Authenticated Encryption Mode for Beyond the Birthday Bound Security

Tetsu Iwata

Dept. of Computational Science and Engineering,
Nagoya University
Furo-cho, Chikusa-ku, Nagoya, 464-8603, Japan
iwata@cse.nagoya-u.ac.jp,
http://www.nuee.nagoya-u.ac.jp/labs/tiwata/

Abstract. In this paper, we propose an authenticated encryption mode for blockciphers. Our authenticated encryption mode, CIP, has provable security bounds which are better than the usual birthday bound security. Besides, the proven security bound for authenticity of CIP is better than any of the previously known schemes. The design is based on the encrypt-then-PRF approach, where the encryption part uses a key stream generation of CENC, and the PRF part combines a hash function based on the inner product and a blockcipher.

Keywords: Blockcipher, modes of operation, authenticated encryption, security proofs, birthday bound.

1 Introduction

Provable security is the standard security goal for blockcipher modes, i.e., encryption modes, message authentication codes, and authenticated encryption modes. For encryption modes, CTR mode and CBC mode are shown to have provable security [1]. The privacy notion we consider is called indistinguishability from random strings [24]. In this notion, the adversary is in the adaptive chosen plaintext attack scenario, and the goal is to distinguish the ciphertext from the random string of the same length. The nonce-based treatment of CTR mode was presented by Rogaway [22], and it was proved that, for any adversary against CTR mode, the success probability is at most $O(\sigma^2/2^n)$ under the assumption that the blockcipher is a secure pseudorandom permutation (PRP), where n is the block length and σ denotes the total ciphertext length in blocks that the adversary obtains. The security bound is known as the *birthday bound.*

Authenticity is achieved by message authentication codes, or MACs. Practical examples of MACs that have provable security include PMAC [7], EMAC [21], and OMAC [10]. We consider the pseudorandom function, or PRF [3], for authenticity which provably implies the adversary's inability to make a forgery. In this notion, the adversary is in the adaptive chosen plaintext attack scenario, and the goal is to distinguish the output of the MAC from that of the random function. It was proved that, for any adversary against PMAC, EMAC, and OMAC, the success probability is at most $O(\sigma^2/2^n)$.

An authenticated encryption mode is a scheme for both privacy and authenticity. It takes a plaintext M and provides both privacy and authenticity for M. There are a number of proposals: the first efficient construction was given by Jutla and the mode is called IAPM [13], OCB mode was proposed by Rogaway et. al. [24], CCM mode [27,12] is the standard of IEEE, EAX mode [6] is based on the generic composition, CWC mode [15] combines CTR mode and Wegman-Carter MAC, and GCM mode [19,20] is the standard of NIST. Other examples include CCFB mode [17], and XCBC [8]. All these modes have provable security with the standard birthday bound.

There are several proposals on MACs that have beyond the birthday bound security. For example, we have RMAC [11] and XOR MAC [2], and there are other proposals which are not based on blockciphers. On the other hand, few proposals are known for encryption modes and authenticated encryption modes. CENC [9] is an example of an encryption mode, and its generalization called NEMO was proposed in [16]. For authenticated encryption modes, CHM [9] is the only example we are aware of.

We view that the beyond the birthday bound security as the standard goal for future modes. AES is designed to be secure even if the adversary obtains nearly 2^{128} input-output pairs, and many other blockciphers have similar security goal. On the other hand, CTR mode, OMAC, or GCM have to re-key before 2^{64} blocks of plaintexts are processed, since otherwise the security is lost. This situation is unfortunate as the security of the blockcipher is significantly lost once it is plugged into the modes, and the current state-of-the-art, CTR mode, OMAC, or GCM, do not fully inherit the security of the blockcipher.

In this paper, we propose an authenticated encryption mode called CIP, CENC with Inner Product hash, to address the security issues in GCM, and CHM. GCM, designed by McGrew and Viega, was selected as the standard of NIST. It is based on CTR mode and Wegman-Carter MAC, and it is fully parallelizable. Likewise, CHM uses CENC for encryption part and Wegman-Carter MAC for PRF part,

While CHM has beyond the birthday bound security, its security bound for authenticity includes the term $M_{\max}/2^{\tau}$, where $M_{\max}$ is the maximum block length of messages, and τ is the tag length. It is a common practice to use small tag length to save communication cost or storage. For example, one may use $\tau = 32$ or 64 with 128-bit blockciphers. However the term, $M_{\max}/2^{\tau}$, is linear in $M_{\max}$, the bound soon becomes non-negligible if τ is small. For example, with $\tau = 32$, if we encrypt only one message of 2^{22} blocks (64MBytes), the security bound is 1/1024, which is not acceptable in general. Therefore, beyond the birthday bound security has little impact when τ is small. GCM also has the same issue, and its security bounds for both privacy and authenticity have the term of the form $M_{\max}/2^{\tau}$.

Our design goal of CIP is to have beyond the birthday bound security, but we insist that it can be used even with small tag length. Besides, we want security proofs with the standard PRP assumption, and we maintain the full parallelizability. CIP follows the encrypt-then-PRF approach [4], which is shown to be a sound way to construct an authenticated encryption mode. We use CENC [9] for encryption part, since it achieves beyond the birthday bound security with very small cost compared to CTR mode. PRF part is a hash function that combines the inner product hash and the blockcipher, which may be seen as the generalization of PMAC [7] to reduce the number of blockcipher calls and still have full parallelizability.

CIP takes a parameter ϖ called frame width, which is supposed to be a small integer (e.g., $2 \le \varpi \le 8$). Our default recommendation is $\varpi = 4$, and with other default parameters, to encrypt a message of l blocks, CIP requires $257l/256$ blockcipher calls for encryption, and l multiplications and $l/\varpi = l/4$ blockcipher calls for PRF, while $\varpi = 4$ blocks of key stream has to be precomputed and stored. CIP requires about l/ϖ more blockcipher calls compared to GCM or CHM. For security, if we use the AES, CIP can encrypt at most 2^{64} plaintexts, and the maximum length of the plaintext is 2^{62} blocks (2^{36}GBytes), and the security bounds are, roughly, $\tilde{\sigma}^3/2^{245} + \tilde{\sigma}/2^{119}$ for privacy, and $\tilde{\sigma}^3/2^{245} + \tilde{\sigma}/2^{118} + 2/2^{\tau}$ for authenticity. This implies $\tilde{\sigma}$ should be sufficiently smaller than 2^{81} blocks (2^{55}GBytes). In particular, the only term that depends on tag length τ is $2/2^{\tau}$, and thus it does not depend on the message length. Therefore, CIP can be used even for short tag length. CIP has security bounds that are better than any of the known schemes we are aware of.

2 Preliminaries

Notation. If x is a string then $|x|$ denotes its length in bits, and $|x|_n$ is its length in n-bit blocks, i.e., $|x|_n = \lceil |x|/n \rceil$. If x and y are two equal-length strings, then $x \oplus y$ denotes the xor of x and y. If x and y are strings, then $x\|y$ or xy denote their concatenation. Let $x \leftarrow y$ denote the assignment of y to x. If X is a set, let $x \overset{R}{\leftarrow} X$ denote the process of uniformly selecting at random an element from X and assigning it to x. For a positive integer n, $\{0,1\}^n$ is the set of all strings of n bits. For positive integers n and ϖ, $(\{0,1\}^n)^{\varpi}$ is the set of all strings of $n\varpi$ bits, and $\{0,1\}^*$ is the set of all strings (including the empty string). For positive integers n and m such that $n \le 2^m - 1$, $\langle n \rangle_m$ is the m-bit binary representation of n. For a bit string x and a positive integer n such that $|x| \ge n$, $\mathsf{first}(n, x)$ and $\mathsf{last}(n, x)$ denote the first n bits of x and the last n bits of x, respectively. For a positive integer n, 0^n and 1^n denote the n-times repetition of 0 and 1, respectively.

Let Perm(n) be the set of all permutations on $\{0,1\}^n$. We say P is a random permutation if $P \overset{R}{\leftarrow} \mathrm{Perm}(n)$. The blockcipher is a function $E : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$, where, for any $K \in \{0,1\}^k$, $E(K, \cdot) = E_K(\cdot)$ is a permutation on $\{0,1\}^n$. The positive integer n is the block length, and k is the key length. Similarly, Func(m, n) denotes the set of all functions from $\{0,1\}^m$ to $\{0,1\}^n$, and R is a random function if $R \overset{R}{\leftarrow} \mathrm{Func}(m, n)$.

The frame, nonce, and counter. CIP takes a positive integer ϖ as a parameter, and it is called a frame width. For fixed positive integer ϖ (say, $\varpi = 4$), a

ϖ-block string is called a frame. Throughout this paper, we assume $\varpi \geq 1$. A nonce N is a bit string, where for each pair of key and plaintext, it is used only once. The length of the nonce is denoted by ℓ_{nonce}, and it is at most the block length. We also use an n-bit counter, ctr. This value is initialized based on the value of the nonce, then it is incremented after each blockcipher invocation. The function for increment is denoted by $\mathsf{inc}(\cdot)$. It takes an n-bit string x (a counter) and returns the incremented x. We assume $\mathsf{inc}(x) = x + 1 \bmod 2^n$, but other implementations also work, e.g., with LFSRs if $x \neq 0^n$.

3 Specification of CIP

In this section, we present our authenticated encryption scheme, CIP. It takes five parameters: a blockcipher, a nonce length, a tag length, and two frame widths.

Fix the blockcipher $E : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$, the nonce length ℓ_{nonce}, the tag length τ, and the frame widths ϖ and w. We require that $\log_2(\lceil k/n \rceil + \varpi) < \ell_{\text{nonce}} < n$, and $1 \leq \tau \leq n$.

CIP consists of two algorithms, the encryption algorithm (CIP.Enc) and the decryption algorithm (CIP.Dec). These algorithms are defined in Fig. 1. The encryption algorithm, CIP.Enc, takes the key $K \in \{0,1\}^k$, the nonce $N \in \{0,1\}^{\ell_{\text{nonce}}}$, and the plaintext M to return the ciphertext C and the tag $\text{Tag} \in \{0,1\}^\tau$. We have $|M| = |C|$, and the length of M is at most $2^{n-\ell_{\text{nonce}}-2}$ blocks. We write $(C, \text{Tag}) \leftarrow \text{CIP.Enc}_K(N, M)$. The decryption algorithm, CIP.Dec, takes K, N, C and Tag to return M or a special symbol $\perp$. We write $M \leftarrow \text{CIP.Dec}_K(N, C, \text{Tag})$ or $\perp \leftarrow \text{CIP.Dec}_K(N, C, \text{Tag})$. Both algorithms internally use the key setup algorithm (CIP.Key), a hash function (CIP.Hash), and the keystream generation algorithm (CIP.KSGen).

We use the standard key derivation for key setup. The input of CIP.Key is the blockcipher key $K \in \{0,1\}^k$, and the output is $(K_H, T_H) \in \{0,1\}^k \times (\{0,1\}^n)^\varpi$, where $T_H = (T_0, \ldots, T_{\varpi-1})$, and

- K_H is the first k bits of

$$E_K(\langle 0 \rangle_{\ell_{\text{nonce}}} \| 1^{n-\ell_{\text{nonce}}}) \| \cdots \| E_K(\langle \lceil k/n \rceil - 1 \rangle_{\ell_{\text{nonce}}} \| 1^{n-\ell_{\text{nonce}}}),$$

  and
- $T_i \leftarrow E_K(\langle \lceil k/n \rceil + i \rangle_{\ell_{\text{nonce}}} \| 1^{n-\ell_{\text{nonce}}})$ for $0 \leq i \leq \varpi - 1$.

These keys are used for CIP.Hash, which is defined in Fig. 2 (See also Fig. 8 for an illustration). It takes the key $(K_H, T_H) \in \{0,1\}^k \times (\{0,1\}^n)^\varpi$, and the input $x \in \{0,1\}^*$, and the output is a hash value $\text{Hash} \in \{0,1\}^n$. The inner product is done in the finite field $\text{GF}(2^n)$ using a canonical polynomial to represent field elements. The suggested canonical polynomial is the lexicographically first polynomial among the irreducible polynomials of degree n that have a minimum number of nonzero coefficients. For $n = 128$ the indicated polynomial is $\mathsf{x}^{128} + \mathsf{x}^7 + \mathsf{x}^2 + \mathsf{x} + 1$. CIP.KSGen, defined in Fig. 3, is equivalent to CENC in [9], and

```
Algorithm CIP.Enc$_K(N, M)$
100    $(K_H, T_H) \leftarrow$ CIP.Key$(K)$
101    $l \leftarrow \lceil |M|/n \rceil$
102    $\texttt{ctr} \leftarrow (N \| 0^{n-\ell_{\text{nonce}}})$
103    $S \leftarrow$ CIP.KSGen$_K(\texttt{ctr}, l+1)$
104    $S_H \leftarrow \mathsf{first}(n, S)$
105    $S_{\text{mask}} \leftarrow \mathsf{last}(n \times l, S)$
106    $C \leftarrow M \oplus \mathsf{first}(|M|, S_{\text{mask}})$
107    Hash $\leftarrow$ CIP.Hash$_{K_H, T_H}(C)$
108    Tag $\leftarrow \mathsf{first}(\tau, \text{Hash} \oplus S_H)$
109    return $(C, \text{Tag})$
```

```
Algorithm CIP.Dec$_K(N, C, \text{Tag})$
200    $(K_H, T_H) \leftarrow$ CIP.Key$(K)$
201    $l \leftarrow \lceil |C|/n \rceil$
202    $\texttt{ctr} \leftarrow (N \| 0^{n-\ell_{\text{nonce}}})$
203    $S \leftarrow$ CIP.KSGen$_K(\texttt{ctr}, l+1)$
204    $S_H \leftarrow \mathsf{first}(n, S)$
205    Hash$' \leftarrow$ CIP.Hash$_{K_H, T_H}(C)$
206    Tag$' \leftarrow \mathsf{first}(\tau, \text{Hash}' \oplus S_H)$
207    if Tag$' \neq$ Tag then return $\perp$
208    $S_{\text{mask}} \leftarrow \mathsf{last}(n \times l, S)$
209    $M \leftarrow C \oplus \mathsf{first}(|C|, S_{\text{mask}})$
210    return $M$
```

Fig. 1. Definition of CIP.Enc (left), and CIP.Dec (right). CIP.KSGen is defined in Fig. 3, and CIP.Hash is defined in Fig. 2.

```
Algorithm CIP.Hash$_{K_H, T_H}(x)$
100    $x \leftarrow x \| 10^{n-1-(|x| \bmod n)}$
101    $l \leftarrow |x|/n$; $(x_0, \ldots, x_{l-1}) \leftarrow x$; $\ell \leftarrow \lceil l/\varpi \rceil$
102    Hash $\leftarrow 0^n$
103    for $i \leftarrow 0$ to $\ell - 2$ do
104        $A_i \leftarrow (x_{i\varpi}, \ldots, x_{(i+1)\varpi-1}) \cdot (T_0, \ldots, T_{\varpi-1})$
105        Hash $\leftarrow$ Hash $\oplus E_{K_H}(A_i \oplus \langle i \rangle_n)$
106    $A_{\ell-1} \leftarrow (x_{(\ell-1)\varpi}, \ldots, x_{l-1}) \cdot (T_0, \ldots, T_{l-(\ell-1)\varpi-1})$
107    Hash $\leftarrow$ Hash $\oplus E_{K_H}(A_{\ell-1} \oplus \langle \ell - 1 \rangle_n)$
108    return Hash
```

Fig. 2. Definition of CIP.Hash. The inner product in lines 104 and 106 is in GF(2^n).

is parameterized by E and w. It takes the blockcipher key K, counter value `ctr`, and an integer l as inputs, and the output is a bit string S of l blocks. See Fig. 9 for an illustration.

Discussion and default parameters. CIP takes five parameters, the blockcipher $E : \{0,1\}^k \times \{0,1\}^n \rightarrow \{0,1\}^n$, the nonce length ℓ_{nonce}, the tag length τ, and the frame widths ϖ and w. With these parameters, CIP can encrypt at most $2^{\ell_{\text{nonce}}}$ plaintexts, and the maximum length of the plaintext is $2^{n-\ell_{\text{nonce}}-2}$ blocks.

Our default parameters are, E is any blockcipher such that $n \geq 128$, $\ell_{\text{nonce}} = n/2$, and $\tau \geq 32$. The values of ϖ and w affect the efficiency and security. Specifically, to hash ϖ blocks of input, CIP.Hash requires ϖ blocks of keys for inner product, ϖ multiplications and one blockcipher call. Thus, large ϖ implies that per message block computation is reduced, while it increases the pre-computation time and register for keys. Therefore there is a trade-off between security, per block efficiency and the pre-computation time/resister size. ϖ is supposed to be a small integer (e.g., $2 \leq \varpi \leq 8$), and our default recommendation is $\varpi = 4$. w follows the recommendation of CENC, and its default value is 256.

```
Algorithm CIP.KSGen_K(ctr, l)
100  for $j \leftarrow 0$ to $\lceil l/w \rceil - 1$ do
101      $L \leftarrow E_K(\mathtt{ctr})$
102      $\mathtt{ctr} \leftarrow \mathsf{inc}(\mathtt{ctr})$
103      for $i \leftarrow 0$ to $w - 1$ do
104          $S_{wj+i} \leftarrow E_K(\mathtt{ctr}) \oplus L$
105          $\mathtt{ctr} \leftarrow \mathsf{inc}(\mathtt{ctr})$
106          if $wj + i = l - 1$ then
107              $S \leftarrow (S_0 \| S_1 \| \cdots \| S_{l-1})$
108              return $S$
```

Fig. 3. Definition of CIP.KSGen, which is equivalent to CENC [9]

With these parameters, if we use the AES, CIP can encrypt at most 2^{64} plaintexts, and the maximum length of the plaintext is 2^{62} blocks (2^{36}GBytes), and the security bounds are $\tilde{\sigma}^3/2^{245}+\tilde{\sigma}/2^{119}$ for privacy, and $\tilde{\sigma}^3/2^{245}+\tilde{\sigma}/2^{118}+2/2^{\tau}$ for authenticity, where $\tilde{\sigma}$ is (roughly) the total number of blocks processed by one key. This implies $\tilde{\sigma}$ should be sufficiently smaller than 2^{81} blocks (2^{55}GBytes).

Information theoretic version. We will derive our security results in the information theoretic setting and in the computational setting. In the former case, a random permutation is used instead of a blockcipher, where we consider that CIP.Key takes a random permutation P as its input, and uses P to derive K_H and T_H by "encrypting" constants. Therefore, P is used in CIP.KSGen (lines 101 and 104 in Fig. 3), and (K_H, T_H) derived from P is used in CIP.Hash. We still use a real blockcipher in lines 105 and 107 in Fig. 2 even in the information theoretic version.

4 Security of CIP

CIP is an authenticated encryption (AE) scheme. We first present its security definitions, and then present our security results.

Security of blockciphers. We follow the PRP notion for blockciphers that was introduced in [18]. An adversary is a probabilistic algorithm with access to one or more oracles. Let A be an adversary with access to an oracle, either the encryption oracle $E_K(\cdot)$ or a random permutation oracle $P(\cdot)$, and returns a bit. We say A is a *PRP-adversary* for E, and define

$$\mathbf{Adv}_E^{\mathrm{prp}}(A) \overset{\mathrm{def}}{=} \left|\Pr(K \overset{R}{\leftarrow} \{0,1\}^k : A^{E_K(\cdot)} = 1) - \Pr(P \overset{R}{\leftarrow} \mathrm{Perm}(n) : A^{P(\cdot)} = 1)\right|.$$

For an adversary A, A's running time is denoted by *time*(A). The running time is its actual running time (relative to some fixed RAM model of computation) and its description size (relative to some standard encoding of algorithms). The details of the big-O notation for the running time reference depend on the RAM model and the choice of encoding.

Privacy of CIP. We follow the security notion from [6]. Let A be an adversary with access to an oracle, either the encryption oracle $\mathrm{CIP.Enc}_K(\cdot,\cdot)$ or $\mathcal{R}(\cdot,\cdot)$, and returns a bit. The $\mathcal{R}(\cdot,\cdot)$ oracle, on input (N, M), returns a random string of length $|\mathrm{CIP.Enc}_K(N, M)|$. We say that A is a PRIV-adversary for CIP. We assume that any PRIV-adversary is nonce-respecting. That is, if $(N_0, M_0), \ldots, (N_{q-1}, M_{q-1})$ are A's oracle queries, then $N_0, \ldots, N_{q-1}$ are always distinct, regardless of oracle responses and regardless of A's internal coins. The advantage of PRIV-adversary A for $\mathrm{CIP} = (\mathrm{CIP.Enc}, \mathrm{CIP.Dec})$ is

$$\mathbf{Adv}^{\mathrm{priv}}_{\mathrm{CIP}}(A) \overset{\mathrm{def}}{=} \left|\Pr(K \overset{R}{\leftarrow} \{0,1\}^k : A^{\mathrm{CIP.Enc}_K(\cdot,\cdot)} = 1) - \Pr(A^{\mathcal{R}(\cdot,\cdot)} = 1)\right|.$$

Privacy results on CIP. Let A be a nonce-respecting PRIV-adversary for CIP, and assume that A makes at most q oracle queries, and the total plaintext length of these queries is at most σ blocks, i.e., if A makes exactly q queries $(N_0, M_0), \ldots, (N_{q-1}, M_{q-1})$, then $\sigma = \lceil |M_0|/n \rceil + \cdots + \lceil |M_{q-1}|/n \rceil$, the total number of blocks of plaintexts. We have the following information theoretic result.

Theorem 1. *Let Perm(n), ℓ_{nonce}, τ, ϖ, and w be the parameters for CIP. Let A be a nonce-respecting PRIV-adversary making at most q oracle queries, and the total plaintext length of these queries is at most σ blocks. Then*

$$\mathbf{Adv}^{\mathrm{priv}}_{\mathrm{CIP}}(A) \le \frac{wr^2\tilde{\sigma}^2}{2^{2n-4}} + \frac{w\tilde{\sigma}^3}{2^{2n-3}} + \frac{r^2}{2^{n+1}} + \frac{w\tilde{\sigma}}{2^n}, \qquad (1)$$

where $r = \lceil k/n \rceil + \varpi$ and $\tilde{\sigma} = \sigma + q(w+1)$.

The proof of Theorem 1 is given in the next section. From Theorem 1, we have the following complexity theoretic result.

Corollary 1. *Let E, ℓ_{nonce}, τ, ϖ, and w be the parameters for CIP. Let A be a nonce-respecting PRIV-adversary making at most q oracle queries, and the total plaintext length of these queries is at most σ blocks. Then there is a PRP-adversary B for E making at most $2\tilde{\sigma}$ oracle queries, time(B) = time$(A) + O(n\tilde{\sigma})$, and $\mathbf{Adv}^{\mathrm{prp}}_{E}(B) \ge \mathbf{Adv}^{\mathrm{priv}}_{\mathrm{CIP}}(A) - wr^2\tilde{\sigma}^2/2^{2n-4} - w\tilde{\sigma}^3/2^{2n-3} - r^2/2^{n+1} - w\tilde{\sigma}/2^n$, where $r = \lceil k/n \rceil + \varpi$ and $\tilde{\sigma} = \sigma + q(w+1)$.*

The proof is standard (e.g., see [9]), and omitted.

Authenticity of CIP. A notion of authenticity of ciphertext for AE schemes was formalized in [24,23] following [14,5,4]. Let A be an adversary with access to an encryption oracle $\mathrm{CIP.Enc}_K(\cdot,\cdot)$ and returns a tuple, $(N^*, C^*, \mathrm{Tag}^*)$, called a forgery attempt. We say that A is an AUTH-adversary for CIP. We assume that any AUTH-adversary is nonce-respecting, where the condition applies only to the adversary's encryption oracle. Thus a nonce used in an encryption-oracle query may be used in a forgery attempt. We say A forges if A returns $(N^*, C^*, \mathrm{Tag}^*)$ such that $\mathrm{CIP.Dec}_K(N^*, C^*, \mathrm{Tag}^*) \not\to \perp$ but A did not make a query (N^*, M^*)

to CIP.Enc$_K(\cdot,\cdot)$ that resulted in a response (C^*, Tag^*). That is, adversary A may never return a forgery attempt $(N^*, C^*, \mathrm{Tag}^*)$ such that the encryption oracle previously returned (C^*, Tag^*) in response to a query (N^*, M^*). Then the advantage of AUTH-adversary A for CIP = (CIP.Enc, CIP.Dec) is

$$\mathbf{Adv}^{\mathrm{auth}}_{\mathrm{CIP}}(A) \overset{\mathrm{def}}{=} \Pr(K \overset{R}{\leftarrow} \{0,1\}^k : A^{\mathrm{CIP.Enc}_K(\cdot,\cdot)} \text{ forges}).$$

Authenticity results on CIP. Let A be an AUTH-adversary for CIP, and assume that A makes at most q oracle queries (including the final forgery attempt), and the total plaintext length of these queries is at most σ blocks. That is, if A makes queries $(N_0, M_0), \ldots, (N_{q-2}, M_{q-2})$, and returns the forgery attempt $(N^*, C^*, \mathrm{Tag}^*)$, then $\sigma = \lceil |M_0|/n \rceil + \cdots + \lceil |M_{q-2}|/n \rceil + \lceil |C^*|/n \rceil$. We have the following information theoretic result.

Theorem 2. *Let Perm(n), ℓ_{nonce}, τ, ϖ, and w be the parameters for CIP. Let A be a nonce-respecting AUTH-adversary making at most q oracle queries, and the total plaintext length of these queries is at most σ blocks. Then, for some D,*

$$\mathbf{Adv}^{\mathrm{auth}}_{\mathrm{CIP}}(A) \le \frac{wr^2\tilde{\sigma}^2}{2^{2n-4}} + \frac{w\tilde{\sigma}^3}{2^{2n-3}} + \frac{r^2}{2^{n+1}} + \frac{w\tilde{\sigma}}{2^n} + \frac{\sigma}{2^{n-1}} + \frac{2}{2^\tau} + \mathbf{Adv}^{\mathrm{prp}}_E(D) \quad (2)$$

where $r = \lceil k/n \rceil + \varpi$, $\tilde{\sigma} = \sigma + q(w+1)$, D makes at most 2σ queries, and time$(D) = O(n\sigma)$.

Note that the left hand side of (2) has $\mathbf{Adv}^{\mathrm{prp}}_E(D)$, since we use a blockcipher in CIP.Hash, while there is no restriction on the running time of A.

The proof of Theorem 2 is given in Section 6. From Theorem 2, we have the following complexity theoretic result.

Corollary 2. *Let E, ℓ_{nonce}, τ, ϖ, and w be the parameters for CIP. Let A be a nonce-respecting AUTH-adversary making at most q oracle queries, and the total plaintext length of these queries is at most σ blocks. Then there is a PRP-adversary B for E making at most $2\tilde{\sigma}$ oracle queries, time(B) = time(A) + $O(n\tilde{\sigma})$, and $\mathbf{Adv}^{\mathrm{prp}}_E(B) \ge \mathbf{Adv}^{\mathrm{auth}}_{\mathrm{CIP}}(A) - wr^2\tilde{\sigma}^2/2^{2n-4} - w\tilde{\sigma}^3/2^{2n-3} - r^2/2^{n+1} - w\tilde{\sigma}/2^n - \sigma/2^{n-1} - 2/2^\tau - \mathbf{Adv}^{\mathrm{prp}}_E(n\sigma, 2\sigma)$, where $r = \lceil k/n \rceil + \varpi$, $\tilde{\sigma} = \sigma + q(w+1)$, and $\mathbf{Adv}^{\mathrm{prp}}_E(n\sigma, 2\sigma)$ is the maximum of $\mathbf{Adv}^{\mathrm{prp}}_E(D)$ over all D such that it makes at most 2σ queries, and time$(D) = O(n\sigma)$.*

The proof is standard (e.g., see [9]), and omitted.

5 Security Proof for Privacy of CIP

We first recall the following tool from [9]. Consider the function family F^+, which corresponds to one frame of CIP.KSGen, and it is defined as follows: Let $P \overset{R}{\leftarrow} \mathrm{Perm}(n)$ be a random permutation, and fix the frame width w. Then $F^+ : \mathrm{Perm}(n) \times \{0,1\}^n \to (\{0,1\}^n)^w$ is $F^+_P(x) = (y[0], \ldots, y[w-1])$, where $y[i] = L \oplus P(\mathsf{inc}^{i+1}(x))$ for $i = 0, \ldots, w-1$ and $L = P(x)$.

Now let A be an adversary. This A is the PRF-adversary for F^+, but we give A additional information, i.e., we allow A to access the blockcipher itself. That is, A is given either a pair of oracles $(P(\cdot), F_P^+(\cdot))$, or a pair of random function oracles $(R_0(\cdot), R_1(\cdot))$, where $R_0 \in \mathrm{Func}(n,n)$ and $R_1 \in \mathrm{Func}(n,nw)$, with the following rules.

– If $W_i \in \{0,1\}^n$ is the i-th query for the first oracle (either $P(\cdot)$ or $R_0(\cdot)$), then $(\ell_{\mathrm{nonce}}+1)$-th bit of W_i must be 1.
– If $x_j \in \{0,1\}^n$ is the j-th query for the second oracle (either $F_P^+(\cdot)$ or $R_1(\cdot)$), then $(\ell_{\mathrm{nonce}}+1)$-th bit of x_j must be 0. That is, input/output samples from the first oracle are not used in $F_P^+(\cdot)$ oracle.
– A does not repeat the same query to its first oracle.
– Let $x_j \in \{0,1\}^n$ denote A's j-th query to its second oracle, and let $X_j = \{x_j, \mathsf{inc}(x_j), \mathsf{inc}^2(x_j), \ldots, \mathsf{inc}^w(x_j)\}$, i.e., X_j is the set of input to P in the j-th query. Now if A makes at most q calls to the second oracle, $X_j \cap X_{j'} = \emptyset$ must hold for any $0 \le j < j' \le q-1$, regardless of oracle responses and regardless of A's internal coins.

Define $\mathbf{Adv}^{\mathrm{prf}}_{\mathrm{Perm}(n),F^+}(A)$ as

$$\Big|\Pr(P \xleftarrow{R} \mathrm{Perm}(n) : A^{P(\cdot),F_P^+(\cdot)} = 1) - \Pr(R_0 \xleftarrow{R} \mathrm{Func}(n,n), R_1 \xleftarrow{R} \mathrm{Func}(n,nw) : A^{R_0(\cdot),R_1(\cdot)} = 1)\Big|$$

and we say A is a PRF-adversary for $(\mathrm{Perm}(n), F^+)$.

We have the following information theoretic result, whose proof is almost the same as that of [9, Theorem 5].

Proposition 1. *Let Perm(n) and w be the parameters for F^+. Let A be the PRF-adversary for $(Perm(n), F^+)$, with the above restrictions, making at most r oracle queries to its first oracle and at most q oracle queries to its second oracle. Then*

$$\mathbf{Adv}^{\mathrm{prf}}_{\mathrm{Perm}(n),F^+}(A) \le \frac{r^2q^2(w+1)^3}{2^{2n-1}} + \frac{q^3(w+1)^4}{2^{2n+1}} + \frac{r(r-1)}{2^{n+1}} + \frac{qw(w+1)}{2^{n+1}}.$$

Now Theorem 1 follows by using Proposition 1. To see this, by using the PRIV-adversary A for CIP as a subroutine, it is possible to construct a PRF-adversary B for $(\mathrm{Perm}(n), F^+)$. B first makes $\lceil k/n \rceil + \varpi$ calls to its first oracle and constructs K_H and T_H, and simulates line 103 of Fig. 1 as in Fig. 4 by making $\tilde{\sigma}/w$ calls to the second oracle.

6 Security Proofs for Authenticity of CIP

6.1 Properties of the Inner Product Hash

We first recall that the inner product hash is ϵ-AXU for small ϵ [26].

```
Algorithm CIP.KSGen.Sim(ctr, l)
100  for $j \leftarrow 0$ to $\lceil l/w \rceil - 1$ do
101      $S_j \leftarrow F_P^+(\mathtt{ctr})$
102      $\mathtt{ctr} \leftarrow \mathsf{inc}^{w+1}(\mathtt{ctr})$
103  $S \leftarrow (S_0, \ldots, S_{\lceil l/w \rceil - 1})$
104  $S \leftarrow \mathsf{first}(n \times l, S)$
105  return $S$
```

Fig. 4. The simulation CIP.KSGen.Sim of CIP.KSGen using F^+

Proposition 2. *Let* $(x_0, \ldots, x_{\varpi-1}), (x'_0, \ldots, x'_{\varpi-1}) \in (\{0,1\}^n)^{\varpi}$ *be two distinct bit strings. Then for any* $y \in \{0,1\}^n$,

$$\Pr(T_H \overset{R}{\leftarrow} (\{0,1\}^n)^{\varpi} : (x_0, \ldots, x_{\varpi-1}) \cdot (T_0, \ldots, T_{\varpi-1}) \\ \oplus (x'_0, \ldots, x'_{\varpi-1}) \cdot (T_0, \ldots, T_{\varpi-1}) = y) = 1/2^n.$$

Proof. We have $x_i \oplus x'_i \neq 0^n$ for some i. Therefore, the coefficient of T_i in $(x_0 \oplus x'_0) \cdot T_0 \oplus \cdots \oplus (x_{\varpi-1} \oplus x'_{\varpi-1}) \cdot T_{\varpi-1} = y$ is non-zero, and for any fixed $T_0, \ldots, T_{i-1}, T_{i+1}, \ldots, T_{\varpi-1}$, exactly one value of T_i satisfies the equality. □

If $y \neq 0^n$, a similar result holds for two bit strings of different block sizes.

Proposition 3. *Let* $\varpi \geq \varpi'$, *and let* $x = (x_0, \ldots, x_{\varpi-1}) \in (\{0,1\}^n)^{\varpi}$ *and* $x' = (x'_0, \ldots, x'_{\varpi'-1}) \in (\{0,1\}^n)^{\varpi'}$ *be two distinct bit strings. Then for any non-zero* $y \in \{0,1\}^n$,

$$\Pr(T_H \overset{R}{\leftarrow} (\{0,1\}^n)^{\varpi} : (x_0, \ldots, x_{\varpi-1}) \cdot (T_0, \ldots, T_{\varpi-1}) \\ \oplus (x'_0, \ldots, x'_{\varpi'-1}) \cdot (T_0, \ldots, T_{\varpi'-1}) = y) = 1/2^n.$$

Proof. The condition can be written as: $(x_0 \oplus x'_0) \cdot T_0 \oplus \cdots \oplus (x_{\varpi'-1} \oplus x'_{\varpi'-1}) \cdot T_{\varpi'-1} \oplus x_{\varpi'} \cdot T_{\varpi'} \oplus \cdots \oplus x_{\varpi-1} \cdot T_{\varpi-1} = y$. If all the coefficients of T_i are zero, then this equation can not be true since y is non-zero. Therefore, we can without loss of generality assume that at least one of coefficients of T_i is non-zero. □

6.2 Properties of the CIP.Hash

We next analyze the properties of CIP.Hash.

Let $x, x' \in \{0,1\}^*$ be two distinct bit strings, where $|x| \geq |x'|$. We show (in Proposition 8) that for any y, $\Pr(\text{CIP.Hash}_{K_H,T_H}(x) \oplus \text{CIP.Hash}_{K_H,T_H}(x') = y)$ is small, where the probability is taken over the choices of K_H and T_H.

We begin by introducing the notation. Let $X \leftarrow x \| 10^{n-1-(|x| \bmod n)}$ and $X' \leftarrow x' \| 10^{n-1-(|x'| \bmod n)}$. We parse them into blocks as $X = (X_0, \ldots, X_{l-1})$ and $X' = (X'_0, \ldots, X'_{l'-1})$, where $l = |X|/n$ and $l' = |X'|/n$. Let $\ell = \lceil l/\varpi \rceil$ and $\ell' = \lceil l'/\varpi \rceil$. We write the i-th frame of X and X' as χ_i and χ'_i, respectively. That is, $\chi_i = (X_{i\varpi}, \ldots, X_{(i+1)\varpi-1})$ for $0 \leq i \leq \ell - 2$, $\chi_{\ell-1} = (X_{(\ell-1)\varpi}, \ldots, X_{l-1})$, $\chi'_i = (X'_{i\varpi}, \ldots, X'_{(i+1)\varpi-1})$ for $0 \leq i \leq \ell' - 2$, and $\chi'_{\ell'-1} = (X_{(\ell'-1)\varpi}, \ldots, X_{l'-1})$.

Further, let $\chi_i \cdot T_H = A_i$ for $0 \le i \le \ell - 2$, $\chi_{\ell-1} \cdot (T_0, \ldots, T_{l-(\ell-1)\varpi-1}) = A_{\ell-1}$, $\chi'_i \cdot T_H = A'_i$ for $0 \le i \le \ell' - 1$, and $\chi'_{\ell'-1} \cdot (T_0, \ldots, T_{l'-(\ell'-1)\varpi-1}) = A_{\ell'-1}$. That is, A_i and A'_i are the results of the inner product in lines 104 and 106 of Fig. 2.

In the following three propositions, we first show that, for some i, $A_i \oplus \langle i \rangle_n$ is unique with high probability in the multi-set $\{A_0 \oplus \langle 0 \rangle_n, \ldots, A_{\ell-1} \oplus \langle \ell-1 \rangle_n, A'_0 \oplus \langle 0 \rangle_n, \ldots, A'_{\ell'-1} \oplus \langle \ell' - 1 \rangle_n\}$.

Proposition 4. *Suppose that $l = l'$. Then there are at least $2^{n\varpi}(1-(2\ell-1)/2^n)$ choices of $T_H \in (\{0,1\}^n)^\varpi$ such that the following is true: for some $0 \le i \le \ell-1$,*

$$A_i \oplus \langle i \rangle_n \neq A_j \oplus \langle j \rangle_n \text{ for all } j \in \{0, \ldots, i-1, i+1, \ldots, \ell-1\}, \text{ and} \tag{3}$$

$$A_i \oplus \langle i \rangle_n \neq A'_j \oplus \langle j \rangle_n \text{ for all } j \in \{0, \ldots, \ell-1\}. \tag{4}$$

Proof. Since $|X| = |X'|$ and $X \neq X'$, we have $\chi_i \neq \chi'_i$ for some i. We show the proof in three cases, (a) $|\chi_{\ell-1}|_n = \varpi$, (b) $|\chi_{\ell-1}|_n < \varpi$ and $0 \le i < \ell - 1$, and (c) $|\chi_{\ell-1}|_n < \varpi$ and $i = \ell - 1$.

We first consider case (a). For any fixed $j \in \{0, \ldots, i-1, i+1, \ldots, \ell-1\}$, the number of T_H that satisfies $A_i \oplus \langle i \rangle_n = A_j \oplus \langle j \rangle_n$ is at most $2^{n\varpi}/2^n$ from Proposition 2. Note that, if $\chi_i = \chi_j$, then there is no T_H that satisfies this condition since $\langle i \rangle_n \oplus \langle j \rangle_n \neq 0^n$. Therefore, we have at most $(\ell-1)2^{n\varpi}/2^n$ values of T_H such that $A_i \oplus \langle i \rangle_n = A_j \oplus \langle j \rangle_n$ holds for some $j \in \{0, \ldots, i-1, i+1, \ldots, \ell-1\}$.

Similarly, the number of T_H which satisfies $A_i \oplus \langle i \rangle_n = A'_j \oplus \langle j \rangle_n$ for some $j \in \{0, \ldots, \ell-1\}$ is at most $\ell 2^{n\varpi}/2^n$. This follows by using Proposition 2 for $j \neq i$, and for $j = i$, we use Proposition 2 and the fact that $\chi_i \neq \chi'_i$.

Therefore, we have at least $2^{n\varpi} - (2\ell-1)2^{n\varpi}/2^n = 2^{n\varpi}(1-(2\ell-1)/2^n)$ choices of $T_H \in (\{0,1\}^n)^\varpi$ which satisfies (3) and (4).

We next consider case (b). From Proposition 2, we have at most $(2\ell-3)2^{n\varpi}/2^n$ values of T_H such that $A_i \oplus \langle i \rangle_n = A_j \oplus \langle j \rangle_n$ for some $j \in \{0, \ldots, i-1, i+1, \ldots, \ell-2\}$, or $A_i \oplus \langle i \rangle_n = A'_j \oplus \langle j \rangle_n$ for some $j \in \{0, \ldots, \ell-2\}$.

From Proposition 3, we have at most $2 \times 2^{n\varpi}/2^n$ values of T_H such that $A_i \oplus \langle i \rangle_n = A_{\ell-1} \oplus \langle \ell-1 \rangle_n$, or $A_i \oplus \langle i \rangle_n = A'_{\ell-1} \oplus \langle \ell-1 \rangle_n$. Note that $\langle i \rangle_n \oplus \langle \ell-1 \rangle_n \neq 0^n$.

Finally, we consider case (c). From Proposition 3, we have at most $(2\ell-2)2^{n\varpi}/2^n$ values of T_H such that $A_{\ell-1} \oplus \langle \ell-1 \rangle_n = A_j \oplus \langle j \rangle_n$ for some $j \in \{0, \ldots, \ell-2\}$, or $A_{\ell-1} \oplus \langle \ell-1 \rangle_n = A'_j \oplus \langle j \rangle_n$ for some $j \in \{0, \ldots, \ell-2\}$.

From Proposition 3 and since $\chi_{\ell-1} \neq \chi'_{\ell-1}$, we have at most $2^{n\varpi}/2^n$ values of T_H such that $A_{\ell-1} \oplus \langle \ell-1 \rangle_n = A'_{\ell-1} \oplus \langle \ell-1 \rangle_n$. □

Proposition 5. *Suppose that $l > l'$ and $\ell > \ell'$. Then there are at least $2^{n\varpi}(1-(\ell+\ell'-1)/2^n)$ choices of $T_H \in (\{0,1\}^n)^\varpi$ such that the following is true:*

$$A_{\ell-1} \oplus \langle \ell-1 \rangle_n \neq A_j \oplus \langle j \rangle_n \text{ for all } j \in \{0, \ldots, \ell-2\}, \text{ and} \tag{5}$$

$$A_{\ell-1} \oplus \langle \ell-1 \rangle_n \neq A'_j \oplus \langle j \rangle_n \text{ for all } j \in \{0, \ldots, \ell'-1\}. \tag{6}$$

Proof. The number is at most $2^{n\varpi}(1-(\ell+\ell'-1)/2^n)$, since if $|\chi_{\ell-1}|_n = \varpi$, the bound follows by using Proposition 2 for each j, and if $|\chi_{\ell-1}|_n < \varpi$, it follows from Proposition 3 and the fact that $\langle \ell-1 \rangle_n \oplus \langle j \rangle_n \neq 0^n$. □

Proposition 6. *Suppose that $l > l'$ and $\ell = \ell'$. Then there are at least $2^{n\varpi}(1 - (\ell + \ell' - 1)/2^n)$ choices of $T_H \in (\{0,1\}^n)^{\varpi}$ which satisfies both (5) and (6).*

Proof. The bound follows by the same argument as in the proof of Proposition 5. The exception is the event $A_{\ell-1} \oplus \langle \ell-1 \rangle_n \neq A'_{\ell-1} \oplus \langle \ell - 1 \rangle_n$, which is equivalent to $A_{\ell-1} \neq A'_{\ell-1}$. In this case, we are interested in the equation $(x_{(\ell-1)\varpi}, \ldots, x_{l-1}) \cdot (T_0, \ldots, T_{l-(\ell-1)\varpi-1}) = (x'_{(\ell-1)\varpi}, \ldots, x'_{l'-1}) \cdot (T_0, \ldots, T_{l'-(\ell-1)\varpi-1})$. We see that the coefficient of $T_{l-(\ell-1)\varpi-1}$ is non-zero (because of padding). Therefore, exactly one value of $T_{l-(\ell-1)\varpi-1}$ satisfies the equality. $\square$

We now consider CIP.Hash that uses a random permutation instead of a blockcipher. Thus, instead of $K_H \overset{R}{\leftarrow} \{0,1\}^k$, we let $P \overset{R}{\leftarrow} \mathrm{Perm}(n)$, and write $\mathrm{CIP.Hash}_{P,T_H}(\cdot)$ instead of $\mathrm{CIP.Hash}_{K_H,T_H}(\cdot)$. Besides, we consider CIP.Hash, where its output bits are truncated to τ bits. The next result proves that this truncated version of CIP.Hash is ϵ-AXU for small ϵ.

Proposition 7. *Let x and x' be two distinct bit strings, where $\ell + \ell' - 1 \leq 2^{n-1}$. For any $1 \leq \tau \leq n$ and any $y \in \{0,1\}^\tau$,*

$$\begin{aligned}&\Pr(P \overset{R}{\leftarrow} \mathrm{Perm}(n), T_H \overset{R}{\leftarrow} (\{0,1\}^n)^{\varpi} : \\ &\qquad \mathsf{first}(\tau, \mathrm{CIP.Hash}_{P,T_H}(x) \oplus \mathrm{CIP.Hash}_{P,T_H}(x')) = y) \leq \frac{\ell + \ell' - 1}{2^n} + \frac{2}{2^\tau}.\end{aligned}$$

Proof. We first choose and fix any T_H. If there is no i such that $A_i \oplus \langle i \rangle_n$ is unique in the multi-set $\{A_0 \oplus \langle 0 \rangle_n, \ldots, A_{\ell-1} \oplus \langle \ell - 1 \rangle_n, A'_0 \oplus \langle 0 \rangle_n, \ldots, A'_{\ell'-1} \oplus \langle \ell' - 1 \rangle_n\}$, then we give up the analysis and regard this as $\mathrm{CIP.Hash}_{P,T_H}(x) \oplus \mathrm{CIP.Hash}_{P,T_H}(x)) = y$ occurs. The probability is at most $(\ell + \ell' - 1)/2^n$ from Proposition 5, 6, and 7, and the first term follows.

Next, we assume for some i, $A_i \oplus \langle i \rangle_n$ is unique in the multi-set. Now since we have fixed T_H, all the inputs to P are now fixed. We next fix the outputs of P except for $A_i \oplus \langle i \rangle_n$. At most $(\ell + \ell' - l)$ input-output pairs are now fixed, and therefore, we have at least $2^n - (\ell + \ell' - l)$ choices for the output of $A_i \oplus \langle i \rangle_n$. Out of these $2^n - (\ell + \ell' - l)$ possible choices, at most $2^{n-\tau}$ values verify $\mathsf{first}(\tau, \mathrm{CIP.Hash}_{K_H,T_H}(x) \oplus \mathrm{CIP.Hash}_{K_H,T_H}(x)) = y$ since the unused $(n - \tau)$ bits may take any value. The probability of this event is at most $2^{n-\tau}/(2^n - (\ell + \ell' - l)) \leq 2/2^\tau$, and the second term follows. $\square$

We now derive the result with a blockcipher E.

Proposition 8. *Let x and x' be two distinct bit strings, where $\ell + \ell' - 1 \leq 2^{n-1}$. For any $1 \leq \tau \leq n$ and any $y \in \{0,1\}^\tau$, there exists a PRP-adversary A for E such that*

$$\begin{aligned}&\Pr(K_H, T_H \overset{R}{\leftarrow} \{0,1\}^k \times (\{0,1\}^n)^{\varpi} : \mathsf{first}(\tau, \mathrm{CIP.Hash}_{K_H,T_H}(x) \\ &\qquad \oplus \mathrm{CIP.Hash}_{K_H,T_H}(x')) = y) \leq \frac{\ell + \ell' - 1}{2^n} + \frac{2}{2^\tau} + \mathbf{Adv}_E^{\mathrm{prp}}(A),\end{aligned}$$

where A makes at most $\ell + \ell'$ queries, and time$(A) = O(n(\ell + \ell'))$.

```
Algorithm CIP.Sim1
Setup:
100    $(K_H, T_H) \overset{R}{\leftarrow} \text{CIP.Key}(R_0)$
If $A$ makes a query $(N_i, M_i)$:
200    $l \leftarrow \lceil |M_i|/n \rceil$
201    $\texttt{ctr} \leftarrow (N_i \| 0^{n-\ell_{\text{nonce}}})$
202    $S \leftarrow \text{CIP.KSGen.Sim1}(\texttt{ctr}, l+1)$
203    $S_H \leftarrow \mathsf{first}(n, S)$
204    $S_{\text{mask}} \leftarrow \mathsf{last}(n \times l, S)$
205    $C_i \leftarrow M_i \oplus \mathsf{first}(|M_i|, S_{\text{mask}})$
206    $\text{Hash}_i \leftarrow \text{CIP.Hash}_{K_H,T_H}(C_i)$
207    $\text{Tag}_i \leftarrow \mathsf{first}(\tau, \text{Hash}_i \oplus S_H)$
208    return $(C_i, \text{Tag}_i)$
```

```
Algorithm CIP.Sim1 (Cont.)
If $A$ returns $(N^*, C^*, \text{Tag}^*)$:
300    $l \leftarrow \lceil |C^*|/n \rceil$
301    $\texttt{ctr} \leftarrow (N^* \| 0^{n-\ell_{\text{nonce}}})$
302    $S \leftarrow \text{CIP.KSGen.Sim1}(\texttt{ctr}, l+1)$
303    $S_H \leftarrow \mathsf{first}(n, S)$
304    $\text{Hash}' \leftarrow \text{CIP.Hash}_{K_H,T_H}(C^*)$
305    $\text{Tag}' \leftarrow \mathsf{first}(\tau, \text{Hash}' \oplus S_H)$
306    if $\text{Tag}' \neq \text{Tag}^*$ then return $\perp$
307    $S_{\text{mask}} \leftarrow \mathsf{last}(n \times l, S)$
308    $M^* \leftarrow C^* \oplus \mathsf{first}(|C^*|, S_{\text{mask}})$
309    return $M^*$
```

Fig. 5. The simulation CIP.Sim1 of CIP. CIP.Hash is defined in Fig. 2.

Proof. Fix x, x' and y, and consider the following A: First, A randomly chooses $T_H \overset{R}{\leftarrow} (\{0,1\}^n)^{\varpi}$. Then A computes the hash values of x and x' following Fig. 2, except that, in lines 105 and 107, blockcipher invocations are replaced with oracle calls. The output of A is 1 iff the xor of their hash values is y. We see that A makes at most $\ell + \ell'$ queries, and

$$\Big| \Pr\Big(\mathsf{first}(\tau, \text{CIP.Hash}_{P,T_H}(x) \oplus \text{CIP.Hash}_{P,T_H}(x')) = y\Big) - \Pr\Big(\mathsf{first}(\tau, \text{CIP.Hash}_{K_H,T_H}(x) \oplus \text{CIP.Hash}_{K_H,T_H}(x')) = y\Big)\Big|$$

is upper bounded by $\mathbf{Adv}_E^{\text{prp}}(A)$. □

6.3 Proof of Theorem 2

We now present the proof of Theorem 2.

Proof (of Theorem 2). First, consider the simulation CIP.Sim1 in Fig. 5 of CIP, where K_H and T_H are generated by using CIP.Key(R_0), i.e., a random function $R_0 \in \text{Func}(n,n)$ is used to encrypt constants, and the keystream generation, CIP.KSGen.Sim1, works as follows: it is exactly the same as Fig. 4, except that it uses a random function $R_1 \in \text{Func}(n, nw)$ instead of F_P^+.

Let $\mathbf{Adv}_{\text{CIP.Sim1}}^{\text{auth}}(A)$ be the success probability of A's forgery, where the oracle is CIP.Sim1, i.e.,

$$\mathbf{Adv}_{\text{CIP.Sim1}}^{\text{auth}}(A) \overset{\text{def}}{=} \Pr(A^{\text{CIP.Sim1}} \text{ forges}),$$

where the probability is taken over the random coins in lines 100, 202, 302 and A's internal coins. We claim that

$$\left|\mathbf{Adv}_{\text{CIP}}^{\text{auth}}(A) - \mathbf{Adv}_{\text{CIP.Sim1}}^{\text{auth}}(A)\right| \tag{7}$$

$$\leq \frac{wr^2\tilde{\sigma}^2}{2^{2n-4}} + \frac{w\tilde{\sigma}^3}{2^{2n-3}} + \frac{r^2}{2^{n+1}} + \frac{w\tilde{\sigma}}{2^n}. \tag{8}$$

To see this, suppose for a contradiction that (7) is larger than (8). Then, by using A as a subroutine, it is possible to construct a PRF-adversary B for $(\mathrm{Perm}(n), F^+)$ making at most r oracle queries to its first oracle and at most $\tilde{\sigma}/w$ oracle queries to its second oracle, where B simulates R_0 and R_1 in Fig. 5 by using its own oracles, and returns 1 if and only if A succeeds in forgery. This implies $\Pr(P \overset{R}{\leftarrow} \mathrm{Perm}(n) : B^{P(\cdot),F_P^+(\cdot)} = 1) = \mathbf{Adv}^{\mathrm{auth}}_{\mathrm{CIP}}(A)$ and $\Pr(R_0 \overset{R}{\leftarrow} \mathrm{Func}(n,n), R_1 \overset{R}{\leftarrow} \mathrm{Func}(n,nw) : B^{R_0(\cdot),R_1(\cdot)} = 1) = \mathbf{Adv}^{\mathrm{auth}}_{\mathrm{CIP.Sim1}}(A)$ and thus, $\mathbf{Adv}^{\mathrm{prf}}_{\mathrm{Perm}(n),F^+}(B)$ is larger than (8), which contradicts Proposition 1.

Now we modify CIP.Sim1 to CIP.Sim2 in Fig. 6.

1. Instead of using CIP.Key in line 100 in Fig. 5, we directly choose (K_H, T_H) randomly.
2. Instead of using CIP.KSGen.Sim1 in line 202 in Fig. 5, we choose an $(l+1)$-block random string. Therefore, we have $S \overset{R}{\leftarrow} \{0,1\}^{n(l+1)}$ in line 201 of Fig. 6. Also, we removed "$\mathtt{ctr} \leftarrow (N_i \| 0^{n-\ell_{\mathrm{nonce}}})$" in line 201 of Fig. 5 because we do not need it.
3. We need a different treatment for a forgery attempt, since we allow the same nonce, i.e., $N^* \in \{N_0, \ldots, N_{q-2}\}$. We make two cases, case $N^* \notin \{N_0, \ldots, N_{q-2}\}$ and case $N^* = N_i$. In the former case, we simply choose a new random S_H in line 301 of Fig. 6. In the latter case, S_H for (N_i, M_i) has to be the same S_H for $(N^*, C^*, \mathrm{Tag}^*)$. Observe that $S_H = \mathrm{Hash}_i \oplus \mathrm{Tag}_i$, and thus, the simulation in line 306 of Fig. 6 is precise. Therefore, the simulation makes no difference in the advantage of A.
4. When A makes a query (N_i, M_i), we return the full n-bit tag, $\mathrm{Tag}_i \in \{0,1\}^n$, instead of a truncated one, while we allow τ-bit tag in the forgery attempt. This only increases the advantage of A.
5. If $\mathrm{Tag}' = \mathrm{Tag}^*$, we return $M^* = C^* \oplus \mathsf{first}(|C^*|, S_{\mathrm{mask}})$. Since the value of M^* has no effect on the advantage (as long as it is not the special symbol $\perp$), we let $M^* \leftarrow 0^{|C^*|}$. This makes no difference in the advantage of A.

Let $\mathbf{Adv}^{\mathrm{auth}}_{\mathrm{CIP.Sim2}}(A) \overset{\mathrm{def}}{=} \Pr(A^{\mathrm{CIP.Sim2}} \text{ forges})$, where the probability is taken over the random coins in lines 100, 201, 301 and A's internal coins. From the above discussion, we have

$$\mathbf{Adv}^{\mathrm{auth}}_{\mathrm{CIP.Sim1}}(A) \le \mathbf{Adv}^{\mathrm{auth}}_{\mathrm{CIP.Sim2}}(A). \tag{9}$$

Now we further modify CIP.Sim2 to CIP.Sim3 in Fig. 7.

1. We do not choose K_H and T_H until we need them (we need them after the forgery attempt).
2. Since C_i is the xor of M_i and a random string of length $|M_i|$, we let $C_i \overset{R}{\leftarrow} \{0,1\}^{|M_i|}$. The distribution of C_i is unchanged, and thus, this makes no difference in the advantage of A.
3. Similarly, since Tag_i includes S_H, which is a truly random string, we let $\mathrm{Tag}_i \overset{R}{\leftarrow} \{0,1\}^n$. The distribution of Tag_i is unchanged, and thus, this makes no difference in the advantage of A (Observe that we do not need K_H and T_H, and we can postpone the selection without changing the distribution of C_i and Tag_i).

```
Algorithm CIP.Sim2
Setup:
100    $K_H \xleftarrow{R} \{0,1\}^k$; $T_H \xleftarrow{R} (\{0,1\}^n)^{\varpi}$
If $A$ makes a query $(N_i, M_i)$:
200    $l \leftarrow \lceil |M_i|/n \rceil$
201    $S \xleftarrow{R} \{0,1\}^{n(l+1)}$
202    $S_H \leftarrow \mathsf{first}(n, S)$
203    $S_{\text{mask}} \leftarrow \mathsf{last}(n \times l, S)$
204    $C_i \leftarrow M_i \oplus \mathsf{first}(|M_i|, S_{\text{mask}})$
205    $\text{Hash}_i \leftarrow \text{CIP.Hash}_{K_H,T_H}(C_i)$
206    $\text{Tag}_i \leftarrow \text{Hash}_i \oplus S_H$
207    return $(C_i, \text{Tag}_i)$
```

```
Algorithm CIP.Sim2 (Cont.)
If $A$ returns $(N^*, C^*, \text{Tag}^*)$:
300    if $N^* \notin \{N_0, \ldots, N_{q-2}\}$ then
301        $S_H \xleftarrow{R} \{0,1\}^n$
302        $\text{Hash}' \leftarrow \text{CIP.Hash}_{K_H,T_H}(C^*)$
303        $\text{Tag}' \leftarrow \mathsf{first}(\tau, \text{Hash}' \oplus S_H)$
304    if $N^* = N_i$ then
305        $\text{Hash}' \leftarrow \text{CIP.Hash}_{K_H,T_H}(C^*)$
306        $\text{Tag}' \leftarrow \text{Hash}' \oplus \text{Hash}_i \oplus \text{Tag}_i$
307        $\text{Tag}' \leftarrow \mathsf{first}(\tau, \text{Tag}')$
308    if $\text{Tag}' \neq \text{Tag}^*$ then return $\perp$
309    $M^* \leftarrow 0^{|C^*|}$
310    return $M^*$
```

Fig. 6. The simulation CIP.Sim2 of CIP

4. If $N^* \in \{N_0, \ldots, N_{q-2}\}$, Tag′ includes the random S_H, and we let $\text{Tag}' \xleftarrow{R} \{0,1\}^\tau$. The distribution of Tag′ is unchanged.
5. If $N^* = N_i$, we need K_H and T_H. We choose them, and the rest is unchanged.

Since the distribution of (C_i, Tag_i) is unchanged, and there is no difference in the advantage of A, we have

$$\mathbf{Adv}^{\text{auth}}_{\text{CIP.Sim2}}(A) = \mathbf{Adv}^{\text{auth}}_{\text{CIP.Sim3}}(A), \tag{10}$$

where $\mathbf{Adv}^{\text{auth}}_{\text{CIP.Sim3}}(A) \overset{\text{def}}{=} \Pr(A^{\text{CIP.Sim3}} \text{ forges})$ and the probability is taken over the random coins in lines 100, 101, 201, 203 and A's internal coins.

We now fix A's internal coins and coins in lines 100 and 101. Then, the query-answer pairs $(N_0, M_0, C_0, \text{Tag}_0), \ldots, (N_{q-2}, M_{q-2}, C_{q-2}, \text{Tag}_{q-2})$ and the forgery attempt (N^*, C^*, Tag^*) are all fixed, and we evaluate $\mathbf{Adv}^{\text{auth}}_{\text{CIP.Sim3}}(A)$ with the coins in lines 201, and 203 only. We evaluate it in the following two cases (Note that we are choosing K_H and T_H *after* fixing $N_i, C_i, \text{Tag}_i, N^*, C^*, \text{Tag}^*$).

– **Case $N^* \notin \{N_0, \ldots, N_{q-2}\}$:** In this case, $\mathbf{Adv}^{\text{auth}}_{\text{CIP.Sim3}}(A) = 1/2^\tau$ since for any fixed Tag^*, $\Pr(\text{Tag}' \xleftarrow{R} \{0,1\}^\tau : \text{Tag}' = \text{Tag}^*) = 1/2^\tau$.
– **Case $N^* = N_i$ and $C^* \neq C_i$:** In this case, we have

$$\begin{aligned}\mathbf{Adv}^{\text{auth}}_{\text{CIP.Sim3}}(A) \leq \Pr(&K_H \xleftarrow{R} \{0,1\}^k, T_H \xleftarrow{R} (\{0,1\}^n)^{\varpi} : \\ &\mathsf{first}(\tau, \text{CIP.Hash}_{K_H,T_H}(C^*) \oplus \text{CIP.Hash}_{K_H,T_H}(C_i)) = y),\end{aligned}$$

where $y = \text{Tag}^* \oplus \text{Tag}_i$. This is at most $(\lceil |C^*|/n \rceil + \lceil |C_i|/n \rceil - 1)/2^n + 2/2^\tau + \mathbf{Adv}^{\text{prp}}_E(D)$ from Proposition 8, and this is upper bounded by $2\sigma/2^n + 2/2^\tau + \mathbf{Adv}^{\text{prp}}_E(D)$, where D makes at most 2σ queries, and $\mathit{time}(D) = O(n\sigma)$.

Therefore, we have

$$\mathbf{Adv}^{\text{auth}}_{\text{CIP.Sim3}}(A) \leq \frac{\sigma}{2^{n-1}} + \frac{2}{2^\tau} + \mathbf{Adv}^{\text{prp}}_E(D). \tag{11}$$

Finally, from (7), (9), (10), and (11), we have (2). □

```
Algorithm CIP.Sim3
If $A$ makes a query $(N_i, M_i)$:
100    $C_i \xleftarrow{R} \{0,1\}^{|M_i|}$
101    $\mathrm{Tag}_i \xleftarrow{R} \{0,1\}^n$
102    return $(C_i, \mathrm{Tag}_i)$
```

```
Algorithm CIP.Sim3 (Cont.)
If $A$ returns $(N^*, C^*, \mathrm{Tag}^*)$:
200    if $N^* \notin \{N_0, \ldots, N_{q-2}\}$ then
201        $\mathrm{Tag}' \xleftarrow{R} \{0,1\}^\tau$
202    if $N^* = N_i$ then
203        $K_H \xleftarrow{R} \{0,1\}^k$; $T_H \xleftarrow{R} (\{0,1\}^n)^\varpi$
204        $\mathrm{Hash}_i \leftarrow \mathrm{CIP.Hash}_{K_H,T_H}(C_i)$
205        $S_H \leftarrow \mathrm{Hash}_i \oplus \mathrm{Tag}_i$
206        $\mathrm{Hash}' \leftarrow \mathrm{CIP.Hash}_{K_H,T_H}(C^*)$
207        $\mathrm{Tag}' \leftarrow \mathrm{Hash}' \oplus S_H$
208        $\mathrm{Tag}' \leftarrow \mathsf{first}(\tau, \mathrm{Tag}')$
209    if $\mathrm{Tag}' \neq \mathrm{Tag}^*$ then return $\perp$
210    $M^* \leftarrow 0^{|C^*|}$
211    return $M^*$
```

Fig. 7. The simulation CIP.Sim3 of CIP

7 Conclusions

We presented an authenticated encryption mode CIP, CENC with Inner Product hash. It has provable security bounds which are better than the usual birthday bound security, and it can be used even when the tag length is short. Our proof is relatively complex, and it would be interesting to see the compact security proofs, possibly by following the "all-in-one" security definition in [25]. It would also be interesting to see schemes with improved security bound and/or efficiency.

Acknowledgement

The author would like to thank anonymous reviewers of Africacrypt 2008 for many insightful and useful comments.

References

1. Bellare, M., Desai, A., Jokipii, E., Rogaway, P.: A concrete security treatment of symmetric encryption. In: Proceedings of The 38th Annual Symposium on Foundations of Computer Science, FOCS 1997, pp. 394–405. IEEE, Los Alamitos (1997)
2. Bellare, M., Guerin, R., Rogaway, P.: XOR MACs: New methods for message authentication using finite pseudorandom functions. In: Coppersmith, D. (ed.) CRYPTO 1995. LNCS, vol. 963, pp. 15–28. Springer, Heidelberg (1995)
3. Bellare, M., Kilian, J., Rogaway, P.: The security of the cipher block chaining message authentication code. JCSS, 61(3), 362–399 (2000); Desmedt, Y.G. (ed.) CRYPTO 1994. LNCS, vol. 839, pp. 341–358. Springer, Heidelberg (1994)
4. Bellare, M., Namprempre, C.: Authenticated encryption: Relations among notions and analysis of the generic composition paradigm. In: Okamoto, T. (ed.) ASIACRYPT 2000. LNCS, vol. 1976, pp. 531–545. Springer, Heidelberg (2000)

5. Bellare, M., Rogaway, P.: Encode-then-encipher encryption: How to exploit nonces or redundancy in plaintexts for efficient cryptography. In: Okamoto, T. (ed.) ASIACRYPT 2000. LNCS, vol. 1976, pp. 317–330. Springer, Heidelberg (2000)
6. Bellare, M., Rogaway, P., Wagner, D.: The EAX mode of operation. In: Roy, B., Meier, W. (eds.) FSE 2004. LNCS, vol. 3017, pp. 389–407. Springer, Heidelberg (2004)
7. Black, J., Rogaway, P.: A block-cipher mode of operation for parallelizable message authentication. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 384–397. Springer, Heidelberg (2002)
8. Gligor, V.G., Donescu, P.: Fast encryption and authentication: XCBC encryption and XECB authentication modes. In: Matsui, M. (ed.) FSE 2001. LNCS, vol. 2355, pp. 92–108. Springer, Heidelberg (2002)
9. Iwata, T.: New blockcipher modes of operation with beyond the birthday bound security. In: Robshaw, M.J.B. (ed.) FSE 2006. LNCS, vol. 4047, pp. 310–317. Springer, Heidelberg (2006), `http://www.nuee.nagoya-u.ac.jp/labs/tiwata/`
10. Iwata, T., Kurosawa, K.: OMAC: One-Key CBC MAC. In: Johansson, T. (ed.) FSE 2003. LNCS, vol. 2887, pp. 129–153. Springer, Heidelberg (2003)
11. Jaulmes, E., Joux, A., Valette, F.: On the security of randomized CBC-MAC beyond the birthday paradox limit: A new construction. In: Daemen, J., Rijmen, V. (eds.) FSE 2002. LNCS, vol. 2365, pp. 237–251. Springer, Heidelberg (2002)
12. Jonsson, J.: On the Security of CTR+CBC-MAC. In: Nyberg, K., Heys, H.M. (eds.) SAC 2002. LNCS, vol. 2595, pp. 76–93. Springer, Heidelberg (2003)
13. Jutla, C.S.: Encryption modes with almost free message integrity. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 529–544. Springer, Heidelberg (2001)
14. Katz, J., Yung, M.: Unforgeable encryption and chosen ciphertext secure modes of operation. In: Schneier, B. (ed.) FSE 2000. LNCS, vol. 1978, pp. 284–299. Springer, Heidelberg (2001)
15. Kohno, T., Viega, J., Whiting, D.: CWC: A high-performance conventional authenticated encryption mode. In: Roy, B., Meier, W. (eds.) FSE 2004. LNCS, vol. 3017, pp. 408–426. Springer, Heidelberg (2004)
16. Lefranc, D., Painchault, P., Rouat, V., Mayer, E.: A generic method to design modes of operation beyond the birthday bound. In: Preproceedings of the 14th annual workshop on Selected Areas in Cryptography, SAC 2007 (2007)
17. Lucks, S.: The two-pass authenticated encryption faster than generic composition. In: Gilbert, H., Handschuh, H. (eds.) FSE 2005. LNCS, vol. 3557, pp. 284–298. Springer, Heidelberg (2005)
18. Luby, M., Rackoff, C.: How to construct pseudorandom permutations from pseudorandom functions. SIAM J. Comput. 17(2), 373–386 (1988)
19. McGrew, D., Viega, J.: The Galois/Counter mode of operation (GCM) (submission to NIST) (2004), `http://csrc.nist.gov/CryptoToolkit/modes/`
20. McGrew, D., Viega, J.: The security and performance of Galois/Counter mode of operation. In: Canteaut, A., Viswanathan, K. (eds.) INDOCRYPT 2004. LNCS, vol. 3348, pp. 343–355. Springer, Heidelberg (2004)
21. Petrank, E., Rackoff, C.: CBC MAC for real-time data sources. Journal of Cryptology 13(3), 315–338 (2000)
22. Rogaway, P.: Nonce-based symmetric encryption. In: Roy, B., Meier, W. (eds.) FSE 2004. LNCS, vol. 3017, pp. 348–358. Springer, Heidelberg (2004)
23. Rogaway, P.: Authenticated-encryption with associated-data. In: Proceedings of the ACM Conference on Computer and Communications Security, ACM CCS 2002, pp. 98–107. ACM, New York (2002)

24. Rogaway, P., Bellare, M., Black, J., Krovetz, T.: OCB: a block-cipher mode of operation for efficient authenticated encryption. ACM Trans. on Information System Security (TISSEC) 6(3), 365–403 (2003); Earlier version in Proceedings of the eighth ACM Conference on Computer and Communications Security, ACM CCS 2001, pp. 196–205, ACM, New York (2001)
25. Rogaway, P., Shrimpton, T.: Deterministic authenticated-encryption: A provable-security treatment of the keywrap problem. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 373–390. Springer, Heidelberg (2006)
26. Wegman, M.N., Carter, J.L.: New hash functions and their use in authentication and set equality. JCSS 22, 256–279 (1981)
27. Whiting, D., Housley, R., Ferguson, N.: Counter with CBC-MAC (CCM) (submission to NIST) (2002), `http://csrc.nist.gov/CryptoToolkit/modes/`

A Figures

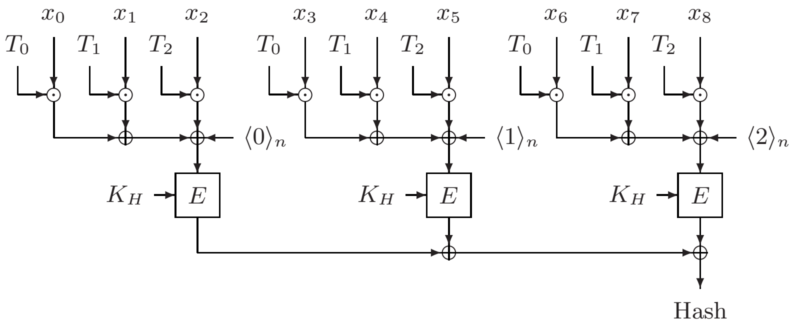

Fig. 8. Illustration of CIP.Hash. This example uses $\varpi = 3$, and $l = 9$.

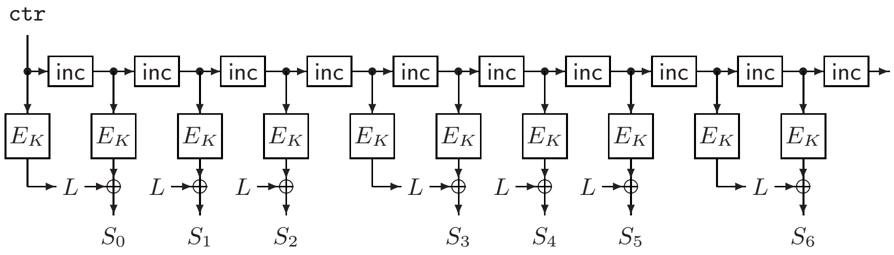

Fig. 9. Illustration of CIP.KSGen. This example uses $w = 3$ and outputs $l = 7$ blocks of keystream $S = (S_0, \ldots, S_6)$.