as follows.

Let $[x_i]_d = x_{i1} x_{i2} \cdots x_{id}$ be the d-bit binary representation of x_i, $[p-1]_d = p_1 p_2 \cdots p_d$ be the d-bit binary representation of $p-1$, then $C_i^{x_i} = C_i^0 (x_{i1} \oplus 1)(x_{i2} \oplus 1) \cdots (x_{id} \oplus 1) \oplus C_i^1 (x_{i1} \oplus 1)(x_{i2} \oplus 1) \cdots x_{id} \oplus \cdots \oplus C_i^{p-1} (x_{i1} \oplus p_1 \oplus 1)(x_{i2} \oplus p_2 \oplus 1) \cdots (x_{id} \oplus p_d \oplus 1)$, where $x_{i1} x_{i2} \cdots x_{id} = [s]_d$ corresponds to selecting $C_i^s, 0 \leq s \leq p-1$.

It follows that, Eq. (2) can be converted into the following equation:

$$\begin{aligned} checksum[j] = \sum_{i=1}^{l} \{ & [P_i^0[j](x_{i1} \oplus 1)(x_{i2} \oplus 1) \cdots (x_{id} \oplus 1) \oplus \cdots \\ & \oplus P_i^{p-1}[j](x_{i1} \oplus p_1 \oplus 1)(x_{i2} \oplus p_2 \oplus 1) \cdots (x_{id} \oplus p_d \oplus 1)] z_i^1 \oplus \\ & [C_i^0[j](x_{i1} \oplus 1)(x_{i2} \oplus 1) \cdots (x_{id} \oplus 1) \oplus \cdots \\ & \oplus C_i^{p-1}[j](x_{i1} \oplus p_1 \oplus 1)(x_{i2} \oplus p_2 \oplus 1) \cdots (x_{id} \oplus p_d \oplus 1)] z_i^2 \}, \end{aligned}$$

where $0 \le j \le n-1, p_1 p_2 \cdots p_d = [p-1]_d$, $z_i^1, z_i^2 \in \{0,1\}$ for all i.

The adversary needs to find $x_{11}, x_{12}, \cdots, x_{ld} \in GF(2)$ such that the above n equations are established. For a system of linear equations in $GF(2)$ with n equations and $ld \ge n$ unknowns, we can find a solution by using Gaussian elimination and the probability that this system of equations has a solution is $1 - 2^{n-ld}$. That is to say, the adversary can forge an output (N, A, C', T) with $C' \neq C$.

B: INT-RUP on Blockcipher-based OCB-IC

To realize OCB-IC with a tweakable blockcipher $\widetilde{E} : \mathcal{K} \times \mathcal{T} \times \{0,1\}^n \to \{0,1\}^n$, where $\mathcal{T} = \{0,1\}^n \times \mathcal{I} \times \mathcal{J}$ is a tweak space, $\mathcal{I}$ is a set of tuples of lager integers, and $\mathcal{J}$ is a set of tuples of small integers, we use a conventional block cipher $E : \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$ to instantiate OCB-IC[$\widetilde{E}$] by the XEX* construction $\widetilde{E} = XEX^*[E, 2^{\mathcal{I}}3^{\mathcal{J}}]$. Overloading the notation, we rewrite this scheme as OCB-IC[E].

The overview of OCB-IC[E] is depicted in Fig. 7. OCB-IC[E] is made up of three algorithms, an encryption algorithm $\mathcal{E}_K$, a decryption algorithm $\mathcal{D}_K$, and a verification algorithm $\mathcal{V}_K$. The detailed description of OCB-IC[E] is shown in Fig. 8. If the underlying block cipher E is a secure strong pseudorandom permutation (SPRP), OCB-IC[E] is proven INT-RUP up to the birthday bound in the nonce-misuse setting.

Theorem 6 (INT-RUP of OCB-IC with a Block Cipher). *Fix a block cipher $E : \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$ and a tweakable blockcipher $\widetilde{E} : \mathcal{K} \times \mathcal{T} \times \{0,1\}^n \to \{0,1\}^n$, where $\mathcal{T} = \{0,1\}^n \times \mathcal{I} \times \mathcal{J}$ is a tweak space, $\mathcal{I}$ is a set of tuples of lager integers, and $\mathcal{J}$ is a set of tuples of small integers. Assume $2^i 3^j \neq 1$ for all $(i,j) \in \mathcal{I} \times \mathcal{J}$. Let $\widetilde{E} = XEX^*[E, 2^{\mathcal{I}}3^{\mathcal{J}}]$, $\mathcal{A}$ be a nonce-misusing adversary, then we have*

$$Adv^{int-rup}_{OCB-IC[E]}(\mathcal{A}) \le Adv^{sprp}_{E}(\mathcal{B}) + 39.5\sigma^2/2^n + q_v q/2^n,$$

where a new adversary $\mathcal{B}$ has an additional running time equal to the time needed to process the queries from $\mathcal{A}$.

Proof Sketch: OCB-IC[E] uses the XEX* construction. Therefore, by Lemma 1, and Theorems 1 and 2, we can easily obtain the bound of INT-RUP on OCB-IC[E].

C: INT-RUP on Blockcipher-based OCB-IPC

Similarity with OCB-IC, we use a conventional block cipher $E : \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$ to instantiate OCB-IPC[$\widetilde{E}$] by $\widetilde{E} = XEX^*[E, 2^{\mathcal{I}}3^{\mathcal{J}}]$, where $\mathcal{I}$ is a set of tuples of lager integers, $\mathcal{J}$ is a set of tuples of small integers. Overloading the notation, we rewrite this scheme as OCB-IPC[E].

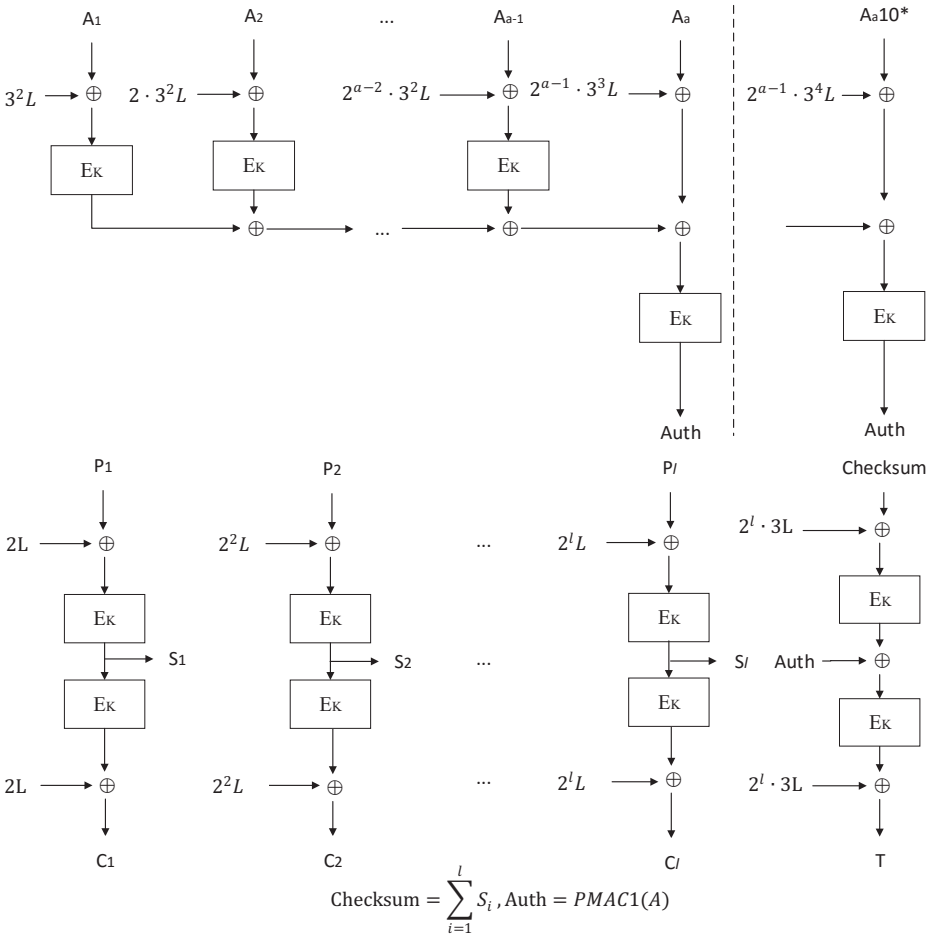


Fig. 7. OCB-IC[E] with a block cipher $E : \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$. This coincides with OCB-IC[$\widetilde{E}$], where $\widetilde{E} = XEX^*[E, 2^{\mathcal{I}}3^{\mathcal{J}}]$, $\mathcal{I}$ is a set of tuples of lager integers, and $\mathcal{J}$ is a set of tuples of small integers, e.g., $\mathcal{I} = \{0, 1, 2, \cdots, 2^n - 1\}$, and $\mathcal{J} = \{0, 1, \cdots, 10\}$. **Top row**: the authentication of associated data A: $Auth = PMAC1(A)$. If the length of associated data $|A|$ is not a positive multiple of n bits, padding 10^* to A so as to $|A10^*|$ is a positive multiple of n bits. The authentication of associated data is achieved by PMAC1 algorithm (XE construction). If there is no associated data, then we set $Auth = 0$. **Bottom row**: the encryption and authentication of the plaintext P (XEX construction). The plaintext is encrypted twice to produce the ciphertext and the XOR-sum of intermediate states is used to generate the tag. We require that the length of the plaintext P is a positive multiple of n bits in OCB-IC[E]. Given an arbitrary-length message $M \in \{0,1\}^*$, it needs to be padded to the plaintext $P = pad(M) = M10^{n-1-(|M| \bmod n)}$ before the encryption algorithm in OCB-IC[E]. Meanwhile, we obtain the message after the decryption algorithm by the unpadding function $unpad(P) = M$.

/*AD Processing*/
Algorithm $Auth(A)$ $//PMAC_K^N(A)$
Partition A into $A_1 \| \cdots \| A_a$
$|A_i| = n, 1 \leq i \leq a-1, 0 < |A_l| \leq n$
for $i = 1$ to $a-1$
$L = E_K(N)$
  $S_i \leftarrow E_K(A_i \oplus 2^{i-1}3^2L)$
if $|A_a| = n$
  $\Sigma \leftarrow S_1 \oplus S_2 \oplus \cdots \oplus S_{a-1} \oplus A_a$
  $Auth = E_K(\Sigma \oplus 2^{a-1}3^3L)$
else
  $\Sigma \leftarrow S_1 \oplus S_2 \oplus \cdots \oplus S_{a-1} \oplus A_a 10^*$
  $Auth = E_K(\Sigma \oplus 2^{a-1}3^4L)$
return Auth

/*Encryption Algorithm*/
Algorithm $OCB-IC.\mathcal{E}_K^N(A,P)$:

Partition P into $P_1 \| \cdots \| P_l$,
$|P_i| = n, 1 \leq i \leq l$
$L = E_K(N)$
for $i = 1$ to l
  $S_i \leftarrow E_K(P_i \oplus 2^iL)$
  $C_i \leftarrow E_K(S_i) \oplus 2^iL$
$C \leftarrow C_1C_2 \cdots C_l$
$\Sigma \leftarrow S_1 \oplus S_2 \oplus \cdots \oplus S_l$
$T_A = E_K(\Sigma \oplus 2^l \cdot 3L) \oplus Auth(A)$
$T = E_K(T_A) \oplus 2^l \cdot 3L$
return $C||T$

/*Decryption Algorithm*/
Algorithm $OCB-IC.\mathcal{D}_K^N(A,C||T)$:

Partition C into $C_1 \| \cdots \| C_l$,
$|C_i| = n, 1 \leq i \leq l$
$L = E_K(N)$
for $i = 1$ to l
  $S_i \leftarrow E_K^{-1}(C_i \oplus 2^iL)$
  $P_i \leftarrow E_K^{-1}(S_i) \oplus 2^iL$
$P \leftarrow P_1P_2 \cdots P_l$
return P

/*Verification Algorithm*/
Algorithm $OCB-IC.\mathcal{V}_K^N(A,C||T)$:

Partition C into $C_1 \| \cdots \| C_l$,
$|C_i| = n, 1 \leq i \leq l$
$L = E_K(N)$
for $i = 1$ to l
  $S_i \leftarrow E_K^{-1}(C_i \oplus 2^iL)$
$\Sigma \leftarrow S_1 \oplus S_2 \oplus \cdots \oplus S_l$
$T_A = E_K(\Sigma \oplus 2^l \cdot 3L) \oplus Auth(A)$
$T' = E_K(T_A) \oplus 2^l \cdot 3L$
if $T' = T$, return $\top$
else return $\bot$

Fig. 8. OCB-IC[E] with a block cipher $E : \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$. This coincides with OCB-IC[$\widetilde{E}$], where $\widetilde{E} = XEX^*[E, 2^{\mathcal{I}}3^{\mathcal{J}}]$, $\mathcal{I}$ is a set of tuples of lager integers, and $\mathcal{J}$ is a set of tuples of small integers, e.g., $\mathcal{I} = \{0,1,2,\cdots,2^n-1\}$, and $\mathcal{J} = \{0,1,\cdots,10\}$. The encryption algorithm $\mathcal{E}_K$ includes the encryption of the plaintext blocks, the authentications of associated data and the plaintext. The decryption algorithm $\mathcal{D}_K$ is straightforward similar to the encryption algorithm except no authentication of the tag at the end of the decryption process. The verification algorithm $\mathcal{V}_K$ outputs $\top$ if the new tag generated by the nonce-associated data-ciphertext pair is equal to the original tag, $\bot$ otherwise.

The overview of OCB-IPC[E] is depicted in Fig. 9. OCB-IPC[E] is made up of three algorithms, an encryption algorithm $\mathcal{E}_K$, a decryption algorithm $\mathcal{D}_K$, and a verification algorithm $\mathcal{V}_K$. The detailed description of OCB-IPC[E] is shown in Fig. 10. If the underlying block cipher E is a secure strong pseudorandom permutation (SPRP), OCB-IPC[E] is INT-RUP in the nonce-misuse setting.

Theorem 7 (INT-RUP of OCB-IPC with a Block Cipher). *Fix a block cipher $E : \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$ and a tweakable blockcipher $\widetilde{E} : \mathcal{K} \times \mathcal{T} \times \{0,1\}^n \to \{0,1\}^n$, where $\mathcal{T} = \{0,1\}^n \times \mathcal{I} \times \mathcal{J}$ is a tweak space, $\mathcal{I}$ is a set of tuples of lager integers, and $\mathcal{J}$ is a set of tuples of small integers. Assume $2^i 3^j \neq 1$ for all $(i,j) \in \mathcal{I} \times \mathcal{J}$. Let $\widetilde{E} = XEX^*[E, 2^{\mathcal{I}}3^{\mathcal{J}}]$, and $\mathcal{A}$ be a nonce-misusing adversary. Then we have*

$$Adv^{int-rup}_{OCB-IPC[E]}(\mathcal{A}) \leq Adv^{sprp}_{E}(\mathcal{B}) + \frac{39.5\sigma^2}{2^n} + \frac{q_v q}{2^{n+1}},$$

where a new adversary $\mathcal{B}$ has an additional running time equal to the time needed to process the queries from $\mathcal{A}$.

Proof Sketch: OCB-IPC[E] uses the XEX* construction. Therefore, by Lemma 1, and Theorems 3 and 4, we can easily obtain the bound of INT-RUP on OCB-IPC[E].

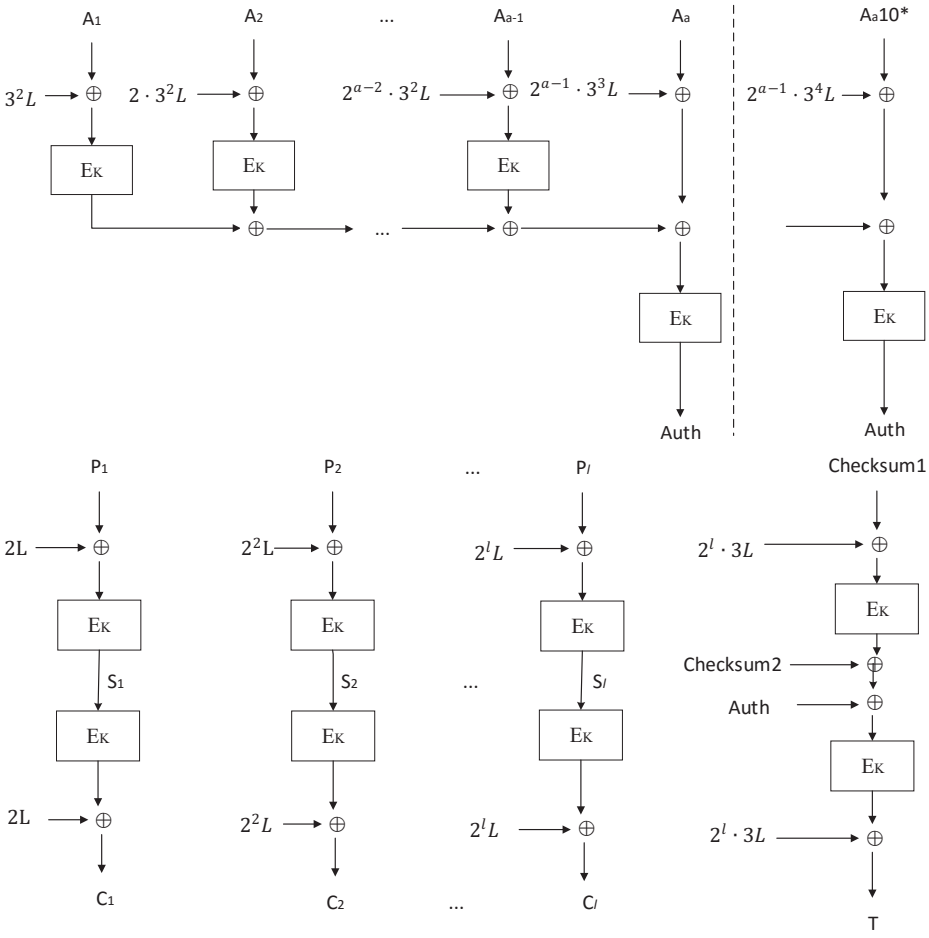


Fig. 9. OCB-IPC[E] with a block cipher $E : \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$. This coincides with OCB-IPC[$\widetilde{E}$], where $\widetilde{E} = XEX^*[E, 2^{\mathcal{I}}3^{\mathcal{J}}]$, $\mathcal{I}$ is a set of tuples of lager integers, and $\mathcal{J}$ is a set of tuples of small integers, e.g., $\mathcal{I} = \{0, 1, 2, \cdots, 2^n - 1\}$, and $\mathcal{J} = \{0, 1, \cdots, 10\}$. **Top row**: the authentication of associated data A: $Auth = PMAC1(A)$ (XE construction). If the length of associated data $|A|$ is not a positive multiple of n bits, padding 10^* to A so as to $|A10^*|$ is a positive multiple of n bits. The authentication of associated data is achieved by PMAC1 algorithm. If there is no associated data, then we set $Auth = 0$. **Bottom row**: the encryption and authentication of the plaintext P (XEX construction). The encryption procedure of the plaintext is the same with OCB-IC. The length of the plaintext P is a positive multiple of n bits. The odd and even parts of intermediate states are used to produce separately a checksum, which is encrypted to generate the final message authentication code. We require that the length of the plaintext P is a positive multiple of n bits in OCB-IPC[E]. Given an arbitrary-length message $M \in \{0,1\}^*$, it needs to be padded to the plaintext $P = pad(M) = M10^{n-1-(|M| \bmod n)}$ before the encryption algorithm in OCB-IPC[E]. Meanwhile, we obtain the message after the decryption algorithm by the corresponding unpadding function $unpad(P) = M$.

/*AD Processing*/
Algorithm $Auth(A)$ $//PMAC1_K^N(A)$
Partition A into $A_1\|\cdots\|A_a$
$|A_i| = n, 1 \le i \le a-1, 0 < |A_l| \le n$
for $i = 1$ to $a-1$
$L = E_K(N)$
    $S_i \leftarrow E_K(A_i \oplus 2^{i-1}3^2L)$
if $|A_a| = n$
    $\Sigma \leftarrow S_1 \oplus S_2 \oplus \cdots \oplus S_{a-1} \oplus A_a$
    $Auth = E_K(\Sigma \oplus 2^{a-1}3^3L)$
else
    $\Sigma \leftarrow S_1 \oplus S_2 \oplus \cdots \oplus S_{a-1} \oplus A_a10^*$
    $Auth = E_K(\Sigma \oplus 2^{a-1}3^4L)$
return Auth

/*Encryption Algorithm*/
Algorithm $OCB-IPC.\mathcal{E}_K^N(A,P)$:
Partition P into $P_1\|\cdots\|P_l$,
$|P_i| = n, 1 \le i \le l$
$L = E_K(N)$
for $i = 1$ to l
    $S_i \leftarrow E_K(P_i \oplus 2^iL)$
    $C_i \leftarrow E_K(S_i) \oplus 2^iL$
$C \leftarrow C_1C_2\cdots C_l$
$\Sigma_1 \leftarrow S_2 \oplus S_4 \oplus \cdots$
$\Sigma_2 \leftarrow S_1 \oplus S_3 \oplus \cdots$
$\Sigma = E_K(\Sigma_1 \oplus 2^l \cdot 3L) \oplus \Sigma_2$
$T = E_K(\Sigma \oplus Auth) \oplus 2^l \cdot 3L$
return $C||T$

/*Decryption Algorithm*/
Algorithm $OCB-IPC.\mathcal{D}_K^N(A,C||T)$:
Partition C into $C_1\|\cdots\|C_l$,
$|C_i| = n, 1 \le i \le l$
$L = E_K(N)$
for $i = 1$ to l
    $S_i \leftarrow E_K^{-1}(C_i \oplus 2^iL)$
    $P_i \leftarrow E_K^{-1}(S_i) \oplus 2^iL$
$P \leftarrow P_1P_2\cdots P_l$
return P

/*Verification Algorithm*/
Algorithm $OCB-IPC.\mathcal{V}_K^N(A,C||T)$:
Partition C into $C_1\|\cdots\|C_l$,
$|C_i| = n, 1 \le i \le l$
$L = E_K(N)$
for $i = 1$ to l
    $S_i \leftarrow E_K^{-1}(C_i \oplus 2^iL)$
$\Sigma_1 \leftarrow S_2 \oplus S_4 \oplus \cdots$
$\Sigma_2 \leftarrow S_1 \oplus S_3 \oplus \cdots$
$\Sigma = E_K(\Sigma_1 \oplus 2^l \cdot 3L) \oplus \Sigma_2$
$T' = E_K(\Sigma \oplus Auth) \oplus 2^l \cdot 3L$
if $T' = T$, return $\top$
else return $\bot$

Fig. 10. OCB-IPC[E] with a block cipher $E : \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$. This coincides with OCB-IPC[$\widetilde{E}$], where $\widetilde{E} = XEX^*[E, 2^{\mathcal{I}}3^{\mathcal{J}}]$, $\mathcal{I}$ is a set of tuples of lager integers, and $\mathcal{J}$ is a set of tuples of small integers, e.g., $\mathcal{I} = \{0,1,2,\cdots,2^n-1\}$, and $\mathcal{J} = \{0,1,\cdots,10\}$. The encryption algorithm $\mathcal{E}_K$ includes the encryption of the plaintext blocks, the authentications of associated data and the plaintext. The authentication of the plaintext is achieved by two PMAC1 algorithms. The authentication of associated data is a PMAC1 algorithm too. The decryption algorithm $\mathcal{D}_K$ is straightforward similar to the encryption algorithm except no authentication of the tag at the end of the decryption process. The verification algorithm $\mathcal{V}_K$ outputs $\top$ if the new tag generated by the associated data-ciphertext pair is equal to the original tag, $\bot$ otherwise.