New Blockcipher Modes of Operation with Beyond the Birthday Bound Security*

Tetsu Iwata

May 20, 2006

Department 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 define and analyze a new blockcipher mode of operation for encryption, CENC, which stands for Cipher-based ENCryption. CENC has the following advantages: (1) beyond the birthday bound security, (2) security proofs with the standard PRP assumption, (3) highly efficient, (4) single blockcipher key, (5) fully parallelizable, (6) allows precomputation of keystream, and (7) allows random access. CENC is based on the new construction of "from PRPs to PRF conversion," which is of independent interest. Based on CENC and a universal hash-based MAC (Wegman-Carter MAC), we also define a new authenticated-encryption with associated-data scheme, CHM, which stands for CENC with Hash-based MAC. The security of CHM is also beyond the birthday bound.

*An extended abstract of this paper appears in *Fast Software Encryption, FSE 2006* [11]. This is the full version.

Contents

1 Introduction

A blockcipher mode of operation, or a mode for short, is an algorithm that provides security goals, such as privacy and/or authenticity, based on blockciphers. The mode for privacy is called an encryption mode.

Of many encryption modes, counter (CTR) mode has a number of desirable advantages, and it works as follows. Let E be a blockcipher whose block length is n bits, and let $\mathtt{ctr}$ be an n-bit counter. For a plaintext $M = (M_0, \ldots, M_{l-1})$ broken into n-bit blocks, let

$$\begin{cases} C_i \leftarrow M_i \oplus S_i, \text{ where } S_i \leftarrow E_K(\mathtt{ctr} + i) \text{ for } 0 \le i \le l-1, \\ \mathtt{ctr} \leftarrow \mathtt{ctr} + l. \end{cases}$$

The ciphertext is $C = (C_0, \ldots, C_{l-1})$, and $S = (S_0, \ldots, S_{l-1})$ is the keystream.

Starting from [3], provable security (or reduction-based security) is the standard security goal for modes. For encryption modes, we consider the strong security notion of privacy called "indistinguishability from random strings" from [23], which provably implies the more standard notions given in [1]. In this strong 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 (where $\mathtt{ctr}$ is not considered part of the ciphertext).

For CTR mode, Bellare, Desai, Jokipii and Rogaway were the first who presented the proof of security [1]. The nonce-based treatment of CTR mode was presented by Rogaway [21]. It was proved that, for any adversary against CTR mode, the success probability is at most $0.5\sigma(\sigma - 1)/2^n$ under the assumption that the blockcipher is a secure pseudorandom permutation (PRP), where σ denotes the total ciphertext length in blocks that the adversary obtains. This is the well-known *birthday bound.*

The above analysis is tight. There *is* an adversary that meets the security bound within a constant factor. The adversary simply searches for a collision in the keystream of σ blocks, and guesses the data is the true ciphertext iff there is no collision. It is easy to show that the success probability is at least $0.3\sigma(\sigma - 1)/2^n$. This implies that, as long as $E_K(\cdot)$ is a permutation, there is no hope that CTR mode achieves beyond the birthday bound security.

In this paper, we design a new blockcipher mode of operation for encryption. The goals are: (1) beyond the birthday bound security, (2) security proofs with the standard PRP assumption, (3) highly efficient, (4) single blockcipher key, (5) fully parallelizable, (6) allows precomputation of keystream, and (7) allows random access. The original CTR mode achieves all the above goals except for the first one, while we improve the security of CTR mode without breaking its important advantages. As for the security assumption, we do not use the ideal blockcipher model. For efficiency, the number of blockcipher calls is close to CTR mode, and we avoid using any heavy operations, e.g., re-keying.

Now in CTR mode, it is known that if $E_K(\cdot)$ is a secure pseudorandom function (PRF), then for any adversary the success probability 0, well beyond the birthday bound. Thus the natural approach to achieve beyond the birthday bound security is to construct a secure PRF from PRPs and use the PRF in CTR mode, where the security of PRF must be beyond the birthday bound. There are several such constructions [4, 10, 16, 2]. The first construction, due to Bellare, Krovetz, and Rogaway is called data-dependent re-keying [4]. It was proved that the construction achieves beyond the birthday bound security in the ideal blockcipher model. The truncation construction was analyzed by Hall et. al., and they also considered the order construction [10]. Lucks [16] and Bellare and Impagriazzo [2] independently analyzed the construction $G_K(x) = E_K(x\|0) \oplus E_K(x\|1)$, where $x \in \{0,1\}^{n-1}$. Lucks also considered a more generalized construction where d blockciphers are xor'ed to output an n-bit block, and a multiple key version, $G_{K_1,K_2}(x) = E_{K_1}(x) \oplus E_{K_2}(x)$, where $x \in \{0,1\}^n$ [16].

By using these constructions in CTR mode, it is possible to construct encryption modes with beyond the birthday bound. However, there is a significant restriction in efficiency, and/or it breaks several important advantages of the original CTR mode. For example, if the construction from [4] is used, we need the ideal blockcipher model for security proofs and have the efficiency problem for re-keying. The constructions from [10] are not very efficient and the truncation construction has relatively small security improvement. If $G_K(x) = E_K(x\|0) \oplus E_K(x\|1)$ is used, $2l$ blockcipher calls are needed to encrypt l plaintext blocks. We see that the main reason for inefficiency is that the output size of these PRFs is one block (or less).

To achieve beyond the birthday bound security, we first show a new "from PRPs to PRF conversion," where the output size of the new PRF is *larger* than the block size. In particular, our PRF outputs w blocks at a time by using $w+1$ blockcipher calls. The parameter, w, is called a frame width, and one frame is equivalent to nw bits. The frame width, w, can be any fixed positive integer. We prove that the adversary's success probability is at most $w\sigma^3/2^{2n-3} + w\sigma/2^n$, where σ is the total number of blocks that the adversary obtains.

Based on the PRF, we show a new encryption mode with beyond the birthday bound security. The new mode is called CENC, which stands for Cipher-based ENCryption. CENC calls $l + \lceil l/w \rceil$ blockciphers to encrypt l plaintext blocks, and the default value is $w = 2^8$, i.e., we need $l + \lceil l/256 \rceil$ blockcipher calls to encrypt l plaintext blocks. Notice that, with the AES ($n = 128$), one frame corresponds to nw bits, which is $128 \times 256 = 4$KBytes, and almost all the traffic on the Internet fits in one frame [8]. This implies we need $l+1$ blockcipher calls for these short data, i.e., the cost is *one* blockcipher call per data compared to CTR mode. As for the security, with $w = 2^8$ and the AES, the security bound of CENC is $\hat{\sigma}^3/2^{248} + \hat{\sigma}/2^{121}$, where $\hat{\sigma}$ is (roughly) the total number of blocks that the adversary obtains. The security of CENC is beyond the birthday bound with the standard PRP assumption. Besides, CENC has desirable advantages of CTR mode. It uses single blockcipher key, it is fully parallelizable, allows precomputation of keystream, and random access is possible.

An authenticated-encryption with associated-data scheme, or AEAD scheme, is a scheme for both privacy and authenticity. It takes a plaintext M and a header H, and provides privacy for M and authenticity for both M and H. There are a number of proposals: we have IAPM [13], OCB mode [23], CCM mode [25, 12], EAX mode [7], CWC mode [15], GCM mode [19, 20], and CCFB mode [17]. Based on CENC and a universal hash-based MAC (Wegman-Carter MAC), we propose a new AEAD scheme called CHM, which stands for CENC with Hash-based MAC. We show that the security of CHM is beyond the birthday bound, which is the first example in literature. The scheme is similar to GCM, achieves higher security with small efficiency loss. It also fixes several undesirable properties of GCM (for example, GCM is not online in the sense that headers must be MACed before starting MAC the ciphertext, and the plaintext length is limited to 64GBytes when used with the AES).

2 Preliminaries

Notation. If x is a string then $|x|$ denotes its length in bits. 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 $x \| y$ denotes 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 w, $(\{0,1\}^n)^w$ is the set of all strings of nw bits, and $\{0,1\}^*$ is the set of all strings (including the empty string). For positive integers n and m such that $n \le 2^m - 1$, $[n]_m$ 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.

Blockciphers and function families. The blockcipher (permutation family) is a function $E : \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$, where, for any $K \in \mathcal{K}$, $E(K, \cdot) = E_K(\cdot)$ is a permutation on $\{0,1\}^n$. The positive integer n is the block length and an n-bit string is called a block. If $\mathcal{K} = \{0,1\}^k$, then k is the key length.

The PRP notion for blockciphers was introduced in [18] and later made concrete in [3]. Let $\mathrm{Perm}(n)$ denote the set of all permutations on $\{0,1\}^n$. This set can be regarded as a blockcipher by considering that each permutation is specified by a unique string. P is a random permutation if $P \overset{R}{\leftarrow} \mathrm{Perm}(n)$. An adversary is a probabilistic algorithm (a program) 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 we define

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

Similarly, the function family is a function $F : \mathcal{K} \times \{0,1\}^m \to \{0,1\}^n$, where, for any $K \in \mathcal{K}$, $F(K, \cdot) = F_K(\cdot)$ is a function from $\{0,1\}^m$ to $\{0,1\}^n$. Let $\mathrm{Func}(m, n)$ denote the set of all functions from $\{0,1\}^m$ to $\{0,1\}^n$. This set can be regarded as a function family by considering that each function in $\mathrm{Func}(m, n)$ is specified by a unique string. R is a random function if $R \overset{R}{\leftarrow} \mathrm{Func}(m, n)$. Let A be an adversary with access to an oracle, either $F_K(\cdot)$ or a random function oracle $R(\cdot)$, and returns a bit. We say A is a *PRF-adversary* for F, and we define

$$\mathbf{Adv}^{\mathrm{prf}}_{F}(A) \overset{\mathrm{def}}{=} \left| \Pr(K \overset{R}{\leftarrow} \mathcal{K} : A^{F_K(\cdot)} = 1) - \Pr(R \overset{R}{\leftarrow} \mathrm{Func}(m, n) : A^{R(\cdot)} = 1) \right| .$$

For an adversary A, A's running time is denoted by $\mathit{time}(A)$. 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.

The frame, nonce, and counter. The modes described in this paper take a positive integer w as a parameter, and it is called a frame width. For fixed positive integer w (say, $w = 2^8$), a w-block string is called a frame. Throughout this paper, we assume $w \ge 1$. A nonce N is a bit string, where for each pair of key and plaintext, it is used only once.

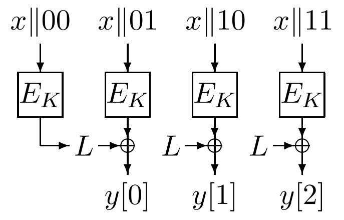

Figure 1: Example illustration of F. In this example, $w = 3$, $\omega = 1 + \lfloor \log_2 w \rfloor = 2$, and $F : \{0,1\}^k \times \{0,1\}^{n-2} \to (\{0,1\}^n)^3$ where $F_K(x) = (y[0], y[1], y[2])$. Here $x \in \{0,1\}^{n-2}$, $y[0] = L \oplus E_K(x\|01)$, $y[1] = L \oplus E_K(x\|10)$, $y[2] = L \oplus E_K(x\|11)$, where $L = E_K(x\|00)$.

The length of the nonce is denoted by ℓ_{nonce}, and it is at most the block length. We also use an n-bit string called a counter, `ctr`. This value is initialized based on the value of the nonce, then it is incremented after each blockcipher invocations. The function for increment is denoted by $\mathsf{inc}(\cdot)$. It takes an n-bit string x (possibly 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 $x \neq 0^n$. For $i > 0$, $\mathsf{inc}^i(\mathtt{ctr})$ means `ctr` is incremented for i times. Since the value is initialized based on the value of the nonce, there is no need to maintain this value across the massages.

3 The Basic Tool: A New Pseudorandom Function F

In this section, we define a new function family F. It takes two parameters, a blockcipher, and a frame width.

Fix the blockcipher $E : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$, and the frame width w. Define $\omega = 1 + \lfloor \log_2 w \rfloor$, i.e., we need ω bits to represent w. Now we define the function family $F : \{0,1\}^k \times \{0,1\}^{n-\omega} \to (\{0,1\}^n)^w$ as $F_K(x) = (y[0], \ldots, y[w-1])$, where $y[i] = L \oplus E_K(\mathsf{inc}^{i+1}(x\|[0]_\omega))$ for $i = 0, \ldots, w-1$ and $L = E_K(x\|[0]_\omega)$. We call L a mask. See Figure 1 for an example.

We have the following information theoretic result on F.

Theorem 1 *Let Perm*(n) *and* w *be the parameters for* F*. Let* A *be a PRF-adversary for* F *making at most* q *oracle queries. Then*

$$\mathbf{Adv}_F^{\text{prf}}(A) \le \frac{(w+1)^4 q^3}{2^{2n+1}} + \frac{w(w+1)q}{2^{n+1}}.$$

Notice that w is a constant and the security bound of Theorem 1 is "beyond the birthday bound." Also, if we set $\sigma = qw$ (i.e., the total number of blocks that the adversary obtains) and measure the security bound in terms of σ, we have $\mathbf{Adv}_F^{\text{prf}}(A) \le w\sigma^3/2^{2n-3} + w\sigma/2^n$, since $1 + w \le 2w$.

The following definition is useful in proving Theorem 1.

Definition 1 *Let* $x = (x_0, \ldots, x_{q-1}) \in (\{0,1\}^{n-\omega})^q$ *be an arbitrary* $(n-\omega)q$*-bit string. We say that* "x is distinct," *if* $x_i \neq x_j$ *for* $0 \le i < j \le q-1$*. Similarly, let* $Y = (Y_0, \ldots, Y_{q-1}) \in (\{0,1\}^{nw})^q$ *be an arbitrary* nqw*-bit string, where* $Y_i = (y_i[0], \ldots, y_i[w-1]) \in (\{0,1\}^n)^w$ *for* $0 \le i \le q-1$*. We say that* "Y is non-zero-distinct," *if there is no equal bit strings in* $\{0^n, y_i[0], \ldots, y_i[w-1]\}$ *for any* i *s.t.* $0 \le i \le q-1$*.*

Note that 0^n is included in the definition for "Y is non-zero-distinct." Suppose that $F_K(x_i) = (y_i[0], \ldots, y_i[w-1])$. Then we always have $y_i[j] \neq 0^n$, and we also see that $y_i[j] \neq y_i[j']$ for $j \neq j'$. We allow, for example, $y_i[j] = y_{i'}[j']$ for $i \neq i'$. Intuitively, Definition 1 is the set of possible input-output pairs, and for these pairs the following lemma, which will be used in the proof of Theorem 1, shows that the distribution is close to uniform. This is the crucial observation for the security improvement. There are no collisions in "one frame," but the collision occurs across the frames.

Lemma 1 *Let $x = (x_0, \ldots, x_{q-1}) \in (\{0,1\}^{n-\omega})^q$ and $Y = (Y_0, \ldots, Y_{q-1}) \in (\{0,1\}^{nw})^q$ be arbitrarily fixed bit strings, where x is distinct and Y is non-zero-distinct. Then*

$$\frac{p_F}{p_R} \geq 1 - \frac{q^3(w+1)^4}{2^{2n+1}}, \tag{1}$$

where $p_F \overset{\text{def}}{=} \Pr(P \overset{R}{\leftarrow} \mathit{Perm}(n) : F_P(x_i) = Y_i$ for $0 \leq i \leq q-1)$ and $p_R \overset{\text{def}}{=} \Pr(R \overset{R}{\leftarrow} \mathit{Func}(n-\omega, nw) : R(x_i) = Y_i$ for $0 \leq i \leq q-1)$.

The proof is based on the counting argument.

Proof (of Lemma 1). We first count the number of $P \in \mathrm{Perm}(n)$ which satisfies $F_P(x_i) = Y_i$ for $0 \leq i \leq q-1$. Let $L_0, \ldots, L_{q-1}$ be n-bit variables. Then the number of $L_0, \ldots, L_{q-1}$ which satisfy $\{L_i, L_i \oplus y_i[0], \ldots, L_i \oplus y_i[w-1]\} \cap \{L_j, L_j \oplus y_j[0], \ldots, L_j \oplus y_j[w-1]\} = \emptyset$ for any $0 \leq i < j \leq q-1$ is at least $\prod_{0 \leq i \leq q-1}(2^n - i(w+1)^2)$, since there are 2^n possibilities for L_0, and once $L_0, \ldots, L_{i-1}$ are fixed, we have at least $2^n - i(w+1)^2$ possibilities for L_i. If we set $L_i = P(x_i \| [0]_\omega)$, then it is possible to set $P(\mathsf{inc}(x_i \| [0]_\omega)) = L_i \oplus y_i[0], \ldots, P(\mathsf{inc}^w(x_i \| [0]_\omega)) = L_i \oplus y_i[w-1]$ uniquely. We have fixed $q(w+1)$ input-output pairs of P, and the remaining $2^n - q(w+1)$ entries can be any value. Therefore, the number of $P \in \mathrm{Perm}(n)$ which satisfies $F_P(x_i) = Y_i$ for $0 \leq i \leq q-1$ is at least $(2^n - q(w+1))! \prod_{0 \leq i \leq q-1}(2^n - i(w+1)^2)$.

Then, the left hand side of (1) is at least

$$\begin{aligned}
&\frac{(2^n)^{qw}(2^n - q(w+1))! \prod_{0 \leq i \leq q-1}(2^n - i(w+1)^2)}{(2^n)!} \\
&\geq \prod_{0 \leq i \leq q-1} \frac{1 - \frac{i(w+1)^2}{2^n}}{\left(1 - \frac{i(w+1)}{2^n}\right)\left(1 - \frac{i(w+1)+1}{2^n}\right) \cdots \left(1 - \frac{i(w+1)+w}{2^n}\right)} \\
&\geq \prod_{0 \leq i \leq q-1} \left(1 - \frac{i(w+1)^2}{2^n}\right)\left(1 + \frac{i(w+1)^2}{2^n} + \frac{w(w+1)}{2^{n+1}}\right).
\end{aligned} \tag{2}$$

We have used the fact that $(1-\alpha)^{-1} \geq 1 + \alpha$ for $|\alpha| < 1$, and the right hand side of (1) is given by simplifying (2). $\square$

We present the proof of Theorem 1 using Lemma 1.

Proof (of Theorem 1). Without loss of generality, we assume that A makes exactly q oracle queries and A does not repeat an oracle query. Also, since A is computationally unbounded, we assume that A is deterministic. Now we can regard A as a function $f_A : (\{0,1\}^{nw})^q \to \{0,1\}$. To see this, let $Y = (Y_0, \ldots, Y_{q-1})$ be an arbitrary nqw-bit string, where each Y_i is nw bits. The first query, x_0, is determined by A. If we return Y_{i-1} as the answer for x_{i-1}, the next query x_i is determined, and finally, if we return Y_{q-1} as the answer for x_{q-1}, the output of A, either 0 or 1, is determined. Therefore, the output of A and the q queries, $x_0, \ldots, x_{q-1}$, are all determined by fixing Y. Note that for any Y, the corresponding sequence of queries $x = (x_0, \ldots, x_{q-1})$ is distinct. Let $\mathbf{v}_{\text{one}} = \{Y \in (\{0,1\}^{nw})^q \mid f_A(Y) = 1\}$, and $\mathbf{v}_{\text{dist}} = \{Y \in (\{0,1\}^{nw})^q \mid Y \text{ is non-zero-distinct}\}$. Observe that $|\mathbf{v}_{\text{dist}}| = ((2^n-1)(2^n-2)\cdots(2^n-w))^q \geq 2^{nwq}(1 - qw(w+1)/2^{n+1})$, and therefore, we have

$$|\mathbf{v}_{\text{one}} \cap \mathbf{v}_{\text{dist}}| \geq |\mathbf{v}_{\text{one}}| - 2^{nwq}qw(w+1)/2^{n+1}. \tag{3}$$

Let $P_R \overset{\text{def}}{=} \Pr(R \overset{R}{\leftarrow} \text{Func}(n-\omega, nw) : A^{R(\cdot)} = 1)$. Then we have

$$P_R = \sum_{Y \in \mathbf{v}_{\text{one}}} p_R = \frac{|\mathbf{v}_{\text{one}}|}{(2^{nw})^q}. \tag{4}$$

On the other hand, let $P_F \overset{\text{def}}{=} \Pr(P \overset{R}{\leftarrow} \text{Perm}(n) : A^{F_P(\cdot)} = 1)$. Then

$$P_F = \sum_{Y \in \mathbf{v}_{\text{one}}} p_F \geq \sum_{Y \in (\mathbf{v}_{\text{one}} \cap \mathbf{v}_{\text{dist}})} p_F \geq \left(1 - \frac{q^3(w+1)^4}{2^{2n+1}}\right) \sum_{Y \in (\mathbf{Y}_{\text{one}} \cap \mathbf{Y}_{\text{dist}})} \frac{1}{(2^{nw})^q}$$

where the last inequality follows from Lemma 1. Then P_F is at least

$$\left(1 - \frac{q^3(w+1)^4}{2^{2n+1}}\right) \frac{|\mathbf{v}_{\text{one}} \cap \mathbf{v}_{\text{dist}}|}{(2^{nw})^q} \geq \left(1 - \frac{q^3(w+1)^4}{2^{2n+1}}\right) \left(P_R - \frac{qw(w+1)}{2^{n+1}}\right)$$

from (3) and (4). Now, we have $P_F \geq P_R - q^3(w+1)^4/2^{2n+1} - qw(w+1)/2^{n+1}$, and by applying the same argument to $1 - P_F$ and $1 - P_R$, we have $1 - P_F \geq 1 - P_R - q^3(w+1)^4/2^{2n+1} - qw(w+1)/2^{n+1}$. □

4 A Relaxed Version F^+

In F, if the input is x, then the mask is always generated with $x \| [0]_\omega$. In this section, we present a slightly relaxed version of F, called F^+, which removes this restriction. Similarly to F, F^+ takes two parameters, a blockcipher $E : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$, and a frame width w.

Now the function family $F^+ : \{0,1\}^k \times \{0,1\}^n \to (\{0,1\}^n)^w$ is defined as $F_K^+(x) = (y[0], \ldots, y[w-1])$, where $y[i] = L \oplus E_K(\mathsf{inc}^{i+1}(x))$ for $i = 0, \ldots, w-1$ and $L = E_K(x)$.

Observe that F^+ takes n-bit x as input, and the mask is generated with x. Also, it is not hard to show that F^+ is a good PRF as long as there is no collision in the input to E.

Let A be an adversary that makes at most q oracle queries and let $x_i \in \{0,1\}^n$ denote A's i-th query. Define $X_i = \{x_i, \mathsf{inc}(x_i), \mathsf{inc}^2(x_i), \ldots, \mathsf{inc}^w(x_i)\}$, i.e., X_i is the set of input to E in the i-th query. We say that A is *input-respecting* if $X_i \cap X_j = \emptyset$ for any $0 \leq i < j \leq q-1$, regardless of oracle responses and regardless of A's internal coins.

We have the following information theoretic result on F^+.

Corollary 1 *Let Perm(n) and w be the parameters for F^+. Let A be a PRF-adversary for F^+ making at most q oracle queries, where A is input-respecting. Then*

$$\mathbf{Adv}_{F^+}^{\text{prf}}(A) \leq \frac{(w+1)^4 q^3}{2^{2n+1}} + \frac{w(w+1)q}{2^{n+1}}.$$

The proof is almost the same as that of Theorem 1, and omitted.

```
Algorithm CENC.Enc_K(N, M)
100  ctr ← (N‖0^{n−ℓ_nonce})
101  l ← ⌈|M|/n⌉
102  S ← CENC.KSGen_K(ctr, l)
103  C ← M ⊕ first(|M|, S)
104  return C
```

```
Algorithm CENC.Dec_K(N, C)
200  ctr ← (N‖0^{n−ℓ_nonce})
201  l ← ⌈|C|/n⌉
202  S ← CENC.KSGen_K(ctr, l)
203  M ← C ⊕ first(|C|, S)
204  return M
```

```
Algorithm CENC.KSGen_K(ctr, l)
300  for j ← 0 to ⌈l/w⌉ − 1 do
301      L ← E_K(ctr)
302      ctr ← inc(ctr)
303      for i ← 0 to w − 1 do
304          S_{wj+i} ← E_K(ctr) ⊕ L
305          ctr ← inc(ctr)
306          if wj + i = l − 1 then
307              S ← (S_0‖S_1‖ ··· ‖S_{l−1})
308              return S
```

Figure 2: Definition of the encryption algorithm CENC.Enc (left top), the decryption algorithm CENC.Dec (left bottom), and the keystream generation algorithm CENC.KSGen (right), which is used in both encryption and decryption.

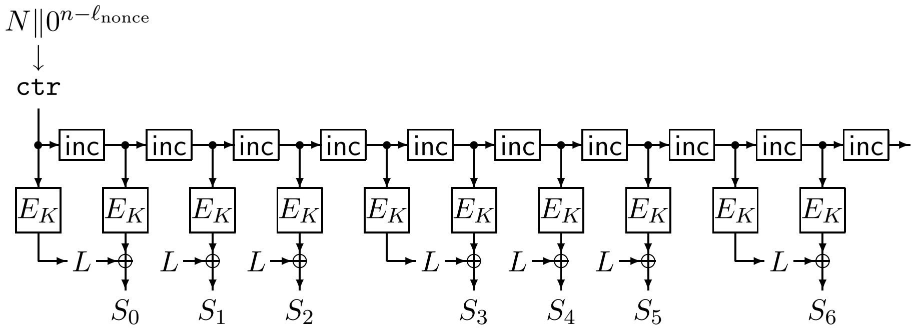

Figure 3: Illustration of the keystream generation algorithm. This example uses $w = 3$ and outputs $l = 7$ blocks of keystream $S = (S_0, \ldots, S_6)$. This S is used in both encryption and decryption. The mask L is updated after generating w blocks of keystream. The counter ctr is incremented for $l + \lceil l/w \rceil = 10$ times, and there are 10 blockcipher invocations.

5 CENC: Cipher-based ENCryption

In this section, we propose a new (nonce-based) encryption scheme, CENC. It takes three parameters, a blockcipher, a nonce length, and a frame width.

Fix the blockcipher $E : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$, the nonce length ℓ_{nonce} and the frame width w, where $1 \le \ell_{\text{nonce}} < n$. CENC consists of two algorithms, the encryption algorithm (CENC.Enc) and the decryption algorithm (CENC.Dec). Both algorithms internally use the keystream generation algorithm (CENC.KSGen). These algorithms are defined in Figure 2. A picture illustrating CENC.KSGen is given in Figure 3.

The encryption algorithm CENC.Enc has the following syntax. CENC.Enc : $\mathsf{Key} \times \mathsf{Nonce} \times \mathsf{Plaintext} \to \mathsf{Ciphertext}$, where Key is $\{0,1\}^k$, Nonce is $\{0,1\}^{\ell_{\text{nonce}}}$, and $\mathsf{Plaintext}$ and $\mathsf{Ciphertext}$ are $\{M \in \{0,1\}^* \mid |M| \le n2^{\ell_{\max}}\}$, i.e., the set of bit strings at most $\ell_{\max}$ blocks, where $\ell_{\max}$ is the largest integer satisfying $\ell_{\max} \le w(2^{n-\ell_{\text{nonce}}} - 1)/(w+1)$. It takes the key K, the nonce N, and the plaintext M to return the ciphertext C. We write $C \leftarrow \text{CENC.Enc}_K(N, M)$. The decryption algorithm CENC.Dec : $\mathsf{Key} \times \mathsf{Nonce} \times \mathsf{Ciphertext} \to \mathsf{Plaintext}$ takes K, N, C to return M. We write $M \leftarrow \text{CENC.Dec}_K(N, C)$. For any K, N, and M, we have $M \leftarrow \text{CENC.Dec}_K(N, \text{CENC.Enc}_K(N, M))$.

CENC.Enc and CENC.Dec call CENC.SKGen to generate the keystream of required length, where the length is in blocks. The encryption (resp. decryption) is just the xor of the plaintext (resp. ciphertext) and the keystream.

The keystream generation algorithm, CENC.KSGen, takes K, the initial counter value $\mathtt{ctr}$, and a non-negative integer l. The output is a keystream S, where the length of S is l blocks. We write $S \leftarrow \text{CENC.KSGen}_K(\mathtt{ctr}, l)$.

In CENC.KSGen, we first generate an n-bit mask, L. $\lceil l/w \rceil$ is the number of frames, incomplete frame counts as one frame. We see that $\lceil l/w \rceil$ masks are generated in line 301. For each mask, w blocks of the keystream are generated in line 304 (except for the last frame, as the last frame may have fewer than w blocks). If l blocks of keystream are generated in line 306, the resulting S is returned in line 308. Observe that the blockcipher is invoked for $l + \lceil l/w \rceil$ times, since we generate $\lceil l/w \rceil$ masks and we have l blocks of keystream, where each block of keystream requires one blockcipher invocation.

Discussion and default parameters. CENC takes the blockcipher $E : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$, the nonce length ℓ_{nonce} ($1 \le \ell_{\text{nonce}} < n$) and the frame width w, as the parameters. With these parameters, CENC can encrypt at most $2^{\ell_{\text{nonce}}}$ plaintexts, and the maximum length of the plaintext is $\ell_{\max}$ blocks. Note that $\ell_{\max}$ is derived by solving $\ell_{\max} + \lceil \ell_{\max}/w \rceil \le 2^{n-\ell_{\text{nonce}}}$ in $\ell_{\max}$, and in general, the bound on $\ell_{\max}$ is $\ell_{\max} \le 2^{n-\ell_{\text{nonce}}-1}$ since $\lceil \ell_{\max}/w \rceil \le \ell_{\max}$. As we will present in Section 6, the security bound of CENC is $(w+1)^4\hat{\sigma}^3/w^3 2^{2n+1} + (w+1)\hat{\sigma}/2^{n+1}$, where $\hat{\sigma}$ is (roughly) the total number of blocks processed by one key.

Our default parameters are, E is any blockcipher such that $n \ge 128$, $\ell_{\text{nonce}} = n/2$, and $w = 2^8 = 256$. For example, if we use the AES, CENC can encrypt at most 2^{64} plaintexts, the maximum length of the plaintext is 2^{63} blocks (2^{37}GBytes), and the security bound is $\hat{\sigma}^3/2^{248} + \hat{\sigma}/2^{121}$ (we used $(w+1)^4/w^3 < 261 < 2^9$), thus $\hat{\sigma}$ should be sufficiently smaller that 2^{82} blocks (2^{56}GBytes).

The frame width, w, should be large enough so that we can implement CENC efficiently. On the other hand, it affects the security bound. We chose $w = 2^8 = 256$, which implies 256 blocks of keystream are generated with 257 blockcipher invocations, thus the cost is about 0.4% compared to CTR mode. We see that the efficiency loss is very small in both software and hardware. Also, the security bound is low enough with this value of w. We do not recommend $w > 2^8$ (when $n = 128$) because of the security loss.

64-bit blockciphers. We do not claim that CENC is generally useful for $n = 64$, since there are restrictions on the nonce length (thus the number of plaintexts), and the plaintext length.

For example, if we use Triple-DES and $(\ell_{\text{nonce}}, w) = (32, 256)$, CENC can encrypt at most 2^{32} plaintexts, and the maximum length of the plaintext is 2^{31} blocks (16GBytes), which may not be enough for general applications (still, it is comparable to CTR mode). In this case, the security bound is $\hat{\sigma}^3/2^{120} + \hat{\sigma}/2^{57}$, which implies $\hat{\sigma}$ should be sufficiently smaller that 2^{40} blocks (2^{13}GBytes).

The limitations of the nonce length and the plaintext length can be removed if we use a counter (instead of a nonce) that is maintained across the plaintexts. This "counter version of CENC” is more suitable for 64-bit blockciphers.

6 Security of CENC

CENC is a symmetric encryption scheme. Before showing the security results on CENC, we first formally define what we mean by symmetric encryption schemes, and what we mean by such schemes to be secure.

Symmetric encryption schemes. A (nonce-based) symmetric encryption scheme is a pair of algorithms $\mathcal{SE} = (\mathcal{E}, \mathcal{D})$ where $\mathcal{E}$ is a deterministic encryption algorithm $\mathcal{E} : \mathsf{Key} \times \mathsf{Nonce} \times \mathsf{Plaintext} \to \mathsf{Ciphertext}$ and $\mathcal{D}$ is a deterministic decryption algorithm $\mathcal{D} : \mathsf{Key} \times \mathsf{Nonce} \times \mathsf{Ciphertext} \to \mathsf{Plaintext}$. The key space Key is a set of keys, and is a nonempty set having a distribution (the uniform distribution when the set is finite). The nonce space Nonce, the plaintext space $\mathsf{Plaintext}$, and the ciphertext space $\mathsf{Ciphertext}$ are nonempty sets of strings. We write $\mathcal{E}_K(N, M)$ for $\mathcal{E}(K, N, M)$ and $\mathcal{D}_K(N, C)$ for $\mathcal{D}(K, N, C)$. We require that $\mathcal{D}_K(N, \mathcal{E}_K(N, M)) = M$ for all $K \in \mathsf{Key}$, $N \in \mathsf{Nonce}$ and $M \in \mathsf{Plaintext}$.

Nonce-respecting adversary. Let A be an adversary with access to an encryption oracle $\mathcal{E}_K(\cdot, \cdot)$. This oracle, on input (N, M), returns the ciphertext $C \leftarrow \mathcal{E}_K(N, M)$. Let $(N_0, M_0), \ldots, (N_{q-1}, M_{q-1})$ denote its oracle queries. The adversary is said to be nonce-respecting if $N_0, \ldots, N_{q-1}$ are always distinct, regardless of oracle responses and regardless of A's internal coins.

Privacy of symmetric encryption schemes. We adopt the strong notion of privacy for nonce-based encryption schemes from [23]. This notion, which we call indistinguishability from random strings, provably implies the more standard notions given in [1].

Let A be an adversary with access to an oracle, either the encryption oracle $\mathcal{E}_K(\cdot, \cdot)$ 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 $|\mathcal{E}_K(N, M)|$. We say that A is a PRIV-adversary for $\mathcal{SE}$. We assume that any PRIV-adversary is nonce-respecting. The advantage of PRIV-adversary A for $\mathcal{SE} = (\mathcal{E}, \mathcal{D})$ having key space Key is

$$\mathbf{Adv}_{\mathcal{SE}}^{\mathrm{priv}}(A) \overset{\mathrm{def}}{=} \left| \Pr(K \overset{R}{\leftarrow} \mathsf{Key} : A^{\mathcal{E}_K(\cdot,\cdot)} = 1) - \Pr(A^{\mathcal{R}(\cdot,\cdot)} = 1) \right| .$$

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

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

$$\mathbf{Adv}_{\mathrm{CENC}}^{\mathrm{priv}}(A) \le \frac{(w+1)^4 \hat{\sigma}^3}{w^3 2^{2n+1}} + \frac{(w+1)\hat{\sigma}}{2^{n+1}}, \tag{5}$$

```
**PRF-adversary** $B$
**If** $A$ **makes a query** $(N_i, M_i)$**:**
100    $\mathtt{ctr} \leftarrow (N_i \| 0^{n-\ell_{\text{nonce}}})$
101    $l \leftarrow \lceil |M_i|/n \rceil$
102    $S \leftarrow \text{CENC.KSGen.Sim}(\mathtt{ctr}, l)$
103    $C_i \leftarrow M_i \oplus \mathsf{first}(|M_i|, S)$
104    **return** $C_i$
**If** $A$ **returns** $b$**:**
200    **output** $b$
```

```
**Algorithm** CENC.KSGen.Sim($\mathtt{ctr}, l$)
300    **for** $j \leftarrow 0$ **to** $\lceil l/w \rceil - 1$ **do**
301        $Y_j \leftarrow \mathcal{O}(\mathtt{ctr})$
302        $\mathtt{ctr} \leftarrow \mathsf{inc}^{w+1}(\mathtt{ctr})$
303    $Y \leftarrow (Y_0, \ldots, Y_{\lceil l/w \rceil - 1})$
304    $Y \leftarrow \mathsf{first}(nl, Y)$
305    **return** $Y$
```

Figure 4: The PRF-adversary B for F^+ based on the PRIV-adversary A for CENC.

where $\hat{\sigma} = \sigma + qw$.

If we use the rough inequality of $w + 1 \le 2w$, then we have the simpler form, $\mathbf{Adv}^{\text{priv}}_{\text{CENC}}(A) \le w\hat{\sigma}^3/2^{2n-3} + w\hat{\sigma}/2^n$.

The proof of Theorem 2 is based on the contradiction argument. If there exists a nonce-respecting PRIV-adversary A such that $\mathbf{Adv}^{\text{priv}}_{\text{CENC}}(A)$ is larger than the right hand side of (5), then we can construct an input-respecting PRF-adversary B for F^+ which contradicts Corollary 1. The proof is given below.

Proof (of Theorem 2). Suppose for a contradiction that $\mathbf{Adv}^{\text{priv}}_{\text{CENC}}(A)$ is larger than the right hand side of (5). Let the oracle $\mathcal{O}$ be either $F^+_P(\cdot)$ or $R(\cdot) \in \text{Func}(n, nw)$. Consider the PRF-adversary B for F^+ in Figure 4, where B uses the nonce-respecting PRIV-adversary A for CENC as a subroutine.

We see that if $\mathcal{O}$ is $F^+_P(\cdot)$, then B gives A a perfect simulation of CENC.Enc, since $F^+_P(\cdot)$ corresponds to “one frame” of CENC.KSGen, and this implies that the outputs of CENC.KSGen.Sim($\mathtt{ctr}, l$) and CENC.KSGen$_P$($\mathtt{ctr}, l$) are the same. Therefore $\Pr(P \xleftarrow{R} \text{Perm}(n) : B^{F^+_P(\cdot)} = 1) = \Pr(P \xleftarrow{R} \text{Perm}(n) : A^{\text{CENC.Enc}_P(\cdot,\cdot)} = 1)$. Also, it is easy to check that B is input-respecting. On the other hand, if $\mathcal{O}$ is $R(\cdot)$, then B gives A a perfect simulation of $\mathcal{R}$. That is, $\Pr(R \xleftarrow{R} \text{Func}(n, nw) : B^{R(\cdot)} = 1) = \Pr(A^{\mathcal{R}(\cdot,\cdot)} = 1)$. Therefore, we have $\mathbf{Adv}^{\text{prf}}_{F^+}(B) = \mathbf{Adv}^{\text{priv}}_{\text{CENC}}(A)$.

Suppose that the queries made by A are $(N_0, M_0), \ldots, (N_{q-1}, M_{q-1})$. If we let $l_i = \lceil |M_i|/n \rceil$, then B makes $\lceil l_0/w \rceil + \cdots + \lceil l_{q-1}/w \rceil$ queries, which is at most $(l_0 + \cdots + l_{q-1})/w + q \le \sigma/w + q = \hat{\sigma}/w$ queries. Note that this holds regardless of the value of $l_0, \ldots, l_{q-1}$. From the assumption for a contradiction, $\mathbf{Adv}^{\text{priv}}_{\text{CENC}}(A)$ is larger than the right hand side of (5), which implies $\mathbf{Adv}^{\text{prf}}_{F^+}(B) > (w+1)^4\hat{\sigma}^3/w^3 2^{2n+1} + (w+1)\hat{\sigma}/2^{n+1}$. This contradicts Corollary 1. □

Given Theorem 2, we have the following complexity theoretic result.

Corollary 2 *Let $E : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$, ℓ_{nonce}, and w be the parameters for CENC. Let A be a nonce-respecting PRIV-adversary for CENC making at most q oracle queries, and the total length of these queries is at most σ blocks. Then there is a PRP-adversary B for E making at most $(w+1)\hat{\sigma}/w$ oracle queries, time(B) = time(A) + $O(n\hat{\sigma}w)$, and $\mathbf{Adv}^{\text{prp}}_{E}(B) \ge \mathbf{Adv}^{\text{priv}}_{\text{CENC}}(A) - w\hat{\sigma}^3/2^{2n-3} - w\hat{\sigma}/2^n$, where $\hat{\sigma} = \sigma + qw$.*

Proof. The proof is standard, and we only check the number of queries made by B. In simulating A's oracle by using B's oracle, suppose that the length of the i-th query made by A is l_i blocks. Then B makes at most $(w+1)(\lceil l_0/w \rceil + \cdots + \lceil l_{q-1}/w \rceil)$ queries, which is at most $(w+1)(\sigma/w + q) = (w+1)\hat{\sigma}/w$ queries. This holds regardless the value of $l_0, \ldots, l_{q-1}$. $\square$

7 CHM: CENC with Hash-based MAC

In this section, we present a new (nonce-based) authenticated-encryption with associated-data (AEAD) scheme, CHM. It takes six parameters, a blockcipher, a nonce length, a tag length, a frame width, and two constants.

Fix the blockcipher $E : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$, the nonce length ℓ_{nonce}, the tag length τ, the frame width w, and two n-bit constants $\texttt{const}_0$ and $\texttt{const}_1$. We require that $1 \le \ell_{\text{nonce}} < n$, $1 \le \tau \le n$, $\texttt{const}_0 \ne \texttt{const}_1$, and $\mathsf{first}(1, \texttt{const}_0) = \mathsf{first}(1, \texttt{const}_1) = 1$ (the most significant bits of $\texttt{const}_0$ and $\texttt{const}_1$ are both 1).

CHM consists of two algorithms, the encryption algorithm (CHM.Enc) and the decryption algorithm (CHM.Dec). These algorithms are defined in Figure 5. Both algorithms use the keystream generation algorithm (CHM.KSGen) and a hash function (CHM.Hash). CHM.KSGen is equivalent to CENC.KSGen defined in Figure 2, and the hash function CHM.Hash is defined in Figure 6.

The syntax of the encryption algorithm is CHM.Enc : $\mathsf{Key} \times \mathsf{Nonce} \times \mathsf{Header} \times \mathsf{Plaintext} \to \mathsf{Ciphertext} \times \mathsf{Tag}$, where the key space Key is $\{0,1\}^k$, the nonce space Nonce is $\{0,1\}^{\ell_{\text{nonce}}}$, and the header space Header is $\{0,1\}^*$. The plaintext space $\mathsf{Plaintext}$ and ciphertext space $\mathsf{Ciphertext}$ are $\{M \in \{0,1\}^* \mid |M| \le n2^{\ell_{\max}}\}$, where $\ell_{\max}$ is the largest integer satisfying $\ell_{\max} \le w(2^{n-\ell_{\text{nonce}}-1} - 1)/(w+1) - 1$. The tag space Tag is $\{0,1\}^\tau$. It takes the key K, the nonce N, the header H, and the plaintext M to return the ciphertext C and the tag T. We write $(C,T) \leftarrow \text{CHM.Enc}_K(N,H,M)$. The decryption algorithm CHM.Dec : $\mathsf{Key} \times \mathsf{Nonce} \times \mathsf{Header} \times \mathsf{Ciphertext} \times \mathsf{Tag} \to \mathsf{Plaintext} \cup \{\mathsf{reject}\}$ takes K, N, H, C and T to return M or a special symbol reject. We write $M \leftarrow \text{CHM.Dec}_K(N,H,C,T)$ or $\mathsf{reject} \leftarrow \text{CHM.Dec}_K(N,H,C,T)$.

CHM is the natural combination of CENC and a universal hash function-based MAC (Wegman-Carter MAC). As a universal hash function, we chose the standard polynomial-based hash, since it is efficient in both software and hardware, and it is well studied. The multiplication is done in the finite field 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 $\mathsf{x}^{128} + \mathsf{x}^7 + \mathsf{x}^2 + \mathsf{x} + 1$.

Discussion and default parameters. CHM takes six parameters, the blockcipher $E : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$, the nonce length ℓ_{nonce}, the tag length τ, the frame width w, and two n-bit constants $\texttt{const}_0$ and $\texttt{const}_1$. With these parameters, CHM can encrypt at most $2^{\ell_{\text{nonce}}}$ plaintext-header pairs, and the maximum length of the plaintext is $\ell_{\max}$ blocks ($\ell_{\max}$ is derived by solving $\ell_{\max} + 1 + \lceil(\ell_{\max}+1)/w\rceil \le 2^{n-\ell_{\text{nonce}}-1}$ in $\ell_{\max}$). As we will present in Section 8, the security bound of CHM is $(w+1)^3\tilde{\sigma}^2/w^2 2^{2n-3} + (w+1)^4\tilde{\sigma}^3/w^3 2^{2n+1} + 1/2^n + (w+1)\tilde{\sigma}/2^{n+1}$ for privacy, and $(w+1)^3\tilde{\sigma}^2/w^2 2^{2n-3} + (w+1)^4\tilde{\sigma}^3/w^3 2^{2n+1} + 1/2^n + (w+1)\tilde{\sigma}/2^{n+1} + (1 + H_{\max} + M_{\max})/2^\tau$ for authenticity, where

```
Algorithm CHM.Enc_K(N, H, M)
100  $S_0 \leftarrow E_K(\mathtt{const}_0)$
101  $S_1 \leftarrow E_K(\mathtt{const}_1)$
102  $l \leftarrow \lceil |M|/n \rceil$
103  $\mathtt{ctr} \leftarrow (0 \| N \| 0^{n-\ell_{\text{nonce}}-1})$
104  $S \leftarrow \text{CHM.KSGen}_K(\mathtt{ctr}, l+1)$
105  $S_2 \leftarrow \mathsf{first}(n, S)$
106  $S_3 \leftarrow \mathsf{last}(nl, S)$
107  $C \leftarrow M \oplus \mathsf{first}(|M|, S_3)$
108  $\text{Hash}_0 \leftarrow \text{CHM.Hash}_{S_0}(C)$
109  $\text{Hash}_1 \leftarrow \text{CHM.Hash}_{S_1}(H)$
110  $T \leftarrow \text{Hash}_0 \oplus \text{Hash}_1 \oplus S_2$
111  $T \leftarrow \mathsf{first}(\tau, T)$
112  return $(C, T)$
```

```
Algorithm CHM.Dec_K(N, H, C, T)
200  $S_0 \leftarrow E_K(\mathtt{const}_0)$
201  $S_1 \leftarrow E_K(\mathtt{const}_1)$
202  $l \leftarrow \lceil |C|/n \rceil$
203  $\mathtt{ctr} \leftarrow (0 \| N \| 0^{n-\ell_{\text{nonce}}-1})$
204  $S \leftarrow \text{CHM.KSGen}_K(\mathtt{ctr}, l+1)$
205  $S_2 \leftarrow \mathsf{first}(n, S)$
206  $\text{Hash}_0 \leftarrow \text{CHM.Hash}_{S_0}(C)$
207  $\text{Hash}_1 \leftarrow \text{CHM.Hash}_{S_1}(H)$
208  $T' \leftarrow \text{Hash}_0 \oplus \text{Hash}_1 \oplus S_2$
209  $T' \leftarrow \mathsf{first}(\tau, T')$
210  if $T' \neq T$ then return reject
211  $S_3 \leftarrow \mathsf{last}(nl, S)$
212  $M \leftarrow C \oplus \mathsf{first}(|C|, S_3)$
213  return $M$
```

Figure 5: Definition of the encryption algorithm CHM.Enc (left), and the decryption algorithm CHM.Dec (right). CHM.KSGen is equivalent to CENC.KSGen in Figure 2, and CHM.Hash is defined in Figure 6.

```
Algorithm CHM.Hash_S(M)
100  $M \leftarrow M \| 10^{n-1-(|M| \bmod n)}$
101  $l \leftarrow |M|/n$
102  $\text{Hash} \leftarrow 0^n$
103  for $i \leftarrow 0$ to $l-1$ do
104      $\text{Hash} \leftarrow (\text{Hash} \oplus M_i) \cdot S$
105  return Hash
```

Figure 6: Definition of CHM.Hash : $\{0,1\}^n \times \{0,1\}^* \to \{0,1\}^n$. M_i is the i-th block of $M \| 10^{n-1-(|M| \bmod n)}$, i.e., $(M_0, \ldots, M_{l-1}) = M \| 10^{n-1-(|M| \bmod n)}$. Multiplication in line 104 is in GF(2^n).

$\tilde{\sigma}$ is (roughly) the total number of blocks processed by one key, $M_{\max}$ is the maximum block length of plaintexts, and $H_{\max}$ is the maximum block length of headers.

Our default parameters are, E is any blockcipher such that $n \geq 128$, $\ell_{\text{nonce}} = n/2 - 1$, $\tau \geq 96$, $w = 2^8 = 256$, $\mathtt{const}_0 = 1^{n-1} \| 0$ and $\mathtt{const}_1 = 1^n$.

With these parameters, if we use the AES, CHM can encrypt at most 2^{63} plaintexts-header pairs, and the maximum length of the plaintext is 2^{63} blocks (2^{37}GBytes), and the security bounds are $\tilde{\sigma}^3/2^{242} + \tilde{\sigma}/2^{120}$ for privacy, and $\tilde{\sigma}^3/2^{242} + \tilde{\sigma}/2^{120} + (1 + H_{\max} + M_{\max})/2^\tau$ for authenticity. This implies $\tilde{\sigma}$ should be sufficiently smaller that 2^{80} blocks (2^{54}GBytes), and $H_{\max}$ and $M_{\max}$ should be small enough so that $(1 + H_{\max} + M_{\max})/2^\tau$ is low enough.

8 Security of CHM

CHM is an authenticated-encryption with associated-data (AEAD) scheme. Before showing the security results on CHM, we first formally define what we mean by AEAD schemes, and what we mean by such schemes to be secure.

An AEAD scheme. A (nonce-based) authenticated-encryption with associated-data (AEAD) scheme is a pair of algorithms $\mathcal{AE} = (\mathcal{E}, \mathcal{D})$ where $\mathcal{E}$ is a deterministic encryption algorithm $\mathcal{E} : \mathsf{Key} \times \mathsf{Nonce} \times \mathsf{Header} \times \mathsf{Plaintext} \to \mathsf{Ciphertext} \times \mathsf{Tag}$ and $\mathcal{D}$ is a deterministic decryption algorithm $\mathcal{D} : \mathsf{Key} \times \mathsf{Nonce} \times \mathsf{Header} \times \mathsf{Ciphertext} \times \mathsf{Tag} \to \mathsf{Plaintext} \cup \{\mathsf{reject}\}$. The key space Key is a set of keys. The nonce space Nonce and the header space Header (also called the space of associated data), the plaintext space $\mathsf{Plaintext}$ and the ciphertext space $\mathsf{Ciphertext}$ are nonempty sets of strings. (We note that there is a more general treatment where $\mathsf{Ciphertext}$ and Tag are not separated. See [7]. We separate them for simplicity.) We write $\mathcal{E}_K(N, H, M)$ for $\mathcal{E}(K, N, H, M)$ and $\mathcal{D}_K(N, H, C, T)$ for $\mathcal{D}(K, N, H, C, T)$. We require that $\mathcal{D}_K(N, H, \mathcal{E}_K(N, H, M)) = M$ for all $K \in \mathsf{Key}$, $N \in \mathsf{Nonce}$, $H \in \mathsf{Header}$ and $M \in \mathsf{Plaintext}$.

Privacy of AEAD schemes. We follow the security notion from [7]. Let A be an adversary with access to an oracle, either the encryption oracle $\mathcal{E}_K(\cdot, \cdot, \cdot)$ or $\mathcal{R}(\cdot, \cdot, \cdot)$, and returns a bit. The $\mathcal{R}(\cdot, \cdot, \cdot)$ oracle, on input (N, H, M), returns a random string of length $|\mathcal{E}_K(N, H, M)|$. We say that A is a PRIV-adversary for $\mathcal{AE}$. We assume that any PRIV-adversary is nonce-respecting (i.e., if $(N_0, H_0, M_0), \ldots, (N_{q-1}, H_{q-1}, M_{q-1})$ is A's oracle queries, $N_0, \ldots, N_{q-1}$ are always distinct, regardless of oracle responses and regardless of A's internal coins). The advantage of PRIV-adversary A for AEAD scheme $\mathcal{AE} = (\mathcal{E}, \mathcal{D})$ having key space Key is

$$\mathbf{Adv}^{\mathrm{priv}}_{\mathcal{AE}}(A) \stackrel{\mathrm{def}}{=} \left| \Pr(K \stackrel{R}{\leftarrow} \mathsf{Key} : A^{\mathcal{E}_K(\cdot,\cdot,\cdot)} = 1) - \Pr(A^{\mathcal{R}(\cdot,\cdot,\cdot)} = 1) \right| .$$

Authenticity of AEAD schemes. A notion of authenticity of ciphertext for AEAD schemes was formalized in [23, 22] following [14, 6, 5]. This time, let A be an adversary with access to an encryption oracle $\mathcal{E}_K(\cdot, \cdot, \cdot)$ and returns a tuple, (N, H, C, T). This tuple is called a forgery attempt. We say that A is an AUTH-adversary for $\mathcal{AE}$. We assume that any AUTH-adversary is nonce-respecting. (The condition is understood to apply 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, H, C, T) such that $\mathcal{D}_K(N, H, C, T) \neq \mathsf{reject}$ but A did not make a query (N, H, M) to $\mathcal{E}_K(\cdot, \cdot, \cdot)$ that resulted in a response (C, T). That is, adversary A may never return a forgery attempt (N, H, C, T) such that the encryption oracle previously returned (C, T) in response to a query (N, H, M). Then the advantage of AUTH-adversary A for AEAD scheme $\mathcal{AE} = (\mathcal{E}, \mathcal{D})$ having key space Key is

$$\mathbf{Adv}^{\mathrm{auth}}_{\mathcal{AE}}(A) \stackrel{\mathrm{def}}{=} \Pr(K \stackrel{R}{\leftarrow} \mathsf{Key} : A^{\mathcal{E}_K(\cdot,\cdot,\cdot)} \text{ forges}).$$

Privacy results on CHM. Let A be a nonce-respecting PRIV-adversary for CHM, and assume that A makes at most q oracle queries, and the total plaintext length of these queries is at most σ blocks, where "the total plaintext length of queries" is defined as follows: if A makes queries $(N_0, H_0, M_0), \ldots, (N_{q-1}, H_{q-1}, M_{q-1})$, then $\sigma = \lceil |M_0|/n \rceil + \cdots + \lceil |M_{q-1}|/n \rceil$, i.e., the total number of blocks of plaintexts. We have the following information theoretic result.

Theorem 3 *Let Perm(n), ℓ_{nonce}, τ, w, $\mathtt{const}_0$ and $\mathtt{const}_1$ be the parameters for CHM. 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*

$$\mathbf{Adv}^{\text{priv}}_{\text{CHM}}(A) \le \frac{(w+1)^3\tilde{\sigma}^2}{w^2 2^{2n-3}} + \frac{(w+1)^4\tilde{\sigma}^3}{w^3 2^{2n+1}} + \frac{1}{2^n} + \frac{(w+1)\tilde{\sigma}}{2^{n+1}}, \tag{6}$$

where $\tilde{\sigma} = \sigma + q(w+1)$.

Note that there is no restriction on the header length. If we use $w+1 \le 2w$, we have the simpler form, $\mathbf{Adv}^{\text{priv}}_{\text{CHM}}(A) \le w\tilde{\sigma}^2/2^{2n-6} + w\tilde{\sigma}^3/2^{2n-3} + 1/2^n + w\tilde{\sigma}/2^n$.

The proof of Theorem 3 is given in Section 9. From Theorem 3, we have the following complexity theoretic result.

Corollary 3 *Let $E : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$, ℓ_{nonce}, τ, w, $\mathtt{const}_0$ and $\mathtt{const}_1$ be the parameters for CHM. 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 $(w+1)\tilde{\sigma}/w$ oracle queries, time$(B) =$ time$(A) + O(n\tilde{\sigma}w)$, and $\mathbf{Adv}^{\text{prp}}_{E}(B) \ge \mathbf{Adv}^{\text{priv}}_{\text{CHM}}(A) - w\tilde{\sigma}^2/2^{2n-6} - w\tilde{\sigma}^3/2^{2n-3} - 1/2^n - w\tilde{\sigma}/2^n$, where $\tilde{\sigma} = \sigma + q(w+1)$.*

Proof. Suppose that the plaintext length of the i-th query made by A is l_i blocks. Then B makes at most $(w+1)(\lceil (l_0+1)/w \rceil + \cdots + \lceil (l_{q-1}+1)/w \rceil)$ queries, which is at most $(w+1)(\sigma/w + q/w + q) = (w+1)\tilde{\sigma}/w$ queries. This holds regardless the value of $l_0, \ldots, l_{q-1}$. The rest of the proof is standard. □

Authenticity results on CHM. Let A be an AUTH-adversary for CHM, and assume that A makes at most q oracle queries (including the final forgery attempt), the total plaintext length of these queries is at most σ blocks, the maximum plaintext length of these queries is at most $M_{\max}$ blocks, and the maximum header length of these queries is at most $H_{\max}$ blocks. Here, if A makes queries $(N_0, H_0, M_0), \ldots, (N_{q-2}, H_{q-2}, M_{q-2})$, and returns the forgery attempt (N^*, H^*, C^*, T^*), then σ, $M_{\max}$ and $H_{\max}$ are defined as

$$\begin{cases} \sigma \overset{\text{def}}{=} \lceil |M_0|/n \rceil + \cdots + \lceil |M_{q-2}|/n \rceil + \lceil |C^*|/n \rceil, \\ M_{\max} \overset{\text{def}}{=} \max\{\lceil |M_0|/n \rceil, \ldots, \lceil |M_{q-2}|/n \rceil, \lceil |C^*|/n \rceil\}, \\ H_{\max} \overset{\text{def}}{=} \max\{\lceil |H_0|/n \rceil, \ldots, \lceil |H_{q-2}|/n \rceil, \lceil |H^*|/n \rceil\}. \end{cases}$$

We say A's query resource is $(q, \sigma, M_{\max}, H_{\max})$. We have the following information theoretic result.

Theorem 4 *Let Perm(n), ℓ_{nonce}, τ, w, $\mathtt{const}_0$ and $\mathtt{const}_1$ be the parameters for CHM. Let A be a nonce-respecting AUTH-adversary whose query resource is $(q, \sigma, M_{\max}, H_{\max})$. Then $\mathbf{Adv}^{\text{auth}}_{\text{CHM}}(A)$ is at most*

$$\frac{(w+1)^3\tilde{\sigma}^2}{w^2 2^{2n-3}} + \frac{(w+1)^4\tilde{\sigma}^3}{w^3 2^{2n+1}} + \frac{1}{2^n} + \frac{(w+1)\tilde{\sigma}}{2^{n+1}} + \frac{1 + H_{\max} + M_{\max}}{2^\tau}, \tag{7}$$

where $\tilde{\sigma} = \sigma + q(w+1)$.

If we use $w+1 \le 2w$, we have the simpler form, $\mathbf{Adv}^{\mathrm{auth}}_{\mathrm{CHM}}(A) \le w\tilde{\sigma}^2/2^{2n-6} + w\tilde{\sigma}^3/2^{2n-3} + 1/2^n + w\tilde{\sigma}/2^n + (1+H_{\max}+M_{\max})/2^{\tau}$.

The proof of Theorem 4 is given in Section 9. From Theorem 4, we have the following complexity theoretic result.

Corollary 4 *Let $E : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$, ℓ_{nonce}, τ, w, $\mathtt{const}_0$, and $\mathtt{const}_1$ be the parameters for CHM. Let A be a nonce-respecting AUTH-adversary whose query resource is $(q,\sigma,M_{\max},H_{\max})$. Then there is a PRP-adversary B for E making at most $(w+1)\tilde{\sigma}/w$ oracle queries, $\mathit{time}(B) = \mathit{time}(A) + O(n\tilde{\sigma}w)$, and $\mathbf{Adv}^{\mathrm{prp}}_{E}(B) \ge \mathbf{Adv}^{\mathrm{auth}}_{\mathrm{CHM}}(A) - w\tilde{\sigma}^2/2^{2n-6} - w\tilde{\sigma}^3/2^{2n-3} - 1/2^n + w\tilde{\sigma}/2^n - (1+H_{\max}+M_{\max})/2^{\tau}$, where $\tilde{\sigma} = \sigma + q(w+1)$.*

The proof is almost the same as that of Corollary 3, and omitted.

9 Security Proofs of CHM

Another security result on the basic tool. To prove Theorem 3 and Theorem 4, consider the function family F^+ defined in Section 4. We show a different security result on F^+.

First, recall the definition of F^+ from Section 4 (but we will concentrate on the information theoretic result). Let $P \overset{R}{\leftarrow} \mathrm{Perm}(n)$ be a random permutation, and fix the frame width w. Let $\omega = 1 + \lfloor \log_2 w \rfloor$. Then $F^+ : \mathrm{Perm}(n) \times \{0,1\}^n \to (\{0,1\}^n)^w$ is $F^+_K(x) = (y[0],\ldots,y[w-1])$, where $y[i] = L \oplus E_K(\mathsf{inc}^{i+1}(x))$ for $i = 0,\ldots,w-1$ and $L = E_K(x)$.

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 $\mathsf{first}(1,W_i) = 1$ must hold.
- 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 $\mathsf{first}(1,x_j) = 0$ must hold. 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.
- A is *input-respecting* with respect to the second oracle (see Section 4).

We say A is *msb-input-respecting* if the above rules hold regardless of oracle responses and regardless of A's internal coins. Define $\mathbf{Adv}^{\mathrm{prf}}_{\mathrm{Perm}(n),F^+}(A)$ as

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

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

We have the following information theoretic result.

Theorem 5 *Let Perm(n) and w be the parameters for F^+. Let A be an msb-input-respecting PRF-adversary for $(Perm(n), F^+)$ making at most r oracle queries to its first oracle and at most q oracle queries to its second oracle. Then*

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

The proof of Theorem 5 is similar to the proof of Theorem 1, but it is not derived directly (and Theorem 1 is not a corollary of Theorem 5 because of the most significant bit restriction). To prove Theorem 5, we need the following lemma.

Lemma 2 *Let $x = (x_0, \ldots, x_{q-1}) \in (\{0,1\}^n)^q$ and $Y = (Y_0, \ldots, Y_{q-1}) \in (\{0,1\}^{nw})^q$ be arbitrarily fixed bit strings, where x is distinct and Y is non-zero-distinct (see Definition 1). Also, let $W = (W_0, \ldots, W_{r-1}) \in (\{0,1\}^n)^r$ and $Z = (Z_0, \ldots, Z_{r-1}) \in (\{0,1\}^n)^r$ be arbitrarily fixed bit strings, where $W_i \neq W_j$ and $Z_i \neq Z_j$ for any $0 \leq i < j \leq r-1$ (i.e., W and Z are distinct). Assume $\{x_0, \ldots, x_{q-1}\} \cap \{W_0, \ldots, W_{r-1}\} = \emptyset$. Then*

$$\frac{p_{F^+}}{p_R} \geq 1 - \frac{r^2q^2(w+1)^3}{2^{2n-1}} - \frac{q^3(w+1)^4}{2^{2n+1}}, \tag{8}$$

where $p_{F^+} \overset{\mathrm{def}}{=} \Pr(P \overset{R}{\leftarrow} Perm(n) : P(W_j) = Z_j$ for $0 \leq j \leq r-1$ and $F^+_P(x_i) = Y_i$ for $0 \leq i \leq q-1)$ and $p_R \overset{\mathrm{def}}{=} \Pr(R_0 \overset{R}{\leftarrow} Func(n,n), R_1 \overset{R}{\leftarrow} Func(n,nw) : R_0(W_j) = Z_j$ for $0 \leq j \leq r-1$ and $R_1(x_i) = Y_i$ for $0 \leq i \leq q-1)$.

Proof. We first count the number of $P \in \mathrm{Perm}(n)$ which satisfies $P(W_j) = Z_j$ for $0 \leq j \leq r-1$ and $F^+_P(x_i) = Y_i$ for $0 \leq i \leq q-1$. Let $Y_i = (y_i[0], \ldots, y_i[w-1]) \in (\{0,1\}^n)^w$ for $0 \leq i \leq q-1$, and let $L_0, \ldots, L_{q-1}$ be n-bit variables. Then the number of $L_0, \ldots, L_{q-1}$ which satisfy

- $\{Z_0, \ldots, Z_{r-1}\} \cap \{L_i, L_i \oplus y_i[0], \ldots, L_i \oplus y_i[w-1]\} = \emptyset$ for any $0 \leq i \leq q-1$, and
- $\{L_i, L_i \oplus y_i[0], \ldots, L_i \oplus y_i[w-1]\} \cap \{L_j, L_j \oplus y_j[0], \ldots, L_j \oplus y_j[w-1]\} = \emptyset$ for any $0 \leq i < j \leq q-1$,

is at least $\prod_{0 \leq i \leq q-1}(2^n - r(w+1) - i(w+1)^2)$, since there are $2^n - r(w+1)$ possibilities for L_0, and once $L_0, \ldots, L_{i-1}$ are fixed, we have at least $2^n - r(w+1) - i(w+1)^2$ possibilities for L_i. If we set $P(W_0) = Z_0, \ldots, P(W_{r-1}) = Z_{r-1}$ and $L_i = P(x_i)$, then it is possible to set $P(\mathsf{inc}(x_i)) = L_i \oplus y_i[0], P(\mathsf{inc}^2(x_i)) = L_i \oplus y_i[1], \ldots, P(\mathsf{inc}^w(x_i)) = L_i \oplus y_i[w-1]$ uniquely. We have fixed $r + q(w+1)$ input-output pairs of P, and the remaining $2^n - r - q(w+1)$ entries can be any value. Therefore, the number of $P \in \mathrm{Perm}(n)$ which satisfies $P(W_j) = Z_j$ for $0 \leq j \leq r-1$ and $F^+_P(x_i) = Y_i$ for $0 \leq i \leq q-1$ is at least $(2^n - r - q(w+1))! \prod_{0 \leq i \leq q-1}(2^n - r(w+1) - i(w+1)^2)$.

Then, the left hand side of (8) is at least

$$\frac{(2^n)^r (2^n)^{qw} (2^n - r - q(w+1))! \prod_{0 \leq i \leq q-1}(2^n - r(w+1) - i(w+1)^2)}{(2^n)!}, \tag{9}$$

and the right hand side of (8) is given by simplifying (9). □

We now present the proof of Theorem 5.

Proof (of Theorem 5). Without loss of generality, we assume that A makes exactly r oracle queries to its first oracle, and exactly q oracle queries to its second oracle, and A does not repeat an oracle query to the same oracle. Also, since A is computationally unbounded, we assume that A is deterministic. Let W_i denote the query for the first oracle, and x_j denote the query for the second oracle. Now we can regard A as a function $f_A : (\{0,1\}^n)^r \times (\{0,1\}^{nw})^q \to \{0,1\}$. To see this, let $Z = (Z_0, \ldots, Z_{r-1})$ be an nr-bit string, and let $Y = (Y_0, \ldots, Y_{q-1})$ be an nqw-bit string, where each Z_i is n bits and Y_i is nw bits. Observe that if we return Z_i as the answer W_i for $0 \le i \le r-1$ and return Y_j as the answer x_j for $0 \le i \le q-1$, the output of A, either 0 or 1, is determined. Therefore, the output of A is determined by fixing Z and Y. Also, note that for any (Z, Y), the corresponding sequence of queries satisfies the conditions in Lemma 2, since A is msb-input-respecting $(\mathsf{first}(1, W_i) \neq \mathsf{first}(1, x_j))$. Let $\mathbf{v}_{\text{one}} = \{(Z, Y) \in (\{0,1\}^n)^r \times (\{0,1\}^{nw})^q \mid f_A(Z, Y) = 1\}$, and $\mathbf{v}_{\text{dist}} = \{(Z, Y) \in (\{0,1\}^n)^r \times \{0,1\}^{nw})^q \mid Z$ is distinct and Y is non-zero-distinct$\}$. Observe that $|\mathbf{v}_{\text{dist}}| = 2^n(2^n - 1) \cdots (2^n - (r-1))((2^n - 1)(2^n - 2) \cdots (2^n - w))^q \ge 2^{n(r+qw)}(1 - r(r-1)/2^{n+1} - qw(w+1)/2^{n+1})$, and therefore, we have

$$|\mathbf{v}_{\text{one}} \cap \mathbf{v}_{\text{dist}}| \ge |\mathbf{v}_{\text{one}}| - 2^{n(r+wq)}(r(r-1)/2^{n+1} + qw(w+1)/2^{n+1}). \tag{10}$$

Let $P_R \overset{\text{def}}{=} \Pr(R_0 \overset{R}{\leftarrow} \text{Func}(n, n), R_1 \overset{R}{\leftarrow} \text{Func}(n, nw) : A^{R_0(\cdot), R_1(\cdot)} = 1)$. Then

$$P_R = \sum_{(Z,Y) \in \mathbf{v}_{\text{one}}} p_R = \frac{|\mathbf{v}_{\text{one}}|}{(2^n)^{r+qw}}. \tag{11}$$

On the other hand, let $P_{F^+} \overset{\text{def}}{=} \Pr(P \overset{R}{\leftarrow} \text{Perm}(n) : A^{P(\cdot), F_P^+(\cdot)} = 1)$. Then we have $P_{F^+} = \sum_{(Z,Y) \in \mathbf{v}_{\text{one}}} p_{F^+} \ge \sum_{(Z,Y) \in (\mathbf{v}_{\text{one}} \cap \mathbf{v}_{\text{dist}})} p_{F^+}$ and therefore,

$$P_{F^+} \ge \left(1 - \frac{r^2 q^2 (w+1)^3}{2^{2n-1}} - \frac{q^3 (w+1)^4}{2^{2n+1}}\right) \sum_{(Z,Y) \in (\mathbf{v}_{\text{one}} \cap \mathbf{v}_{\text{dist}})} \frac{1}{(2^n)^{r+qw}}$$

from Lemma 2 and the fact that $p_R = 1/(2^n)^{r+qw}$. By using (10) and (11),

$$P_{F^+} \quad \ge \quad P_R - \frac{r^2 q^2 (w+1)^3}{2^{2n-1}} - \frac{q^3 (w+1)^4}{2^{2n+1}} - \frac{r(r-1)}{2^{n+1}} - \frac{qw(w+1)}{2^{n+1}}.$$

Finally, we have upper bound on P_{F^+} by applying the same argument to $1 - P_{F^+}$ and $1 - P_R$. This concludes the proof of Theorem 5. □

We now present the proof of Theorem 3.

Proof (of Theorem 3). Suppose for a contradiction that $\mathbf{Adv}^{\text{priv}}_{\text{CHM}}(A)$ is larger than the right hand side of (6). Let the oracles $(\mathcal{O}_0, \mathcal{O}_1)$ be either $(P(\cdot), F_P^+(\cdot))$ or $(R_0(\cdot), R_1(\cdot)) \in \text{Func}(n, n) \times \text{Func}(n, nw)$. Consider the PRF-adversary B for $(\text{Perm}(n), F^+)$ in Figure 7, where B uses A as a subroutine.

First, it is easy to see that B is msb-input-respecting. Next, we see that if $(\mathcal{O}_0, \mathcal{O}_1)$ is $(P(\cdot), F_P^+(\cdot))$, then B gives A a perfect simulation of CHM.Enc_P, and therefore we have $\Pr(P \overset{R}{\leftarrow} \text{Perm}(n) : B^{P(\cdot), F_P^+(\cdot)} = 1) = \Pr(P \overset{R}{\leftarrow} \text{Perm}(n) : A^{\text{CHM.Enc}_P(\cdot,\cdot,\cdot)} = 1)$. On the other hand, if $(\mathcal{O}_0, \mathcal{O}_1)$ is $(R_0(\cdot), R_1(\cdot))$, then B gives A a perfect simulation of $\mathcal{R}$, and we have $\Pr(R_0 \overset{R}{\leftarrow} \text{Func}(n, n), R_1 \overset{R}{\leftarrow} \text{Func}(n, nw) : B^{R_0(\cdot), R_1(\cdot)} = 1) = \Pr(A^{\mathcal{R}(\cdot,\cdot,\cdot)} = 1)$. From the above discussion, we have $\mathbf{Adv}^{\text{prf}}_{\text{Perm}(n), F^+}(B) = \mathbf{Adv}^{\text{priv}}_{\text{CHM}}(A)$.

```
PRF-adversary $B$
Setup:
100    $S_0 \leftarrow \mathcal{O}_0(\texttt{const}_0)$
101    $S_1 \leftarrow \mathcal{O}_0(\texttt{const}_1)$
If $A$ makes a query $(N_i, H_i, M_i)$:
200    $l \leftarrow \lceil |M_i|/n \rceil$
201    $\texttt{ctr} \leftarrow (0 \| N_i \| 0^{n-\ell_{\text{nonce}}-1})$
202    $S \leftarrow \text{CHM.KSGen.Sim}(\texttt{ctr}, l+1)$
203    $S_2 \leftarrow \mathsf{first}(n, S)$
204    $S_3 \leftarrow \mathsf{last}(nl, S)$
205    $C_i \leftarrow M_i \oplus \mathsf{first}(|M_i|, S_3)$
206    $\text{Hash}_0 \leftarrow \text{CHM.Hash}_{S_0}(C_i)$
207    $\text{Hash}_1 \leftarrow \text{CHM.Hash}_{S_1}(H_i)$
208    $T_i \leftarrow \text{Hash}_0 \oplus \text{Hash}_1 \oplus S_2$
209    $T_i \leftarrow \mathsf{first}(\tau, T_i)$
210    return $(C_i, T_i)$
```

```
PRF-adversary $B$ (Cont.)
If $A$ returns $b$:
300    output $b$
```

```
Algorithm CHM.KSGen.Sim($\texttt{ctr}, l$)
400    for $j \leftarrow 0$ to $\lceil l/w \rceil - 1$ do
401        $Y_j \leftarrow \mathcal{O}_1(\texttt{ctr})$
402        $\texttt{ctr} \leftarrow \mathsf{inc}^{w+1}(\texttt{ctr})$
403    $Y \leftarrow (Y_0, \ldots, Y_{\lceil l/w \rceil - 1})$
404    $Y \leftarrow \mathsf{first}(nl, Y)$
405    return $Y$
```

Figure 7: The PRF-adversary B for $(\text{Perm}(n), F^+)$ based on the PRIV-adversary A for CHM.

Observe that B makes $r = 2$ queries to its first oracle, and for the second oracle, assume the queries made by A are $(N_0, H_0, M_0), \ldots, (N_{q-1}, H_{q-1}, M_{q-1})$. If we let $l_i = \lceil |M_i|/n \rceil$, then B makes $\lceil (l_0+1)/w \rceil + \cdots + \lceil (l_{q-1}+1)/w \rceil$ queries, which is at most $(l_0 + \cdots + l_{q-1})/w + q/w + q \le (q+\sigma)/w + q = \tilde{\sigma}/w$ queries. Note that this holds regardless the value of $l_0, \ldots, l_{q-1}$.

From the assumption for a contradiction, $\mathbf{Adv}^{\text{priv}}_{\text{CHM}}(A)$ is larger than the right hand side of (6), which implies $\mathbf{Adv}^{\text{prf}}_{\text{Perm}(n),F^+}(B) > (w+1)^3\tilde{\sigma}^2/w^2 2^{2n-3} + (w+1)^4\tilde{\sigma}^3/w^3 2^{2n+1} + 1/2^n + (w+1)\tilde{\sigma}/2^{n+1}$. This contradicts Theorem 5. □

Before proving Theorem 4, we recall the following well known fact on the property of CHM.Hash [24].

Proposition 1 *Let $x, x' \in \{0,1\}^*$ be two distinct bit strings, and let $l = \lceil |x|/n \rceil$ and $l' = \lceil |x'|/n \rceil$ be their block length. Then for any $\tau \le n$ and any τ-bit string $\texttt{const}_\tau \in \{0,1\}^\tau$, we have $\Pr(S \xleftarrow{R} \{0,1\}^n : \mathit{first}(\tau, \mathit{CHM.Hash}_S(x) \oplus \mathit{CHM.Hash}_S(x')) = \texttt{const}_\tau) \le \max\{l, l'\}/2^\tau$.*

We now present the proof of Theorem 4.

Proof (of Theorem 4). First, consider the simulation CHM.Sim1 in Figure 8 of CHM, where S_0 and S_1 are generated by the random function $R_0 \in \text{Func}(n,n)$, and the keystream generation, CHM.KSGen uses the random function $R_1 \in \text{Func}(n, nw)$.

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

$$\mathbf{Adv}^{\text{auth}}_{\text{CHM.Sim1}}(A) \overset{\text{def}}{=} \Pr(R_0 \xleftarrow{R} \text{Func}(n,n), R_1 \xleftarrow{R} \text{Func}(n,nw) : A^{\text{CHM.Sim1}} \text{ forges}).$$

```
Algorithm CHM.Sim1
Setup:
100   $S_0 \leftarrow R_0(\mathtt{const}_0)$
101   $S_1 \leftarrow R_0(\mathtt{const}_1)$
If $A$ makes a query $(N_i, H_i, M_i)$:
200   $l \leftarrow \lceil |M_i|/n \rceil$
201   $\mathtt{ctr} \leftarrow (0 \| N_i \| 0^{n-\ell_{\text{nonce}}-1})$
202   $S \leftarrow \text{CHM.KSGen.Sim1}(\mathtt{ctr}, l+1)$
203   $S_2 \leftarrow \mathsf{first}(n, S)$
204   $S_3 \leftarrow \mathsf{last}(nl, S)$
205   $C_i \leftarrow M_i \oplus \mathsf{first}(|M_i|, S_3)$
206   $\text{Hash}_0 \leftarrow \text{CHM.Hash}_{S_0}(C_i)$
207   $\text{Hash}_1 \leftarrow \text{CHM.Hash}_{S_1}(H_i)$
208   $T_i \leftarrow \text{Hash}_0 \oplus \text{Hash}_1 \oplus S_2$
209   $T_i \leftarrow \mathsf{first}(\tau, T_i)$
210   return $(C_i, T_i)$
```

```
Algorithm CHM.Sim1 (Cont.)
If $A$ returns $(N^*, H^*, C^*, T^*)$:
300   $l \leftarrow \lceil |C^*|/n \rceil$
301   $\mathtt{ctr} \leftarrow (0 \| N^* \| 0^{n-\ell_{\text{nonce}}-1})$
302   $S \leftarrow \text{CHM.KSGen.Sim1}(\mathtt{ctr}, l+1)$
303   $S_2 \leftarrow \mathsf{first}(n, S)$
304   $\text{Hash}_0 \leftarrow \text{CHM.Hash}_{S_0}(C^*)$
305   $\text{Hash}_1 \leftarrow \text{CHM.Hash}_{S_1}(H^*)$
306   $T' \leftarrow \text{Hash}_0 \oplus \text{Hash}_1 \oplus S_2$
307   $T' \leftarrow \mathsf{first}(\tau, T')$
308   if $T' \neq T^*$ then return reject
309   $S_3 \leftarrow \mathsf{last}(nl, S)$
310   $M^* \leftarrow C^* \oplus \mathsf{first}(|C^*|, S_3)$
311   return $M^*$
```

```
Algorithm CHM.KSGen.Sim1($\mathtt{ctr}, l$)
400   for $j \leftarrow 0$ to $\lceil l/w \rceil - 1$ do
401       $Y_j \leftarrow R_1(\mathtt{ctr})$
402       $\mathtt{ctr} \leftarrow \mathsf{inc}^{w+1}(\mathtt{ctr})$
403   $Y \leftarrow (Y_0, \ldots, Y_{\lceil l/w \rceil - 1})$
404   $Y \leftarrow \mathsf{first}(nl, Y)$
405   return $Y$
```

Figure 8: The simulation CHM.Sim1 of CHM. $R_0 \in \text{Func}(n, n)$ and $R_1 \in \text{Func}(n, nw)$ are random functions. CHM.Hash is defined in Figure 6.

We claim that

$$\left| \mathbf{Adv}^{\text{auth}}_{\text{CHM}}(A) - \mathbf{Adv}^{\text{auth}}_{\text{CHM.Sim1}}(A) \right| \tag{12}$$

$$\leq \frac{(w+1)^3 \tilde{\sigma}^2}{w^2 2^{2n-3}} + \frac{(w+1)^4 \tilde{\sigma}^3}{w^3 2^{2n+1}} + \frac{1}{2^n} + \frac{(w+1)\tilde{\sigma}}{2^{n+1}}. \tag{13}$$

To see this, suppose for a contradiction that (12) is larger than (13). Then, by using A as a subroutine, it is possible to construct an msb-input-respecting PRF-adversary B for $(\text{Perm}(n), F^+)$ making at most 2 oracle queries to its first oracle and at most $\tilde{\sigma}/w$ oracle queries to its second oracle, where B simply simulates R_0 and R_1 in Figure 8 by using its own oracles, and returns 1 if and only if A succeeds in forgery. This implies $\Pr(P \overset{R}{\leftarrow} \text{Perm}(n) : B^{P(\cdot), F^+_P(\cdot)} = 1) = \mathbf{Adv}^{\text{auth}}_{\text{CHM}}(A)$ and $\Pr(R_0 \overset{R}{\leftarrow} \text{Func}(n, n), R_1 \overset{R}{\leftarrow} \text{Func}(n, nw) : B^{R_0(\cdot), R_1(\cdot)} = 1) = \mathbf{Adv}^{\text{auth}}_{\text{CHM.Sim1}}(A)$ and thus, $\mathbf{Adv}^{\text{prf}}_{\text{Perm}(n), F^+}(B)$ is larger than (13), which contradicts Theorem 5.

Now we modify CENC.Sim1 to CENC.Sim2 in Figure 9.

1. Instead of using a random function R_0, we choose two n-bit random strings (lines 100 and 101). This makes no difference in the advantage of A.
2. Similarly, instead of using a random function R_1, we choose an nw-bit random string each time R_1 is called. This implies Y in line 405 of Figure 8 is an nl-bit random

```
Algorithm CHM.Sim2
Setup:
100    $S_0 \xleftarrow{R} \{0,1\}^n$
101    $S_1 \xleftarrow{R} \{0,1\}^n$
If $A$ makes a query $(N_i, H_i, M_i)$:
200    $l \leftarrow \lceil |M_i|/n \rceil$
201    $S \xleftarrow{R} \{0,1\}^{n(l+1)}$
202    $S_2 \leftarrow \mathsf{first}(n, S)$
203    $S_3 \leftarrow \mathsf{last}(nl, S)$
204    $C_i \leftarrow M_i \oplus \mathsf{first}(|M_i|, S_3)$
205    $\mathrm{Hash}_0 \leftarrow \mathrm{CHM.Hash}_{S_0}(C_i)$
206    $\mathrm{Hash}_1 \leftarrow \mathrm{CHM.Hash}_{S_1}(H_i)$
207    $T_i \leftarrow \mathrm{Hash}_0 \oplus \mathrm{Hash}_1 \oplus S_2$
208    $T_i \leftarrow \mathsf{first}(\tau, T_i)$
209    return $(C_i, T_i)$
```

```
Algorithm CHM.Sim2 (Cont.)
If $A$ returns $(N^*, H^*, C^*, T^*)$:
300    if $N^* \notin \{N_0, \ldots, N_{q-1}\}$ then
301        $S_2 \xleftarrow{R} \{0,1\}^n$
302        $\mathrm{Hash}_0 \leftarrow \mathrm{CHM.Hash}_{S_0}(C^*)$
303        $\mathrm{Hash}_1 \leftarrow \mathrm{CHM.Hash}_{S_1}(H^*)$
304        $T' \leftarrow \mathrm{Hash}_0 \oplus \mathrm{Hash}_1 \oplus S_2$
305        $T' \leftarrow \mathsf{first}(\tau, T')$
306    if $N^* = N_i$ then
307        $\mathrm{Hash}_0 \leftarrow \mathrm{CHM.Hash}_{S_0}(C_i)$
308        $\mathrm{Hash}_1 \leftarrow \mathrm{CHM.Hash}_{S_1}(H_i)$
309        $S'_2 \leftarrow \mathsf{first}(\tau, \mathrm{Hash}_0 \oplus \mathrm{Hash}_1) \oplus T_i$
310        $\mathrm{Hash}^*_0 \leftarrow \mathrm{CHM.Hash}_{S_0}(C^*)$
311        $\mathrm{Hash}^*_1 \leftarrow \mathrm{CHM.Hash}_{S_1}(H^*)$
312        $T' \leftarrow \mathsf{first}(\tau, \mathrm{Hash}^*_0 \oplus \mathrm{Hash}^*_1) \oplus S'_2$
313    if $T' \neq T^*$ then return reject
314    $M^* \leftarrow 0^{|C^*|}$
315    return $M^*$
```

Figure 9: The simulation CHM.Sim2 of CHM.

string, and therefore, we have $S \xleftarrow{R} \{0,1\}^{n(l+1)}$ in line 201 of Figure 9. Also, we removed "$\mathtt{ctr} \leftarrow (0\|N_i\|0^{n-\ell_{\text{nonce}}-1})$" in line 201 of Figure 8 because we do not need it.

3. We need a different treatment for a forgery attempt, since we allow the same nonce, i.e., $N^* \in \{N_0, \ldots, N_{q-1}\}$. We make two cases, case $N^* \notin \{N_0, \ldots, N_{q-1}\}$ and case $N^* = N_i$. In the former case, we simply choose a new random S (thus S_2) in line 301 of Figure 9. In the latter case, S_2 for (N_i, H_i, M_i) has to be the same S_2 for (N^*, H^*, C^*, T^*). Observe that S'_2 in line 309 of Figure 9 is the "S_2 for (N_i, H_i, M_i)." Thus, the simulation is precise, and this makes no difference in the advantage of A.

4. If $T' = T^*$, we return $M^* = C^* \oplus \mathsf{first}(|C^*|, S_3)$. Since the value of M^* has no effect on the advantage (as long as it is not the special symbol reject), we return $M^* \leftarrow 0^{|C^*|}$. Again, this makes no difference in the advantage of A.

Let $\mathbf{Adv}^{\mathrm{auth}}_{\mathrm{CHM.Sim2}}(A) \overset{\mathrm{def}}{=} \Pr(A^{\mathrm{CHM.Sim2}}$ forges), where the probability is taken over the random coins in lines 100, 101, 201, 301 and A's internal coins. From the above discussion, we have

$$\mathbf{Adv}^{\mathrm{auth}}_{\mathrm{CHM.Sim1}}(A) = \mathbf{Adv}^{\mathrm{auth}}_{\mathrm{CHM.Sim2}}(A). \tag{14}$$

Now we further modify CENC.Sim2 to CENC.Sim3 in Figure 10.

1. We postpone to choose S_0 and S_1 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 simply let $C_i \xleftarrow{R} \{0,1\}^{|M_i|}$. The distribution of C_i is unchanged, and thus, this makes no difference in the advantage of A.

```
Algorithm CHM.Sim3
If $A$ makes a query $(N_i, H_i, M_i)$:
100    $C_i \xleftarrow{R} \{0,1\}^{|M_i|}$
101    $T_i \xleftarrow{R} \{0,1\}^{\tau}$
102    return $(C_i, T_i)$
```

```
Algorithm CHM.Sim3 (Cont.)
If $A$ returns $(N^*, H^*, C^*, T^*)$:
200    if $N^* \notin \{N_0, \ldots, N_{q-1}\}$ then
201        $T' \xleftarrow{R} \{0,1\}^{\tau}$
202    if $N^* = N_i$ then
203        $S_0 \xleftarrow{R} \{0,1\}^n$
204        $S_1 \xleftarrow{R} \{0,1\}^n$
205        $\mathrm{Hash}_0 \leftarrow \mathrm{CHM.Hash}_{S_0}(C_i)$
206        $\mathrm{Hash}_1 \leftarrow \mathrm{CHM.Hash}_{S_1}(H_i)$
207        $S_2' \leftarrow \mathsf{first}(\tau, \mathrm{Hash}_0 \oplus \mathrm{Hash}_1) \oplus T_i$
208        $\mathrm{Hash}_0^* \leftarrow \mathrm{CHM.Hash}_{S_0}(C^*)$
209        $\mathrm{Hash}_1^* \leftarrow \mathrm{CHM.Hash}_{S_1}(H^*)$
210        $T' \leftarrow \mathsf{first}(\tau, \mathrm{Hash}_0^* \oplus \mathrm{Hash}_1^*) \oplus S_2'$
211    if $T' \neq T^*$ then return reject
212    $M^* \leftarrow 0^{|C^*|}$
213    return $M^*$
```

Figure 10: The simulation CHM.Sim3 of CHM.

3. Similarly, since T_i includes S_2, which is a random string of length τ, we simply let $T_i \xleftarrow{R} \{0,1\}^{\tau}$. The distribution of T_i is unchanged, and thus, this makes no difference in the advantage of A (Observe that we do not need S_0 and S_1, and we can postpone the selection without changing the distribution of C_i and T_i).

4. If $N^* \in \{N_0, \ldots, N_{q-1}\}$, T' includes S_2, which is a random string of length τ, and we simply let $T' \xleftarrow{R} \{0,1\}^{\tau}$. The distribution of T' is unchanged.

5. If $N^* = N_i$, we need S_0 and S_1. We choose them, and the rest is unchanged.

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

$$\mathbf{Adv}^{\mathrm{auth}}_{\mathrm{CHM.Sim2}}(A) = \mathbf{Adv}^{\mathrm{auth}}_{\mathrm{CHM.Sim3}}(A), \tag{15}$$

where $\mathbf{Adv}^{\mathrm{auth}}_{\mathrm{CHM.Sim3}}(A) \stackrel{\mathrm{def}}{=} \Pr(A^{\mathrm{CHM.Sim3}} \text{ forges})$ and the probability is taken over the random coins in lines 100, 101, 201, 203, 204 and A's internal coins.

Consider arbitrarily fixed A's internal coins and coins in lines 100 and 101. Then, the query-answer pairs $(N_0, H_0, M_0, C_0, T_0), \ldots, (N_{q-1}, H_{q-1}, M_{q-1}, C_{q-1}, T_{q-1})$ and the forgery attempt (N^*, H^*, C^*, T^*) are all fixed, and we consider $\Pr(A^{\mathrm{CHM.Sim3}} \text{ forges})$ with the coins in lines 201, 203 and 204 only. We evaluate $\mathbf{Adv}^{\mathrm{auth}}_{\mathrm{CHM.Sim3}}(A)$ in the following three cases (It is important to note that we are choosing S_0 and S_1 *after* fixing $N_i, H_i, C_i, T_i, N^*, H^*, C^*, T^*$).

- **Case** $N^* \notin \{N_0, \ldots, N_{q-1}\}$**:** In this case, $\mathbf{Adv}^{\mathrm{auth}}_{\mathrm{CHM.Sim3}}(A) = 1/2^{\tau}$ since for fixed T^*, $\Pr(T' \xleftarrow{R} \{0,1\}^{\tau} : T' = T^*) = 1/2^{\tau}$.

- **Case** $N^* = N_i$ **and** $C^* \neq C_i$**:** In this case, fix any S_1. Then $\mathbf{Adv}^{\mathrm{auth}}_{\mathrm{CHM.Sim3}}(A)$ is at most $\Pr(S_0 \xleftarrow{R} \{0,1\}^n : \mathsf{first}(\tau, \mathrm{CHM.Hash}_{S_0}(C^*) \oplus \mathrm{CHM.Hash}_{S_0}(C_i)) = \mathtt{const}_{\tau})$, where $\mathtt{const}_{\tau} = \mathsf{first}(\tau, \mathrm{CHM.Hash}_{S_1}(H^*) \oplus \mathrm{CHM.Hash}_{S_1}(H_i)) \oplus T^* \oplus T_i$. This is at most $\max\{|C^*|, |C_i|\}/2^{\tau}$ from Proposition 1.

- **Case $N^* = N_i$ and $H^* \neq H_i$:** This case is similar to the above. We first fix any S_0. Then $\mathbf{Adv}^{\mathrm{auth}}_{\mathrm{CHM.Sim3}}(A)$ is at most $\Pr(S_1 \overset{R}{\leftarrow} \{0,1\}^n : \mathsf{first}(\tau, \mathrm{CHM.Hash}_{S_1}(H^*) \oplus \mathrm{CHM.Hash}_{S_1}(H_i)) = \mathtt{const}_\tau)$. In this case, $\mathtt{const}_\tau = \mathsf{first}(\tau, \mathrm{CHM.Hash}_{S_0}(C^*) \oplus \mathrm{CHM.Hash}_{S_0}(C_i)) \oplus T^* \oplus T_i$. This probability is at most $\max\{|H^*|, |H_i|\}/2^\tau$ from Proposition 1.

Therefore, we have

$$\mathbf{Adv}^{\mathrm{auth}}_{\mathrm{CHM.Sim3}}(A) \leq \frac{\max\{1, H_{\max}, M_{\max}\}}{2^\tau}. \tag{16}$$

Finally, from (12), (13), (14), (15), and (16), we have (7). □

10 Discussions

Counter-based versions. CENC and CHM use a nonce, and it is natural to consider their counter-based versions. Call them CENC-C and CHM-C, respectively. They use an n-bit counter maintained across the plaintexts (usually by the sender). The drawback is the difficulty of implementation and it is relatively harder to use them properly, which is the reason why we have concentrated on the nonce-based schemes. The advantage of CENC-C and CHM-C is that, the nonce length and the maximum plaintext length restrictions are removed, while the security is unchanged (further, non-adaptive version of PRP is enough for the security proofs). The restrictions only come from the security bound (instead of the schemes). Thus, if carefully implemented and properly used, these counter versions are suitable especially for 64-bit blockciphers

Tightness of the security bounds. For CTR mode, the security bound is tight up to a constant factor. However, for CENC and CHM (and the PRF F in Section 3), we do not know the tightness of our security bounds. The tightness is an open question. For example, if we take CENC, the bound is $O(w\hat{\sigma}^3/2^{2n} + w\hat{\sigma}/2^n)$. The question is the existence of an adversary A that breaks the privacy of CENC with about $\hat{\sigma} = 2^{82}$ data (without breaking the pseudorandomness of the AES), *or* the proof that the security is better than the above. We conjecture that the bound of CENC can be improved to $O(w\hat{\sigma}/2^n)$, possibly by using the technique from [2][1].

Acknowledgement

The author would like to thank Kazumaro Aoki, Fumihiko Sano, and Akashi Satoh for useful comments. Part of this paper was written while the author was at Ibaraki University, Japan.

References

[1] M. Bellare, A. Desai, E. Jokipii, and P. Rogaway. A concrete security treatment of symmetric encryption. Proceedings of *The 38th Annual Symposium on Foundations of Computer Science, FOCS '97,* pp. 394–405, IEEE, 1997.

[1]However, it is not possible to check the details of the proof of [2], since only a sketch is given.

[2] M. Bellare, and R. Impagliazzo. A tool for obtaining tighter security analyses of pseudorandom function based constructions, with application to PRP $\rightarrow$ PRF convention. *Cryptology ePrint Archive,* Report 1999/024, Available at `http://eprint.iacr.org/`, 1999.

[3] M. Bellare, J. Kilian, and P. Rogaway. The security of the cipher block chaining message authentication code. *JCSS,* vol. 61, no. 3, pp. 362–399, 2000. Earlier version in *Advances in Cryptology—CRYPTO '94,* LNCS 839, pp. 341–358, Springer-Verlag, 1994.

[4] M. Bellare, T. Krovetz, and P. Rogaway. Luby-Rackoff backwards: Increasing security by making block ciphers non-invertible. *Advances in Cryptology—EUROCRYPT '98,* LNCS 1403, pp. 266–280, Springer-Verlag, 1998.

[5] M. Bellare, and C. Namprempre. Authenticated encryption: Relations among notions and analysis of the generic composition paradigm. *Advances in Cryptology—ASIACRYPT 2000,* LNCS 1976, pp. 531–545, Springer-Verlag, 2000.

[6] M. Bellare, and P. Rogaway. Encode-then-encipher encryption: How to exploit nonces or redundancy in plaintexts for efficient cryptography. *Advances in Cryptology—ASIACRYPT 2000,* LNCS 1976, pp. 317–330, Springer-Verlag, 2000.

[7] M. Bellare, P. Rogaway, and D. Wagner. The EAX mode of operation. *Fast Software Encryption, FSE 2004,* LNCS 3017, pp. 389–407, Springer-Verlag, 2004.

[8] K. Claffy, G. Miller, and K. Thompson. The nature of the beast: Recent traffic measurements from an Internet backbone. Proceedings of *INET '98.* Available at `http://www.caida.org/outreach/papers/1998/Inet98`.

[9] D. Delov, C. Dwork, and M. Naor. Non-malleable cryptography. *SIAM J. Comput.,* vol. 30, no. 2, pp. 391–437, 2000.

[10] C. Hall, D. Wagner, J. Kelsey, and B. Schneier. Building PRFs from PRPs. *Advances in Cryptology—CRYPTO '98,* LNCS 1462, pp. 370–389, Springer-Verlag, 1998.

[11] T. Iwata. New blockcipher modes of operation with beyond the birthday bound security. *Fast Software Encryption, FSE 2006,* LNCS 4047, Springer-Verlag, 2006.

[12] J. Jonsson. On the Security of CTR + CBC-MAC. *Selected Areas in Cryptography, 9th Annual Workshop (SAC 2002),* LNCS 2595, pp. 76–93. Springer-Verlag, 2002.

[13] C.S. Jutla. Encryption modes with almost free message integrity. *Advances in Cryptology—EUROCRYPT 2001,* LNCS 2045, pp. 529–544, Springer-Verlag, 2001.

[14] J. Katz, and M. Yung. Unforgeable encryption and chosen ciphertext secure modes of operation. *Fast Software Encryption, FSE 2000,* LNCS 1978, pp. 284–299, Springer-Verlag, 2000.

[15] T. Kohno, J. Viega, and D. Whiting. CWC: A high-performance conventional authenticated encryption mode. *Fast Software Encryption, FSE 2004,* LNCS 3017, pp. 408–426, Springer-Verlag, 2004.

[16] S. Lucks. The sum of PRPs is a secure PRF. *Advances in Cryptology—EUROCRYPT 2000,* LNCS 1807, pp. 470–484, Springer-Verlag, 2000.

[17] S. Lucks. The two-pass authenticated encryption faster than generic composition. *Fast Software Encryption, FSE 2005,* LNCS 3557, pp. 284–298, Springer-Verlag, 2005.

[18] M. Luby and C. Rackoff. How to construct pseudorandom permutations from pseudorandom functions. *SIAM J. Comput.,* vol. 17, no. 2, pp. 373–386, 1988.

[19] D. McGrew, and J. Viega. The Galois/Counter mode of operation (GCM). Submission to NIST. Available at `http://csrc.nist.gov/CryptoToolkit/modes/`, 2004.

[20] D. McGrew, and J. Viega. The security and performance of Galois/Counter mode of operation. *Progress in Cryptology—INDOCRYPT 2004,* LNCS 3348, pp. 343–355, Springer-Verlag, 2004.

[21] P. Rogaway. Nonce-based symmetric encryption. *Fast Software Encryption, FSE 2004,* LNCS 3017, pp. 348–358, Springer-Verlag, 2004.

[22] P. Rogaway. Authenticated-encryption with associated-data. *Proceedings of the ACM Conference on Computer and Communications Security, ACM CCS 2002,* pp. 98–107, ACM, 2002.

[23] P. Rogaway, M. Bellare, J. Black, and T. Krovetz. OCB: a block-cipher mode of operation for efficient authenticated encryption. *ACM Trans. on Information System Security (TISSEC),* vol. 6, no. 3, pp. 365–403, 2003. Earlier version in *Proceedings of the eighth ACM Conference on Computer and Communications Security, ACM CCS 2001,* pp. 196–205, ACM, 2001.

[24] M.N. Wegman, and J.L. Carter. New hash functions and their use in authentication and set equality. *JCSS,* vol. 22, pp. 256–279, 1981.

[25] D. Whiting, R. Housley, and N. Ferguson. Counter with CBC-MAC (CCM). Submission to NIST. Available at `http://csrc.nist.gov/CryptoToolkit/modes/`, 2002.