On Security of TrCBC

Debrup Chakraborty and Samir Kundu

Indian Statistical Institute
203 B.T. Road. Kolkata 700108, India
debrup.chakraborty@gmail.com, samirkundu3@gmail.com

Abstract. TrCBC is a variant of CBC-MAC which appeared in *Information Processing Letters*, 112(7):302-307, 2012. The authors claimed TrCBC to be a secure message authentication code (MAC) with some interesting properties. If TrCBC is instantiated with a block cipher with block length n, then it requires $\lceil \lambda/n \rceil$ block cipher calls for authenticating a λ-bit message and requires a single key, which is the block cipher key. The authors state that TrCBC can have tag lengths of size less than $n/2$. We show that with high probability, an adversary can forge TrCBC with tag length $n/2-1$ with just three queries. The attack that we show can be applied to forge a large class of messages. The authors proved TrCBC to be a pseudorandom function (PRF). A scrutiny of the claimed PRF bound shows that for some recommended values of tag lengths, the bound turns out to be quite large. Thus, the security theorem does not imply security of TrCBC for all recommended tag lengths.

1 Introduction

CBC-MAC and its variants are widely used and are parts of different standards. It is known that the basic CBC-MAC is not secure as a variable input length MAC; more precisely, CBC-MAC is only secure if the message space is prefix-free [8], i.e., the message space does not contain any two messages where one is a prefix of the other. In case the message space is not prefix-free, a simple length extension attack [1] can be performed to obtain a forgery with probability 1. But, CBC-MAC is optimal in terms of the number of block-cipher calls. If CBC-MAC is instantiated with a block-cipher of block length n, then to authenticate a message of λ bits, it requires a single key (which is the key of the underlying block cipher) and $\lceil \lambda/n \rceil$ block-cipher calls. This makes CBC-MAC a good choice for applications where only fixed-length messages are required to be authenticated.

Over the years, several modifications over the basic CBC-MAC have been proposed to accommodate general message spaces. Some notable constructions in this direction are EMAC [8], XCBC[2], CMAC [3], TMAC [6], OMAC [5], GCBC1 [7], GCBC2 [7]. All these variants require some extra overhead compared to the basic CBC-MAC either in terms of the number of keys used and/or in the number of block-cipher calls required.

TrCBC is a variant of CBC-MAC proposed in [9]. The motivation of TrCBC construction was to provide a CBC-MAC like message authentication code which works for general message spaces but whose overhead in terms of the number of keys and block cipher calls is exactly the same as the CBC-MAC. In [9], it is claimed that TrCBC achieves this with the limitation that it can produce only short tags whose lengths are less than $n/2$-bits, where

n is the block-length of the underlying block cipher. The main idea of the construction is to truncate the output of CBC-MAC. Truncated CBC-MACs have been analyzed in [4] through a construction called TCBC which is quite different from TrCBC. We discuss more about TCBC later in the paper.

In this paper we show that TrCBC is insecure. A variant of the length extension attack can be mounted on TrCBC which produces $(n/2-1)$-bit tags with a success probability of $1/4$. The basic attack can be extended to make it work for a large class of messages.

The authors of [9] also prove an upper bound for the PRF advantage of an adversary for TrCBC. The PRF bound that the authors' proof does not suggest TrCBC to be a PRF where the tag length is $(n/2-1)$-bits. We analyze the PRF bound and conclude that the security theorem for TrCBC does not really imply that TrCBC is a secure MAC for all suggested tag lengths.

2 Notations and Preliminaries

General Notations. For a natural number N, $[N]$ denotes the set $\{1, \ldots, \mathrm{N}\}$. For a set $\mathcal{X}$, $X \xleftarrow{\$} \mathcal{X}$ denotes that X is chosen uniformly at random from $\mathcal{X}$. The set of all binary strings is denoted by $\{0,1\}^*$, which includes the empty string ϵ. The set of all n-bit strings is denoted by $\{0,1\}^n$. $\mathsf{Perm(n)}$ denotes the set of all permutations over $\{0,1\}^n$ and $\mathsf{Func}(*,\tau)$ denotes the set of all functions from $\{0,1\}^*$ to $\{0,1\}^\tau$. For $x, y \in \{0,1\}^*$, $x\|y$ denotes the concatenation of x and y. For $x \in \{0,1\}^*$ the length of x is denoted by $|x|$. For any $x \in \{0,1\}^*$, $x \neq \epsilon$, $\mathsf{parse}_n(x)$ parses x as $x_1\|x_2\|\ldots\|x_\ell$ where for $i \in [\ell-1]$, $|x_i| = n$ and $0 < |x_\ell| \leq n$. We define $\mathsf{parse}_n(\epsilon) = x_1$, where $x_1 = \epsilon$, i.e, $|x_1| = 0$. For $x \in \{0,1\}^*$ and $|x| \leq n$, $\mathsf{Pad}(x) = x\|10^{n-|x|-1}$, if $|x| < n$, and $\mathsf{Pad}(x) = x$, if $|x| = n$. For $x \in \{0,1\}^*$ and $|x| \geq \tau$, $\mathsf{MSB}_\tau(x)$ and $\mathsf{LSB}_\tau(x)$ stand for the most and least significant τ bits of x respectively. We call a string $x \in \{0,1\}^n$ as a block. We call $z \in \{0,1\}^*$ as full block if $|z|$ is a multiple of n and as incomplete block otherwise.

Block ciphers. A block cipher is a function $E : \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$, where $\mathcal{K}$ is the key space, which generally contains fixed length strings; n is called the block length of the block cipher. For $K \in \mathcal{K}$ and $X \in \{0,1\}^n$, we will generally denote $E(K,X)$ by $E_K(X)$. A block cipher also has an inverse denoted by $E_K^{-1}()$, but we will not have any occasion to use the inverse in this work.

Pseudo-Random Function(PRF). Let $\mathsf{Func}(*,\tau)$ be the set of all functions from $\{0,1\}^*$ to $\{0,1\}^\tau$, and let $\{F_k\}_{k\in\mathcal{K}}$ be a family of functions where for each $k \in \mathcal{K}$, $F_k : \{0,1\}^* \to \{0,1\}^\tau$. We define the PRF-advantage of an adversary $\mathcal{A}$ as

$$\mathsf{Adv}^{\mathsf{prf}}_F(\mathcal{A}) = \left|\Pr[k \xleftarrow{\$} \mathcal{K} : \mathcal{A}^{F_k(\cdot)} \Rightarrow 1] - \Pr[\rho \xleftarrow{\$} \mathsf{Func}(*,\tau) : \mathcal{A}^{\rho(\cdot)} \Rightarrow 1]\right|.$$

We say that $\{F_k\}_{k\in\mathcal{K}}$ is a pseudorandom family of functions (or, sometimes F is a pseudorandom function) if for all efficient adversaries $\mathcal{A}$, $\mathsf{Adv}^{\mathsf{prf}}_F(\mathcal{A})$ is small.

Message Authentication Codes. A message authentication code (MAC) is a map $F : \mathcal{K} \times \mathcal{M} \to \{0,1\}^\tau$, where $\mathcal{K}$ is the key space and $\mathcal{M}$ the message space. We often write $F_K(\cdot)$ to denote $F(K, \cdot)$. The output of a MAC is called the tag, and τ is called the tag length. The security of a MAC F is defined using an interaction of F with an adversary $\mathcal{A}$ [1]. It is assumed that $\mathcal{A}$ has an oracle access to $F_K()$, where $K \xleftarrow{\$} \mathcal{K}$. For a query $x \in \mathcal{M}$ of $\mathcal{A}$ the oracle responds by sending $y = F_K(x)$. Let, $\mathcal{A}$ query $x_1, x_2, \ldots, x_q$ and gets $y_1, y_2, \ldots, y_q$ as responses from the oracle. These queries are performed adaptively. Finally, $\mathcal{A}$ outputs a pair (x^*, y^*), where $x^* \notin \{x_1, x_2, \ldots, x_q\}$. This pair is called a forgery and it is said that $\mathcal{A}$ has successfully *forged* F if $F_K(x^*) = y^*$. The auth-advantage of $\mathcal{A}$ is defined as

$$\mathsf{Adv}_F^{\mathsf{auth}}(\mathcal{A}) = \Pr[K \xleftarrow{\$} \mathcal{K} : \mathcal{A} \text{ forges}].$$

We say that F is (ϵ, t) secure if for every adversary $\mathcal{A}$, which runs for time at most t, $\mathsf{Adv}_F^{\mathsf{auth}}(\mathcal{A}) \leq \epsilon$. It is well known that for any arbitrary adversary $\mathcal{A}$ for the MAC F there exists a PRF adversary $\mathcal{B}$ for F such that

$$\mathsf{Adv}_F^{\mathsf{auth}}(\mathcal{A}) \leq \mathsf{Adv}_F^{\mathsf{prf}}(\mathcal{B}) + \frac{1}{2^\tau}, \tag{1}$$

where $\mathcal{B}$ and $\mathcal{A}$ both run almost for the same time and ask almost the same number of queries.

A dominant paradigm of designing MACs is by using block ciphers, and CBC-MAC and its variants are some notable examples of block cipher based MACs.

CBC-MAC : Consider the map $\mathsf{CBC} : \mathcal{K} \times \mathcal{M} \to \{0,1\}^n$, where $\mathcal{M} \subseteq \cup_{i>0}\{0,1\}^{ni}$. For $M \in \mathcal{M}$, let $\mathsf{parse}(M) = M_1 || M_2 || \cdots || M_\ell$ and $C_0 = 0^n$, $C_i = E_K(M_i \oplus C_{i-1})$ for $i \in [\ell]$, where $E_K : \{0,1\}^n \to \{0,1\}^n$ is a block cipher. We define $\mathsf{CBC}(K, M) = C_\ell$. We often denote $\mathsf{CBC}(K, M)$ by $\mathsf{CBC}_K(M)$. A schematic view of the function $\mathsf{CBC}_K(M)$ is shown in Figure 2. The function CBC is called the $\mathsf{CBC\text{-}MAC}$ and it is a secure MAC if the underlying block cipher E is a pseudorandom function and the message space $\mathcal{M}$ is prefix-free, i.e., for any two distinct $x, y \in \mathcal{M}$, x is not a prefix of y. For practical purposes, CBC is used in scenarios where the message space contains strings of fixed length; such message spaces are prefix-free.

Let $X_1, X_2, \ldots, X_\ell \in \{0,1\}^n$, then it is easy to see that for any $1 < k < \ell$,

$$\mathsf{CBC}_K\left(X_1 \| \cdots \| X_k \| \cdots \| X_\ell\right) = \mathsf{CBC}_K\left(\mathsf{CBC}_K(X_1 \| \cdots \| X_k) \oplus X_{k+1} \| X_{k+2} \| \cdots \| X_\ell\right). \tag{2}$$

Thus, if $\mathsf{CBC}_K(X_1 \| \cdots \| X_k) = T$, then

$$\mathsf{CBC}_K\left(X_1 \| \cdots \| X_k \| \cdots \| X_\ell\right) = \mathsf{CBC}_K\left(T \oplus X_{k+1} \| X_{k+2} \| \cdots \| X_\ell\right).$$

This property can be easily translated into a forgery attack: an adversary queries $X_1 \| \cdots \| X_k$ and gets response as T; further, it queries $T \oplus X_{k+1} \| X_{k+2} \| \cdots \| X_\ell$ and gets the response T_1; and finally it produces $(X_1 \| \cdots \| X_k \| \cdots \| X_\ell, T_1)$ as a forgery. From (2), it is easy to verify that the forgery will be successful with probability 1. This specific attack is called the *length extension attack* and this cannot be mounted if the message space is prefix-free.

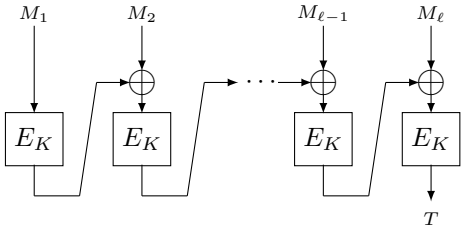

Fig. 1. The function $\mathsf{CBC}_K(M)$. E_K is a block cipher of block size n and $M = M_1||\ldots||M_\ell$, where $|M_i| = n$, for $i \in [\ell]$.

3 The Scheme TrCBC

TrCBC instantiated with a block cipher $E : \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$ is described in details in Figure 2. A schematic diagram of the same is shown in Figure 3. TrCBC takes a random key $K \xleftarrow{\$} \mathcal{K}$ and a message $M \in \{0,1\}^*$ as a input and returns a tag $T \in \{0,1\}^\tau$ of length $\tau < n/2$.

MAC Algorithm: $\mathsf{TrCBC}_K(M)$
Input: $K \xleftarrow{\$} \mathcal{K}$, $M \in \{0,1\}^*$.
Output: $T \in \{0,1\}^\tau$, where $\tau < n/2$.

01. $M_1 \| \cdots \| M_\ell \leftarrow \mathsf{parse_n}(M)$;
02. $Y \leftarrow 0^n$;
03. **for** $i \leftarrow 1$ *to* $\ell - 1$ **do**
04. $\quad X \leftarrow Y \oplus M_i$;
05. $\quad Y \leftarrow E_K(X)$;
06. **end for**
07. **if** $|M_\ell| = n$ **then**
08. $\quad X \leftarrow Y \oplus M_\ell$;
09. $\quad Y \leftarrow E_K(X)$;
10. $\quad T \leftarrow \mathsf{MSB}_\tau(Y)$;
11. **else**
12. $\quad X \leftarrow Y \oplus \mathsf{Pad}(M_\ell)$;
13. $\quad Y \leftarrow E_K(X)$;
14. $\quad T \leftarrow \mathsf{LSB}_\tau(Y)$;
15. **end if**
16. **return** T.

Fig. 2. Specification of TrCBC instantiated with an n-bit block cipher E_K.

A simplified view of TrCBC in terms of the function CBC would be useful. Let $M \in \{0,1\}^*$ where $|M| = \lambda > 0$. Let $x_1||x_2||\ldots||x_\ell = \mathsf{parse}(M)$ and let $r = |x_\ell|$. Notice that

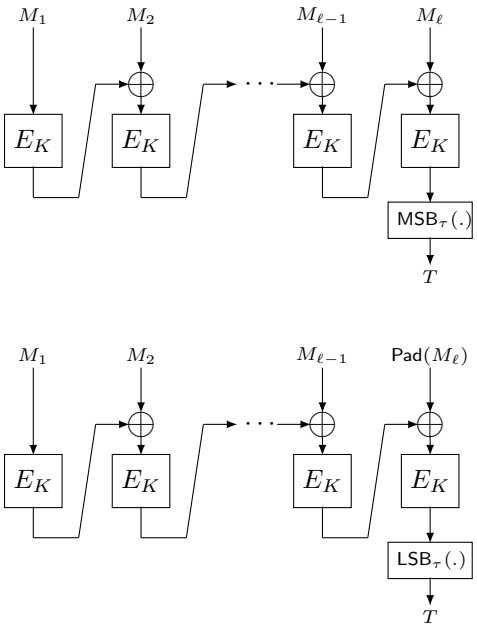

Fig. 3. The TrCBC construction. The first figure is for the full block messages, i.e., the message length is a multiple of the block size n, and the second figure is for messages whose length is not a multiple of n. $\mathsf{Pad}(M_\ell) = M_\ell \| 10^{n-|M_\ell|-1}$ and $\tau < n/2$.

$r = \lambda - (\ell - 1)n = \lambda - (\lceil \lambda/n \rceil - 1)n$. We define TrCBC as

$$\mathsf{TrCBC}_K(M) = \begin{cases} \mathsf{MSB}_\tau\left(\mathsf{CBC}_K(M)\right) & \text{if } r = n. \\ \mathsf{LSB}_\tau\left(\mathsf{CBC}_K(M||10^{n-r-1})\right) & \text{if } r < n. \end{cases} \tag{3}$$

4 An Attack on TrCBC

It was claimed in [9] that TrCBC is a secure MAC if the underlying block cipher is a pseudorandom permutation. We show that an adversary making just three queries to the MAC oracle can successfully forge TrCBC with probability 1/4.

We consider TrCBC instantiated with a block cipher of block length n (which is even). We fix the tag length $\tau = n/2 - 1$. Let x_1, x_2, x_3 be fixed but arbitrary strings such that $|x_1| = |x_3| = n$ and $|x_2| = n - 2$. We set $M_1 = x_1$, $M_2 = x_2||10$ and $M_3 = x_3$. The three queries which the adversary asks, along with the responses, are as follows.

1. Query $X^{(1)} = M_1||M_2$, and get T_1 as response.
2. Query $X^{(2)} = M_1||x_2$, and get T_2 as response.
3. Query $X^{(3)} = M_1||M_2||M_3$, and get T_3 as response.

Finally, the adversary submits (M^*, T^*) as the forgery, where

$$M^* = (T_1 \| b_1^* b_2^* \| T_2) \oplus M_3, \quad T^* = T_3 \quad \text{where} \quad b_1^*, b_2^* \xleftarrow{\$} \{0,1\}.$$

Note that (M^*, T^*) is a valid forgery, as M^* which is a single block message has never been queried to the oracle. We are left to show that this forgery is successful with high probability. We claim that for any choice of $K \in \mathcal{K}$,

$$\Pr[\mathsf{TrCBC}_K(M^*) = T^*] = 1/4,$$

where the probability is over the choice of b_1^*, b_2^*. We substantiate our claim below.

From the TrCBC construction and our simplified description of TrCBC in Eq.(3) we get,

$$T_1 = \mathsf{TrCBC}_K(M_1||M_2) = \mathsf{MSB}_\tau\left(\mathsf{CBC}_K\left(M_1 \| M_2\right)\right) \tag{4}$$

$$T_2 = \mathsf{TrCBC}_K(M_1||x_2) = \mathsf{LSB}_\tau\left(\mathsf{CBC}_K\left(M_1 \| x_2 \| 10\right)\right) = \mathsf{LSB}_\tau\left(\mathsf{CBC}_K\left(M_1 \| M_2\right)\right) \tag{5}$$

$$T_3 = \mathsf{TrCBC}_K(M_1||M_2||M_3) = \mathsf{MSB}_\tau\left(\mathsf{CBC}_K\left(M_1||M_2||M_3\right)\right). \tag{6}$$

As $|T_1| = |T_2| = \tau = n/2 - 1$, we have for some $b_1, b_2 \in \{0,1\}$,

$$T_1 \| b_1 b_2 \| T_2 = \mathsf{CBC}_K(M_1||M_2). \tag{7}$$

Hence, using (7) and (2)

$$\begin{aligned}\mathsf{MSB}_\tau\left(\mathsf{CBC}_K(T_1 \| b_1 b_2 \| T_2 \oplus M_3)\right) &= \mathsf{MSB}_\tau\left(\mathsf{CBC}_K\left(\mathsf{CBC}_K(M_1||M_2) \oplus M_3\right)\right) \\ &= \mathsf{MSB}_\tau\left(\mathsf{CBC}_K\left(M_1||M_2||M_3\right)\right) \\ &= T_3,\end{aligned}$$

and

$$\mathsf{TrCBC}_K(M^*) = \mathsf{MSB}_\tau\left(\mathsf{CBC}_K((T_1 \| b_1^* b_2^* \| T_2) \oplus M_3)\right).$$

As b_1^*, b_2^* are chosen uniformly at random from $\{0,1\}$, so with probability $1/4$, we have $(b_1^*, b_2^*) = (b_1, b_2)$, and thus

$$\Pr[\mathsf{TrCBC}_K(M^*) = T_3] = \frac{1}{4},$$

as claimed.

5 Discussions

The source of insecurity. The following are the main characteristics of TrCBC:

1. For full block messages, TrCBC is exactly the CBC-MAC scheme, except that instead of the full output only a part of the output is produced as the tag, in particular $\tau < n/2$ most significant bits only forms the tag.

2. For messages which are not full block, a deterministic padding is applied and the CBC-MAC of the padded message is computed and the least significant τ bits are output as a tag.

The idea behind such a design seems to be separating the outputs for full block and incomplete block messages and the authors thought that a small tag length would prevent a length extension type of attack. But, as only a deterministic padding scheme is applied, hence for any message $M \in \{0,1\}^{mn}$ where the last block is not 0^n almost all bits of $\mathsf{CBC}_K(M)$ can be recovered with just two queries to TrCBC. To see this, let $M = M'||x$, where $|x| = n$ and $x \neq 0^n$. Let $x = a_n \cdots a_2 a_1$, where $a_i \in \{0,1\}$, and j be the smallest element in $[n]$ such that $a_j = 1$. Let $M_1 = M'||a_n a_{n-1} \cdots a_{j-1}$. Then, following the padding scheme in TrCBC we have,

$$\mathsf{TrCBC}_K(M) = \mathsf{MSB}_\tau(\mathsf{CBC}_K(M)),$$
$$\mathsf{TrCBC}_K(M_1) = \mathsf{LSB}_\tau(\mathsf{CBC}_K(M)).$$

Our attack essentially uses the above property of TrCBC to recover 2τ many bits of $\mathsf{CBC}_K(M)$. This property can be further used to forge a large class of messages, which we will describe next.

A generic attack. Consider a message $X = X_1||X_2||\dots||X_\ell$, for $\ell \geq 2$ and suppose there exists $k \in [\ell - 1]$ such that $X_k \neq 0^n$, i.e, $X_1||X_2||\dots||X_{\ell-1}$ is not the all zero string. Let, $X_k = x\|10^m$, where the first 1 (from the left) in X_k occurs in the m^{th} place. As before, we fix the tag length $\tau = n/2 - 1$. Let an adversary query with the three queries specified below:

1. $X^{(1)} = X_1||X_2||\dots||X_k$.
2. $X^{(2)} = X_1||X_2||\dots\|X_{k-1}||x$.
3. $X^{(3)} = X_1||X_2||\dots||X_\ell$.

Let the responses to the above three queries be T_1, T_2, T_2 respectively, and let

$$M^* = ((T_1\|b_1^*b_2^*\|T_2) \oplus X_{k+1})\|X_{k+2}\|\dots\|X_\ell,$$

where $b_1^*, b_2^* \xleftarrow{\$} \{0,1\}$. Then following the same arguments as in Section 4 it is easy to verify that (M^*, T_3) will be a forgery with a success probability $1/4$.

Provable security of TrCBC. In [9] the authors claim TrCBC to be PRF. We restate the theorem in [9] next.

Theorem 1. *Let $R \xleftarrow{\$} \mathsf{Func}(*, \tau)$ and $P \xleftarrow{\$} \mathsf{Perm}(n)$. Let TrCBC_P be the construction where the block cipher in TrCBC is replaced by P. $\mathcal{A}$ is an adversary who asks at most q queries, having an aggregate length of at most σ blocks, then*

$$\left|\Pr\left[\mathcal{A}^{\mathsf{TrCBC}_P(.)} \Rightarrow 1\right] - \Pr\left[\mathcal{A}^{R(.)} \Rightarrow 1\right]\right| \leq \frac{\sigma(\sigma-1)}{2^{n+1}} + \frac{\sigma(\sigma-1)}{2^{n-2\tau+1}}.$$

Theorem 1 claims a bound on the PRF advantage of an adversary attacking TrCBC. If we investigate the bound a bit closely, it is clear that the bound on the advantage can be really large for some suggested parameter values of TrCBC. For example, for $\tau = n/2 - 1$, the dominant term in the bound is $\sigma(\sigma - 1)/8$, thus for any $\sigma > 3$ the bound becomes meaningless. As a PRF advantage (which is a difference of two probabilities), being less than 1 is a trivial information. Thus, though the theorem is correct, the bound does not guarantee that TrCBC is a PRF for all suggested parameter values and hence the provable security theorem, though correct, does not imply security of TrCBC for all suggested tag lengths.

The inadequacy of the theorem gets more clear when we see it in light of our attack. For our basic version of the attack, the adversary uses only three queries with query lengths 2 blocks, 2 blocks and 3 blocks respectively. Thus the query complexity (the total aggregate query length) of our adversary is $\sigma = 7$ and it has a large forgery advantage of $1/4$. Based on Theorem 1 and Eq. (1), the forgery advantage of an adversary with query complexity 7 would be at most

$$\frac{7(7-1)}{2^{n+1}} + \frac{7(7-1)}{2^3} + \frac{1}{2^{n/2-1}},$$

which is greater than 5 irrespective of the value of n. Thus, technically our attack does not refute the provable security claim.

It is worth investigating for which tag length(s) the bound implies security of TrCBC. According to Eq. (1), the forgery advantage of any adversary $\mathcal{A}$ with query complexity σ attacking TrCBC will be upper bounded by

$$\frac{\sigma(\sigma-1)}{2^{n+1}} + \frac{\sigma(\sigma-1)}{2^{n-2\tau+1}} + \frac{1}{2^{\tau}}.$$

Taking $\tau = n/2 - \alpha$, where $1 \leq \alpha < n/2$ we have the bound as

$$\frac{\sigma(\sigma-1)}{2^{n+1}} + \frac{\sigma(\sigma-1)}{2^{2\alpha+1}} + \frac{1}{2^{n/2-\alpha}} > \frac{1}{2^{2\alpha+1}} + \frac{1}{2^{n/2-\alpha}}. \tag{8}$$

A simple computation shows that the expression on the right-hand side of Eq. (8) attains the minima at $\alpha = n/6$, which suggests that the best-suited value of τ will be $n/2 - n/6 = n/3$.

As suggested by the authors, the allowed value of τ is less than $n/2$. It is common knowledge that shorter tags give lesser security; hence from a user's perspective, the maximum length of a tag, which is supported by the MAC and the application at hand, is chosen. Thus, given the specification of TrCBC it would be alluring for a user to use the largest possible tag length, which is $n/2 - 1$, and as we show, this choice can be disastrous. Thus the provable security guarantee of TrCBC which the authors provide through Theorem 1 is very confusing without a proper interpretation of the bound.

Our analysis shows that the PRF bound suggests maximum security when the tag length is $n/3$. If we consider a block cipher with a block length of 128 bits, this translates

to tags of length around 42 bits. For most applications, such a short tag lengths would not be tolerated. But TrCBC can provide adequate security when instantiated with block ciphers with large block lengths (say 256 bits) and when the tag length is appropriately selected.

The case of TCBC . Security properties of truncated CBC-MAC has been studied in details in [4]. In [4] a scheme called TCBC is described as

$$\mathsf{TCBC}_K(X) = \mathsf{MSB}_\tau\left(\mathsf{CBC}_K\left(\mathsf{pad1}(X)\right)\right).$$

Where $\mathsf{pad1}(x)$ appends a 1 followed by sufficiently many zeros to X to make the length of the resulting string a multiple of n. In particular if $x_1||x_2||\ldots||x_\ell = \mathsf{parse}_n(X)$ then

$$\mathsf{pad1}(X) = \begin{cases} X||10^{n-|x_\ell|-1} & \text{if } |x_\ell| < n. \\ X||10^{n-1} & \text{if } |x_\ell| = n. \end{cases}$$

It has been proved in [4] that TCBC is a secure pseudorandom function. In particular, if TCBC is instantiated with a random permutation, then any adversary making q queries with length at most $\lambda < 2^{n/4}$ cannot distinguish TCBC from a random function with probability more than $\epsilon(\lambda, q) = O(\frac{q(q+\lambda)}{2^{n-\tau}} + \frac{\lambda q^2}{2^n})$.

It is important to note the differences between TrCBC and TCBC. The padding scheme of TCBC injectively maps any string in $\{0,1\}^*$ to the set of strings $\cup_{i\geq 1}\{0,1\}^{ni}$, whereas the padding scheme for TrCBC is not injective. Also, for any message it is not possible for an adversary to know more than τ bits of the final output of TCBC, but as we already showed for TrCBC it is possible to know 2τ many bits of the output for a large class of messages and this helps in the forgery attack. Finally, TCBC requires one more block cipher call than TrCBC for full block messages.

6 Conclusion

We revisited the security of TrCBC. Our study shows that TrCBC is not secure for all suggested tag lengths. In particular, for a tag length of $n/2 - 1$, we showed a concrete and practical attack with high success probability, which uses only three queries to the MAC. The security theorem for TrCBC, though correct, does not imply security of TrCBC for all suggested parameters. Our study re-confirms the need to study claimed security bounds in security theorems for cryptographic constructions before choosing safe parameter values for the system.

We do not see any easy way to fix TrCBC such that it retains the interesting requirement of a single key and $\lceil \lambda/n \rceil$ many block cipher calls for authenticating a λ-bit message with a good security margin. It is worth mentioning here that GCBC1/GCBC2 [7] achieves this to a large extent, i.e., when $\lambda > n$ GCBC1/GCBC2 indeed produces a secure MAC with a single key and $\lceil \lambda/n \rceil$ block cipher calls.

References

1. Mihir Bellare, Joe Kilian, and Phillip Rogaway. The security of the cipher block chaining message authentication code. *Journal of Computer and System Sciences*, 61(3):362–399, 2000.
2. John Black and Phillip Rogaway. CBC MACs for arbitrary-length messages: The three-key constructions. In Mihir Bellare, editor, *Advances in Cryptology - CRYPTO 2000, 20th Annual International Cryptology Conference, Santa Barbara, California, USA, August 20-24, 2000, Proceedings*, volume 1880 of *Lecture Notes in Computer Science*, pages 197–215. Springer, 2000.
3. Morris Dworkin. The CMAC mode for authentication. *Recommendation for Block Cipher Modes of Operation*, 2005.
4. Peter Gazi, Krzysztof Pietrzak, and Stefano Tessaro. The exact PRF security of truncation: Tight bounds for keyed sponges and truncated CBC. In Rosario Gennaro and Matthew Robshaw, editors, *Advances in Cryptology - CRYPTO 2015 - 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part I*, volume 9215 of *Lecture Notes in Computer Science*, pages 368–387. Springer, 2015.
5. Tetsu Iwata and Kaoru Kurosawa. OMAC: one-key CBC MAC. In Thomas Johansson, editor, *Fast Software Encryption, 10th International Workshop, FSE 2003, Lund, Sweden, February 24-26, 2003, Revised Papers*, volume 2887 of *Lecture Notes in Computer Science*, pages 129–153. Springer, 2003.
6. Kaoru Kurosawa and Tetsu Iwata. TMAC: two-key CBC MAC. In Marc Joye, editor, *Topics in Cryptology - CT-RSA 2003, The Cryptographers' Track at the RSA Conference 2003, San Francisco, CA, USA, April 13-17, 2003, Proceedings*, volume 2612 of *Lecture Notes in Computer Science*, pages 33–49. Springer, 2003.
7. Mridul Nandi. Fast and secure CBC-Type MAC algorithms. In Orr Dunkelman, editor, *Fast Software Encryption, 16th International Workshop, FSE 2009, Leuven, Belgium, February 22-25, 2009, Revised Selected Papers*, volume 5665 of *Lecture Notes in Computer Science*, pages 375–393. Springer, 2009.
8. Erez Petrank and Charles Rackoff. CBC MAC for real-time data sources. *J. Cryptol.*, 13(3):315–338, 2000.
9. Liting Zhang, Wenling Wu, Peng Wang, and Bo Liang. TrCBC: Another look at CBC-MAC. *Inf. Process. Lett.*, 112(7):302–307, 2012.