Impact of ANSI X9.24-1:2009 Key Check Value on ISO/IEC 9797-1:2011 MACs

Tetsu Iwata[1(✉)] and Lei Wang[2]

[1] Nagoya University, Nagoya, Japan
iwata@cse.nagoya-u.ac.jp
[2] Nanyang Technological University, Singapore, Singapore
wang.lei@ntu.edu.sg

Abstract. ANSI X9.24-1:2009 specifies the key check value, which is used to verify the integrity of the blockcipher key. This value is defined as the most significant bits of the ciphertext of the zero block, and is assumed to be publicly known data for verification. ISO/IEC 9797-1:2011 illustrates a total of ten CBC MACs, where one of these MACs, the basic CBC MAC, is widely known to be insecure. In this paper, we consider the remaining nine CBC MACs and derive the quantitative security impact of using the key check value. We first show attacks against five MACs by taking advantage of the knowledge of the key check value. We then *prove* that the analysis is tight, in a concrete security paradigm. For the remaining four MACs, we prove that *the standard birthday bound still holds* even with the presence of the key check value. As a result, we obtain a complete characterization of the impact of using ANSI X9.24-1 key check value with the ISO/IEC 9797-1 MACs.

Keywords: ANSI X9.24-1:2009 · Key check value · ISO/IEC 9797-1:2011 · CBC MAC · Proof of security

1 Introduction

Background. A Message Authentication Code, or a MAC, is a fundamental cryptographic primitive to ensure the authenticity of messages. A MAC is a keyed function that takes a message as its input and produces a fixed length string called a tag. The tag is then used to verify the integrity of the message, i.e., the message is indeed originated from the intended party who shares the key.

There are several ways of constructing MACs, and CBC MAC is a widely used MAC based on a blockcipher. While the basic CBC MAC has a proof of security when it is used for messages of one fixed length [4,6], it is known that it allows the so called length-extension attack when the message length can vary. To avoid the weakness, several variants of CBC MAC were proposed. ISO/IEC 9797-1:2011 [14] specifies six different versions of CBC MACs, where each MAC is defined by specifying the final iteration and the output transformation. The six MACs are further classified by specifying the key derivation method and the

C. Cid and C. Rechberger (Eds.): FSE 2014, LNCS 8540, pp. 303–322, 2015.
DOI: 10.1007/978-3-662-46706-0_16

padding method, and Annex C in [14] illustrates a total of ten different CBC MACs depending on the choice of these methods.

ANSI X9.24-1:2009 [2] specifies the manual and automated management of keying material used for financial services. The services include point-of-sale (POS) transactions (debit and credit), automated teller machine (ATM) transactions, messages among terminals and financial institutions, and interchange messages among acquirers, switches, and card issuers [2]. Annex C in [2] suggests the use of the *key check value* for the integrity verification of the blockcipher key. Let $E_K : \{0,1\}^n \to \{0,1\}^n$ be a blockcipher with a key K. ANSI X9.24-1 suggests to use the most significant s bits of $E_K(0^n)$, a ciphertext of the zero block, as the key check value for the key K, where E is DES or TDES [3] and s is 16 or 24. That is, the key check value, KCV, is defined as $\mathsf{KCV} = \mathsf{msb}_s(E_K(0^n))$. The value KCV is then used to verify the integrity of K, or as the ID for K. Therefore, KCV is treated as a public value and it can be transmitted, sent, or stored in clear. This implies that an adversary has chance to learn this value.

In some MACs and other blockcipher modes of operation, the value $E_K(0^n)$ is used to derive the "sub-key." For example, MAC5 in [14], also known as CMAC [11], uses this value as a sub-key that has to be kept secret from the adversaries. Other examples that use $E_K(0^n)$ as a sub-key include GCM [12,22], PMAC [7,28], and OCB [28,30]. MAC5 has the proof of security [15,16], while disclosing a part of $E_K(0^n)$ is not taken into account in the proof, and it is anticipated that there is some security loss when it is used with the key check value. Indeed, [2,11] give an explicit warning that the value $E_K(0^n)$ has to be kept secret, but there is no prior work that derives the security loss in a *quantitative* way, which is the main question solved in the present paper.

Contributions. We consider the security of ten MACs, CBC MAC and its variants, specified in [14], in the presence of the key check value. We first consider MAC2.1, which is also known as EMAC [9]. Petrank and Rackoff showed that it is a secure MAC [25]. The security bound without the use of the key check value is $O(\sigma^2/2^n)$ [25] assuming that the underlying blockcipher satisfies an appropriate security notion, where σ is the total length of queries in blocks made by the adversary, and n is the block size of the underlying blockcipher in bits. This implies that the adversary needs to make $\Omega(2^{n/2})$ queries in order to make a forgery. In its specification, the value $E_K(0^n)$ does not appear as in the case for MAC5. However, based on a similar technique to the one in [18], we present attacks with $O(2^{(n-s)/2})$ queries when it is used with the s-bit key check value. For MAC5, the security bound without the use of the key check value is $O(\sigma^2/2^n)$ [15,16]. We show that almost the same attacks against MAC2.1 can be used to attack MAC5 with $O(2^{(n-s)/2})$ queries when it is used with the s-bit key check value. We point out that similar attacks can be used against MAC2.2 (EMAC [9]), MAC3 (ANSI retail MAC [1]), and MAC6.2 (FCBC [8]).

We then formalize a security notion of a variant of a pseudorandom function that captures the key check value, and for these five MACs, we *prove* that the analysis is tight. That is, under the appropriate assumption about the underlying blockcipher, we show that the adversary actually needs to make $\Omega(2^{(n-s)/2})$

queries in order to mount the attacks. For the remaining four MACs, MAC1.2, MAC4.1 (MacDES [19]), MAC4.2 (MacDES [19]), and MAC6.1 (FCBC [8]), the situation is quite different. Even if the key check value consists of the entire n bits of $E_K(0^n)$, we show that the standard birthday bound still holds, and there is almost no security loss for these MACs. As a result, we obtain a complete characterization of the impact of using ANSI X9.24-1 key check value with the ISO/IEC 9797-1 MACs. See Table 1 for the summary of this paper. In the table, the block size of the underlying blockcpher is n bits, s is the bit length of the key check value, and σ is the total length of queries in blocks made by the adversary. All results for MAC1.1 are widely known. Results on existential forgeries for MAC1.2, MAC4.1, MAC4.2, and MAC6.1 follow from [14,27]. The attacks are possible without using the key check value. We note that there are other attacks for all these MACs, e.g., a key recovery attack. See [14,29] for more details.

We also discuss several generic ways to avoid the security loss in the presence of the key check value.

Remarks. We argue that our security bounds, both $O(\sigma^2/2^{n-s})$ and $O(\sigma^2/2^n)$, are non-trivial, in the sense that there are blockcipher modes of operation that become completely insecure if the key check value is used. For instance consider the CTR encryption mode where the initial counter value starts with 0^n. It is not hard to see that the adversary succeeds in distinguishing between a ciphertext and a random string of the same length as the ciphertext even if the key check value consists of one bit.

We remark that the presence of the key check value can be considered as a special case of leakage of the internal state. Leakage resilient MACs are proposed, e.g., in [10,21]. In contrast to designing new schemes, the purpose of this paper is to analyze the security of widely standardized and deployed MACs when used with the key check value, a common practice in financial applications.

We also remark that this paper does not show the analysis of MACs in the older version of ISO/IEC 9797-1 that was published in 1999 [13]. Specifically, MAC5 and MAC6 in the 2011 version are different from those in the 1999 version. We leave the analyses of these MACs as open questions.

2 Preliminaries

For a finite set $\mathcal{X}$, $X \xleftarrow{\$} \mathcal{X}$ means that an element X is chosen from $\mathcal{X}$ uniformly at random. Let $\{0,1\}^*$ be the set of all finite bit strings including the empty string. If $X, Y \in \{0,1\}^*$ are equal-length strings then $X \oplus Y$ is their bitwise xor. If $X, Y \in \{0,1\}^*$ are strings then $X \,\|\, Y$, or simply XY, denote their concatenation. If $X \in \{0,1\}^*$ is a string then $|X|$ denotes its length in bits. Throughout the paper we fix the *block size* n. Typical values of n are 64 and 128. We let $\{0,1\}^n$ be a set of all bit strings of n bits. For a string $X \in \{0,1\}^{n\ell}$ with $\ell \geq 1$, the *partition* $X[1] \cdots X[\ell]$ of X are defined as the unique strings satisfying the conditions $X[1] \,\|\, \cdots \,\|\, X[\ell] = X$ and $|X[1]| = \cdots = |X[\ell]| = n$. We write $X[1] \cdots X[\ell] \xleftarrow{n} X$. For an n-bit string $X = X_n \cdots X_2 X_1 \in \{0,1\}^n$, $X \ll 1 = X_{n-1} \cdots X_2 X_1 0$ is the

Table 1. Summary of the results. All results are in the presence of the key check value. The figures in the "known" column indicate the required number of known message and tag pairs, and "chosen" indicates the number of chosen message and tag pairs.

MAC	Existential forgery			Selective forgery			Security bound
	Known	Chosen	Reference	Known	Chosen	Reference	[Sect. 6]
MAC1.1	1	0	folklore	1	1	folklore	—
MAC1.2	$O(2^{n/2})$	1	[14,27]	—	—		$O(\sigma^2/2^n)$
MAC2.1	1	$O(2^{(n-s)/2})$	[Sect. 5]	0	$O(2^{(n-s)/2})$	[Sect. 5]	$O(\sigma^2/2^{n-s})$
MAC2.2	1	$O(2^{(n-s)/2})$	[Sect. 5]	0	$O(2^{(n-s)/2})$	[Sect. 5]	$O(\sigma^2/2^{n-s})$
MAC3	1	$O(2^{(n-s)/2})$	[Sect. 5]	0	$O(2^{(n-s)/2})$	[Sect. 5]	$O(\sigma^2/2^{n-s})$
MAC4.1	$O(2^{n/2})$	1	[14,27]	—	—		$O(\sigma^2/2^n)$
MAC4.2	$O(2^{n/2})$	1	[14,27]	—	—		$O(\sigma^2/2^n)$
MAC5	1	$O(2^{(n-s)/2})$	[Sect. 5]	0	$O(2^{(n-s)/2})$	[Sect. 5]	$O(\sigma^2/2^{n-s})$
MAC6.1	$O(2^{n/2})$	1	[14,27]	—	—		$O(\sigma^2/2^n)$
MAC6.2	1	$O(2^{(n-s)/2})$	[Sect. 5]	0	$O(2^{(n-s)/2})$	[Sect. 5]	$O(\sigma^2/2^{n-s})$

left shift of X by 1 bit. For a positive integer ℓ and a string X such that $\ell \leq |X|$, $\mathsf{msb}_\ell(X)$ is the leftmost ℓ bits of the string X, and $\mathsf{lsb}_\ell(X)$ is the rightmost ℓ bits of X. For non-negative integers ℓ and n such that $\ell < 2^n$, $\mathsf{bin}_n(\ell)$ is an n-bit binary representation of ℓ.

A blockcipher is a family of permutations. We write $E : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ for a blockcipher, where $K \in \{0,1\}^k$ is a key and $E_K(\cdot) = E(K,\cdot)$ is the permutation over $\{0,1\}^n$ specified by K. We write $\mathrm{Perm}(n)$ for the set of all permutations over $\{0,1\}^n$, and $\mathrm{Rand}(n)$ for the set of all functions from $\{0,1\}^n$ to $\{0,1\}^n$.

A MAC is a keyed function $\mathcal{M}_K$. It takes an input message $M \in \{0,1\}^*$ and outputs a tag $T \in \{0,1\}^\tau$. The value τ is called a tag length, and we consider the case $\tau = n$. An adversary $\mathcal{A}$ against the MAC is an algorithm that has access to the MAC oracle, $\mathcal{M}_K$, and outputs a *forgery*, which is a pair of a message and a tag, (M^*, T^*). $\mathcal{A}$ can be a probabilistic algorithm or a deterministic algorithm. We say $\mathcal{A}$ *forges* if T^* was not previously returned from the MAC oracle in a response to a query M^*.

3 ISO/IEC 9797-1:2011 MACs

Let $E : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ be a blockcipher. The basic CBC MAC is defined in Fig. 1. It takes a message M as input, where $|M|$ is a positive multiple of n, and returns an n-bit tag T. We write $T \leftarrow \mathrm{CBC}_K(M)$. The specification of the MACs defined in ISO/IEC 9797-1:2011 [14] is in Fig. 2, and illustrated in Fig. 3. The subroutines KD, double, $\mathsf{pad1}$, $\mathsf{pad2}$, and $\mathsf{pad3}$ are specified in Fig. 1. KD is a key derivation function and is used in MAC6.1. double is a doubling operation in $\mathrm{GF}(2^n)$ and is used in MAC5. We note that L is defined as $\mathsf{double}(E_K(0^n))$. $\mathsf{pad1}$, $\mathsf{pad2}$, and $\mathsf{pad3}$ are functions for padding. Our description may seem to be different from the ones in [14] in appearance, but they are carefully distilled for simpler presentation, and they are the same algorithms as specified in [14].

Algorithm $\mathrm{CBC}_K(M)$

1. $Y[0] \leftarrow 0^n$
2. $M[1] \cdots M[m] \xleftarrow{n} M$
3. **for** $i \leftarrow 1$ **to** m **do**
4. $\quad X[i] \leftarrow Y[i-1] \oplus M[i]$
5. $\quad Y[i] \leftarrow E_K(X[i])$
6. $T \leftarrow Y[m]$
7. **return** T

Subroutine $\mathsf{KD}(K)$

1. $\ell \leftarrow \lceil k/n \rceil$
2. **for** $i \leftarrow 1$ **to** ℓ **do**
3. $\quad K'[i] \leftarrow E_K(\mathsf{bin}_n(i))$
4. $\quad K''[i] \leftarrow E_K(\mathsf{bin}_n(\ell + i))$
5. $K' \leftarrow \mathsf{msb}_k(K'[1] \cdots K'[\ell])$
6. $K'' \leftarrow \mathsf{msb}_k(K''[1] \cdots K''[\ell])$
7. **return** (K', K'')

Subroutine $\mathsf{double}(L)$

1. **if** $\mathsf{msb}_1(L) = 0$ **then**
2. $\quad L \leftarrow L \ll 1$
3. **else**
4. $\quad L \leftarrow (L \ll 1) \oplus \mathsf{constant}_n$
5. **return** L

Subroutine $\mathsf{pad1}(M)$

1. $M \leftarrow M \,\|\, 1 \,\|\, 0^{n-1-(|M| \bmod n)}$
2. **return** M

Subroutine $\mathsf{pad2}(M)$

1. **if** $(|M| > 0) \wedge (|M| \bmod n = 0)$ **then**
2. $\quad M \leftarrow \mathsf{bin}_n(|M|) \| M$
3. **else**
4. $\quad M \leftarrow \mathsf{bin}_n(|M|) \| M \| 0^{n-(|M| \bmod n)}$
5. **return** M

Subroutine $\mathsf{pad3}(L, M)$

1. **if** $(|M| = 0) \vee (|M| \bmod n > 0)$ **then**
2. $\quad M \leftarrow \mathsf{pad1}(M)$
3. $\quad L \leftarrow \mathsf{double}(L)$
4. $M[1] \cdots M[m] \xleftarrow{n} M$
5. $M[m] \leftarrow M[m] \oplus L$
6. **return** $M[1] \cdots M[m]$

Fig. 1. Pseudocode of the basic CBC MAC, and the subroutines used in the definition of MACs in ISO/IEC 9797-1. In $\mathsf{KD}(K)$, $\ell + i$ is the arithmetic addition of ℓ and i. In $\mathsf{double}(L)$, $\mathsf{constant}_n$ is an n-bit constant that depends on n. For example, $\mathsf{constant}_{64} = \mathtt{0x0} \cdots \mathtt{01b}$ and $\mathsf{constant}_{128} = \mathtt{0x0} \cdots \mathtt{087}$. When M is the empty string, we have $\mathsf{pad2}(M) = 0^{2n}$.

A total of six MACs are specified in [14]. The ten MACs in Fig. 2 are taken from [14, Annex C, Table C.1], which are specified as the concrete choices of the final iteration, the output transformation, and the padding method. For all MACs except for MAC4.1, they take a message $M \in \{0,1\}^*$ as their input, while MAC4.1 takes a message M such that $|M| \geq n$ as input. All these ten MACs return an n-bit tag T (we only consider the case where the tag consists of a full n-bit string). The key derivation method is not specified in [14, Annex C, Table C.1]. In MAC2.1, following the examples in [14, Annex B.3], we let $K' \leftarrow K \oplus (\mathtt{0xf0f0} \cdots \mathtt{f0})$. We also have a similar key derivation in MAC4.1 and MAC4.2, and in these algorithms, $k = |K|$ is assumed to be a multiple of 8. In MAC6.1, following [14, Annex B.7, NOTE 2 in Sect. 7.7], we let $(K', K'') \leftarrow \mathsf{KD}(K)$.

4 ANSI X9.24:2009 Key Check Value

In [2], the key check value is defined as follows.

Algorithm $\mathrm{MAC1.1}_K(M)$

1. $M \leftarrow \mathsf{pad1}(M)$
2. $T \leftarrow \mathrm{CBC}_K(M)$
3. **return** T

Algorithm $\mathrm{MAC1.2}_K(M)$

1. $M \leftarrow \mathsf{pad2}(M)$
2. $T \leftarrow \mathrm{CBC}_K(M)$
3. **return** T

Algorithm $\mathrm{MAC2.1}_K(M)$

1. $K' \leftarrow K \oplus (\mathtt{0xf0f0} \cdots \mathtt{f0})$
2. $M \leftarrow \mathsf{pad1}(M)$
3. $S \leftarrow \mathrm{CBC}_K(M)$
4. $T \leftarrow E_{K'}(S)$
5. **return** T

Algorithm $\mathrm{MAC2.2}_{K,K'}(M)$

1. $M \leftarrow \mathsf{pad1}(M)$
2. $S \leftarrow \mathrm{CBC}_K(M)$
3. $T \leftarrow E_{K'}(S)$
4. **return** T

Algorithm $\mathrm{MAC3}_{K,K'}(M)$

1. $M \leftarrow \mathsf{pad1}(M)$
2. $S \leftarrow \mathrm{CBC}_K(M)$
3. $T \leftarrow E_K(E_{K'}^{-1}(S))$
4. **return** T

Algorithm $\mathrm{MAC4.1}_{K,K'}(M)$

1. $K'' \leftarrow K' \oplus (\mathtt{0xf0f0} \cdots \mathtt{f0})$
2. $M[1] \cdots M[m] \xleftarrow{n} \mathsf{pad1}(M)$
3. $M[2] \leftarrow E_{K''}(E_K(M[1])) \oplus M[2]$
4. $S \leftarrow \mathrm{CBC}_K(M[2] \cdots M[m])$
5. $T \leftarrow E_{K'}(S)$
6. **return** T

Algorithm $\mathrm{MAC4.2}_{K,K'}(M)$

1. $K'' \leftarrow K' \oplus (\mathtt{0xf0f0} \cdots \mathtt{f0})$
2. $M[1] \cdots M[m] \xleftarrow{n} \mathsf{pad2}(M)$
3. $M[2] \leftarrow E_{K''}(E_K(M[1])) \oplus M[2]$
4. $S \leftarrow \mathrm{CBC}_K(M[2] \cdots M[m])$
5. $T \leftarrow E_{K'}(S)$
6. **return** T

Algorithm $\mathrm{MAC5}_K(M)$

1. $L \leftarrow \mathsf{double}(E_K(0^n))$
2. $M \leftarrow \mathsf{pad3}(L, M)$
3. $T \leftarrow \mathrm{CBC}_K(M)$
4. **return** T

Algorithm $\mathrm{MAC6.1}_K(M)$

1. $(K', K'') \leftarrow \mathsf{KD}(K)$
2. $M[1] \cdots M[m] \xleftarrow{n} \mathsf{pad1}(M)$
3. $S \leftarrow 0^n$
4. **if** $m \geq 2$ **then**
5. $\quad S \leftarrow \mathrm{CBC}_{K'}(M[1] \cdots M[m-1])$
6. $T \leftarrow E_{K''}(S \oplus M[m])$
7. **return** T

Algorithm $\mathrm{MAC6.2}_{K,K'}(M)$

1. $M[1] \cdots M[m] \xleftarrow{n} \mathsf{pad1}(M)$
2. $S \leftarrow 0^n$
3. **if** $m \geq 2$ **then**
4. $\quad S \leftarrow \mathrm{CBC}_K(M[1] \cdots M[m-1])$
5. $T \leftarrow E_{K'}(S \oplus M[m])$
6. **return** T

Fig. 2. Pseudocode of the ISO/IEC 9797-1 MACs

"The optional check values, as mentioned in notes 2 and 3 above, are the left-most six hexadecimal digits from the ciphertext produced by using the DEA in ECB mode to encrypt to 64-bit binary zero value with the subject key or key component. The check value process may be simplified operationally, while still retaining reliability, by limiting the check value to the left-most four or six hexadecimal digits of the ciphertext. (Using the truncated check value may provide additional security in that the ciphertext which could be used for exhaustive key determination would be unavailable.)"

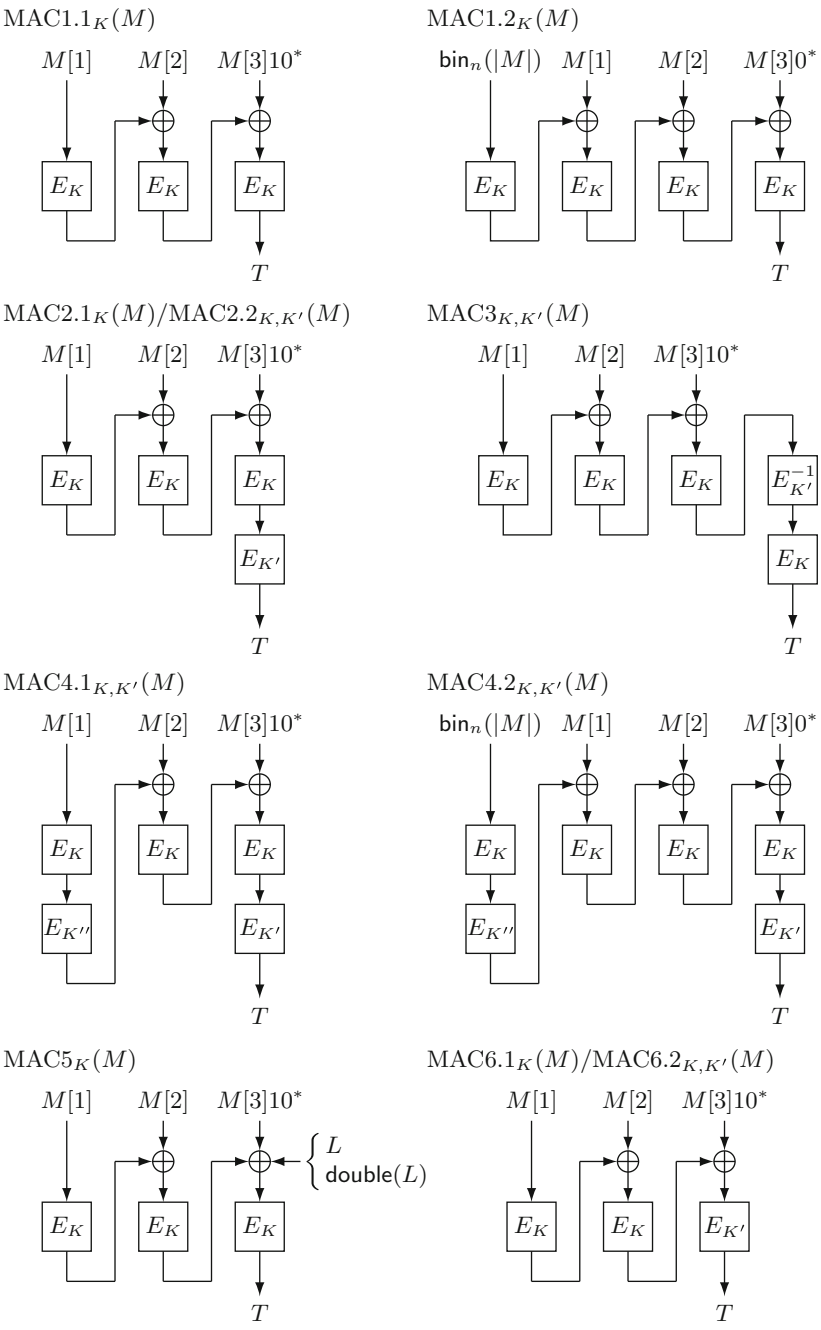

Fig. 3. Illustrations of the ISO/IEC 9797-1 MACs for $M = M[1]M[2]M[3]$, where $|M[1]| = |M[2]| = n$ and $1 \leq |M[3]| \leq n - 1$. In MAC6.1, $(E_K, E_{K'})$ is replaced with $(E_{K'}, E_{K''})$.

The key check value for the 56-bit key K is thus $\mathsf{msb}_{24}(\mathrm{DES}_K(0^{64}))$ or $\mathsf{msb}_{16}(\mathrm{DES}_K(0^{64}))$. The key check value is used to verify the integrity of K or as the ID for K in financial services including banking systems. The value is inherently a public value for verification, and it may be transmitted, sent or stored in clear, which implies that an adversary can learn this value.

In [2], the key check value is defined for DES or TDES, 64-bit blockciphers. However, other documents do not limit the block size being 64 bits. For example, the key check value of AES, a 128-bit blockcipher, is mentioned in [23]. MACs in [14] can be used with AES, and the document gives a warning about the use of the key check value. A similar warning can be found in [11,12], where AES can be used. Although it is not clear how the key check value is defined for AES in these documents, in this paper, for generality, we naturally extend the definition to any blockcipher E and allow other lengths of the key check value. For a blockcipher $E_K : \{0,1\}^n \to \{0,1\}^n$ with key $K \in \{0,1\}^k$ and an integer s such that $0 \le s \le n$, we define the key check value, KCV, as

$$\mathsf{KCV} = \mathsf{msb}_s(E_K(0^n)).$$

We leave s as a parameter that can be defined by a user of the blockcipher.

Now suppose that a MAC internally uses the blockcipher E and the key space of the MAC is $(\{0,1\}^k)^w$ for some integer $w \ge 1$. That is, a total of w blockcipher keys $K_1, \ldots, K_w \in \{0,1\}^k$ are used in the MAC, and it is built from $E_{K_1}, \ldots, E_{K_w}$. Then we define the key check value of the MAC as

$$\mathsf{KCV} = (\mathsf{msb}_s(E_{K_1}(0^n)), \ldots, \mathsf{msb}_s(E_{K_w}(0^n))).$$

Specifically, if the MAC uses single blockcipher key K, which corresponds to MAC1.1, MAC1.2, MAC2.1, MAC5, and MAC6.1, then the adversary is given $\mathsf{KCV} = \mathsf{msb}_s(E_K(0^n))$. For the remaining MACs that use two blockcipher keys K and K', i.e., for MAC2.2, MAC3, MAC4.1, MAC4.2, and MAC6.2, the adversary is given $\mathsf{KCV} = (\mathsf{msb}_s(E_K(0^n)), \mathsf{msb}_s(E_{K'}(0^n)))$.

5 Security Analysis

5.1 Security Analysis for Case $s = n$

We first consider the most extreme case $s = n$ to illustrate the impact.

Existential Forgery Against MAC2.1. Let $\mathsf{KCV} = \mathsf{msb}_s(E_K(0^n))$ be the key check value given to the adversary $\mathcal{A}$, where $s = n$. Let M be any message, and $T = \mathrm{MAC2.1}_K(M)$ be the corresponding tag. $\mathcal{A}$ is given the known message and tag pair (M, T). If $|M| \ge n$, then $\mathcal{A}$ defines M^* as

$$M^* = 0^n \,\|\, \mathsf{KCV} \,\|\, \cdots \,\|\, \mathsf{KCV} \,\|\, \mathsf{KCV} \oplus \mathsf{msb}_n(M) \,\|\, \mathsf{lsb}_{|M|-n}(M).$$

We see that all these messages, regardless of the number of the intermediate KCV blocks, share the same tag T, and therefore, (M^*, T) is a valid forgery.

We next consider the case $0 \leq |M| < n$. We assume that $\mathsf{pad1}(M) \neq \mathsf{KCV}$. Then there exists some M' such that $0 \leq |M'| < n$ and $\mathsf{pad1}(M') = \mathsf{pad1}(M) \oplus \mathsf{KCV}$. We then define M^* as

$$M^* = 0^n \,\|\, \mathsf{KCV} \,\|\, \cdots \,\|\, \mathsf{KCV} \,\|\, M', \tag{1}$$

and it can be easily verified that (M^*, T) is a valid forgery. If $\mathsf{pad1}(M) = \mathsf{KCV}$, then M' in (1), and hence M^*, cannot be defined and the attack does not work. However, there is at most one such M, and thus $\mathcal{A}$ can simply ask for the second known message and tag pair to mount the attack.

Therefore, MAC2.1 allows existential forgeries if the adversary knows any message and tag pair (M, T), where $\mathsf{pad1}(M) \neq \mathsf{KCV}$.

Selective Forgery Against MAC2.1. Almost the same idea can be used for a selective forgery against MAC2.1. Let M^* be a message that $\mathcal{A}$ tries to forge. We show that $\mathcal{A}$ is able to obtain the correct tag for M^* with one chosen message and tag pair, provided that $\mathsf{pad1}(M^*) \neq \mathsf{KCV}$. Now $\mathcal{A}$ defines M as

$$M = \begin{cases} 0^n \,\|\, \mathsf{KCV} \oplus \mathsf{msb}_n(M^*) \,\|\, \mathsf{lsb}_{|M^*|-n}(M^*) & \text{if } |M^*| \geq n, \\ 0^n \,\|\, M' & \text{else,} \end{cases}$$

where M' is a string that satisfies $0 \leq |M'| < n$ and $\mathsf{pad1}(M') = \mathsf{pad1}(M^*) \oplus \mathsf{KCV}$. Then $\mathcal{A}$ asks M to its MAC oracle and obtains $T = \mathrm{MAC2.1}_K(M)$. We see that (M^*, T) is a valid forgery.

We remark that if $\mathsf{pad1}(M^*) = \mathsf{KCV}$, then this particular M^* does not seem to allow attacks.

Existential/Selective Forgeries Against MAC5. As in the case for MAC2.1, $\mathcal{A}$ is given $\mathsf{KCV} = E_K(0^n)$. We see that existential and selective forgeries against MAC2.1 are irrelevant to the operations on the last message block, and almost the same attacks can be applied against MAC5.

Besides these attacks, there are other trivial attacks on MAC5. Recall that $\mathsf{KCV} = E_K(0^n)$ is used to compute the value of L. With KCV, $\mathcal{A}$ can compute both $L = \mathsf{double}(E_K(0^n))$ and $\mathsf{double}(\mathsf{double}(E_K(0^n)))$. This implies that, from $\mathcal{A}$'s view point, MAC5 is essentially reduced to MAC1.1, and therefore almost the same attacks against MAC1.1 in Appendix A work for MAC5.

There are subtle differences due to the padding rule of MAC5, but the modification is rather straightforward and we thus omit the details.

Existential/Selective Forgeries Against Other MACs. We point out that very similar attacks against MAC2.1 can be applied on MAC2.2, MAC3, and MAC6.2. For MAC2.2, $\mathcal{A}$ is given $\mathsf{KCV} = (E_K(0^n), E_{K'}(0^n))$, but the attacks are possible by using only $E_K(0^n)$. For MAC3, we see that the attacks against MAC2.1 are irrelevant to the operations on the last message block, and thus the same attacks can be applied. With the same reasoning these attacks can be applied on MAC6.2.

5.2 Security Analysis for Case $s < n$

We next consider the case $s < n$.

Existential/Selective Forgeries Against MAC2.1. The adversary $\mathcal{A}$ has access to the MAC2.1$_K(\cdot)$ oracle. Let $\mathsf{KCV} = \mathsf{msb}_s(E_K(0^n))$ be the key check value given to $\mathcal{A}$. We first derive the value of $E_K(0^n)$.

Let $r = 2^{(n-s)/2}$. $\mathcal{A}$ first chooses r random strings $\mathsf{rand}_1, \ldots, \mathsf{rand}_r$, where $|\mathsf{rand}_i| = n - s$ for all $1 \le i \le r$ and $\mathsf{rand}_i \ne \mathsf{rand}_j$ for all $1 \le i < j \le r$. Similarly, $\mathcal{A}$ chooses r random strings $\mathsf{rand}'_1, \ldots, \mathsf{rand}'_r$, where $|\mathsf{rand}'_i| = n - s$ for all $1 \le i \le r$ and $\mathsf{rand}'_i \ne \mathsf{rand}'_j$ for all $1 \le i < j \le r$. Let $M_i = 0^n \,\|\, (0^s \,\|\, \mathsf{rand}_i)$ and $M'_i = (\mathsf{KCV} \,\|\, \mathsf{rand}'_i)$. $\mathcal{A}$ then makes $2r$ queries, $M_1, \ldots, M_r, M'_1, \ldots, M'_r$, to its oracle and obtains $T_1, \ldots, T_r, T'_1, \ldots, T'_r$, where $T_i = \text{MAC2.1}_K(M_i)$ and $T'_i = \text{MAC2.1}_K(M'_i)$.

Then it is easy to see that we have $\mathsf{pad1}(M_i) = 0^n \,\|\, (0^s \,\|\, \mathsf{rand}_i) \,\|\, 10^{n-1}$ and $\mathsf{pad1}(M'_i) = (\mathsf{KCV} \,\|\, \mathsf{rand}'_i) \,\|\, 10^{n-1}$. Let $(X_i[1], X_i[2], X_i[3])$ be the input sequence of E_K in the computation of MAC2.1$_K(M_i)$, and $(Y_i[1], Y_i[2], Y_i[3])$ be the corresponding output sequence. Similarly, let $(X'_i[1], X'_i[2])$ and $(Y'_i[1], Y'_i[2])$ be the input and output sequences of E_K in the computation of MAC2.1$_K(M'_i)$. We have

$$\begin{cases} X_i[1] = 0^n, X_i[2] = Y_i[1] \oplus (0^s \,\|\, \mathsf{rand}_i), X_i[3] = Y_i[2] \oplus 10^{n-1}, \\ Y_i[1] = E_K(X_i[1]), Y_i[2] = E_K(X_i[2]), Y_i[3] = E_K(X_i[3]). \end{cases}$$

Similarly, we have

$$\begin{cases} X'_i[1] = (\mathsf{KCV} \,\|\, \mathsf{rand}'_i), X'_i[2] = Y'_i[1] \oplus 10^{n-1}, \\ Y'_i[1] = E_K(X'_i[1]), Y'_i[2] = E_K(X'_i[2]). \end{cases}$$

Note that $Y_i[3] = E_K(X_i[3]) = S_i$ and $Y'_i[2] = E_K(X'_i[2]) = S'_i$. We also note that $E_{K'}(S_i) = T_i$ and $E_{K'}(S'_i) = T'_i$ hold.

We claim that, with a high probability, there exists a pair of indices (j, j') such that $T_j = T'_{j'}$. To see this, we have $T_j = T'_{j'}$ if and only if $X_j[3] = X'_{j'}[2]$ since E_K and $E_{K'}$ are permutations. Now $X_j[3] = X'_{j'}[2]$ holds if and only if $Y_j[2] = Y'_{j'}[1]$ since the same value, 10^{n-1}, is xor-ed. Then $Y_j[2] = Y'_{j'}[1]$ holds if and only if $X_j[2] = X'_{j'}[1]$ from the invertibility of E_K, and this is equivalent to $E_K(0^n) \oplus (0^s \,\|\, \mathsf{rand}_j) = (\mathsf{KCV} \,\|\, \mathsf{rand}'_{j'})$. Now the last condition is equivalent to $\mathsf{lsb}_{n-s}(E_K(0^n)) \oplus \mathsf{rand}_j = \mathsf{rand}'_{j'}$, since $\mathsf{KCV} = \mathsf{msb}_s(E_K(0^n))$, and from the standard birthday paradox, we have the claim.

Let (j, j') be the pair of indices such that $T_j = T'_{j'}$. Observe that $\mathcal{A}$ can now retrieve the value of $E_K(0^n)$ since we have $E_K(0^n) = (\mathsf{KCV} \,\|\, \mathsf{rand}_j \oplus \mathsf{rand}'_{j'})$. With the knowledge of $E_K(0^n)$, $\mathcal{A}$ can produce arbitrarily number of existential forgeries with one known message and tag pair, and can produce a selective forgery with one chosen message and tag pair, as described in Sect. 5.1. Therefore, MAC2.1 allows existential forgeries with $2 \cdot 2^{(n-s)/2}$ chosen messages and one known message, and it allows selective forgery with $1 + 2 \cdot 2^{(n-s)/2}$ chosen messages.

Existential/Selective Forgeries Against MAC5. As in the case for MAC2.1, $\mathcal{A}$ is given $\mathsf{KCV} = \mathsf{msb}_s(E_K(0^n))$. We see that exactly the same procedure can be used to retrieve the value of $E_K(0^n)$, which is also used to compute a value of L. Therefore, MAC5 allows existential forgeries with $2 \cdot 2^{(n-s)/2}$ chosen messages and one known message, and it allows selective forgery with $1+2 \cdot 2^{(n-s)/2}$ chosen messages.

Besides these attacks, since L is now known to the adversary, the standard length-extension attack in Appendix A can be used to attack MAC5.

Existential/Selective Forgeries Against Other MACs. We remark that similar attacks can be used against MAC2.2, MAC3, and MAC6.2. These MACs allow existential forgeries with $2 \cdot 2^{(n-s)/2}$ chosen messages and one known message, and they allow selective forgeries with $1 + 2 \cdot 2^{(n-s)/2}$ chosen messages.

We note that the attacks presented in this section cannot be used against MAC1.2, MAC4.1, MAC4.2, and MAC6.1 even if $s = n$.

6 Provable Security Results

We present the provable security results for the nine ISO/IEC 9797-1 MACs.

Security Definition for MACs. Let $\mathcal{M}_{K_1,\ldots,K_w} : \{0,1\}^* \to \{0,1\}^n$ be a MAC with key space $(\{0,1\}^k)^w$ for some $w \geq 1$. An adversary $\mathcal{A}$ is an algorithm that outputs a bit. We consider the following game. First, $\mathcal{A}$ is given the key check value $\mathsf{KCV} = (\mathsf{msb}_s(E_{K_1}(0^n)), \ldots, \mathsf{msb}_s(E_{K_w}(0^n)))$. Then $\mathcal{A}$ is given access to an oracle, which is either the MAC oracle or the ideal random oracle. The MAC oracle $\mathcal{M}_{K_1,\ldots,K_w}$ takes a message M as the input and returns $T = \mathcal{M}_{K_1,\ldots,K_w}(M)$. The random oracle $\mathcal{R}$ takes a message M to return a random string T. We define

$$\mathbf{Adv}^{\text{prf-kcv}}_{\mathcal{M}}(\mathcal{A}) = \Pr\left[\mathcal{A} \leftarrow \mathsf{KCV}, \mathcal{A}^{\mathcal{M}_{K_1,\ldots,K_w}(\cdot)} \Rightarrow 1\right] - \Pr\left[\mathcal{A} \leftarrow \mathsf{KCV}, \mathcal{A}^{\mathcal{R}(\cdot)} \Rightarrow 1\right],$$

where the first probability is taken over the choices of $K_1, \ldots, K_w$ and $\mathcal{A}$'s coin, and the last is over the choices of $K_1, \ldots, K_w$ used for KCV, the random oracle, and $\mathcal{A}$'s coin.

Specifically, if $\mathcal{M}$ uses single blockcipher key K, i.e., if $\mathcal{M} \in \{\text{MAC1.1}, \text{MAC1.2}, \text{MAC2.1}, \text{MAC5}, \text{MAC6.1}\}$, then $\mathcal{A}$ is given $\mathsf{KCV} = \mathsf{msb}_s(E_K(0^n))$, and we consider

$$\mathbf{Adv}^{\text{prf-kcv}}_{\mathcal{M}}(\mathcal{A}) = \Pr\left[\mathcal{A} \leftarrow \mathsf{KCV}, \mathcal{A}^{\mathcal{M}_K(\cdot)} \Rightarrow 1\right] - \Pr\left[\mathcal{A} \leftarrow \mathsf{KCV}, \mathcal{A}^{\mathcal{R}(\cdot)} \Rightarrow 1\right].$$

For $\mathcal{M}$ with two blockcipher keys K and K', i.e., for $\mathcal{M} \in \{\text{MAC2.2}, \text{MAC3}, \text{MAC4.1}, \text{MAC4.2}, \text{MAC6.2}\}$, then $\mathsf{KCV} = (\mathsf{msb}_s(E_K(0^n)), \mathsf{msb}_s(E_{K'}(0^n)))$ and we consider

$$\mathbf{Adv}^{\text{prf-kcv}}_{\mathcal{M}}(\mathcal{A}) = \Pr\left[\mathcal{A} \leftarrow \mathsf{KCV}, \mathcal{A}^{\mathcal{M}_{K,K'}(\cdot)} \Rightarrow 1\right] - \Pr\left[\mathcal{A} \leftarrow \mathsf{KCV}, \mathcal{A}^{\mathcal{R}(\cdot)} \Rightarrow 1\right].$$

We write $\mathbf{Adv}^{\text{prf-kcv}}_{\mathcal{M}}(t,q,\sigma) = \max_{\mathcal{A}} \mathbf{Adv}^{\text{prf-kcv}}_{\mathcal{M}}(\mathcal{A})$, where the maximum is taken over adversaries $\mathcal{A}$ whose time complexity, number of queries, and query complexity are at most t, q, and σ, respectively. For the time complexity, we fix a model of computation and a choice of encoding, and it includes the running time and the code size. The query complexity is the total length in blocks of the *padded* queries made to the oracle. For instance if we consider an adversary $\mathcal{A}$ attacking MAC1.2 and if $\mathcal{A}$ makes queries $M_1, \ldots, M_q$, then the query complexity is $\sum_{1 \le i \le q} |\mathsf{pad2}(M_i)|/n$. Without loss of generality, we exclude the trivial queries, and we apply this convention to all adversaries in this paper.

We note that the above definitions capture the security of a MAC as a pseudorandom function, or a PRF, in the presence of KCV. It is well known that PRFs are secure MACs, see e.g. [4].

Security Definition for Blockciphers. We consider three security notions for the underlying blockcipher [5,20]. Let $E_K : \{0,1\}^n \to \{0,1\}^n$ be a blockcipher, E_K^{-1} be its inverse, $P, P' : \{0,1\}^n \to \{0,1\}^n$ be two independent random permutations, and P^{-1} be the inverse of P. An adversary $\mathcal{A}$ is an algorithm that outputs a bit. For $\mathcal{A}$, we define

$$\mathbf{Adv}^{\text{prp}}_{E}(\mathcal{A}) = \Pr\left[\mathcal{A}^{E_K(\cdot)} \Rightarrow 1\right] - \Pr\left[\mathcal{A}^{P(\cdot)} \Rightarrow 1\right],$$
$$\mathbf{Adv}^{\text{sprp}}_{E}(\mathcal{A}) = \Pr\left[\mathcal{A}^{E_K(\cdot),E_K^{-1}(\cdot)} \Rightarrow 1\right] - \Pr\left[\mathcal{A}^{P(\cdot),P^{-1}(\cdot)} \Rightarrow 1\right],$$
$$\mathbf{Adv}^{\text{prp-rka}}_{E}(\mathcal{A}) = \Pr\left[\mathcal{A}^{E_K(\cdot),E_{K'}(\cdot)} \Rightarrow 1\right] - \Pr\left[\mathcal{A}^{P(\cdot),P'(\cdot)} \Rightarrow 1\right],$$

where $K' = K \oplus (\texttt{0xf0f0}\cdots\texttt{f0})$. The last one is a particular form of related key attacks [5]. We fix a model of computation and a choice of encoding, and write $\mathbf{Adv}^{\text{prp}}_{E}(t,\sigma) = \max_{\mathcal{A}} \mathbf{Adv}^{\text{prp}}_{E}(\mathcal{A})$, $\mathbf{Adv}^{\text{sprp}}_{E}(t,\sigma) = \max_{\mathcal{A}} \mathbf{Adv}^{\text{sprp}}_{E}(\mathcal{A})$, and $\mathbf{Adv}^{\text{prp-rka}}_{E}(t,\sigma) = \max_{\mathcal{A}} \mathbf{Adv}^{\text{prp-rka}}_{E}(\mathcal{A})$, where the maximum is taken over adversaries $\mathcal{A}$ whose time complexity is at most t and whose query complexity is at most σ. The query complexity is the total number of queries made to the oracles.

Theorem Statement. Let $\mathcal{M}[E]$ be a MAC $\mathcal{M}$, where $E : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ is used as the underlying blockcipher. We have the following result.

Theorem 1. *Fix t, q, and σ, where $q, \sigma \ge 1$. Then the following bounds hold.*

$$\mathbf{Adv}^{\text{prf-kcv}}_{\text{MAC1.2}[E]}(t,q,\sigma) \le \mathbf{Adv}^{\text{prp}}_{E}(t',\sigma+1) + n/2^{n/2} + 7.5\sigma^2/2^n, \tag{2}$$
$$\mathbf{Adv}^{\text{prf-kcv}}_{\text{MAC2.1}[E]}(t,q,\sigma) \le \mathbf{Adv}^{\text{prp-rka}}_{E}(t',q+\sigma+1) + 3.5\sigma^2/2^{n-s}, \tag{3}$$
$$\mathbf{Adv}^{\text{prf-kcv}}_{\text{MAC2.2}[E]}(t,q,\sigma) \le 2\mathbf{Adv}^{\text{prp}}_{E}(t',\sigma+1) + 8\sigma^2/2^{n-s}, \tag{4}$$
$$\mathbf{Adv}^{\text{prf-kcv}}_{\text{MAC3}[E]}(t,q,\sigma) \le 2\mathbf{Adv}^{\text{sprp}}_{E}(t',q+\sigma+1) + 23.5\sigma^2/2^{n-s}, \tag{5}$$
$$\mathbf{Adv}^{\text{prf-kcv}}_{\text{MAC4.1}[E]}(t,q,\sigma) \le 2\mathbf{Adv}^{\text{prp-rka}}_{E}(t',2\sigma+1) + 11.5\sigma^2/2^n, \tag{6}$$

$$\mathbf{Adv}^{\text{prf-kcv}}_{\text{MAC4.2}[E]}(t,q,\sigma) \le 2\mathbf{Adv}^{\text{prp-rka}}_{E}(t',2\sigma+1) + 11.5\sigma^2/2^n, \tag{7}$$

$$\mathbf{Adv}^{\text{prf-kcv}}_{\text{MAC5}[E]}(t,q,\sigma) \le \mathbf{Adv}^{\text{prp}}_{E}(t',\sigma+1) + 5\sigma^2/2^{n-s}, \tag{8}$$

$$\mathbf{Adv}^{\text{prf-kcv}}_{\text{MAC6.1}[E]}(t,q,\sigma) \le 2\mathbf{Adv}^{\text{prp}}_{E}(t',\sigma+1) + \mathbf{Adv}^{\text{prp}}_{E}(t'',2\ell+1) + 8\sigma^2/2^n + 4.5\ell^2/2^n, \tag{9}$$

$$\mathbf{Adv}^{\text{prf-kcv}}_{\text{MAC6.2}[E]}(t,q,\sigma) \le 2\mathbf{Adv}^{\text{prp}}_{E}(t',\sigma+1) + 8\sigma^2/2^{n-s}, \tag{10}$$

where $t' = t + O(\sigma)$, $t'' = t + O(\ell+\sigma)$, *and* $\ell = \lceil k/n \rceil$.

A proof overview is presented in Sect. 7, and the proof is presented in [17].

Discussions. For the assumption about the underlying blockcipher, we require the security against related key attacks for MAC2.1, MAC4.1, and MAC4.2. These are the MACs that use $K' = K \oplus (\texttt{0xf0f0}\cdots\texttt{f0})$. MAC3 is the only MAC that requires the strong pseudorandomness assumption, as it uses the inverse of the blockcipher. The remaining MACs, MAC1.2, MAC2.2, MAC5, MAC6.1, and MAC6.2, need the standard pseudorandomness assumption.

For the security bound, we see that MAC1.2, MAC4.1, MAC4.2, and MAC6.1 have the standard birthday bound that does not depend on s. This implies that the security bounds for these MACs remain unchanged even if the key check value consists of the entire n bits. Other MACs, MAC2.1, MAC2.2, MAC3, MAC5, and MAC6.2, have the security loss by $s/2$ bits. With respect to the term $n/2^{n/2}$ in MAC1.2, we do not know if it can be removed, but there is an attack with the suggested success probability with two queries and the birthday query complexity. See Appendix B. We also note that the use of ℓ in MAC6.1 comes from the use of the key derivation function.

We argue that, if s stays relatively small as specified in [2], depending on applications, these MACs can still be used in practice. See Table 2 for the expected number of blocks of queries to attack these MACs.

7 Proof Overview of Theorem 1

Although nine MACs in Theorem 1 share the same basic structure of CBC MAC, the security proofs are different in details. Our proof of Theorem 1 can be divided into five cases, MAC1.2, MAC2.1, MAC3, MAC4.1, and MAC5. The proofs for MAC2.2, MAC6.1, and MAC6.2 are similar to that of MAC2.1. The proof for MAC4.2 follows from that of MAC4.1. The first step is to replace the blockcipher with a random permutation. This will introduce $\mathbf{Adv}^{\text{prp}}_{E}(t',\sigma')$, $\mathbf{Adv}^{\text{prp-rka}}_{E}(t',\sigma')$, or $\mathbf{Adv}^{\text{sprp}}_{E}(t',\sigma')$ depending on the usage of the underlying blockcipher. We then replace the random permutation with a random function. This will introduce a term $O(\sigma^2/2^n)$. The rest of the proofs are different depending on the MACs.

Table 2. Required number of blocks of queries to mount attacks against MAC2.1, MAC2.2, MAC3, MAC5, and MAC6.2

$n = 64$			$n = 128$				
$s = 0$	$s = 16$	$s = 24$	$s = 0$	$s = 16$	$s = 24$	$s = 32$	$s = 48$
2^{32}	2^{24}	2^{20}	2^{64}	2^{56}	2^{52}	2^{48}	2^{40}

Case MAC1.2. The analysis of MAC1.2 is quite involved. We define a number of oracles. Let M be an ℓ-bit message. After applying the padding, we have $M[1] \cdots M[m] \xleftarrow{n} \mathsf{pad2}(M)$, where $m = \lceil \ell/n \rceil + 1$. Then we define an oracle that is only used to encrypt the j-th block $M[j]$. That is, our oracles are parameterized by ℓ and j, and a specific oracle is used only for encrypting the j-th block of an ℓ-bit message M. By doing so, we eliminate the interaction between KCV and the MAC computation part, except for a rare case of computing a tag for the empty string. We proceed by showing that MAC1.2, instantiated with such oracles, is indistinguishable from the MAC1.2 that is based on a single random function. We then show that CBC MAC that uses independent random functions for every block is indistinguishable from a random function.

Case MAC2.1, MAC2.2, MAC6.1, and MAC6.2. For MAC2.1, we make use of the following lemma, where $\mathrm{CBC}_F(M)$ is the CBC MAC value of M where a random function $F : \{0,1\}^n \to \{0,1\}^n$ is used as the underlying blockcipher.

Lemma 1. *For any* $\mathsf{KCV} \in \{0,1\}^s$, $M \in \{0,1\}^{mn}$, *and* $M' \in \{0,1\}^{m'n}$, *where* $m, m' \geq 1$ *and* $M \neq M'$, *we have*

$$\Pr[\mathrm{CBC}_F(M) = \mathrm{CBC}_F(M') \mid \mathsf{msb}_s(F(0^n)) = \mathsf{KCV}] \leq \frac{mm' + \max\{m, m'\}}{2^{n-s}}.$$

Proof. To simplify the notation, let E_1 be the event $\mathrm{CBC}_F(M) = \mathrm{CBC}_F(M')$. Similarly, let E_2 be the event $\mathsf{msb}_s(F(0^n)) = \mathsf{KCV}$. Now [8, Lemma 3] shows that $\Pr[\mathsf{E}_1] \leq (mm' + \max\{m, m'\})/2^n$. We also have $\Pr[\mathsf{E}_2] = 1/2^s$. We have the claimed bound from the two probabilities and the Bayes' theorem as

$$\Pr[\mathsf{E}_1 \mid \mathsf{E}_2] = \frac{\Pr[\mathsf{E}_1 \wedge \mathsf{E}_2]}{\Pr[\mathsf{E}_2]} \leq \frac{\Pr[\mathsf{E}_1]}{\Pr[\mathsf{E}_2]} \leq \frac{mm' + \max\{m, m'\}}{2^{n-s}},$$

and this completes the proof. □

The proof of MAC2.1 is obtained by bounding the probability that we have a collision at the input of the last random function, which can be derived by using Lemma 1. We use the following lemma in the proof of MAC2.2.

Lemma 2. *For any* $\mathsf{KCV} \in \{0,1\}^s$, $\mathsf{constant} \in \{0,1\}^n$, *and* $M \in \{0,1\}^{mn}$, *where* $m \geq 1$, *we have*

$$\Pr[\mathrm{CBC}_F(M) = \mathsf{constant} \mid \mathsf{msb}_s(F(0^n)) = \mathsf{KCV}] \leq \frac{2(m+1)}{2^{n-s}}.$$

Proof. Let $\tilde{M} \leftarrow M \,\|\, \mathsf{constant}$ and $\tilde{M'} \leftarrow 0^n$. We see that if $\mathrm{CBC}_F(M) = \mathsf{constant}$ holds, then we have $\mathrm{CBC}_F(\tilde{M}) = \mathrm{CBC}_F(\tilde{M'})$. By applying Lemma 1 to $\tilde{M}$ and $\tilde{M'}$, the upper bound of $\Pr[\mathrm{CBC}_F(M) = \mathsf{constant} \mid \mathsf{msb}_s(F(0^n)) = \mathsf{KCV}]$ is obtained as

$$\Pr[\mathrm{CBC}_F(\tilde{M}) = \mathrm{CBC}_F(\tilde{M'}) \mid \mathsf{msb}_s(F(0^n)) = \mathsf{KCV}] \leq \frac{2(m+1)}{2^{n-s}},$$

which completes the proof. □

The proof of MAC2.2 closely follows that of MAC2.1, and we consider the additional event that we have 0^n at the input of the last random function. We use Lemma 2 to derive a bound on the probability. The proof for MAC6.2 is similar to that of MAC2.2, and the proof of MAC6.1 is obtained by using the result of MAC6.2 without the key check value.

Case MAC3. Let F be a random function that replaces E_K, and P' be a random permutation that replaces $E_{K'}$. Let $Q(X) = F(P'^{-1}(F(X)))$ and G be a random function. The core of the proof of MAC3 lies in proving that three oracles $\mathcal{Q} = (P'(\cdot), F(\cdot), Q(\cdot))$ are indistinguishable from three oracles $\mathcal{G} = (P'(\cdot), F(\cdot), G(\cdot))$. We show this when the domain of the first oracle, P', is restricted to $\{0^n\}$. We only need P' to generate the key check value, and hence it is sufficient for our purpose. We then replace the call of $Q(X)$ for the final block by $G(X)$, and the rest of the proof follows from those of MAC2.2 and MAC6.2.

Case MAC4.1 and MAC4.2. We define five oracles, $\mathcal{Q} = (Q_1(\cdot), \ldots, Q_5(\cdot))$. We use Q_1 and Q_2 to obtain the key check value. For a query M, we let $M[1] \cdots M[m] \xleftarrow{n} \mathsf{pad1}(M)$, and we use Q_3 to encrypt $M[1]$, Q_4 to encrypt blocks that correspond to $M[2], \ldots, M[m-1]$, and Q_5 to encrypt the last block that corresponds to $M[m]$. We show that these oracles can be used to simulate MAC4.1, and we also show that they are indistinguishable from five independent random functions. Therefore, this eliminates the interaction between the key check value and the MAC computation. The rest of the proof is similar to that of MAC2.1, and the proof of MAC4.2 follows from that of MAC4.1.

Case MAC5. For MAC5, we define seven oracles, $\mathcal{Q} = (Q_1(\cdot), \ldots, Q_7(\cdot))$. We use Q_1 for the key check value. Q_2 is used for the first block, Q_3 is used for the middle blocks, and we use four oracles $Q_4, \ldots, Q_7$ for the final block, depending on the length of the input. We show that these oracles can be used to simulate MAC5, and that they are indistinguishable from seven independent random functions. Then the MAC computation becomes independent from the key check value, and the proof follows.

8 Possible Fixes

There are applications where the security loss from using the key check value is not an issue. For instance the key check value may be computed on a master

key, and MACs are computed with session keys that are derived from the master key in a cryptographically strong way, and the master key may never be used to compute the MAC.

For other applications that need to fix the issue of reduction in security, one possible option is to change the specification of the scheme, or the other is to change the definition of the key check value. Since the latter seems to be impractical in view of the long history and the wide spread deployment of the standard, we discuss the former option. We present two generic solutions, meaning that they do not harm the provable security of the mode of operation, and they work for MACs, encryption modes, and authenticated encryption modes.

We can *always* use the key derivation function used in MAC6.1 even when the underlying blockcipher uses n-bit keys. Specifically, consider the case of MAC5, or CMAC, with 128-bit key AES. In this case, $\mathrm{AES}_K(0^n)$ is used as the key check value and $\mathrm{AES}_K(\cdot)$ is used in the actual computation of the tag. Instead, one can use $\mathrm{AES}_K(0^n)$ as the key check value, derive K' as $K' \leftarrow \mathrm{AES}_K(0^{n-1}1)$, and use $\mathrm{AES}_{K'}(\cdot)$ in computing the tag. Under the assumption that AES is a pseudorandom permutation, the key check value and $\mathrm{AES}_{K'}(\cdot)$ are independent, and thus the original security proof of MAC5 carries over.

The above solution introduces an additional key scheduling process, and we present another solution without it. Let K and K' be two independent keys for a blockcipher E. If we use $E_K(0^n)$ to derive the key check value and $E_{K'}(\cdot)$ for the mode of operation, then this clearly does not harm the provable security. Now consider a blockcipher E'_K defined as $E'_K(X) = E_K(X \oplus L) \oplus L$, where $L = E_K(0^{n-1}1)$. Then similarly to XEX construction [28], under the assumption that E is a strong pseudorandom permutation, the pairs of oracles $(E_K(\cdot), E_{K'}(\cdot), E_{K'}^{-1}(\cdot))$ and $(E_K(\cdot), E'_K(\cdot), E'^{-1}_K(\cdot))$ are indistinguishable if the first (leftmost) oracle takes only one value, 0^n, as the input. Therefore, we can use $E_K(0^n)$ to derive the key check value, and use $E'_K(\cdot)$ and $E'^{-1}_K(\cdot)$ for the mode of operation. If the mode of operation is provably secure with the pseudorandomness assumption, we can use E''_K defined as $E''_K(X) = E_K(X \oplus L)$, where $L = E_K(0^{n-1}1)$, instead of E'_K. In this case, similarly to the proof of XE construction [28], $(E_K(\cdot), E_{K'}(\cdot))$ and $(E_K(\cdot), E''_K(\cdot))$ are indistinguishable if the first oracle takes only 0^n as the input. Therefore, we can use $E_K(0^n)$ to derive the key check value, and use $E''_K(\cdot)$ for the mode of operation.

9 Conclusions

We have investigated the use of ANSI X9.24-1 key check value with the MACs specified in ISO/IEC 9797-1. MAC1.1 is widely known to be insecure, and we showed attacks against five MACs, out of nine MACs, by taking advantage of the knowledge of the key check value. We also showed that, for these five MACs, the analysis is tight and the attack cannot be improved. The results suggest that using the key check value *does* result in a security loss by $s/2$ bits, but it does *not* result in a total security loss. This indicates that, depending on the applications and the length of the key check value, they can still be used in practice even in the presence of the key check value, as the security impact is limited as long as s

is not large, say 16 or 24 as suggested in [2]. For the remaining four MACs, the security impact of using the key check value is small, even if the key check value consists of the entire block. We also presented possible ways to fix the issue of the security loss.

It would be interesting to see the impact of the key check value on the security of other blockcipher modes of operation e.g., MAC5 and MAC6 in the 1999 version of ISO/IEC 9797-1 [13], and it would also be interesting to see a more efficient way to fix the issue of the security loss, possibly a solution that depends on the mode of operation. Finally, some stronger security bounds than the standard birthday bound are known for several MACs [24,26], and it would be interesting to see if similar bounds can be obtained in the presence of the key check value.

Acknowledgments. The authors would like to thank Morris Dworkin for informing them of the issue of the key check value on CMAC, which motivated the work. The authors also would like to thank participants of Dagstuhl Seminar 09031 (Symmetric Cryptography), participants of ASK 2011 (The First Asian Workshop on Symmetric Key Cryptography), and the anonymous FSE 2014 reviewers for comments. The work by Tetsu Iwata was carried out in part while visiting Nanyang Technological University, Singapore, and was supported in part by MEXT KAKENHI, Grant-in-Aid for Young Scientists (A), 22680001, and in part by the Naito Science & Engineering Foundation. The work by Lei Wang was supported by the Singapore National Research Foundation Fellowship 2012 (NRF-NRFF2012-06).

References

1. ANSI: Financial Institution Retail Message Authentication (American Bankers Association). ANSI X9.19 (1986)
2. ANSI: Retail Financial Services Symmetric Key Management Part 1: Using Symmetric Techniques. ANSI X9.24-1:2009 (2009)
3. Barker, W.C., Barker, E.: Recommendation for the Triple Data Encryption Algorithm (TDEA) Block Cipher. NIST Special Publication 800-67, Revision 1 (2012)
4. Bellare, M., Kilian, J., Rogaway, P.: The Security of the Cipher Block Chaining Message Authentication Code. J. Comput. Syst. Sci. **61**(3), 362–399 (2000)
5. Bellare, M., Kohno, T.: A Theoretical Treatment of Related-Key Attacks: RKA-PRPs, RKA-PRFs, and Applications. In: Biham, E. (ed.) EUROCRYPT 2003. LNCS, vol. 2656, pp. 491–506. Springer, Heidelberg (2003)
6. Bellare, M., Pietrzak, K., Rogaway, P.: Improved Security Analyses for CBC MACs. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 527–545. Springer, Heidelberg (2005)
7. Black, J., Rogaway, P.: A Block-Cipher Mode of Operation for Parallelizable Message Authentication. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 384–397. Springer, Heidelberg (2002)
8. Black, J., Rogaway, P.: CBC MACs for Arbitrary-Length Messages: the Three-key Constructions. J. Cryptol. **18**(2), 111–131 (2005)
9. Bosselaers, A., Preneel, B. (eds.): Integrity Primitives for Secure Information Systems. LNCS, vol. 1007. Springer, Heidelberg (1995)
10. Dodis, Y., Pietrzak, K.: Leakage-Resilient Pseudorandom Functions and Side-Channel Attacks on Feistel Networks. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 21–40. Springer, Heidelberg (2010)

11. Dworkin, M.: Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication, NIST Special Publication 800-38B (2005)
12. Dworkin, M.: Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC, NIST Special Publication 800-38D (2007)
13. ISO/IEC: Information Technology – Security Techniques – Message Authentication Codes (MACs) – Part 1: Mechanisms Using a Block Cipher. ISO/IEC 9797-1:1999 (1999)
14. ISO/IEC: Information Technology – Security Techniques – Message Authentication Codes (MACs) – Part 1: Mechanisms Using a Block Cipher. ISO/IEC 9797-1:2011 (2011)
15. Iwata, T., Kurosawa, K.: OMAC: One-Key CBC MAC. In: Johansson, T. (ed.) FSE 2003. LNCS, vol. 2887, pp. 129–153. Springer, Heidelberg (2003)
16. Iwata, T., Kurosawa, K.: Stronger Security Bounds for OMAC, TMAC, and XCBC. In: Johansson, T., Maitra, S. (eds.) INDOCRYPT 2003. LNCS, vol. 2904, pp. 402–415. Springer, Heidelberg (2003)
17. Iwata, T., Wang, L.: Impact of ANSI X9.24-1:2009 Key Check Value on ISO/IEC 9797-1:2011 MACs. Cryptology ePrint Archive, Report 2014/183 (2014). Full version of this paper http://eprint.iacr.org/
18. Knudsen, L.: Chosen-Text Attack on CBC-MAC. Electron. Lett. **33**(1), 48–49 (1997)
19. Knudsen, L., Preneel, B.: MacDES: MAC Algorithm Based on DES. Electron. Lett. **34**(9), 871–873 (1998)
20. Luby, M., Rackoff, C.: How to Construct Pseudorandom Permutations from Pseudorandom Functions. SIAM J. Comput. **17**(2), 373–386 (1988)
21. Martin, D., Oswald, E., Stam, M.: A Leakage Resilient MAC. Cryptology ePrint Archive, Report 2013/292 (2013). http://eprint.iacr.org/
22. McGrew, D.A., Viega, J.: The Security and Performance of the Galois/Counter Mode (GCM) of Operation. In: Canteaut, A., Viswanathan, K. (eds.) INDOCRYPT 2004. LNCS, vol. 3348, pp. 343–355. Springer, Heidelberg (2004)
23. Nachtigall, E.: ICSF Verify Key Check Value. IBM Techdocs Library, Doc: PRS4840 (2011)
24. Nandi, M.: Improved Security Analysis for OMAC as a Pseudorandom Function. J. Math. Cryptol. **3**(2), 133–148 (2009)
25. Petrank, E., Rackoff, C.: CBC MAC for Real-Time Data Sources. J. Cryptol. **13**(3), 315–338 (2000)
26. Pietrzak, K.: A Tight Bound for EMAC. In: Bugliesi, M., Preneel, B., Sassone, V., Wegener, I. (eds.) ICALP 2006. LNCS, vol. 4052, pp. 168–179. Springer, Heidelberg (2006)
27. Preneel, B., van Oorschot, P.C.: On the Security of Iterated Message Authentication Codes. IEEE Trans. Inf. Theory **45**(1), 188–199 (1999)
28. Rogaway, P.: Efficient Instantiations of Tweakable Blockciphers and Refinements to Modes OCB and PMAC. In: Lee, P.J. (ed.) ASIACRYPT 2004. LNCS, vol. 3329, pp. 16–31. Springer, Heidelberg (2004)
29. Rogaway, P.: Evaluation of Some Blockcipher Modes of Operation. Investigation Reports on Cryptographic Techniques in FY 2010 (2011). http://www.cryptrec.go.jp/english/
30. Rogaway, P., Bellare, M., Black, J.: OCB: A Block-cipher Mode of Operation for Efficient Authenticated Encryption. ACM Trans. Inf. Syst. Secur. **6**(3), 365–403 (2003)

A Attacks Against MAC1.1

It is widely known that MAC1.1 is not secure for variable length messages. These attacks are known as the length-extension attack. We here recall these attacks for completeness.

Existential Forgery Against MAC1.1. Let (M, T) be a known message and tag pair, where $T = \mathrm{MAC1.1}_K(M)$. We show that existential forgeries are possible provided that $\mathsf{pad1}(M) \neq T$ holds. If $|M| \geq n$ then define

$$M^* = \mathsf{pad1}(M) \,\|\, M' \,\|\, \cdots \,\|\, M' \,\|\, T \oplus \mathsf{msb}_n(M) \,\|\, \mathsf{lsb}_{|M|-n}(M),$$

where $M' = T \oplus \mathsf{msb}_n(\mathsf{pad1}(M)) \,\|\, \mathsf{lsb}_{|\mathsf{pad1}(M)|-n}(\mathsf{pad1}(M))$. If $0 \leq |M| < n$ then define

$$M^* = \mathsf{pad1}(M) \,\|\, T \oplus \mathsf{pad1}(M) \,\|\, \cdots \,\|\, T \oplus \mathsf{pad1}(M) \,\|\, M'',$$

where M'' is a string that satisfies $0 \leq |M''| < n$ and $\mathsf{pad1}(M'') = T \oplus \mathsf{pad1}(M)$. We see that T is the correct tag for all M^* defined above.

Selective Forgery Against MAC1.1. Let M^* be the message that A tries to forge. The following selective forgery attack uses one known message and tag pair and one chosen message and tag pair. Let (M_1, T_1) be the known message and tag pair, where $T_1 = \mathrm{MAC1.1}_K(M_1)$. We assume that $M_1 \neq M^*$ and $T_1 \neq \mathsf{pad1}(M^*)$. If $M_1 = M^*$ holds, then the attack fails, and if $T_1 = \mathsf{pad1}(M^*)$, then A can ask for a different known message and tag pair. Let M_2 be

$$M_2 = \begin{cases} \mathsf{pad1}(M_1) \,\|\, T_1 \oplus \mathsf{msb}_n(M^*) \,\|\, \mathsf{lsb}_{|M^*|-n}(M^*) & \text{if } |M^*| \geq n \\ \mathsf{pad1}(M_1) \,\|\, M' & \text{else} \end{cases}$$

where M' is a string that satisfies $0 \leq |M'| < n$ and $\mathsf{pad1}(M') = T_1 \oplus \mathsf{pad1}(M^*)$. Next, A asks M_2 to its MAC oracle and obtains $T_2 = \mathrm{MAC1.1}_K(M_2)$. We see that T_2 is a valid tag for M^*.

B Existential Forgery Against MAC1.2

Let $s = n$. We show an existential forgery against MAC1.2 with a success probability of about $n/2^{n/2}$, and the query complexity of about $2^{n/2}$. Our adversary makes one query to the MAC oracle and one verification query. Now $\mathcal{A}$ is given $\mathsf{KCV} = E_K(0^n)$. Denote the integer representation of KCV as $\mathsf{int}(\mathsf{KCV})$, i.e., $\mathsf{int}(\mathsf{KCV})$ is an integer Z such that $\mathsf{bin}_n(Z) = \mathsf{KCV}$. Let ε be the empty string, and $V = \mathrm{MAC1.2}_K(\varepsilon)$ be the corresponding tag. Note that $\mathsf{pad2}(\varepsilon) = 0^n \| 0^n$ and hence $V = E_K(0^n \oplus E_K(0^n)) = E_K(\mathsf{KCV})$. $\mathcal{A}$ makes a query ε to the MAC1.2 oracle and receives the value of V. Next, $\mathcal{A}$ defines M as

$$M = V \,\|\, \mathsf{KCV} \,\|\, \cdots \,\|\, \mathsf{KCV} \,\|\, 0^m,$$

where $0 < m \leq n$ and $\mathsf{int}(\mathsf{KCV}) = |M|$. Then, it holds that

$$\mathsf{pad2}(M) = \mathsf{KCV} \,\|\, V \,\|\, \mathsf{KCV} \,\|\, \cdots \,\|\, \mathsf{KCV} \,\|\, 0^n.$$

Recall that $V = E_K(\mathsf{KCV})$ and $\mathsf{KCV} = E_K(0^n)$. We see that V is the correct tag for M defined as above.

Now we evaluate the complexity of the above attack. It is dominated by the verification of (M, V), and hence the query complexity is about $\mathsf{int}(\mathsf{KCV})/n$. Since we are interested in attacks with complexity below $2^{n/2}$, it is necessary that we have $\mathsf{int}(\mathsf{KCV}) < n2^{n/2}$, which holds with a probability of $n/2^{n/2}$ as the value of KCV is uniformly random.

Overall this existential forgery attack can be applied to MAC1.2 with a probability of $n/2^{n/2}$.