On the Security of CTR + CBC-MAC

Jakob Jonsson*

jakob_jonsson@yahoo.se

Abstract. We analyze the security of the CTR + CBC-MAC (CCM) encryption mode. This mode, proposed by Doug Whiting, Russ Housley, and Niels Ferguson, combines the CTR ("counter") encryption mode with CBC-MAC message authentication and is based on a block cipher such as AES. We present concrete lower bounds for the security of CCM in terms of the security of the underlying block cipher. The conclusion is that CCM provides a level of privacy and authenticity that is in line with other proposed modes such as OCB.

Keywords: AES, authenticated encryption, modes of operation.

1 Introduction

Background. Block ciphers are popular building blocks in cryptographic algorithms intended to provide information services such as privacy and authenticity. Such block-cipher based algorithms are referred to as *modes of operation*. Examples of encryption modes of operation are Cipher Block Chaining Mode (CBC) [21], Electronic Codebook Mode (ECB) [21], and Counter Mode (CTR) [8]. Since each of these modes provides privacy only and not authenticity, most applications require that the mode be combined with an authentication mechanism, typically a MAC algorithm [20] based on a hash function such as SHA-1 [22].

As it turns out, there are secure and insecure ways of combining a secure encryption mode with a secure MAC algorithm; certain constructions are easily broken due to bad interactions between the components (see [4,18] for discussion). While there are generic constructions with a provable security in terms of the underlying components (e.g., schemes based on the first-encrypt-then-authenticate paradigm as described in [18]), implementers tend to pick other combinations for reasons that are beyond the scope of this paper. Some of these combinations have turned out to be secure for the specific components chosen, while other combinations have been broken (see again [18]).

An interesting line of research the last couple of years has been the development of block cipher modes of operation that simultaneously provide privacy and authenticity. We will refer to such modes of operation as *combined modes* (a less ambiguous and frequently used term is *authenticated-encryption modes*). Ideally, such a mode should be *provably* secure, roughly meaning that there exists a mathematical proof that the scheme cannot be broken unless a weakness can be found in the underlying block cipher.

* This work was completed at RSA Laboratories Europe in Stockholm.

K. Nyberg and H. Heys (Eds.): SAC 2002, LNCS 2595, pp. 76–93, 2003.

Our goal. In this paper we provide a formal analysis of a combined mode denoted CCM, which is shorthand for **C**TR + **C**BC-**M**AC. As the full name indicates, CCM combines the CTR encryption mode with the CBC-MAC [13] authentication mode; it does so using one single block cipher encryption key. CCM is proposed for IEEE 802.11 [27] and NIST Modes of Operation [28].

Some attractive properties of CCM mode are as follows.

1. CCM readily handles arbitrary messages in which certain parts are intended to be authenticated only and not encrypted; this is done without any additional ciphertext overhead. Many other combined modes require certain enhancements to fully achieve this property.
2. The underlying block cipher is used only in the forward "encryption" direction and not in the reverse "decryption" direction; this is true both for CCM encryption and CCM decryption. This feature makes CCM an attractive candidate for applications where a small code size is desirable. Also, this makes it possible to define CCM in terms of an arbitrary pseudo-random function that is not necessarily reversible. In this respect, CCM is more versatile than many other proposed modes.
3. CCM is based on well-known technology; CTR and CBC-MAC were introduced long ago. The two modes being widely scrutinized and documented may help avoid potential implementation loopholes. Also, highly optimized and well-trusted implementations of CBC have been around for years.
4. According to [27], all intellectual property rights to CCM have been released into the public domain.

As is pointed out in [25], CCM being based on well-trusted components is not in itself an argument for the security of CCM: While the underlying modes CTR and CBC-MAC are known to be provably secure under certain assumptions (see [2] and [3,24]), the two modes share the same block cipher encryption key within CCM. In particular, the results in [18] do not apply. Our object is to demonstrate that CCM is as secure as the two-key variant covered in [18]. We stress that our analysis applies to the typical situation where messages of variable length (as opposed to a fixed prescribed length as in [3]) are to be processed.

Property 2 turns out to be of significant help in the analysis; thanks to this property we can give a security proof for CCM in terms of a pseudo-random function that does not necessarily have a well-defined inverse. By standard arguments, the proof is then easily translated into a security proof for CCM in terms of a pseudo-random permutation (i.e., a block cipher) that does have an inverse. A direct proof in terms of a block cipher would most certainly be very tricky due to biases caused by the absence of output collisions in permutations.

Related work and further directions. A number of different combined modes have been proposed; these modes have in common that they add "redundancy" to the message to be encrypted. The approach, employed in proposals such as IAPM [16], OCB [26], IACBC [15], and IGE [10], is to concatenate the message with a non-cryptographic checksum before encryption. In some cases the checksum is an xor sum of the blocks in the message, while in other cases a constant block

will do. The encryption method is typically a refinement of a standard mode such as CBC (e.g., IACBC and IGE) or ECB (e.g., IAPM and OCB). See [1] for a general treatment of the "encrypt-with-redundancy" paradigm.

The purpose of concatenating a checksum to the message is to make it hard for an adversary to modify a ciphertext in a manner consistent with the underlying message. The above mentioned modes of operation are all equipped with security proofs assuring that this is indeed hard. CCM employs the same paradigm but uses a cryptographic tag rather than a checksum. This makes CCM less efficient than the other variants, but instead CCM achieves benefits such as properties 1 and 2 listed above.

While our definition of a combined mode is in terms of a block cipher, another possibility would be to use a hash function such as SHA-1 as the underlying primitive. It has been demonstrated [11] that certain standard hash functions (e.g., SHA-1) are easily turned into block ciphers with attractive properties. Also, any hash function can be used as a building block in a stream cipher; consider the MGF construction in [12] based on ideas from [5]. We will not pursue this discussion further in this context and confine ourselves with acknowledging the problem as an interesting area of research.

Notation

For each integer $k > 0$, $\{0,1\}^k$ denotes the set of bit strings of length k. The length of a bit string X is denoted $|X|$. For integers $j \geq 0$ and $k > 0$ ($j < 2^k$), $(j)_k$ is the k-bit representation of j (e.g., $(13)_6 = \texttt{001101}$). For a bit string X of bit length k, we will sometimes write $(X)_k$ instead of X to indicate explicitly that the bit length of X is k. For any set S, define S^* as the union $\bigcup_{k\geq 0} S^k$; S^* is the set of all finite sequences (including the empty sequence) of elements from S. The concatenation of two bit strings X and Y is denoted $X\,.\,Y$.

2 Scheme Description

CTR + CBC-MAC [27] (from now on denoted CCM) is a combined mode providing privacy and authenticity. We stress that CCM as defined in this section is a generalization of the proposal [27]; the special case defined in [27] is described in Section 2.3. CCM is based on a pseudo-random function

$$E : \{0,1\}^{k_0} \times \{0,1\}^{k_b} \to \{0,1\}^{k_b} \ ;$$

E takes as input a key of bit length k_0 and a block to be encrypted of bit length k_b and outputs an encrypted block of the same bit length k_b. We will write $E_K(X) = E(K,X)$.

We anticipate that most practical applications of CCM will be based on a traditional block cipher E, which means that E_K is a permutation (thus invertible) for each K. For example, in the proposal [27] the underlying function is AES [7,23]. However, as we have already pointed out, E_K does not have to be a permutation; the function is used only in the forward encryption direction and never in the reverse decryption direction.

2.1 Overview of CCM

Before the CCM encryption scheme can be used, the parties that are going to exchange secret information must agree on a secret key K. A detailed description of possible key exchange methods is beyond the scope of this paper, but it is assumed that the key is selected uniformly (or close to uniformly) at random from the set $\{0,1\}^{k_0}$ of all possible keys.

The CCM encryption operation takes as input a *nonce* N of fixed bit length $k_n < k_b$, a *message* m, and *additional data* a to be authenticated.[1] The additional data a is only authenticated and not encrypted, whereas the message m is both authenticated and encrypted; an authentication *tag* is derived from (N, m, a) via CBC-MAC and encrypted together with m in CTR mode. The tag is of fixed length $k_t \leq k_b$. The encryption operation outputs a ciphertext c of bit length $|m| + k_t$.

The nonce N is *non-repeating* ("fresh") in the sense that it must not have been used as a nonce in a previous application of the CCM encryption operation during the lifetime of a key. Typically, there are certain restrictions on the inputs to the CCM encryption operation. For example, the lengths of the message and the additional data might be upper-bounded by some constant. Also, some applications may require that the bit length of the message and the additional data be a multiple of 8 or the block length. An input (N, m, a) satisfying all requirements is *valid*; the set of all valid inputs is a subset $\mathcal{V}$ of the set of all possible triples (N, m, a) of bit strings.

CBC-MAC computation. In the first step of the encryption operation, we compute a CBC-MAC tag of a string derived from the input. Since CBC-MAC acts on blocks of bit length k_b, it cannot be applied directly to the CCM input (which is a triple of bit strings with lengths not necessarily multiples of k_b). For this reason we need to introduce an *encoding function*

$$\beta : \mathcal{V} \to \mathcal{W}^*,$$

where $\mathcal{W} = \{0,1\}^{k_b}$ ($\mathcal{W}$ is the set of *blocks*); the output from β is a string of blocks. On a valid input $(N, m, a) \in \mathcal{V}$, the encoding function β derives a string

$$B_0 \,.\, B_1 \,.\, \cdots \,.\, B_r$$

of *CBC-MAC blocks* $B_0, \ldots, B_r$. A tag T is derived by applying CBC-MAC to these blocks; see the algorithm description in Section 2.2 for details. The first block B_0 is the *CBC-MAC pre-IV*.[2]

We require that the following hold for the encoding function β.

1. N is uniquely determined by the first block B_0 of $\beta(N, m, a)$.
2. For any two valid and distinct inputs (N, m, a) and (N', m', a') with corresponding CBC-MAC blocks

[1] As a convention, upper-case letters denote fixed-length strings, whereas lower-case letters denote variable-length strings.

[2] We may view T as the CBC-MAC tag of $B_1 \,.\, \cdots \,.\, B_r$ with IV $E_K(B_0)$.

$$\beta(N, m, a) = B_0 \,.\, B_1 \,.\, \cdots \,.\, B_r \ ,$$
$$\beta(N', m', a') = B'_0 \,.\, B'_1 \,.\, \cdots \,.\, B'_s \ ,$$

B_i and B'_i are distinct for some $i \leq \min\{r, s\}$; the function β is *prefix-free*.

While maybe not absolutely necessary for security (compare to [3,24,14]), the first condition is a convenient way of making the security analysis more streamlined; each new application of the CCM encryption operation will employ a fresh CBC-MAC pre-IV B_0. The second condition is not arbitrarily chosen; Petrank and Rackoff [24] have observed that CBC-MAC has attractive security properties when applied to a prefix-free message space. Note that this condition implies that (N, m, a) is uniquely determined by $\beta(N, m, a)$.

CTR encryption. In the second step of the encryption operation, we encrypt the message m and the CBC-MAC tag T in CTR mode. We use a *CTR block generator* π with four arguments $(i, N, |m|, a)$ such that the nonce N and the *counter* i (but not necessarily the message length $|m|$ and the the additional data a) are uniquely determined by the CTR block $\pi(i, N, |m|, a)$. Here, $N \in \{0,1\}^{k_n}$ and $0 \leq i \leq \mu_{\max}$, where $\mu_{\max}$ is a scheme-specific parameter bounding the maximal number of blocks in a message (note that $(\mu_{\max} + 1) \cdot 2^{k_n} \leq 2^{k_b}$). This gives the theoretical upper bound $\mu_{\max} \cdot 2^{k_n}$ on the total number of message blocks that can be encrypted during the lifetime of a key. There might be scheme-specific restrictions on the nonce that make the actual upper bound considerably smaller than the theoretical upper bound.

On input (N, m, a), the CTR input blocks $A_0, A_1, A_2, \ldots$ are defined as

$$A_i = \pi(i, N, |m|, a) \ .$$

The k_t leftmost bits of $E_K(A_0)$ are used for encryption of the CBC-MAC tag T, while the $|m|$ leftmost bits of the string $E_K(A_1) \,.\, E_K(A_2) \,.\, E_K(A_3) \,.\, \cdots$ are used for encryption of the message m. Let $\beta_0(N, m, a)$ be equal to the first block B_0 of $\beta(N, m, a)$. We require that

$$\pi(i, N, |m|, a) \neq \beta_0(N', m', a')$$

for all valid $(N, m, a), (N', m', a')$ and $0 \leq i \leq \mu_{\max}$. This is achieved if, e.g., the leftmost bit of the output from π is always 0, whereas the leftmost bit of the output from β_0 is always 1.

The nonce being non-repeating implies that *all CTR input blocks A_i and all CBC-MAC pre-IVs B_0 used during the lifetime of a key are distinct.*

2.2 CCM Specification

CCM encryption can be summarized as follows. First, the CBC-MAC tag T of $\beta(N, m, a)$ is computed. Second, the message m is encrypted in CTR mode with CTR blocks generated from the nonce N via π. Finally, the tag T is encrypted with a single CTR block.

Formally, CCM encryption is defined as follows.

CCM-ENCRYPT(N, m, a)

1. CBC-MAC computation:
   - Let $B_0 \,.\, B_1 \,.\, \cdots \,.\, B_r = \beta(N, m, a)$.
   - Let $Y_0 = E_K(B_0)$.
   - For $1 \le i \le r$, let $Y_i = E_K(Y_{i-1} \oplus B_i)$.
   - Let T be equal to the k_t leftmost bits of Y_r.
2. CTR encryption:
   - Let $\mu = \lceil |m|/k_b \rceil$.
   - For $0 \le i \le \mu$, let $A_i = \pi(i, N, |m|, a)$.
   - For $0 \le i \le \mu$, let $S_i = E_K(A_i)$.
   - Let s_m be equal to the $|m|$ leftmost bits of $S_1 \,.\, S_2 \,.\, \cdots \,.\, S_\mu$ and let S_T be equal to the k_t leftmost bits of S_0.
   - Let $c = [m \oplus s_m] \,.\, [T \oplus S_T]$.
3. Output c.

CCM decryption of a ciphertext c with the nonce N and the additional data a is defined in the obvious manner: First, apply the reverse of step 2 to c to obtain a message m and a CBC-MAC tag T (the CTR block generator π is applied to $(i, N, |c| - k_t, a)$). Next, apply CBC-MAC to $\beta(N, m, a)$ as in step 1 to obtain a CBC-MAC tag T' equal to the k_t leftmost bits of Y_r. If $T = T'$, then c is valid and m is output. Otherwise, c is not valid and an error is output. Note that the decryption operation must not release the message or any part of it until the tag has been verified. This is to prevent a chosen-ciphertext adversary from deriving useful information from invalid decryption queries.

2.3 Example

In the proposals [27] to IEEE 802.11 and [28] to NIST, CCM is based on AES with block length $k_b = 128$ and key length k_0 equal to 128, 192, or 256. All strings are assumed to be of length a multiple of 8. Before CCM can be used, we need to fix k_t, k_n, and $\mu_{\max}$. In [27,28], the tag length k_t is a multiple of 16 between 32 and 128, while the nonce length k_n is a multiple of 8 between 56 and 112. For formatting reasons, the number of octets in a message must not exceed $2^{120-k_n} - 1$; put $k_{\max} = 120 - k_n$. Note that $\mu_{\max} = 2^{k_{\max}-4}$; each block contains 2^4 octets. An input (N, m, a) is valid if and only if $N \in \{0,1\}^{k_n}$, $0 \le |a|/8 < 2^{16}$, and $0 \le |m|/8 < 2^{k_{\max}}$. [3]

The encoding function β is defined as follows on input (N, m, a). The first block B_0 is equal to

$$(0b)_2 \,.\, (k_t/16 - 1)_3 \,.\, (k_{\max}/8 - 1)_3 \,.\, (N)_{k_n} \,.\, (|m|/8)_{k_{\max}} \,.$$

[3] For simplicity, we assume that the octet length of the additional data a is small enough to fit within two octets; the proposal can handle larger values as well.

The bit b is equal to 0 if a is the empty string and 1 otherwise. If $b = 1$, then the two leftmost octets of B_1 are equal to $(|a|/8)_{16}$. Let L_a be $(|a|/8)_{16}$ if $|a| > 0$ and the empty string otherwise. Then

$$\beta(N, m, a) = B_0 \,.\, L_a \,.\, a \,.\, Z_1 \,.\, m \,.\, Z_2 \ .$$

Here, Z_1 and Z_2 are short (possibly empty) strings of zeros such that $|L_a \,.\, a \,.\, Z_1|$ and $|m \,.\, Z_2|$ are multiples of the block length 128. Note that N is uniquely determined by B_0 and that β is prefix-free; no proper prefix of $\beta(N, m, a)$ is a valid output from β, and the input (N, m, a) is uniquely determined by $\beta(N, m, a)$. Namely, the inclusion of the exact octet length of a and m in $\beta(N, m, a)$ makes it possible to extract a and m from $a \,.\, Z_1 \,.\, m \,.\, Z_2$ in an unambiguous manner.

The CTR block generator π depends only on the nonce and the counter and is defined as

$$\pi(i, N) = (00000)_5 \ .\ (k_{\max}/8 - 1)_3 \ .\ (N)_{k_n} \ .\ (i)_{k_{\max}} \ .$$

This cannot be equal to a CBC-MAC pre-IV B_0; the first five bits in B_0 are not all zeros since $k_t/16 - 1$ is nonzero.

3 Security Analysis of CCM

In this section we analyze the security of CCM. There are two aspects of security in our setting:

- Privacy: It should be infeasible for an adversary to derive any information from the ciphertexts without access to the secret key.
- Authenticity: It should be infeasible for an adversary to forge a valid ciphertext without access to the secret key.

In Section 3.1 we argue heuristically for the security of CCM. Formal definitions are provided in Section 3.2, while the main theorems are given in Section 3.3.

3.1 Heuristic Security Argument

Before analyzing CCM in greater detail, we provide a rough outline of the security properties of CCM; see next section for a detailed description of the attack models. Note that the discussion in this section is only heuristic and leaves out quite a few technical details that must not be ignored in a formal analysis. Throughout this section, we assume that the underlying permutation E_k is chosen uniformly at random from the set of all permutations.

First, consider privacy. In our setting, the goal for the adversary is to distinguish the ciphertexts from "random gibberish" (a bit string chosen uniformly at random from the set of all possible bit strings of a specified length). Let (N, m, a) be an input to the encryption operation. This operation first computes a tag T and then encrypts the message m and the tag in CTR mode. Since N is required to be fresh, the CTR input blocks and the CBC-MAC pre-IVs are new.

In particular, the output ciphertext will be very close to random gibberish even if the adversary knows the plaintext.

As we will see in the formal analysis, there are only two ways for the adversary to be successful. First, the adversary may mount a "birthday" attack against the CTR output blocks. Namely, since E_k is a block cipher and since all input blocks are distinct, there are no collisions among the CTR output blocks. However, with probability approximately $O(q^2) \cdot 2^{-b}$ (q is the number of applications of the underlying block cipher), true random gibberish will contain such block collisions. Second, the adversary may hope for an anomaly to occur within the CBC-MAC computations (e.g., an internal collision or a CBC-MAC tag that coincides with some CTR output block). In our formal analysis, we will demonstrate that the probability of any such anomaly is bounded by $O(q^2) \cdot 2^{-b}$.

Next, consider authenticity. We have already concluded that it is hard to distinguish the ciphertexts from random gibberish. In addition, it turns out that it is hard to tell anything nontrivial about the internal CBC-MAC input and output blocks even if all plaintexts are known. We will prove later that the probability that the adversary is able to extract any useful information about the internal blocks is bounded by $O(q^2) \cdot 2^{-b}$.

Unless q is very large, the adversary knows close to nothing about the internal blocks, which implies that it is close to impossible to modify any previous encryption query without having the encrypted tag modified in an unpredictable manner. Namely, since β is prefix-free, any forgery attempt is uniquely determined by the corresponding sequence $B_0 \,.\, B_1 \,.\, \cdots$ of CBC-MAC blocks. Specifically, if there is a previous encryption query with the same initial blocks as the present forgery attempt, it is still the case that there is some position on which the CBC-MAC blocks differ. The conclusion is that it is hard to guess the tag with probability better than 2^{-k_t}; whatever modification the adversary tries to make, she cannot predict the consequences.

3.2 Security Concepts

Our definitions are based on work in [2,4,6,17] and are analogous to those in [26]. CCM is a member of the family of *nonce-using symmetric encryption schemes*. Such a scheme is defined by a 4-tuple $(\mathcal{K}, \mathcal{E}, \mathcal{D}, k_n)$. Here, k_n is an integer (the nonce length) and $\mathcal{K}$ is the *key space*. In our setting, $\mathcal{E}$ and $\mathcal{D}$ are functions

$$\mathcal{K} \times \{0,1\}^{k_n} \times \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^* \cup \{\phi\}$$

(ϕ = "Error") such that

$$\mathcal{D}(K,N,c,a) = \begin{cases} m \text{ if } c = \mathcal{E}(K,N,m,a) \text{ for some (unique) } m; \\ \phi \text{ if } c \neq \mathcal{E}(K,N,m,a) \text{ for all } m. \end{cases}$$

We will write $\mathcal{E}_K(N,m,a) = \mathcal{E}(K,N,m,a)$ and similarly for $\mathcal{D}$. We assume that the bit length of $\mathcal{E}_K(N,m,a)$ is uniquely determined by the bit lengths of m and a (the bit length of N is fixed to k_n).

We define privacy and authenticity of a nonce-using symmetric encryption scheme $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D}, k_n)$ in terms of two attack experiments. In each of the two experiments, a key K is first chosen uniformly at random from $\mathcal{K}$. We proceed as follows in a manner similar to the approach in [26], except that we allow the adversary against authenticity to make several forgery attempts.

Privacy. In the privacy experiment, the adversary $\mathcal{A}$ has access to an encryption oracle $\mathcal{O}$ that on input (N, m, a) returns a ciphertext c. $\mathcal{A}$ may send arbitrary queries to the oracle, except that the same nonce must not be used in more than one query; such a query is immediately rejected by the oracle. Thus we restrict our attention to *nonce-respecting* adversaries.

The encryption oracle is chosen from a set of two possible oracles via a fair coin flip $b \in \{0, 1\}$. If $b = 1$, then the oracle is the true oracle $\mathcal{E}_K$. If $b = 0$, then the oracle is a random oracle $\mathcal{R}$ that on input (N, m, a) returns a string of length $|\mathcal{E}_K(N, m, a)|$ chosen uniformly at random. By assumption, $|\mathcal{E}_K(N, m, a)|$ depends only on $|m|$ and $|a|$. The goal for $\mathcal{A}$ is to guess the bit b; a correct guess would mean that she is able to distinguish $\mathcal{E}_K$ from a true random number generator. We define the advantage of $\mathcal{A}$ against the privacy of Π as

$$\mathrm{Adv}^{\mathsf{priv}}_{\Pi}(\mathcal{A}) = \left| \Pr_{K \leftarrow \mathcal{K}}(1 \leftarrow \mathcal{A}^{\mathcal{E}_K}) - \Pr(1 \leftarrow \mathcal{A}^{\mathcal{R}}) \right| \ ;$$

$1 \leftarrow \mathcal{A}^{\mathcal{O}}$ denotes the event that $\mathcal{A}$ outputs 1 conditioned that the underlying oracle is $\mathcal{O}$. □

Authenticity. In the authenticity experiment, $\mathcal{A}$ has access to the true encryption oracle $\mathcal{E}_K$ and to the true decryption oracle $\mathcal{D}_K$. Queries to the decryption oracle will be referred to as *forgery attempts*. As in the privacy experiment, we assume that the adversary is nonce-respecting when making encryption queries. However, there are no such restrictions on forgery attempts. The goal for $\mathcal{A}$ is to produce a forgery attempt (N^*, c^*, a^*) such that $\mathcal{D}_K(N^*, c^*, a^*) \neq \phi$; if this is true, then $\mathcal{A}$ *forges*. The only restriction on (N^*, c^*, a^*) is that there must not be any previous encryption query (N^*, m, a^*) with response c^*. However, N^* may well be part of a previous encryption query and one or several previous forgery attempts. $\mathcal{A}$ may send her encryption queries and forgery attempts in any order and at any time during the experiment. We define the advantage of $\mathcal{A}$ against the authenticity of Π as

$$\mathrm{Adv}^{\mathsf{auth}}_{\Pi}(\mathcal{A}) = \Pr_{K \leftarrow \mathcal{K}}(\mathcal{A}^{\mathcal{E}_K} \text{ forges}) \ . \ \square$$

The reason for accepting several forgery attempts (as opposed to one single forgery attempt at the end of the experiment) is that we want to analyze how the number of forgery attempts affects the success probability of an adversary. Specifically, we want to show that the adversary does not gain more than negligibly from making multiple forgery attempts if the tag length is considerably larger than $k_b/2$.

Our goal is to relate the security of CCM to the hardness of distinguishing the underlying function E_K from a random function (and, if E is a block cipher, from a random permutation). Let $\mathsf{Rand}(k_b)$ be the set of all functions $f : \{0,1\}^{k_b} \to \{0,1\}^{k_b}$ and let $\mathsf{Perm}(k_b)$ be the subset of $\mathsf{Rand}(k_b)$ consisting of all permutations. First consider indistinguishability from a random function.

PRF indistinguishability. The attack experiment for E_K is very similar to the privacy experiment for Π above. In the first step of the experiment a key K is chosen uniformly at random from $\mathcal{K}$ and an oracle $\mathcal{O}$ is chosen via a fair coin flip b. If $b = 0$, then $\mathcal{O}$ is a function ρ selected uniformly at random from $\mathsf{Rand}(k_b)$. If $b = 1$, then $\mathcal{O}$ is E_K. The adversary $\mathcal{B}$ is given access to the oracle $\mathcal{O}$ and is allowed to send arbitrary queries. The goal for the adversary is to guess the bit b. We define

$$\mathrm{Adv}_E^{\mathsf{prf}}(\mathcal{B}) = \left| \Pr_{K \leftarrow \mathcal{K}}(1 \leftarrow \mathcal{B}^{E_K}) - \Pr_{\rho \leftarrow \mathsf{Rand}(k_b)}(1 \leftarrow \mathcal{B}^{\rho}) \right| . \square$$

PRP indistinguishability. We now consider indistinguishability from a random permutation. The only modification from the previous experiment is that ρ is selected from $\mathsf{Perm}(k_b)$ instead of $\mathsf{Rand}(k_b)$; we define

$$\mathrm{Adv}_E^{\mathsf{prp}}(\mathcal{B}) = \left| \Pr_{K \leftarrow \mathcal{K}}(1 \leftarrow \mathcal{B}^{E_K}) - \Pr_{\rho \leftarrow \mathsf{Perm}(k_b)}(1 \leftarrow \mathcal{B}^{\rho}) \right| . \square$$

Before we proceed, we state a useful result regarding the correspondence between PRF and PRP indistinguishability; see [3] for a proof.

Lemma 1. *Let E be a block cipher; E_K is a permutation for each key $K \in \mathcal{K}$. Then for any PRF distinguisher $\mathcal{B}$ making q queries to his oracle, there is a PRP distinguisher $\hat{\mathcal{B}}$ making q queries to his oracle such that*

$$Adv_E^{\mathsf{prf}}(\mathcal{B}) \leq Adv_E^{\mathsf{prp}}(\hat{\mathcal{B}}) + q(q-1) \cdot 2^{-k_b - 1} .$$

The running time for $\hat{\mathcal{B}}$ is the running time for $\mathcal{B}$ plus the time needed to transport q queries and q responses between $\mathcal{B}$ and $\hat{\mathcal{B}}$'s oracle. $\square$

3.3 Security Results

We now present lower bounds for the security of CCM in terms of the underlying function E_K. First consider authenticity. For an encryption query $Q = (N, m, a)$, define

$$l_Q = \left\lceil \frac{|\beta(N, m, a)| + |m|}{k_b} \right\rceil + 1 ;$$

l_Q is the total number of applications of the block cipher needed to respond to the query Q. For a forgery attempt $Q^* = (N^*, c^*, a^*)$ corresponding to the message m^*, define

$$l_{Q^*} = \left\lceil \frac{|\beta(N^*, m^*, a^*)| + |c^*|}{k_b} \right\rceil + 1 ;$$

l_{Q^*} is the total number of applications of the block cipher needed to decrypt c^* and check whether c^* is valid.

Theorem 1. *Let $\mathcal{A}$ be an adversary against the authenticity of CCM. Let q_E be the number of encryption queries and let $Q_1, Q_2, \ldots, Q_{q_E}$ denote the queries. Let q_F be the number of forgery attempts and let $Q_1^*, Q_2^*, \ldots, Q_{q_F}^*$ denote the attempts. Put*

$$l_E = \sum_i l_{Q_i} \quad \text{and} \quad l_F = \sum_i l_{Q_i^*} \ .$$

Then there is a PRF distinguisher $\mathcal{B}$ such that

$$\mathrm{Adv}^{\mathsf{auth}}_{\mathsf{CCM}}(\mathcal{A}) \leq \mathrm{Adv}^{\mathsf{prf}}_{E}(\mathcal{B}) + q_F \cdot 2^{-k_t} + (l_E + l_F)^2 \cdot 2^{-k_b - 1} \ .$$

Thus, by Lemma 1, if E is a block cipher, then there is a PRP distinguisher $\hat{\mathcal{B}}$ such that

$$\mathrm{Adv}^{\mathsf{auth}}_{\mathsf{CCM}}(\mathcal{A}) \leq \mathrm{Adv}^{\mathsf{prp}}_{E}(\hat{\mathcal{B}}) + q_F \cdot 2^{-k_t} + (l_E + l_F)^2 \cdot 2^{-k_b} \ .$$

Both distinguishers have an additional running time equal to the time needed to process the queries from $\mathcal{A}$. This includes making $l_E + l_F$ oracle queries and xoring $l_E - q_E + l_F - q_F$ pairs of blocks of size k_b.

The proof of Theorem 1 is given in Appendix A.

Now consider privacy.

Theorem 2. *Let $\mathcal{A}$ be an adversary against the privacy of CCM. Let q_E and l_E be defined as in Theorem 1. Then there is a PRF distinguisher $\mathcal{B}$ such that*

$$\mathrm{Adv}^{\mathsf{priv}}_{\mathsf{CCM}}(\mathcal{A}) \leq \mathrm{Adv}^{\mathsf{prf}}_{E}(\mathcal{B}) + l_E^2 \cdot 2^{-k_b - 1} \ .$$

Thus, by Lemma 1, if E is a block cipher, then there is a PRP distinguisher $\hat{\mathcal{B}}$ such that

$$\mathrm{Adv}^{\mathsf{priv}}_{\mathsf{CCM}}(\mathcal{A}) \leq \mathrm{Adv}^{\mathsf{prp}}_{E}(\hat{\mathcal{B}}) + l_E^2 \cdot 2^{-k_b} \ .$$

Both distinguishers have an additional running time equal to the time needed to process the queries from $\mathcal{A}$. This includes making l_E oracle queries and xoring $l_E - q_E$ pairs of blocks of size k_b.

The proof of Theorem 2 is given in Appendix B.

3.4 Possible Extensions Beyond the Birthday Paradox

The security bounds in the previous section include a term of the form $c \cdot l^2 \cdot 2^{-k_b}$, where c is a small constant, l is the number of applications of the underlying block cipher, and k_b is the block size. This term is closely related to the "birthday paradox", which states that a collision $X_i = X_j$ is likely to be found among l uniformly random bit strings $X_1, \ldots, X_l$ of length k_b if l is approximately $2^{k_b/2}$.

In some situations, it would be desirable to have assurance that CCM remains secure beyond the birthday paradox bound. While this cannot be true for the

confidentiality of CCM, we still hope to find an authenticity bound that is linear rather than quadratic in the number of applications of the block cipher.

As a comparison, consider the recent paper [14], which elaborates on a variant of CBC-MAC called RMAC. The RMAC construction is a message authentication code based on CBC-MAC that is provably secure against birthday paradox attacks. RMAC is similar to CCM in that both schemes encrypt the CBC-MAC tag. While the encryption method in RMAC is substantially stronger than that in CCM, we still conjecture that the CTR encryption of the CBC-MAC tag is strong enough to thwart birthday attacks.

We summarize the problem to be solved as follows.

Problem 1 *Let notations be as in Theorem 1 with $\mathcal{A}$ being an adversary against the authenticity of CCM, and assume that E is a block cipher. Is there a PRP distinguisher $\hat{\mathcal{B}}$ with approximately the same running time as $\mathcal{A}$ such that*

$$\mathrm{Adv}^{\mathsf{auth}}_{\mathsf{CCM}}(\mathcal{A}) \leq \mathrm{Adv}^{\mathsf{prp}}_{E}(\hat{\mathcal{B}}) + q_F^{1+o(1)} \cdot 2^{-k_t} + (l_E + l_F)^{1+o(1)} \cdot 2^{-k_b} \ ?$$

Ferguson [9] has demonstrated that the corresponding conjecture for OCB mode [26] is false; there is an attack against the authenticity of OCB with complexity very close to the lower security bound established in [26]. In this context, it is worth mentioning TAE, an interesting variant of OCB with excellent security bounds based on a "tweakable" block cipher; see [19] for details. It remains to be examined whether there exists a variant of CCM based on a "tweakable" block cipher with just as good security bounds.

Acknowledgments

I thank Burt Kaliski, Tadayoshi Kohno, and the CCM designers Niels Ferguson, Russ Housley, and Doug Whiting for valuable comments and fruitful discussions.

References

1. J. H. An and M. Bellare. Does Encryption with Redundancy Provide Authenticity? *Advances in Cryptology – EUROCRYPT 2001*, pp. 512 – 528, Springer Verlag, 2001.
2. M. Bellare, A. Desai, E. Jokipii, and P. Rogaway. A Concrete Security Treatment of Symmetric Encryption: Analysis of the DES Modes of Operation. *Proceedings of 38th Annual Symposium on Foundations of Computer Science (FOCS 97)*, IEEE, 1997.
3. M. Bellare, J. Kilian, P. Rogaway. The Security of the Cipher Block Chaining Message Authentication Code. *Journal of Computer and System Sciences*, 61 (3), 362 – 399, 2000.
4. M. Bellare and C. Namprempre. Authenticated Encryption: Relations Among Notions and Analysis of the Generic Composition Paradigm. *Advances in Cryptology – ASIACRYPT 2000*, pp. 531 – 545, Springer-Verlag, 2000.
5. M. Bellare and P. Rogaway. Optimal Asymmetric Encryption – How to Encrypt with RSA. *Advances in Cryptology – Eurocrypt '94*, pp. 92 – 111, Springer Verlag, 1994.

6. M. Bellare and P. Rogaway. Encode-Then-Encipher Encryption: How to Exploit Nonces or Redundancy in Plaintexts for Efficient Encryption. *Advances in Cryptology – ASIACRYPT 2000*, pp. 317 – 330, Springer-Verlag, 2000.
7. J. Daemen and V. Rijmen. *AES Proposal: Rijndael*. Contribution to NIST, September 1999. Available from csrc.nist.gov/encryption/aes/rijndael/.
8. W. Diffie and M. Hellman. Privacy and Authentication: An Introduction to Cryptography. *Proceedings of the IEEE*, 67, pp. 397 – 427, 1979.
9. N. Ferguson. *Collision Attacks on OCB*. Preprint, February 2002.
10. V. Gligor, P. Donescu. *Infinite Garble Extension*. Contribution to NIST, 2000. Available from csrc.nist.gov/encryption/modes/proposedmodes/.
11. H. Handschuh and D. Naccache. *SHACAL*. Contribution to the NESSIE project, 2000.
12. IEEE Std 1363-2000. *Standard Specifications for Public Key Cryptography*. IEEE, 2000.
13. *ISO/IEC 9797: Information Technology – Security Techniques – Data Integrity Mechanism Using a Cryptographic Check Function Employing a Block Cipher Algorithm*. Second edition, 1994.
14. É. Jaulmes, A Joux and F. Valette. On the Security of Randomized CBC-MAC Beyond the Birthday Paradox Limit – A New Construction. *Fast Software Encryption, 9th International Workshop, FSE 2002*, to appear.
15. C. S. Jutla. *Encryption Modes with Almost Free Message Integrity*. Contribution to NIST, 2000. Available from csrc.nist.gov/encryption/modes/proposedmodes/.
16. C. S. Jutla. *Parallelizable Encryption Mode with Almost Free Message Integrity*. Contribution to NIST, 2000.
Available from csrc.nist.gov/encryption/modes/proposedmodes/.
17. J. Katz and M. Yung. Unforgeable Encryption and Chosen-Ciphertext-Secure Modes of Operation. *Fast Software Encryption 2000*, pp. 284-299, 2000.
18. H. Krawczyk. The Order of Encryption and Authentication for Protecting Communications (or: How Secure Is SSL?). *Advances in Cryptology – CRYPTO 2001*, pp. 310 – 331, Springer Verlag, 2001.
19. M. Liskov, R. L. Rivest and D. Wagner. *Tweakable Block Ciphers. Advances in Cryptology – CRYPTO 2002*, Springer Verlag, 2002.
20. A. Menezes, P. van Oorschot and S. Vanstone. *Handbook of Applied Cryptography*. CRC Press, 1996.
21. National Institute of Standards and Technology (NIST). *FIPS Publication 81: DES Modes of Operation*. December 1980.
22. National Institute of Standards and Technology (NIST). *FIPS Publication 180-1: Secure Hash Standard (SHS)*. April 1995.
23. National Institute of Standards and Technology (NIST). *FIPS Publication 197: Advanced Encryption Standard (AES)*. November 2001.
24. E. Petrank, C. Rackoff. CBC MAC for Real-Time Data Sources. *Journal of Cryptology*, 13 (3), pp. 315–338, 2000.
25. P. Rogaway. *IEEE 802.11-01/156r0: Some Comments on WHF Mode*. March 2002. Available from www.cs.ucdavis.edu/~rogaway/ocb/ocb-doc.htm.
26. P. Rogaway, M. Bellare, J. Black and T. Krovetz. OCB: A Block-Cipher Mode of Operation for Efficient Authenticated Encryption. *8th ACM Conference on Computer and Communications Security (CCS-8)*, pp. 196-205. ACM Press, 2001.
27. D. Whiting, R. Housley and N. Ferguson. *IEEE 802.11-02/001r2: AES Encryption & Authentication Using CTR Mode & CBC-MAC*. March 2002.

28. D. Whiting, R. Housley and N. Ferguson. *Counter with CBC-MAC (CCM), AES Mode of Operation.* Contribution to NIST, May 2002.
Available from http://csrc.nist.gov/encryption/modes/proposedmodes/.

A Proof of Theorem 1

We want to relate an adversary $\mathcal{A}$ against the authenticity of CCM to a PRF distinguisher $\mathcal{B}$ attacking the underlying function E_K. $\mathcal{B}$ is given access to an oracle $\mathcal{O}$ equal to either E_K or a random function ρ, each with probability 1/2. $\mathcal{B}$ is able to respond to encryption queries from $\mathcal{A}$ if $\mathcal{O} = E_K$. In this case $\mathcal{B}$ provides a perfect simulation of $\mathcal{E}_K$; the responses are exactly the same as those provided by the true encryption oracle $\mathcal{E}_K$.

However, if $\mathcal{O} = \rho$, then the simulation is no longer perfect. Still, the experiment can be run; $\mathcal{A}$ may or may not be able forge at the end. Let $\mathcal{B}$ output 1 if $\mathcal{A}$ forges (with respect to an oracle based on $\mathcal{O}$) and 0 otherwise. Note that

$$\begin{aligned}\mathrm{Adv}^{\mathsf{auth}}_{\mathsf{CCM}}(\mathcal{A}) &= \Pr_{K \leftarrow \mathcal{K}}(1 \leftarrow \mathcal{B}^{E_K}) \le \mathrm{Adv}^{\mathsf{prf}}_{E}(\mathcal{B}) + \Pr_{\rho \leftarrow \mathsf{Rand}(k_b)}(1 \leftarrow \mathcal{B}^{\rho}) \\ &= \mathrm{Adv}^{\mathsf{prf}}_{E}(\mathcal{B}) + \Pr_{\rho \leftarrow \mathsf{Rand}(k_b)}(\mathcal{A}^{\rho} \text{ forges}) \ ;\end{aligned}$$

$\mathcal{A}^{\rho}$ denotes $\mathcal{A}$ with a CCM encryption oracle based on ρ. Thus we need to give a bound for the probability that an adversary $\mathcal{A}$ forges conditioned that the underlying function is selected uniformly at random from $\mathsf{Rand}(k_b)$. To achieve such a bound, we need to demonstrate how to simulate $\mathcal{A}$'s encryption oracle. This is achieved if we are able to simulate the underlying random function ρ. We will also need to simulate the decryption oracle to check whether the forgery attempts by the adversary are valid or not. However, for the moment we restrict our attention to the encryption oracle simulation.

A true simulation of the function ρ would be as follows. The algorithm RF-Simulation (RF = Random Function) takes as input the block X to be processed and a list $\mathcal{L}$ containing all pairs (X', Y') corresponding to previous applications of ρ; $Y' = \rho(X')$.

RF-Simulation$(X, \mathcal{L})$

1. If X is the first block in some entry on the list $\mathcal{L}$, then $Y = \rho(X)$ is already defined; output Y and exit.
2. Generate a string Y uniformly at random.
3. Add (X, Y) to the list $\mathcal{L}$ and output Y.

To simplify analysis, we introduce a modified simulation by skipping step 1 and the list $\mathcal{L}$; RO-Simulation (RO = Random Output) is defined as follows, in the easiest possible manner.

RO-Simulation(X)

1. Generate a string Y uniformly at random and output Y.

This simulation fails if and only if some block appears twice as input to RO-Simulation and the two corresponding outputs are distinct (this makes the definition of ρ inconsistent). Let RO-InColl be the even that some input X appears twice; $\Pr(\text{RO-InColl})$ is at least the probability of a simulation failure. Before we can estimate the probability of RO-InColl, we need to examine the structure of the input blocks to RO-Simulation. This is done as follows.

Let $\mathcal{X}$ be the multi-set of all input blocks to ρ needed when responding to encryption queries. $\mathcal{X}$ being a multi-set means that if some input block appears several times during the simulation, then this block appears just as many times in $\mathcal{X}$. We claim that the following holds for RO-Simulation.

Claim. The set $\mathcal{X}$ of input blocks can be divided into two sets $\mathcal{X}_1$ and $\mathcal{X}_2$. $\mathcal{X}_1$ is the set of input blocks derived from the nonces (i.e., the CTR blocks and the CBC-MAC pre-IV), while $\mathcal{X}_2$ is the set of input blocks occurring in the internal CBC-MAC computations. All blocks in the first set $\mathcal{X}_1$ are distinct and known to the adversary. All blocks in the second set $\mathcal{X}_2$ are, from the view of the adversary, mutually independent and uniformly distributed among all possible blocks. Also, the sets $\mathcal{X}_1$ and $\mathcal{X}_2$ are independent. In particular, the ciphertexts given to the adversary leak no information about the blocks in the set $\mathcal{X}_2$.

Proof of Claim. We use induction over the number of encryption queries to prove that the claim holds. Thus assume that after a certain number $q \geq 0$ of queries, the set $\mathcal{X}'$ containing the corresponding input blocks can be divided into two sets $\mathcal{X}'_1$ and $\mathcal{X}'_2$ as above. Consider a new encryption query (N, m, a) with blocks $A_0, \ldots, A_\mu, B_0$ derived from N. Since N is a new nonce never used before, the set $\mathcal{X}'_1$ does not contain any of the blocks $A_0, \ldots, A_\mu, B_0$. Also, N is independent from all elements in $\mathcal{X}'_2$ since these are completely unknown to the adversary and since N is generated in a predetermined fashion. Thus adding $A_0, \ldots, A_\mu, B_0$ to $\mathcal{X}'_1$ does not violate the desired properties.

Now, consider the CBC-MAC computation of

$$\beta(N, m, a) = B_0 \,.\, B_1 \,.\, \cdots \,.\, B_r \;.$$

By construction, the first output block Y_0 in the CBC-MAC computation is a uniformly distributed string. Namely, RO-Simulation does not check for input collisions (with the true simulation, Y_0 would not necessarily be uniformly distributed). In particular, Y_0 is independent from B_1, which implies that $Y_0 \oplus B_1$ is independent from other input blocks and uniformly distributed. Similarly, Y_{i-1} is uniformly distributed for $i > 1$, which implies that $Y_{i-1} \oplus B_i$ is independent from other input blocks and uniformly distributed. Also, the CBC-MAC tag T (which is Y_r truncated to k_t bits) and the part of the CTR output stream to which T is xored are both uniformly distributed and do not leak any information about the internal blocks. Hence, all internal CBC-MAC input blocks can be added to $\mathcal{X}'_2$ without violating the desired properties.

Using induction, we conclude that $\mathcal{X}$ has the desired property. □

As a consequence, we may now compute the probability of the event RO-InColl. Let $x_1 = |\mathcal{X}_1|$ and $x_2 = |\mathcal{X}_2|$ (the sets are still viewed as multi-sets); $x_1 + x_2 = l_E$. The probability that there is an input block that appears twice in the simulation is bounded by

$$\mathsf{RO\text{-}InColl} \leq (x_1 x_2 + x_2(x_2 - 1)/2) \cdot 2^{-k_b} \leq l_E(l_E - 1)/2 \cdot 2^{-k_b} \ ; \qquad (1)$$

This follows immediately from the facts that the elements in $\mathcal{X}_1$ are all different and that the elements in $\mathcal{X}_2$ are uniformly distributed, mutually independent, and independent from $\mathcal{X}_1$. Thus, from the view of the adversary, every two elements in $\mathcal{X}$ coincide with probability either 0 (if both elements are in $\mathcal{X}_1$) or 2^{-k_b} (otherwise).

Note that total secrecy of the elements in $\mathcal{X}_2$ is essential for the proof; otherwise the adversary might be able to increase the failure probability for the simulation by selecting new encryption queries on the basis of leaked information about $\mathcal{X}_2$.

Now consider forgery attempts. We may assume that $\mathcal{B}$ always responds with ϕ on any forgery attempt, even if the forgery attempt turns out to be valid with respect to $\mathcal{B}$'s oracle. Namely, if $\mathcal{A}$ forges with a specific query Q^*, then $\mathcal{A}$ wins, so $\mathcal{B}$'s response does not matter. If Q^* is invalid, then $\mathcal{B}$'s response coincides with the true oracle.

Using this flawed simulation of the decryption oracle, we obtain a model where the actual success of a specific forgery attempt does not depend on the responses to other forgery attempts (as these are always ϕ), only on the responses to the encryption queries. The conclusion is that we may analyze each forgery attempt separately. In addition, we may assume that all forgery attempts are made at the end of the attack (see [14] for further discussion), with the following exception:

In case $\mathcal{A}$ makes a forgery attempt Q^* corresponding to a certain triple (N, m, a) and later queries the very same triple (N, m, a) at the encryption oracle (e.g., by pure coincidence or by choosing a very short m that can be guessed), we cannot make the assumption that $\mathcal{A}$ waits until the end before she queries Q^*. Namely, she is not allowed to make forgery attempts corresponding to previous encryption queries in this manner. Still, by construction, the responses to later encryption queries are completely unpredictable, which implies that Q^* cannot be valid with better probability than 2^{-k_t}; all encrypted tags are equally likely.

From now on, consider forgery attempts that do not correspond to later encryption queries in the manner just described. For such forgery attempts, we may assume that all encryption queries are already made. We need to compute the probability that a specific forgery attempt is successful. Let $Q^* = (N^*, c^*, a^*)$ be such an attempt and let m^* be the corresponding message (note that we may not know m^* yet).

If N^* is not part of any encryption query, then we may apply the decryption operation in the same manner as we applied the encryption operation on encryption queries, thus generating a uniformly distributed output block for each

application of ρ. With the same argument as before, it is easily seen that this simulation will fail with probability at most

$$(l_{Q^*}(l_{Q^*}-1)/2 + l_E l_{Q^*}) \cdot 2^{-k_b} \ . \tag{2}$$

Namely, each of the input blocks in the decryption computation is equal to some input block appearing in an encryption query with probability at most $l_E \cdot 2^{-k_b}$; there are l_{Q^*} such blocks. With probability at most $l_{Q^*}(l_{Q^*}-1)/2 \cdot 2^{-k_b}$, there is an input collision within the decryption computation. The success probability for the adversary is obviously 2^{-k_t} since each encrypted tag has the same probability.

Now, assume that N^* is part of an encryption query $Q = (N^*, m, a)$. Extract m^* from c^* by applying ρ to the relevant CTR blocks derived from N^*. Some of these blocks might be part of the computations needed to respond to the encryption query Q; ρ is already defined on these blocks. Define E_k on the other blocks in the usual manner as uniformly distributed output blocks. The resulting message m^* is clearly independent from the set $\mathcal{X}_2$.

We identify two cases; the first block B_0 in $\beta(N^*, m, a)$ and the first block B_0^* in $\beta(N^*, m^*, a^*)$ are either equal or different.

- First, assume that $B_0 \neq B_0^*$. This implies that B_0^* is not part of $\mathcal{X}_1$. In particular, we may simulate the CBC-MAC computation in the same random manner as usual. Again, the error probability for this simulation is bounded by (2) and the success probability for the adversary is 2^{-k_t}. Namely, the CBC-MAC tag is uniformly distributed and independent from the block that is xored to it to form the encrypted tag T.
- Second, assume that $B_0 = B_0^*$. Write

$$\beta(N^*, m, a) = B_0 \,.\, \cdots \,.\, B_r \ ;$$
$$\beta(N^*, m^*, a^*) = B_0^* \,.\, \cdots \,.\, B_{r^*}^* \ .$$

  Let $i \leq \min\{r, r^*\}$ be the smallest index such that B_i and B_i^* are different; such an i exists since β is prefix-free. Let Y_{i-1} be the corresponding block that is xored to B_i and B_i^*, respectively, within the CBC-MAC computation. Put $X_i = Y_{i-1} \oplus B_i$ and $X_i^* = Y_{i-1} \oplus B_i^*$. By assumption, X_i is an element in $\mathcal{X}_2$ that is completely unknown to the adversary and independent from all other elements in $\mathcal{X}$. Also, B_i^* is independent from all elements in $\mathcal{X}_2$ (namely, the adversary has no information about $\mathcal{X}_2$). In particular, X_i^* is uniformly distributed and independent from all elements in $\mathcal{X} \setminus \{X_i\}$ and different from X_i. Thus we may proceed in the usual manner from this point with the CBC-MAC computation of $B_0^* \,.\, B_1^* \,.\, \cdots \,.\, B_r^*$, generating random output blocks $Y_i^*, \ldots, Y_{r^*}^*$. As before, the error probability for this simulation is bounded by (2), and the success probability for the adversary is 2^{-k_t}.

Summing over all forgery attempts, we conclude that the success probability for the adversary within this model based on RO-Simulation is at most $q_F \cdot 2^{-k_t}$ and that the probability of decryption oracle simulation failure is bounded by

$$\sum_i \left(l_{Q_i^*}(l_{Q_i^*}-1)/2 + l_E l_{Q_i^*}\right) \cdot 2^{-k_b} \leq (l_F(l_F-1)/2 + l_E l_F) \cdot 2^{-k_b} \ ; \tag{3}$$

this follows from (2). The total probability of simulation failure is bounded by the sum of (1) and (3), from which we conclude that the probability of success for the adversary within the true model based on RF-Simulation is bounded by

$$\begin{aligned} & q_F \cdot 2^{-k_t} + l_E(l_E - 1)/2 \cdot 2^{-k_b} + (l_F(l_F - 1)/2 + l_E l_F) \cdot 2^{-k_b} \\ < \; & q_F \cdot 2^{-k_t} + (l_E + l_F)^2 \cdot 2^{-k_b - 1} \ , \end{aligned}$$

which is the desired bound. □

B Proof of Theorem 2

We want to relate an adversary $\mathcal{A}$ against the privacy of CCM to a PRF distinguisher $\mathcal{B}$ attacking the underlying function E_K. The prerequisites are the same as in the proof of Theorem 1. In addition, introduce a random oracle $\mathcal{R}'$ for $\mathcal{B}$ that on any input block X outputs a uniformly random output block Y (without checking for consistency with previous queries). This oracle is the oracle simulated by RO-Simulation in the proof of Theorem 1. It is easily seen that $\mathcal{B}$ is able to provide a perfect simulation of $\mathcal{R}$ with probability 1 using the oracle $\mathcal{R}'$. Thus we obtain that

$$\begin{aligned} \mathrm{Adv}^{\mathsf{priv}}_{\mathsf{CCM}}(\mathcal{A}) &= \left| \Pr_{K \leftarrow \mathcal{K}}(1 \leftarrow \mathcal{B}^{E_K}) - \Pr(1 \leftarrow \mathcal{B}^{\mathcal{R}'}) \right| \\ &\leq \mathrm{Adv}^{\mathsf{prf}}_{E}(\mathcal{B}) + \left| \Pr_{\rho \leftarrow \mathsf{Rand}(k_b)}(1 \leftarrow \mathcal{B}^{\rho}) - \Pr(1 \leftarrow \mathcal{B}^{\mathcal{R}'}) \right| \ . \end{aligned}$$

Hence we need to compute the probability that ρ, selected uniformly at random from $\mathsf{Rand}(k_b)$, does not provide a perfect simulation of the random oracle $\mathcal{R}'$. This is the case only if $\mathcal{B}$ is asked to query the same input block twice. From the proof of Theorem 1 we learn that this happens with probability at most $l_E^2 \cdot 2^{-k_b}$, which gives the desired bound. □