A Tweakable Enciphering Mode

Shai Halevi[1] and Phillip Rogaway[2]

[1] IBM T.J. Watson Research Center, Yorktown-Heights, NY 10598, USA
shaih@watson.ibm.com

[2] Dept. of Computer Science, University of California, Davis, CA 95616, USA, and
Dept. of Computer Science, Fac. of Science, Chiang Mai University, 50200, Thailand
rogaway@cs.ucdavis.edu http://www.cs.ucdavis.edu/~rogaway

Abstract. We describe a block-cipher mode of operation, CMC, that turns an n-bit block cipher into a tweakable enciphering scheme that acts on strings of mn bits, where $m \geq 2$. When the underlying block cipher is secure in the sense of a strong pseudorandom permutation (PRP), our scheme is secure in the sense of tweakable, strong PRP. Such an object can be used to encipher the sectors of a disk, in-place, offering security as good as can be obtained in this setting. CMC makes a pass of CBC encryption, xors in a mask, and then makes a pass of CBC decryption; no universal hashing, nor any other non-trivial operation beyond the block-cipher calls, is employed. Besides proving the security of CMC we initiate a more general investigation of tweakable enciphering schemes, considering issues like the non-malleability of these objects.

1 Introduction

Enciphering schemes. Suppose you want to encrypt the contents of a disk, but the encryption is to be performed by a low-level device, such as a disk controller, that knows nothing of higher-level concepts like files and directories. The disk is partitioned into fixed-length sectors and the encrypting device is given one sector at a time, in arbitrary order, to encrypt or decrypt. The device needs to operate on sectors as they arrive, independently of the rest. Each ciphertext must have the same length as its plaintext, typically 512 bytes. When the plaintext disk sector P is put to the disk media at location T what is stored on the media should be a ciphertext $C = \mathcal{E}_K^T(P)$ that depends not only on the plaintext P and the key K, but also on the location T, which we call the *tweak*. Including the dependency on T allows that identical plaintext sectors stored at different places on the disk will have computationally unrelated ciphertexts.

The envisioned attack-model is a chosen plaintext/ciphertext attack: the adversary can learn the ciphertext C for any plaintext P and tweak T, and it can learn the plaintext P for any ciphertext C and tweak T. Informally, we want a *tweakable*, *strong*, *pseudorandom permutation* (PRP) that operates on a *wide blocksize* (like 512 bytes). We call such an object an *enciphering scheme*. We want to construct the enciphering scheme from a standard block cipher, such as AES, giving a *mode of operation*. The problem is one of current interest for standardization [11]. We seek an algorithm that is simple, and is efficient in both hardware and software.

Naor-Reingold approach. Naor and Reingold give an elegant approach for making a strong PRP on N bits from a block cipher on $n < N$ bits [17, 18]. Their *hash–encipher–hash* paradigm involves applying to the input an *invertible blockwise-universal hash-function*, enciphering the result (say in ECB mode), and then applying yet another invertible blockwise-universal hash-function. Their work stops short of fully specifying a mode of operation, but in [17] they come closer, showing how to make the invertible blockwise-universal hash-function out of an xor-universal hash-function. So the problem, one might imagine, is simply to *instantiate* the approach [17], selecting an appropriate xor-universal hash function from the literature.

It turns out not to be so simple. Despite many attempts to construct a desirable hash function to use with the hash–encipher–hash approach, we could find no desirable realization. We wanted a hash function that was simple and more efficient, per byte, across hardware and software, than AES. The collision bound should be about 2^{-128} (degrading with the length of messages). Many techniques were explored, but nothing with the desired constellation of characteristics was ever found. We concluded that while making a wide-blocksize, strong PRP had "in principal" been reduced to a layer of block-cipher calls plus two "cheap" layers of universal hashing, the story, in practice, was that the "cheap" hashing layers would come to dominate the total cost in hardware, software, or both.

Our contributions. Our main contribution is a simple, practical, completely-specified enciphering mode. CMC starts with a block cipher $E: \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$ and turns it into an enciphering scheme CMC[E]: $\mathcal{K}' \times \mathcal{T} \times \mathcal{M} \to \mathcal{M}$ where $\mathcal{T} = \{0,1\}^n$ and $\mathcal{M}$ contains strings with any number (at least two) of n-bit blocks. See Figs. 1 and 2 for a preview. CMC stands for CBC–Mask–CBC.

CMC uses $2m + 1$ block-cipher calls. No "non-elementary" operations are used—in particular, no form of universal hashing is employed. The mode is highly symmetric: deciphering is the same as enciphering except that one uses the inverse block cipher E_K^{-1} in place of E_K. We prove that CMC[E] is secure, in the sense of a tweakable, strong PRP. This assumes that E itself is secure as a strong PRP. The actual results are quantitative, with the usual quadratic degradation in security.

Apart from the specific scheme, we investigate, more generally, the underlying goal. We show that being secure as a tweakable, strong, PRP implies the appropriate versions of indistinguishability [2, 9] and non-malleability [3, 8] under a chosen-ciphertext attack. Following Liskov, Rivest and Wagner [14], we show how tweaks can be cheaply added to the untweaked version of the primitive.

Joux's attack. In an earlier, unpublished, manuscript we described a different version of CMC mode [19]. Although the algorithmic change between the old and new mode is small, its consequences are not: the old mode was *wrong*, as recently shown by Antoine Joux [12]. His simple and clever attack is described in Appendix A. In the same appendix we describe the bug in the proof that corresponds to the attack. This paper fixes the mode and its proof.

Other prior work. Efforts to construct a block cipher with a large blocksize from one with a smaller blocksize go back to Luby and Rackoff [15], whose work can be viewed as building a $2n$-bit block cipher from an n-bit one. They also put forward the notion of a PRP and a strong ("super") PRP. The concrete-security treatment of PRPs begins with Bellare, Kilian, and Rogaway [4]. The notion of a tweakable block-cipher is due to Liskov, Rivest and Wagner [14]. Earlier work by Schroeppel describes a block cipher that was already designed to incorporate a tweak [20]. The first attempt to directly construct an nm-bit block cipher from an n-bit one is due to Zheng, Matsumoto and Imai [21], who give a Feistel-type construction. Bellare and Rogaway [5] give an enciphering mode that works on messages of varying lengths but is not a strong PRP. Another enciphering scheme that is potentially a strong PRP appears in unpublished work of Bleichenbacher and Desai [6]. Yet another suggestion we have seen [11] is forward-then-backwards PCBC mode [16]. The mode is easily broken in the sense of a strong PRP, but the possibility of a simple, two-layer, CBC-like mode helped to motivate us. A different approach for disk-sector encipherment is to build a wide-blocksize block cipher from scratch. Such attempts include BEAR, LION, and Mercy [1, 7].

Afterwards. Recent work by the authors has focused on providing a fully parallelizable enciphering scheme having serial efficiency comparable to that of CMC. We shall report on that work elsewhere. The full version of the current paper appears as [10].

2 Preliminaries

Basics. A *message space* $\mathcal{M}$ is a set of strings $\mathcal{M} = \bigcup_{i\in I}\{0,1\}^i$ for some nonempty index set $I \subseteq \mathbb{N}$. A *length-preserving permutation* is a map $\pi\colon \mathcal{M} \to \mathcal{M}$ where $\mathcal{M}$ is a message space and π is a permutation and $|\pi(P)| = |P|$ for all $P \in \mathcal{M}$. A *tweakable enciphering scheme,* or simply an *enciphering scheme*, is a function $\mathcal{E}\colon \mathcal{K} \times \mathcal{T} \times \mathcal{M} \to \mathcal{M}$ where $\mathcal{K}$ (the key set) is a finite nonempty set and $\mathcal{T}$ (the tweak set) is a nonempty set and $\mathcal{M}$ is a message space and for every $K \in \mathcal{K}$ and $T \in \mathcal{T}$ we have that $\mathcal{E}(K, T, \cdot) = \mathcal{E}_K^T(\cdot)$ is a length-preserving permutation. An *untweakable enciphering scheme* is a function $\mathbb{E}\colon \mathcal{K} \times \mathcal{M} \to \mathcal{M}$ where $\mathcal{K}$ is a finite nonempty set and $\mathcal{M}$ is message space and $\mathbb{E}(K, \cdot) = \mathbb{E}_K(\cdot)$ is a length-preserving permutation for every $K \in \mathcal{K}$. A *block cipher* is a function $E\colon \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$ where $n \geq 1$ and $\mathcal{K}$ is a finite nonempty set and $E(K, \cdot) = E_K(\cdot)$ is a permutation for each $K \in \mathcal{K}$. The number n is the *blocksize*. An untweakable enciphering scheme can be regarded as a tweakable enciphering scheme with tweak set $\mathcal{T} = \{\varepsilon\}$ and a block cipher can be regarded as a tweakable enciphering scheme with tweak set $\mathcal{T} = \{\varepsilon\}$ and message space $\mathcal{M} = \{0,1\}^n$. The inverse of an enciphering scheme $\mathcal{E}$ is the enciphering scheme $\mathcal{D} = \mathcal{E}^{-1}$ where $X = \mathcal{D}_K^T(Y)$ if and only if $\mathcal{E}_K^T(X) = Y$. An *adversary* A is a (possibly probabilistic) algorithm with access to some oracles. Oracles are written as superscripts.

By convention, the running time of an algorithm includes its description size. We let $\mathrm{Time}_f(\mu)$ be a function that bounds the worst-case time to compute f on strings that total μ bits. We write $\widetilde{O}(f)$ for $O(f(n)\lg(f(n))$. Constants inside of O and $\widetilde{O}$ notations are absolute constants, depending only on details of the model of computation. If X and Y are strings of possibly different lengths we let $X \leftarrow\!\!\oplus\ Y$ be the string one gets by xoring the shorter string into the *beginning* of the longer string, leaving the rest of the longer string alone.

Security notions. The definitions here are adapted from [4, 14, 15]. When $\mathcal{M}$ is a message space and $\mathcal{T}$ is a nonempty set we let $\mathrm{Perm}(\mathcal{M})$ denote the set of all functions $\pi\colon \mathcal{M} \to \mathcal{M}$ that are length-preserving permutations, and we let $\mathrm{Perm}^{\mathcal{T}}(\mathcal{M})$ denote the set of functions $\pi\colon \mathcal{T} \times \mathcal{M} \to \mathcal{M}$ for which $\pi(T, \cdot)$ is a length-preserving permutation for all $T \in \mathcal{T}$.

Let $\mathcal{E}\colon \mathcal{K} \times \mathcal{T} \times \mathcal{M} \to \mathcal{M}$ be an enciphering scheme and A be an adversary. We define the *advantage* of A in distinguishing $\mathcal{E}$ from a random, tweakable, length-preserving permutation and its inverse as

$$\begin{aligned}\mathbf{Adv}_{\mathcal{E}}^{\pm\widetilde{\mathrm{prp}}}(A) \stackrel{\mathrm{def}}{=} \Pr\Big[K \stackrel{\$}{\leftarrow} \mathcal{K}:\ \ A^{\mathcal{E}_K(\cdot,\cdot)\ \mathcal{E}_K^{-1}(\cdot,\cdot)} \Rightarrow 1\Big] \\ -\ \Pr\Big[\pi \stackrel{\$}{\leftarrow} \mathrm{Perm}^{\mathcal{T}}(\mathcal{M}):\ \ A^{\pi(\cdot,\cdot)\ \pi^{-1}(\cdot,\cdot)} \Rightarrow 1\Big]\end{aligned}$$

The notation above shows, in the brackets, an experiment to the left of the colon and an event to the right of the colon. We are looking at the probability of the indicated event after performing the specified experiment. By $A \Rightarrow 1$ we mean the event that A outputs the bit 1. Often we omit writing the experiment, the oracle, or the placeholder-arguments of the oracle. The tilde above the "prp" serves as a reminder that the prp is tweakable, while the $\pm$ symbol in front of the "prp" serves as a reminder that this is the "strong" (i.e., chosen plaintext/ciphertext attack) notion of security. Thus we omit the tilde for untweakable enciphering schemes and block ciphers, and we omit the $\pm$ sign to mean that the adversary is given only the first oracle from each pair.

For each "advantage notion" $\mathbf{Adv}_{\Pi}^{\mathrm{xxx}}$ we write $\mathbf{Adv}_{\Pi}^{\mathrm{xxx}}(\mathcal{R})$ for the maximal value of $\mathbf{Adv}_{\Pi}^{\mathrm{xxx}}(A)$ over all adversaries A that use resources at most $\mathcal{R}$. Resources of interest are the running time t, the number of queries q, the total length of all queries μ (sometimes written as $\mu = n\sigma$ when μ is a multiple of some number n), and the length of the adversary's output ς. The name of an argument (t, t', q, etc.) will be enough to make clear what resource it refers to.

Pointless queries. There is no loss of generality in the definitions above to assume that regardless of responses that adversary A might receive from an arbitrary pair of oracles, it never repeats a query (T, P) to its left oracle, never repeats a query (T, C) to its right oracle, never asks its right oracle a query (T, C) if it earlier received a response of C to a query (T, P) from its left oracle, and never asks its left oracle a query (T, P) if it earlier received a response of P to a query (T, C) from its right oracle. We call such queries *pointless* because the adversary "knows" the answer that it should receive. A query is called *valid* if it is well-formed and not pointless. A sequence of queries and their responses

Algorithm $\mathcal{E}_{K\widetilde{K}}^{T}(P_1 \cdots P_m)$	**Algorithm** $\mathcal{D}_{K\widetilde{K}}^{T}(C_1 \cdots C_m)$
$\mathbb{T} \leftarrow E_{\widetilde{K}}(T)$	$\mathbb{T} \leftarrow E_{\widetilde{K}}(T)$
$PPP_0 \leftarrow \mathbb{T}$	$CCC_0 \leftarrow \mathbb{T}$
for $i \leftarrow 1$ **to** m **do**	**for** $i \leftarrow 1$ **to** m **do**
$\quad PP_i \leftarrow P_i \oplus PPP_{i-1}$	$\quad CC_i \leftarrow C_i \oplus CCC_{i-1}$
$\quad PPP_i \leftarrow E_K(PP_i)$	$\quad CCC_i \leftarrow E_K^{-1}(CC_i)$
$M \leftarrow 2\,(PPP_1 \oplus PPP_m)$	$M \leftarrow 2\,(CCC_1 \oplus CCC_m)$
for $i \in [1\,..\,m]$ **do**	**for** $i \in [1\,..\,m]$ **do**
$\quad CCC_i \leftarrow PPP_{m+1-i} \oplus M$	$\quad PPP_i \leftarrow CCC_{m+1-i} \oplus M$
$CCC_0 \leftarrow 0^n$	$PPP_0 \leftarrow 0^n$
for $i \in [1\,..\,m]$ **do**	**for** $i \in [1\,..\,m]$ **do**
$\quad CC_i \leftarrow E_K(CCC_i)$	$\quad PP_i \leftarrow E_K^{-1}(PPP_i)$
$\quad C_i \leftarrow CC_i \oplus CCC_{i-1}$	$\quad P_i \leftarrow PP_i \oplus PPP_{i-1}$
$C_1 \leftarrow C_1 \oplus \mathbb{T}$	$P_1 \leftarrow P_1 \oplus \mathbb{T}$
return $C_1 \cdots C_m$	**return** $P_1 \cdots P_m$

Fig. 1. Enciphering (left) and deciphering (right) under $\mathcal{E} = \mathrm{CMC}[E]$, where $E\colon \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$ is a block cipher. The tweak is $T \in \{0,1\}^n$ and the plaintext is $P = P_1 \cdots P_m$ and the ciphertext is $C = C_1 \cdots C_m$.

is valid if every query in the sequence is valid. We assume that adversaries ask only valid queries.

The finite field $GF(2^n)$. We may think of an n-bit string $L = L_{n-1} \ldots L_1 L_0 \in \{0,1\}^n$ in any of the following ways: as an abstract point in the finite field $\mathrm{GF}(2^n)$; as the number in $[0..2^n - 1]$ whose n-bit binary representation is L; and as the polynomial $L(\mathsf{x}) = L_{n-1}\mathsf{x}^{n-1} + \cdots + L_1\mathsf{x} + L_0$. To add two points, $A \oplus B$, take their bitwise xor. To multiply two points we must fix an irreducible polynomial $P_n(\mathsf{x})$ having binary coefficients and degree n: say the lexicographically first polynomial among the irreducible degree-n polynomials having a minimum number of nonzero coefficients. For $n = 128$, the indicated polynomial is $P_{128}(\mathsf{x}) = \mathsf{x}^{128} + \mathsf{x}^7 + \mathsf{x}^2 + \mathsf{x} + 1$. Now multiply $A(\mathsf{x})$ and $B(\mathsf{x})$ by forming the degree $2n-2$ (or less) polynomial that is their product and taking the remainder when this polynomial is divided by $P_n(\mathsf{x})$.

Often there are simpler ways to multiply in $\mathrm{GF}(2^n)$ than the definition above might seem to suggest. In particular, given L it is easy to "double" L. We illustrate the procedure for $n = 128$, in which case $2L = L \ll 1$ if $\mathsf{firstbit}(L) = 0$, and $2L = (L \ll 1) \oplus \mathrm{Const87}$ if $\mathsf{firstbit}(L) = 1$, where Const87 is $0^{120}10000111$. Here $\mathsf{firstbit}(L)$ means L_{n-1} and $L \ll 1$ means $L_{n-2}L_{n-3} \cdots L_1 L_0\, 0$.

3 Specification of CMC Mode

We construct from block cipher $E\colon \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$ a tweakable enciphering scheme that we denote by CMC-E or $\mathrm{CMC}[E]$. The enciphering scheme has key space $\mathcal{K} \times \mathcal{K}$. It has tweak space $\mathcal{T} = \{0,1\}^n$. The message space $\mathcal{M} = \bigcup_{m \ge 2} \{0,1\}^{mn}$ contains any string having any number m of n-bit blocks,

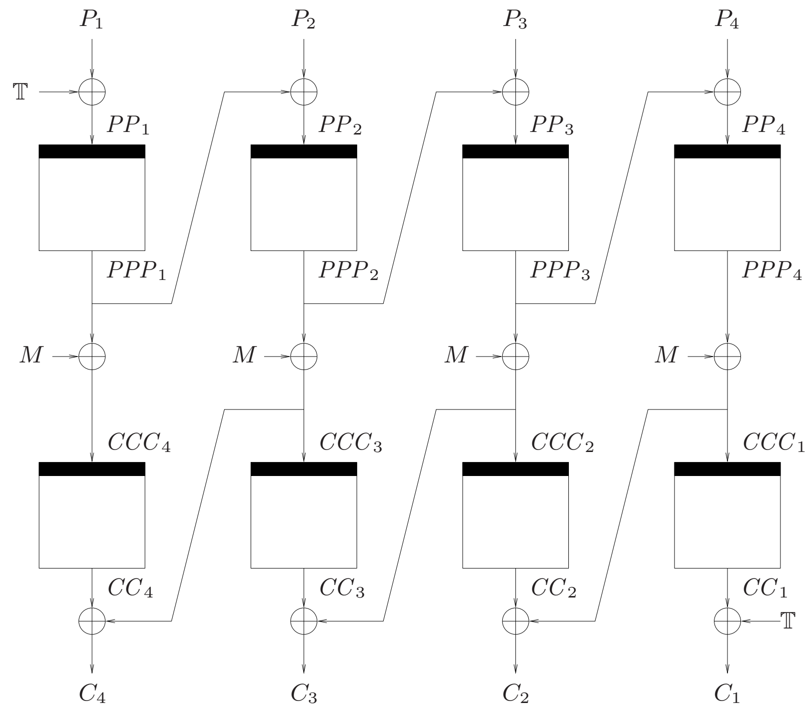

Fig. 2. Enciphering under CMC mode for a message of $m = 4$ blocks. The boxes represent E_K. We set mask $M = 2\,(PPP_1 \oplus PPP_m)$. This value can also be computed as $M = 2\,(CCC_1 \oplus CCC_m)$. We set $\mathbb{T} = E_{\widetilde{K}}(T)$ where T is the tweak.

where $m \geq 2$. We specify in Fig. 1 both the forward direction of our construction, $\mathcal{E} = \text{CMC-}E$, and its inverse $\mathcal{D}$. An illustration of CMC mode is given in Fig. 2. In the figures, all capitalized variables except for K and $\widetilde{K}$ are n-bit strings (keys K and $\widetilde{K}$ are elements of $\mathcal{K}$). Variable names P, C, and M are meant to suggest *plaintext*, *ciphertext*, and *mask*. When we write $\mathcal{E}_K^T(P_1 \cdots P_m)$ we mean that the incoming plaintext $P = P_1 \cdots P_m$ is silently partitioned into n-bit strings $P_1, \ldots, P_m$ (and similarly when we write $\mathcal{D}_K^T(C_1 \cdots C_m)$). It is an error to provide $\mathcal{E}$ (or $\mathcal{D}$) with a plaintext (or ciphertext) that is not mn bits for some $m \geq 2$.

4 Discussion

Basic observations. Deciphering $C = \mathcal{E}_{K\widetilde{K}}^T(P)$ produces the same mask M as enciphering P because $CCC_1 \oplus CCC_m = (PPP_1 \oplus M) \oplus (PPP_m \oplus M) = PPP_1 \oplus PPP_m$. Also note that the multiply by two in computing M cannot be dispensed with; if it were, CCC_m would not depend on PPP_1 so the mode could not be a PRP.

The CMC core. Consider the untweakable enciphering scheme $\mathbb{CMC}$ one gets by ignoring T and setting $\mathbb{T}$ to 0^n in Figs. 1 and 2. The CMC algorithm can then be viewed as taking $\mathbb{CMC}$ and "adding in" a tweak according to the construction $\mathrm{CMC}^{T}_{K\widetilde{K}}(P) = \mathbb{T} \leftarrow\!\!\oplus\ \mathbb{CMC}_K(P \leftarrow\!\!\oplus\ \mathbb{T})$ where $\mathbb{T} = E_{\widetilde{K}}(T)$. A similar approach to modifying an untweakable enciphering scheme to create a tweakable one was used by Liskov, Rivest, and Wagner [14, Theorem 2]. See Section 6.

Symmetry. Encryption under CMC is the same as decryption under CMC except that E_K is swapped with E_K^{-1} (apart from the computation of $\mathbb{T}$). Pictorially, this high degree of symmetry can be seen by observing that if the picture in Fig. 2 is rotated 180 degrees it is unchanged, apart from swapping letters P and C. Symmetry is a useful design heuristic in trying to achieve strong PRP security, as the goal itself provides the adversary with capabilities that are invariant with respect to replacing an enciphering scheme $\mathcal{E}$ by its inverse $\mathcal{D}$.

Notice that output blocks in CMC mode are taken in reverse order from the input blocks (meaning that $CCC_i = PPP_{m+1-i} \oplus M$ instead of $CCC_i = PPP_i \oplus M$). This was done for purposes of symmetry: if one had numbered output blocks in the "forward" direction then deciphering would be quite different from enciphering. As an added benefit, the reverse-numbering may improve the cache-interaction characteristics of CMC by improving locality of reference. That said, an application is always free to write its output according to whatever convention it wishes, and an application with limited memory may prefer to write its output as $C_m \cdots C_1$.

Re-orienting the bottom layer. It is tempting to orient the second block-cipher layer in the opposite direction as the first, thinking that this improves symmetry. But if one were to use E_K^{-1} in the second layer then CMC would become an involution, and thus easily distinguishable from a random permutation.

Limitations. CMC has the following limitations: (1) The mode is not parallelizable. (2) The sector size must be a multiple of the blocksize. (3) In order to make due with $2m + 1$ block-cipher calls one needs $\Theta(nm)$ bits of extra memory. Alternatively, one can use $\Theta(n)$ bits of memory, but then one needs $3m + 1$ block-cipher calls and one should output the blocks in reverse order. (4) The key for CMC is longer than the key for the underlying block cipher; to keep things simple, we have done nothing to "collapse keys" for this mode. (5) Both directions of the block cipher are used to decipher, due to the one block-cipher call used for producing $\mathbb{T}$ from T.

All of the above limitations could potentially be addressed. Further limitations are inherent characteristics of the type of object that is being constructed. Namely: (a) a good PRP necessarily achieves less than semantic security: repetitions of plaintexts that share a tweak are manifest in the ciphertexts. (b) A PRP must process the entire plaintext before emitting the first bit of ciphertext (and it must process the entire ciphertext before emitting the first block of plaintext). Depending on the context, these limitations can be significant.

5 Security of CMC

The concrete security of the CMC is summarized in the following theorem. The theorem relates the advantage that an adversary has in attacking CMC-E to the advantage that an adversary can get in attacking the underlying block cipher E.

Theorem 1. [CMC security] *Fix $n, t, q \geq 1$, $m \geq 2$, and a block cipher $E\colon \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$. Let message space $\mathcal{M} = \{0,1\}^{mn}$ and let $\sigma = mq$. Let* CMC *and* $\mathbb{CMC}$ *be the modes with the indicated message space. Then*

$$\mathbf{Adv}^{\pm\mathrm{prp}}_{\mathbb{CMC}[\mathrm{Perm}(n)]}(n\sigma) \leq \frac{5\,\sigma^2}{2^n} \tag{1}$$

$$\mathbf{Adv}^{\pm\widetilde{\mathrm{prp}}}_{\mathrm{CMC}[\mathrm{Perm}(n)]}(n\sigma) \leq \frac{7\,\sigma^2}{2^n} \tag{2}$$

$$\mathbf{Adv}^{\pm\widetilde{\mathrm{prp}}}_{\mathrm{CMC}[E]}(t, n\sigma) \leq \frac{7\,\sigma^2}{2^n} + 2\,\mathbf{Adv}^{\pm\mathrm{prp}}_{E}(t', 2\sigma) \tag{3}$$

where $t' = t + O(n\sigma)$. □

Although we defined CMC and $\mathbb{CMC}$ to have message space $\bigcup_{m\geq 2}\{0,1\}^{mn}$ the theorem restricts messages to one particular length, mn bits for some m. In other words, proven security is for a fixed-input-length (FIL) cipher and not a variable-input-length (VIL) one. We believe that, in fact, security also holds in the sense of a VIL cipher, but we do not at this time provide a proof. All other results in this paper are done for arbitrary (VIL) message spaces.

The heart of Theorem 1 is Equation (1), which is sketched in Appendix C. Equation (2) follows immediately using Theorem 2, as given below. Equation (3) embodies the standard way to pass from the information-theoretic setting to the complexity-theoretic one.

Since the proof of Equation (1) is long (and only a small portion of it is included in this proceedings version), let us try to get across some basic intuition for it. Refer to Fig. 2 (but ignore the $\mathbb{T}$, as we are only considering $\mathbb{CMC}$). Suppose the adversary asks to encipher some new four-block plaintext P. Plaintext P must be different from all previous plaintexts, so it has some first block where it is different, say P_3. This will usually result in PP_3 being *new*—some value not formerly acted on by the block cipher π. This, in turn, will result in PPP_3 being nearly uniform, and this will propagate to the right, so that PPP_4 will be nearly uniform as well. The values PPP_1 and PPP_2 will usually have been different from each other, and they'll usually be different from the freshly chosen PPP_3 and PPP_4 values. Now $M = 2(PPP_1 \oplus PPP_4)$ and so M will be nearly uniform due to the presence of PPP_4. When we add M to the PPP_i values we will get a bunch of sums CCC_i that are almost always new and distinct. This in turn will cause the vector of CC_i-values to be uniform, which will cause C to be uniform. The argument for a decryption query is symmetric.

Though it is ultimately the above intuition that the proof formalizes, one must be careful, as the experience with the Joux-attack drives home [12]. One must be sure that an adversary cannot, by cutting and pasting parts of plaintexts and ciphertexts, force any nontrivial repetitions in intermediate values.

6 Transforming an Untweakable Enciphering Scheme to a Tweakable One

Let $E\colon \widetilde{\mathcal{K}} \times \{0,1\}^n \to \{0,1\}^n$ be a block cipher and let $\mathbb{E}\colon \mathcal{K} \times \mathcal{M} \to \mathcal{M}$ be an untweakable enciphering scheme where the message space $\mathcal{M}$ contains no string of length less than n bits. We construct a tweakable enciphering scheme $\mathcal{E} = \mathbb{E} \triangleleft E$ where $\mathcal{E}\colon (\mathcal{K} \times \widetilde{\mathcal{K}}) \times \{0,1\}^n \times \mathcal{M} \to \mathcal{M}$. The construction is $\mathcal{E}^T_{K\widetilde{K}}(M) = \mathbb{T} \leftarrow\!\!\oplus\, \mathbb{E}_K(M \leftarrow\!\!\oplus\, \mathbb{T})$ where $\mathbb{T} = E_{\widetilde{K}}(T)$. (Recall that $\leftarrow\!\!\oplus$ just means to xor in the shorter string at the beginning.) Notice that the cost of adding in the tweak is one block-cipher call and two n-bit xors, regardless of the length of the sector being enciphered or deciphered. Also notice that $\mathrm{CMC} = \mathbb{CMC} \triangleleft E$.

The specified construction is similar to that of Liskov, Rivest and Wagner [14, Theorem 2] but, instead of a PRP E, those authors used an xor-universal hash function. One can view a secure block cipher as being "computationally" xor-universal, and try to conclude the security of the construction in that way. But we have also broadened the context to include enciphering schemes whose input is not a string of some fixed length, and so it seems better to prove the result from scratch. We show that $\mathcal{E} = \mathbb{E} \triangleleft E$ is secure (as a tweakable, strong, enciphering scheme) as long as $\mathbb{E}$ is secure (as an untweakable, strong enciphering scheme) and E is secure (as a PRP). The proof is given in the full version of this paper [10].

Theorem 2. [Adding in a tweak] *Let $E\colon \widetilde{\mathcal{K}} \times \{0,1\}^n \to \{0,1\}^n$ be a block cipher and let $\mathbb{E}\colon \mathcal{K} \times \mathcal{M} \to \mathcal{M}$ be an untweakable enciphering scheme whose message space $\mathcal{M}$ has a shortest string of $N \geq n$ bits. Then*

$$\mathbf{Adv}^{\pm\widetilde{\mathrm{prp}}}_{\mathbb{E}\triangleleft E}(t,q,\mu) \leq \frac{q^2}{2^n} + \frac{q^2}{2^N} + \mathbf{Adv}^{\pm\mathrm{prp}}_{\mathbb{E}}(t',q,\mu) + \mathbf{Adv}^{\mathrm{prp}}_{E}(t',q) \qquad (4)$$

where $t' = t + \widetilde{O}(\mu + q\mathrm{Time}_E + \mathrm{Time}_{\mathbb{E}}(\mu))$. □

7 Indistinguishability and Nonmalleability of Tweakable Enciphering Schemes

The definition we have given for the security of an enciphering scheme is simple and natural, but it is also quite far removed from any natural way to say that an encryption scheme does what it should do. In this section we explore two notions of security that speak more directly about the privacy and integrity of an enciphering scheme. First we give a definition of *indistinguishability* and then we give a definition for the *nonmalleability*. We show that, as one would expect, security in the sense of a tweakable PRP implies both of these notions, and by tight reductions.

Indistinguishability. To define the indistinguishability of a tweakable enciphering scheme $\mathcal{E}\colon \mathcal{K} \times \mathcal{T} \times \mathcal{M} \to \mathcal{M}$ we adapt the left-or-right notion from [2]. We imagine the following game. At the onset of the game we select at random a key K from $\mathcal{K}$ and a bit b. The adversary is then given access to two oracles, $\mathcal{E} = \mathcal{E}^b_K$ and $\mathcal{D} = \mathcal{D}^b_K$. The attacker can query the $\mathcal{E}$-oracle with any 4-tuple

When query	gets an answer of	then these queries are no longer allowed:
$\mathcal{E}(T_0, P_0;\ T_1, P_1)$	C	$\mathcal{D}(T_0, C,\ \cdot, \cdot)$ $\quad \mathcal{D}(\cdot, \cdot,\ T_1, C)$ $\mathcal{E}(T_0, P_0,\ \cdot, \cdot)$ $\quad \mathcal{E}(\cdot, \cdot,\ T_1, P_1)$
$\mathcal{D}(T_0, C_0,\ T_1, C_1)$	P	$\mathcal{E}(T_0, P,\ \cdot, \cdot)$ $\quad \mathcal{E}(\cdot, \cdot,\ T_1, P)$ $\mathcal{D}(T_0, C_0,\ \cdot, \cdot)$ $\quad \mathcal{D}(\cdot, \cdot,\ T_1, C_1)$

Table 1. Disallowed queries. The dot refers to an arbitrary argument—all are disallowed.

$(T_0, P_0,\ T_1, P_1)$ where $T_0, T_1 \in \mathcal{T}$ and P_0 and P_1 are equal-length strings in $\mathcal{M}$. The oracle returns $\mathcal{E}_K(T_b, P_b)$. Alternatively, the adversary can query the $\mathcal{D}$ oracle with a 4-tuple $(T_0, C_0,\ T_1, C_1)$ where $T_0, T_1 \in \mathcal{T}$ and C_0 and C_1 are equal-length strings in $\mathcal{M}$. The oracle returns $\mathcal{D}_K(T_b, C_b)$ where $\mathcal{D}$ is the inverse of $\mathcal{E}$. The adversary wants to identify the bit b. We must disallow the adversary from asking queries that will allow it to win trivially. The disallowed queries are given in Table 1.

The advantage of the adversary in guessing the bit b is defined by

$$\mathbf{Adv}^{\pm\widetilde{\mathrm{ind}}}_{\mathcal{E}}(A) \stackrel{\mathrm{def}}{=} \Pr[K \stackrel{\$}{\leftarrow} \mathcal{K}:\ A^{\mathcal{E}^1_K\ \mathcal{D}^1_K} \Rightarrow 1\,] \ - \ \Pr[K \stackrel{\$}{\leftarrow} \mathcal{K}:\ A^{\mathcal{E}^0_K\ \mathcal{D}^0_K} \Rightarrow 1\,]$$

We now show a tight equivalence between the PRP-security of a tweakable enciphering scheme and its indistinguishability. In Theorem 3 we show that PRP-security implies indistinguishability, and in Theorem 4 we show the converse. The proofs are in the full version of this paper [10].

Theorem 3. [$\pm\widetilde{\mathbf{prp}}$**-security** $\Rightarrow$ $\pm\widetilde{\mathbf{ind}}$**-security**] *Let* $\mathcal{E}\colon \mathcal{K} \times \mathcal{T} \times \mathcal{M} \to \mathcal{M}$ *be an enciphering scheme whose message space* $\mathcal{M}$ *consists of strings of length at least* n *bits. Then for any* t, q, μ,

$$\mathbf{Adv}^{\pm\widetilde{\mathrm{ind}}}_{\mathcal{E}}(t, q, 2\mu) \le 2\,\mathbf{Adv}^{\pm\widetilde{\mathrm{prp}}}_{\mathcal{E}}(t', q, \mu) + \frac{2q^2}{2^n - q}$$

where $t' = t + O(\mu)$. □

Theorem 4. [$\pm\widetilde{\mathbf{ind}}$**-security** $\Rightarrow$ $\pm\widetilde{\mathbf{prp}}$**-security**] *Let* $\mathcal{E}\colon \mathcal{K} \times \mathcal{T} \times \mathcal{M} \to \mathcal{M}$ *be an enciphering scheme. Then for any* t, q, μ, *we have* $\mathbf{Adv}^{\pm\widetilde{\mathrm{prp}}}_{\mathcal{E}}(t, q, \mu) \le \mathbf{Adv}^{\pm\widetilde{\mathrm{ind}}}_{\mathcal{E}}(t', q, 2\mu)$, *where* $t' = t + \widetilde{O}(\mu)$. □

Nonmalleability. Nonmalleability is an important cryptographic goal that was first identified and investigated by Dolev, Dwork, and Naor [8]. Informally, an encryption scheme is nonmalleable if an adversary cannot modify a ciphertext C to create a ciphertext C^* where the plaintext P^* of C^* is related to the plaintext P of C. In this section we define the nonmalleability of a tweakable enciphering scheme with respect to a chosen-ciphertext attack and we show that $\pm\widetilde{\mathrm{prp}}$-security implies nonmalleability. The result mirrors the well-known result that indistinguishability of a probabilistic encryption scheme under a chosen-ciphertext attack implies its nonmalleability under the same kind of attack [3, 8].

Fix an enciphering scheme $\mathcal{E}\colon \mathcal{K}\times\mathcal{T}\times\mathcal{M}\to\mathcal{M}$ and an adversary A. Consider running A with two oracles: an enciphering oracle $\mathcal{E}_K(\cdot,\cdot)$ and a deciphering oracle $\mathcal{D}_K(\cdot,\cdot)$, where $\mathcal{D}=\mathcal{E}^{-1}$ and K is chosen randomly from $\mathcal{K}$. After A has made all of its oracle queries and halted, we define a number of sets:

- *Known plaintexts.* For every $T\in\mathcal{T}$ we define the $\mathcal{P}^T$ as the set of all P such that A asked $\mathcal{E}_K$ to encipher (T,P) or A asked $\mathcal{D}_K$ to decipher some (T,C) and A got back an answer of P. Thus $\mathcal{P}^T$ is the set of all plaintexts P associated to T that the adversary already "knows".
- *Known ciphertexts.* For every $T\in\mathcal{T}$ we define $\mathcal{C}^T$ as the set of all C such A asked $\mathcal{D}_K$ to decipher (T,C) or A asked $\mathcal{E}_K$ to encipher some (T,P) and A got back an answer of C. Thus $\mathcal{C}^T$ is the set of all ciphertexts C associated to T that the adversary already "knows".
- *Plausible plaintexts.* For every $T\in\mathcal{T}$ and $C\in\mathcal{M}$ we define $\mathcal{P}^T(C)$ as the singleton set $\{\mathcal{D}_K^T(C)\}$ if $C\in\mathcal{C}^T$ and as $\{0,1\}^{|C|}\setminus\mathcal{P}^T$ otherwise. Thus $\mathcal{P}^T(C)$ is the set of all plaintexts P for which the adversary should regard it as plausible that $C=\mathcal{E}_K^T(P)$.

With enciphering scheme $\mathcal{E}\colon \mathcal{K}\times\mathcal{T}\times\mathcal{M}\to\mathcal{M}$ and adversary A still fixed, we consider the following two games, which we call games Real and Ideal. Both games begin by choosing a random key $K\xleftarrow{\$}\mathcal{K}$ and letting the adversary A interact with oracles $\mathcal{E}_K$ and $\mathcal{D}_K$ where $\mathcal{D}=\mathcal{E}^{-1}$. Just before termination, after the adversary has asked all the queries that it will ask, it outputs a three-tuple (T,C,f) where $T\in\mathcal{T}$ and $C\in\mathcal{M}$ and f is the encoding of a predicate $f\colon\mathcal{M}\to\{0,1\}$ (we do not distinguish between the predicate and its encoding). Now for game Real we set $P\leftarrow\mathcal{D}_K^T(C)$ and for game Ideal we set $P\xleftarrow{\$}\mathcal{P}^T(C)$. Finally, we look at the event that $f(P)=1$. Formally, we define the advantage of A, in the sense of nonmalleability under a chosen-ciphertext attack, as follows:

$$\begin{aligned}\mathbf{Adv}_{\mathcal{E}}^{\pm\widetilde{\mathrm{nm}}}(A) = {} & \Pr[K\xleftarrow{\$}\mathcal{K};\ (T,C,f)\xleftarrow{\$}A^{\mathcal{E}_K(\cdot,\cdot)\ \mathcal{D}_K(\cdot,\cdot)};\ P\leftarrow\mathcal{D}_K^T(C)\colon f(P)=1]- \\ & \Pr[K\xleftarrow{\$}\mathcal{K};\ (T,C,f)\xleftarrow{\$}A^{\mathcal{E}_K(\cdot,\cdot)\ \mathcal{D}_K(\cdot,\cdot)};\ P\xleftarrow{\$}\mathcal{P}^T(C)\colon f(P)=1]\end{aligned}$$

We emphasize that in game Ideal (the second experiment) the set $\mathcal{P}^T(C)$ depends on the oracle queries asked by A and the answers returned to it (even though this is not reflected in the notation). For the resource-bounded version of $\mathbf{Adv}_{\mathcal{E}}^{\pm\widetilde{\mathrm{nm}}}$ we let the running time t include the running time to compute $f(P)$. We have the following result, the proof of which appears in the full version of this paper [10].

Theorem 5. [$\pm\widetilde{\mathbf{prp}}$**-security** $\Rightarrow$ $\pm\widetilde{\mathbf{nm}}$**-security**] *Let $\mathcal{E}\colon\mathcal{K}\times\mathcal{T}\times\mathcal{M}\to\mathcal{M}$ be an enciphering scheme. Then for any t,q,μ,ς*

$$\mathbf{Adv}_{\mathcal{E}}^{\pm\widetilde{\mathrm{nm}}}(t,q,\mu,\varsigma)\le 2\,\mathbf{Adv}_{\mathcal{E}}^{\pm\widetilde{\mathrm{prp}}}(t',q+1,\mu+\varsigma)$$

where $t'=t+\widetilde{O}(\mu+\varsigma)$. □

Acknowledgments

The authors thank Jim Hughes for posing this problem to each of us and motivating our work on it. Shai thanks Hugo Krawczyk and Charanjit Jutla for discussions regarding candidates for implementing the Naor-Reingold approach and regarding the security proof for CMC. Phil thanks John Black for useful conversations on this problem, and Mihir Bellare, who promptly broke his first two-layer attempts. Phil received support from NSF grant CCR-0085961 and a gift from CISCO Systems. This work was carried out while Phil was at Chiang Mai University, Thailand.

References

1. R. Anderson and E. Biham. Two practical and provably secure block ciphers: BEAR and LION. In *Fast Software Encryption, Third International Workshop*, volume 1039 of *Lecture Notes in Computer Science*, pages 113–120, 1996. www.cs.technion.ac.il/∼biham/.
2. M. Bellare, A. Desai, E. Jokipii, and P. Rogaway. A concrete security treatment of symmetric encryption: Analysis of the DES modes of operation. In *Proceedings of 38th Annual Symposium on Foundations of Computer Science (FOCS 97)*, 1997.
3. M. Bellare, A. Desai, D. Pointcheval, and P. Rogaway. Relations among notions of security for public-key encryption schemes. In H. Krawczyk, editor, *Advances in Cryptology – CRYPTO '98*, volume 1462 of *Lecture Notes in Computer Science*, pages 232–249. Springer-Verlag, 1998.
4. M. Bellare, J. Kilian, and P. Rogaway. The security of the cipher block chaining message authentication code. *Journal of Computer and System Sciences*, 61(3):362–399, 2000. www.cs.ucdavis.edu/∼rogaway.
5. M. Bellare and P. Rogaway. On the construction of variable-input-length ciphers. In *Fast Software Encryption—6th International Workshop—FSE '99*, volume 1635 of *Lecture Notes in Computer Science*, pages 231–244. Springer-Verlag, 1999. www.cs.ucdavis.edu/∼rogaway.
6. D. Bleichenbacher and A. Desai. A construction of a super-pseudorandom cipher. Manuscript, February 1999.
7. P. Crowley. Mercy: A fast large block cipher for disk sector encryption. In B. Schneier, editor, *Fast Software Encryption: 7th International Workshop*, volume 1978 of *Lecture Notes in Computer Science*, pages 49–63, New York, USA, Apr. 2000. Springer-Verlag. www.ciphergoth.org/crypto/mercy.
8. D. Dolev, C. Dwork, and M. Naor. Non-malleable cryptography. *SIAM Journal on Computing*, 30(2):391–437, 2000. Earlier version in STOC 91.
9. S. Goldwasser and S. Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28:270–299, Apr. 1984.
10. S. Halevi and P. Rogaway. A tweakable enciphering mode. Manuscript, full version of this paper, May 2003. www.cs.ucdavis.edu/∼rogaway.
11. J. Hughes. Chair of the IEEE Security in Storage Working Group. Working group homepage at www.siswg.org. Call for algorithms can be found at www.mail-archive.com/cryptography@wasabisystems.com/msg02102.html, May 2002.
12. A. Joux. Cryptanalysis of the EMD mode of operation. In *Advances in Cryptology – EUROCRYPT '03*, volume 2656 of *Lecture Notes in Computer Science*. Springer-Verlag, 2003.

13. J. Kilian and P. Rogaway. How to protect DES against exhaustive key search. *Journal of Cryptology*, 14(1):17–35, 2001. Earlier version in CRYPTO '96. www.cs.ucdavis.edu/∼rogaway.
14. M. Liskov, R. Rivest, and D. Wagner. Tweakable block ciphers. In *Advances in Cryptology – CRYPTO '02*, Lecture Notes in Computer Science. Springer-Verlag, 2002. www.cs.berkeley.edu/∼daw/.
15. M. Luby and C. Rackoff. How to construct pseudorandom permutations from pseudorandom functions. *SIAM J. of Computation*, 17(2), April 1988.
16. C. Meyer and S. Matyas. *Cryptography: A new dimension in computer security.* John Wiley and Sons, 1982.
17. M. Naor and O. Reingold. A pseudo-random encryption mode. Manuscript, available from www.wisdom.weizmann.ac.il/∼naor/.
18. M. Naor and O. Reingold. On the construction of pseudo-random permutations: Luby-Rackoff revisited. *Journal of Cryptology*, 12(1):29–66, 1999. (Earlier version in STOC '97.) Available from www.wisdom.weizmann.ac.il/∼naor/.
19. P. Rogaway. The EMD mode of operation (a tweaked, wide-blocksize, strong PRP). Cryptology ePrint Archive, Report 2002/148, Oct. 2002. Early (buggy) version of the CMC algorithm. http://eprint.iacr.org/.
20. R. Schroeppel. The hasty pudding cipher. AES candidate submitted to NIST. www.cs.arizona.edu/∼rcs/hpc, 1999.
21. Y. Zheng, T. Matsumoto, and H. Imai. On the construction of block ciphers provably secure and not relying on any unproved hypotheses. In *Advances in Cryptology – CRYPTO '89*, volume 435 of *Lecture Notes in Computer Science*, pages 461–480. Springer-Verlag, 1989.

A The Joux Attack

In an early, unpublished version of the current paper [19] the scheme CMC, then called EMD, worked a little bit differently: instead of computing $\mathbb{T} = \mathbb{E}_K(T)$ and xoring $\mathbb{T}$ into P_1 and C_1, we simply xored T into the mask M, setting $M = 2(PPP_1 \oplus PPP_m) \oplus T$. We claimed—incorrectly—that the scheme was secure (as a tweakable, strong PRP). Antoine Joux [12] noticed that the scheme was wrong, pointing out that it is easy to distinguish the mode and its inverse from a tweakable truly random permutation and its inverse. Below is (a slightly simplified variant of) his attack:

1. The adversary picks an arbitrary tweak T and an arbitrary 4-block plaintext $P_1P_2P_3P_4$. It encrypts $(T,\ P_1P_2P_3P_4)$, obtaining ciphertext $C_1C_2C_3C_4$, and it encrypts $(T+1,\ P_1P_2P_3P_4)$, obtaining a different ciphertext $C_1'C_2'C_3'C_4'$.
2. The adversary now decrypts $(T, C_1(C_2'+1)(C_3+1)C_4)$, obtaining plaintext $P_1''P_2''P_3''P_4''$.

If $P_1'' = P_1$ then the adversary outputs 1 (it guesses that it has a "real" enciphering oracle; otherwise, the adversary answers 0 (it knows that it has a "fake" enciphering oracle). It is easy to see that this attack has advantage of nearly 1.

What went wrong? Clearly the provided proof had a bug. The bug turns out not to be a particularly interesting one. On the 14-th page of the proof [19] begins a detailed case analysis. The case denoted X1–X5 was incorrect: two random variables are said to rarely collide, but with an appropriate choice of constants the random variables become degenerate (constants) and *always* collide. The same happens for case Y1–Y5. The current paper restructures the case analysis.

Our earlier manuscript [19] also mentioned a parallelizable mode that we called EME. Joux also provides an attack on EME, using the tweak in a manner similar to the attack on CMC. We later found that, as opposed to CMC, the EME scheme remains insecure even as an untweakable PRP. Thus one cannot repair EME simply by using a different method of incorporating the tweak.

B A Useful Lemma — $\pm\widetilde{\textbf{prp}}$-security $\Leftrightarrow$ $\pm\widetilde{\textbf{rnd}}$-security

Before proving security for CMC, we provide a little lemma that says that a (tweakable) truly random permutation and its inverse looks very much like a pair of oracles that just return random bits (assuming you never ask pointless queries). Let $\mathcal{E}\colon \mathcal{K}\times\mathcal{T}\times\mathcal{M}\to\mathcal{M}$ be a tweaked block-cipher and let $\mathcal{D}$ be its inverse. The advantage of distinguishing $\mathcal{E}$ from random bits, $\mathbf{Adv}_{\mathcal{E}}^{\pm\widetilde{\mathrm{rnd}}}$, is

$$\mathbf{Adv}_{\mathcal{E}}^{\pm\widetilde{\mathrm{rnd}}}(A) = \Pr[K \xleftarrow{\$} \mathcal{K}:\ \ A^{\mathcal{E}_K(\cdot,\cdot)\ \mathcal{D}_K(\cdot,\cdot)} \Rightarrow 1\,] - \Pr[\,A^{\$(\cdot,\cdot)\ \$(\cdot,\cdot)} \Rightarrow 1\,]$$

where $\$(T,M)$ returns a random string of length $|M|$. We insist that A makes no pointless queries, regardless of oracle responses, and A asks no query (T,M) outside of $\mathcal{T}\times\mathcal{M}$. We extend the definition above in the usual way to its resource-bounded versions. We have the following:

Lemma 1. **[$\pm\widetilde{\textbf{prp}}$-security $\approx$ $\pm\widetilde{\textbf{rnd}}$-security]** *Let $\mathcal{E}\colon \mathcal{K}\times\mathcal{T}\times\mathcal{M}\to\mathcal{M}$ be a tweaked block-cipher and let $q\geq 1$. Then $|\mathbf{Adv}_{\mathcal{E}}^{\pm\widetilde{\mathrm{prp}}}(q) - \mathbf{Adv}_{\mathcal{E}}^{\pm\widetilde{\mathrm{rnd}}}(q)| \leq q(q-1)/2^{N+1}$, where N is the length of a shortest string in $\mathcal{M}$.*

The proof, which is standard, appears in the full paper [10].

C Sketch of Theorem 1 — Security of $\mathbb{CMC}$

Our proof of security for $\mathbb{CMC}$ is divided into two parts: (1) a game-substitution argument, reducing the analysis of $\mathbb{CMC}$ to the analysis of a simpler probabilistic game; and (2) analyzing that game. We also use Lemma 1 from above.

THE GAME-SUBSTITUTION SEQUENCE. Let n, m, and q all be fixed, and $\sigma = mq$. Let A be an adversary that asks q oracle queries (none pointless), each of nm bits. Our first major goal is to describe a probability space, NON2, this probability space depending on constants derived from A, and to define an event on the probability space, denoted NON2 sets *bad*, for which $\mathbf{Adv}_{\mathbb{CMC}[\mathrm{Perm}(n)]}^{\pm\widetilde{\mathrm{prp}}}(A) \leq 2\cdot\Pr[\,\text{NON2 sets } bad\,]+\sigma^2/2^n$. Later we bound $\Pr[\,\text{NON2 sets } bad\,]$ and, putting that together with Lemma 1, we will get Equation (1) of Theorem 1. The rest of Theorem 1 follows easily, as explained in Section 5. Game NON2 is obtained by a game-substitution argument, as carried out in works like [13]. The goal is to simplify the rather complicated setting of A adaptively querying its oracles and

Initialization:
$bad \leftarrow \mathsf{false}$; Domain $\leftarrow$ Range $\leftarrow \emptyset$; **for** all $X \in \{0,1\}^n$ **do** $\pi(X) \leftarrow \mathsf{undef}$

Respond to the s-th adversary query as follows:

An encipher query, $\mathsf{Enc}(P_1^s \cdots P_m^s)$:
$u[s] \leftarrow$ the largest value in $[0 \,..\, m]$ s.t. $P_1^s \cdots P_{u[s]}^s = P_1^r \cdots P_{u[s]}^r$ for some $r < s$
$PPP_0^s \leftarrow CCC_0^s \leftarrow 0^n$; **for** $i \leftarrow 1$ **to** $u[s]$ **do** $PP_i^s \leftarrow P_i^s \oplus PPP_{i-1}^s$, $PPP_i^s \leftarrow PPP_i^r$
for $i \leftarrow u[s]+1$ **to** m **do**
  $PP_i^s \leftarrow P_i^s \oplus PPP_{i-1}^s$
  $PPP_i^s \xleftarrow{\$} \{0,1\}^n$; **if** $PPP_i^s \in$ Range **then** $bad \leftarrow \mathsf{true}$, $\boxed{PPP_i^s \xleftarrow{\$} \overline{\text{Range}}}$
  if $PP_i^s \in$ Domain **then** $bad \leftarrow \mathsf{true}$, $\boxed{PPP_i^s \leftarrow \pi(PP_i^s)}$
  $\pi(PP_i^s) \leftarrow PPP_i^s$, Domain $\leftarrow$ Domain $\cup \{PP_i^s\}$, Range $\leftarrow$ Range $\cup \{PPP_i^s\}$
$M^s \leftarrow 2\,(PPP_1^s \oplus PPP_m^s)$; **for** $i \in [1 \,..\, m]$ **do** $CCC_i^s \leftarrow PPP_{m+1-i}^s \oplus M^s$
for $i \leftarrow 1$ **to** m **do**
  $CC_i^s \xleftarrow{\$} \{0,1\}^n$; **if** $CC_i^s \in$ Range **then** $bad \leftarrow \mathsf{true}$, $\boxed{CC_i^s \xleftarrow{\$} \overline{\text{Range}}}$
  if $CCC_i^s \in \text{Domain}(\pi)$ **then** $bad \leftarrow \mathsf{true}$, $\boxed{CC_i^s \leftarrow \pi(CCC_i^s)}$
  $C_i^s \leftarrow CC_i^s \oplus CCC_{i-1}^s$
  $\pi(CCC_i^s) \leftarrow CC_i^s$, Domain $\leftarrow$ Domain $\cup \{CCC_i^s\}$, Range $\leftarrow$ Range $\cup \{CC_i^s\}$
return $C_1 \cdots C_m$

A decipher query, $\mathsf{Dec}(C_1^s \cdots C_m^s)$, is handled similarly

Fig. 3. Game CMC1 provides a perfect simulation of $\mathbb{CMC}[\text{Perm}(n)]$. The boxed statements are events where we need to reset a previously chosen value.

to arrive at a simpler setting where there is no adversary and no interaction—just a program that flips coins and a flag *bad* that does or does not get set.

The various games. We describe the attack of A against $\mathbb{CMC}[\text{Perm}(n)]$ as a probabilistic game in which the permutation π is chosen "on the fly", as needed to answer the queries of A. Initially, the partial function $\pi\colon \{0,1\}^n \to \{0,1\}^n$ is everywhere undefined. When we need $\pi(X)$ and π isn't yet defined at X we choose this value randomly among the available range values. When we need $\pi^{-1}(Y)$ and there is no X for which $\pi(X)$ has been set to Y we likewise choose X at random from the available domain values. As we fill in π its domain and its range thus grow. In the game we keep track of the domain and range of π by maintaining two sets, Domain and Range, that include all the points for which π is already defined. We let $\overline{\text{Domain}}$ and $\overline{\text{Range}}$ be the complement of these sets relative to $\{0,1\}^n$. The game, denoted CMC1, is shown in Fig. 3. Since game CMC1 accurately represent the attack scenario, we have that

$$\Pr[\, A^{\mathbb{E}_\pi \, \mathbb{D}_\pi} \Rightarrow 1\,] = \Pr[\, A^{\text{CMC1}} \Rightarrow 1\,] \tag{5}$$

The basic idea in the proof is that the bad events that we need to analyze are "accidental collisions", where a value that was supposed to be "new" happens to be equal to one of the values currently in the sets Domain and Range. The full proof therefore goes through a sequence of intermediate games—RND1, RND2, RND3, NON1, and NON2—that are designed to help us reason about the probability of these "accidental collisions" (and therefore the advantage an adversary can get in distinguishing CMC1 from a random permutation and its inverse).

Specifically, in game RND1 we omit all the boxed "resetting events" that immediately follow the setting of the flag *bad* (this is the usual trick under the game-substitution approach). In games RND2 and RND3 we just re-arrange the code without effecting the distribution of any of the variables in the game. In game NON1 we eliminate the interaction, essentially by letting the adversary specify not only the queries in the game, but also the answers to these queries (with some minor restrictions). Finally, in game NON2 we use the symmetry of CMC, arguing that it is sufficient to analyze only half of the "collision events" since the other half is completely symmetric. These games are designed so that:

- $\Pr[\, A^{\mathrm{CMC1}} \Rightarrow 1\,] - \Pr[\, A^{\mathrm{RND1}} \Rightarrow 1\,] \le \Pr[\, A^{\mathrm{RND1}} \text{ sets } \mathit{bad}\,]$
- $\Pr[\, A^{\mathrm{RND1}} \Rightarrow 1\,] = \Pr[\, A^{\mathrm{RND2}} \Rightarrow 1\,] = \Pr[\, A^{\widetilde{\pm\mathrm{rnd}}} \Rightarrow 1\,]$
- $\Pr[\, A^{\mathrm{RND1}} \text{ sets } \mathit{bad}\,] = \Pr[\, A^{\mathrm{RND2}} \text{ sets } \mathit{bad}\,] = \Pr[\, A^{\mathrm{RND3}} \text{ sets } \mathit{bad}\,]$ $\le \Pr[\,\mathrm{NON1} \text{ sets } \mathit{bad}\,] + \frac{q(q-1)}{2^{n+1}} \le 2 \cdot \Pr[\,\mathrm{NON2} \text{ sets } \mathit{bad}\,] + \frac{q(q-1)}{2^{n+1}}$

Combining these statements with Equation (5) and Lemma 1 we have reduced the problem of bounding the adversary's advantage to answering a question about game NON2. We now look at game NON2, which is shown in Fig. 4.

Game NON2 (the name suggests "noninteractive") depends on a fixed transcript $\boldsymbol{\tau} = \langle \mathbf{ty}, \mathbf{P}, \mathbf{C} \rangle$ with $\mathbf{ty} = (\mathsf{ty}^1, \cdots, \mathsf{ty}^q)$, $\mathbf{P} = (\mathsf{P}^1, \cdots, \mathsf{P}^q)$, and $\mathbf{C} = (\mathsf{C}^1, \cdots, \mathsf{C}^q)$ where $\mathsf{ty}^s \in \{\mathsf{Enc}, \mathsf{Dec}\}$ and $\mathsf{P}^s = \mathsf{P}^s_1 \cdots \mathsf{P}^s_m$ and $\mathsf{C}^s = \mathsf{C}^s_1 \cdots \mathsf{C}^s_m$ for $|\mathsf{P}^r_i| = |\mathsf{C}^r_i| = n$. This fixed transcript may not specify any "immediate collisions" or "pointless queries"; we call such a transcript *allowed*. Formally, saying that $\boldsymbol{\tau}$ is allowed means that for all $r < s$ we have the following: if $\mathsf{ty}^s = \mathsf{Enc}$ then (i) $\mathsf{P}^s \ne \mathsf{P}^r$ and (ii) $\mathsf{C}^s_1 \ne \mathsf{C}^r_1$; while if $\mathsf{ty}^s = \mathsf{Dec}$ then (i) $\mathsf{C}^s \ne \mathsf{C}^r$ and (ii) $\mathsf{P}^s_1 \ne \mathsf{P}^r_1$. Now fix an allowed transcript $\boldsymbol{\tau}$ that maximizes the probability of the flag *bad* being set. This one transcript $\boldsymbol{\tau}$ is hardwired into game NON2.

Analysis of game NON2. It is helpful to view the multiset $\mathfrak{D}$ as a set of formal variables (rather than a multiset containing the values that these variables assume). Namely, whenever in game NON2 we set $\mathfrak{D} \leftarrow \mathfrak{D} \cup \{X\}$ for some variable X, we would think of it as setting $\mathfrak{D} \leftarrow \mathfrak{D} \cup \{\text{“}X\text{”}\}$ where "X" is the name of that formal variable. Viewed in this light, our goal now is to bound the probability that two formal variables in $\mathfrak{D}$ assume the same value in the execution of NON2. We observe that the formal variables in $\mathfrak{D}$ are uniquely determined by $\boldsymbol{\tau}$—they don't depend on the random choices made in the game NON2; specifically,

$$
\begin{aligned}
\mathfrak{D} = \{PP^s_i \mid\ \mathsf{ty}^s = \mathsf{Dec}\} \ \cup\ \{PP^s_i \mid\ \mathsf{ty}^s = \mathsf{Enc} \text{ and } i > \mathsf{u}[s]\} \ \cup \\
\{CCC^s_i \mid\ \mathsf{ty}^s = \mathsf{Enc}\} \ \cup\ \{CCC^s_i \mid\ \mathsf{ty}^s = \mathsf{Dec} \text{ and } i > \mathsf{u}[s]\}
\end{aligned}
$$

We view the formal variables in $\mathfrak{D}$ as *ordered* according to when they are assigned a value in the execution of game NON2. This ordering too is fixed, depending only on the fixed transcript $\boldsymbol{\tau}$. The crucial claim is the following:

$\mathfrak{D} \leftarrow \emptyset$ // Multiset
for $s \leftarrow 1$ **to** q **do**
  if $ty^s = \mathsf{Enc}$ **then**
    $\mathsf{u}[s] \leftarrow$ largest value in $[0 \mathinner{..} m]$ s.t. $\mathsf{P}_1^s \cdots \mathsf{P}_{\mathsf{u}[s]}^s = \mathsf{P}_1^r \cdots \mathsf{P}_{\mathsf{u}[s]}^r$ for some $r < s$
    $PPP_0^s \leftarrow CCC_0^s \leftarrow 0^n$
    for $i \leftarrow 2$ **to** $\mathsf{u}[s]$ **do** $PP_i^s \leftarrow \mathsf{P}_i^s \oplus PPP_{i-1}^s,\quad PPP_i^s \leftarrow PPP_i^r$
    for $i \leftarrow \mathsf{u}[s]+1$ **to** m **do**
      $PP_i^s \leftarrow \mathsf{P}_i^s \oplus PPP_{i-1}^s$; $\mathfrak{D} \leftarrow \mathfrak{D} \cup \{PP_i^s\}$
      $\boxed{PPP_i^s \xleftarrow{\$} \{0,1\}^n}$
    for $i \in [1 \mathinner{..} m]$ **do** $CCC_i^s \leftarrow PPP_{m+1-i}^s \oplus 2\,(PPP_1^s \oplus PPP_m^s)$
      $\mathfrak{D} \leftarrow \mathfrak{D} \cup \{CCC_i^s\}$
  else ($ty^s = \mathsf{Dec}$)
    $\mathsf{u}[s] \leftarrow$ largest value in $[0 \mathinner{..} m]$ s.t. $\mathsf{C}_1^s \cdots \mathsf{C}_{\mathsf{u}[s]}^s = \mathsf{C}_1^r \cdots \mathsf{C}_{\mathsf{u}[s]}^r$ for some $r < s$
    $CCC_0^s \leftarrow PPP_0^s \leftarrow 0^n$
    for $i \leftarrow 1$ **to** $\mathsf{u}[s]$ **do** $CC_i^s \leftarrow \mathsf{C}_i^s \oplus CCC_{i-1}^s,\quad CCC_i^s \leftarrow CCC_i^r$
    for $i \in [\mathsf{u}[s]+1 \mathinner{..} m]$ **do** $\boxed{CCC_i^s \xleftarrow{\$} \{0,1\}^n}$; $\mathfrak{D} \leftarrow \mathfrak{D} \cup \{CCC_i^s\}$
    for $i \in [1 \mathinner{..} m]$ **do** $PPP_i^s \leftarrow CCC_{m+1-i}^s \oplus 2\,(CCC_1^s \oplus CCC_m^s)$
    for $i \in [1 \mathinner{..} m]$ **do** $PP_i^s \leftarrow \mathsf{P}_i^s \oplus PPP_{i-1}^s$; $\mathfrak{D} \leftarrow \mathfrak{D} \cup \{PP_i^s\}$
$bad \leftarrow$ (some value appears more than once in $\mathfrak{D}$)

Fig. 4. Game NON2. The boxed statements are the random choices in this game.

Claim. For any two distinct variables $X, X' \in \mathfrak{D}$ we have $\Pr[X = X'] \le 2^{-n}$ □

The proof, consisting of an exhaustive case analysis, can be found in the full version of this paper [10]. The idea is to look at the "free variables" that are directly chosen at random in game NON2 (the boxed statements in Fig. 4), and to show that the sum of any two variables in $\mathfrak{D}$ depends linearly on at least one free variable. We now show how this claim finishes the proof of Theorem 1. As there are no more than 2σ variables in $\mathfrak{D}$, we use the union bound to conclude $\Pr[\,\text{NON2 sets } bad\,] \le \binom{2\sigma}{2}/2^n$. Combining the results given so far we have that:

$$\begin{aligned}
\mathbf{Adv}^{\pm\widetilde{\mathrm{prp}}}_{\mathbb{CMC}[\mathrm{Perm}(n)]}(A) &\le \mathbf{Adv}^{\pm\widetilde{\mathrm{rnd}}}_{\mathbb{CMC}[\mathrm{Perm}(n)]}(A) + q(q-1)/2^{2n+1} \\
&\le 2 \cdot \Pr[\text{NON2 sets } bad] + q(q-1)/2^{n+1} + q(q-1)/2^{2n+1} \\
&\le 2 \cdot \binom{2\sigma}{2}/2^n + q(q-1)/2^{n+1} + q(q-1)/2^{2n+1} \ \le 5\sigma^2/2^n
\end{aligned}$$

This completes the proof, assuming the missing claim from above. ∎