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, Japan, iwata@cse.nagoya-u.ac.jp
[2] Nanyang Technological University, 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 [15] 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 padding method, and Annex C in [15] 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.

* A preliminary version of this paper appears in the pre-proceedings of FSE 2014. This is the full version.

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 [15], also known as CMAC [12], 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 [22,13], PMAC [8,28], and OCB [30,28]. MAC5 has the proof of security [16,17], 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, [2,12] 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 [15], in the presence of the key check value. We first consider MAC2.1, which is also known as EMAC [10]. 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 $O(\sigma^2/2^n)$ [16,17]. 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 [10]), MAC3 (ANSI retail MAC [1]), and MAC6.2 (FCBC [9]).

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 [9]), 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 [27,15]. 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 [15] and [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 [11,21]. In contrast to

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	[27,15]	—	—		$O(\sigma^2/2^n)$
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	[27,15]	—	—		$O(\sigma^2/2^n)$
MAC4.2	$O(2^{n/2})$	1	[27,15]	—	—		$O(\sigma^2/2^n)$
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	[27,15]	—	—		$O(\sigma^2/2^n)$
MAC6.2	1	$O(2^{(n-s)/2})$	[Sect. 5]	0	$O(2^{(n-s)/2})$	[Sect. 5]	$O(\sigma^2/2^{n-s})$

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 [14]. 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 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^*.

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 $\mathsf{pad2}(M) = 0^{2n}$.

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 [15] 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 [15] in appearance, but they are carefully distilled for simpler presentation, and they are the same algorithms as specified in [15].

A total of six MACs are specified in [15]. The ten MACs in Fig. 2 are taken from [15, 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 [15, Annex C, Table C.1]. In MAC2.1, following the examples in [15, 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 [15, 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 MAC1.1$_K(M)$	**Algorithm** MAC1.2$_K(M)$
1. $M \leftarrow \mathsf{pad1}(M)$ 2. $T \leftarrow \mathrm{CBC}_K(M)$ 3. **return** T	1. $M \leftarrow \mathsf{pad2}(M)$ 2. $T \leftarrow \mathrm{CBC}_K(M)$ 3. **return** T
Algorithm MAC2.1$_K(M)$	**Algorithm** MAC2.2$_{K,K'}(M)$
1. $K' \leftarrow K \oplus (\texttt{0xf0f0}\cdots\texttt{f0})$ 2. $M \leftarrow \mathsf{pad1}(M)$ 3. $S \leftarrow \mathrm{CBC}_K(M)$ 4. $T \leftarrow E_{K'}(S)$ 5. **return** T	1. $M \leftarrow \mathsf{pad1}(M)$ 2. $S \leftarrow \mathrm{CBC}_K(M)$ 3. $T \leftarrow E_{K'}(S)$ 4. **return** T
Algorithm MAC3$_{K,K'}(M)$	**Algorithm** MAC4.1$_{K,K'}(M)$
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	1. $K'' \leftarrow K' \oplus (\texttt{0xf0f0}\cdots\texttt{f0})$ 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 MAC4.2$_{K,K'}(M)$	**Algorithm** MAC5$_K(M)$
1. $K'' \leftarrow K' \oplus (\texttt{0xf0f0}\cdots\texttt{f0})$ 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	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 MAC6.1$_K(M)$	**Algorithm** MAC6.2$_{K,K'}(M)$
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	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.)"

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 [15] 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 [12,13], where AES can be used. Although it is not clear how the key check value is defined for AES

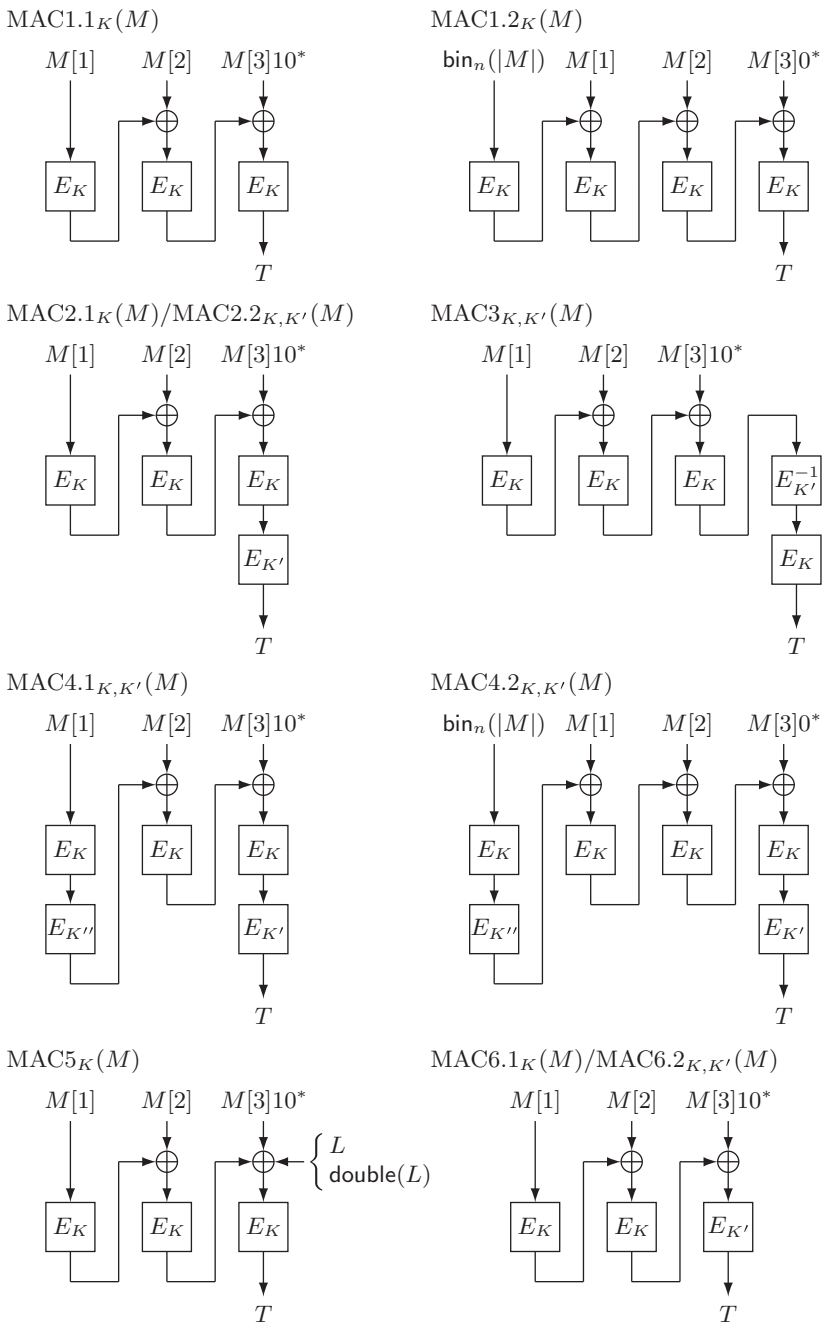

Fig. 3. Illustrations of the ISO/IEC 9797-1 MACs for $M = M[1]M[2]M[3]$, where $|M[1]| = |M[2]| = n$ and $1 \leq |M[3]| \leq n-1$. In MAC6.1, $(E_K, E_{K'})$ is replaced with $(E_{K'}, E_{K''})$.

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 $T = \text{MAC2.1}_K(M)$ 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 $0 \le |M| < n$. We assume that $\mathsf{pad1}(M) \ne \mathsf{KCV}$. Then there exists some M' such that $0 \le |M'| < n$ 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 $\mathsf{pad1}(M) \ne \mathsf{KCV}$.

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 $\mathsf{pad1}(M^*) \ne \mathsf{KCV}$. Now $\mathcal{A}$ defines M as

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

where M' is a string that satisfies $0 \le |M'| < n$ and $\mathsf{pad1}(M') = \mathsf{pad1}(M^*) \oplus \mathsf{KCV}$. Then $\mathcal{A}$ asks M to its MAC oracle and obtains $T = \text{MAC2.1}_K(M)$. 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 $\mathsf{rand}_i \neq \mathsf{rand}_j$ 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 $\mathsf{rand}'_i \neq \mathsf{rand}'_j$ 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 [20,5]. 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 $K' = K \oplus (\mathtt{0xf0f0} \cdots \mathtt{f0})$. 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 Appendices C–G.

Table 2. Required number of blocks of queries to mount attacks against MAC 2.1, 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}

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 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 $\mathbf{Adv}^{\mathrm{prp}}_{E}(t', \sigma')$, $\mathbf{Adv}^{\text{prp-rka}}_{E}(t', \sigma')$, or $\mathbf{Adv}^{\mathrm{sprp}}_{E}(t', \sigma')$ 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.

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 [9, 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 $(E_K(\cdot), E'_K(\cdot), E_K'^{-1}(\cdot))$ 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 $E_K'^{-1}(\cdot)$ 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 [14], 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. Lecture Notes in Computer Science, vol. 2656, pp. 491–506. Springer (2003)
6. Bellare, M., Pietrzak, K., Rogaway, P.: Improved Security Analyses for CBC MACs. In: Shoup, V. (ed.) CRYPTO. Lecture Notes in Computer Science, vol. 3621, pp. 527–545. Springer (2005)
7. Bellare, M., Rogaway, P.: The Security of Triple Encryption and a Framework for Code-Based Game-Playing Proofs. In: Vaudenay, S. (ed.) EUROCRYPT. Lecture Notes in Computer Science, vol. 4004, pp. 409–426. Springer (2006)
8. Black, J., Rogaway, P.: A Block-Cipher Mode of Operation for Parallelizable Message Authentication. In: Knudsen, L.R. (ed.) EUROCRYPT. Lecture Notes in Computer Science, vol. 2332, pp. 384–397. Springer (2002)
9. Black, J., Rogaway, P.: CBC MACs for Arbitrary-Length Messages: The Three-Key Constructions. J. Cryptology 18(2), 111–131 (2005)
10. Bosselaers, A., Preneel, B. (eds.): Integrity Primitives for Secure Information Systems, Final Report of RACE Integrity Primitives Evaluation RIPE-RACE 1040, Lecture Notes in Computer Science, vol. 1007. Springer (1995)
11. Dodis, Y., Pietrzak, K.: Leakage-Resilient Pseudorandom Functions and Side-Channel Attacks on Feistel Networks. In: Rabin, T. (ed.) CRYPTO. Lecture Notes in Computer Science, vol. 6223, pp. 21–40. Springer (2010)
12. Dworkin, M.: Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication. NIST Special Publication 800-38B (2005)
13. Dworkin, M.: Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC. NIST Special Publication 800-38D (2007)

14. ISO/IEC: Information Technology – Security Techniques – Message Authentication Codes (MACs) – Part 1: Mechanisms Using a Block Cipher. ISO/IEC 9797-1:1999 (1999)
15. ISO/IEC: Information Technology – Security Techniques – Message Authentication Codes (MACs) – Part 1: Mechanisms Using a Block Cipher. ISO/IEC 9797-1:2011 (2011)
16. Iwata, T., Kurosawa, K.: OMAC: One-Key CBC MAC. In: Johansson, T. (ed.) FSE. Lecture Notes in Computer Science, vol. 2887, pp. 129–153. Springer (2003)
17. Iwata, T., Kurosawa, K.: Stronger Security Bounds for OMAC, TMAC, and XCBC. In: Johansson, T., Maitra, S. (eds.) INDOCRYPT. Lecture Notes in Computer Science, vol. 2904, pp. 402–415. Springer (2003)
18. Knudsen, L.: Chosen-Text Attack on CBC-MAC. Electronics Letters 33(1), 48–49 (1997)
19. Knudsen, L., Preneel, B.: MacDES: MAC Algorithm Based on DES. Electronics Letters 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. Lecture Notes in Computer Science, vol. 3348, pp. 343–355. Springer (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. Mathematical Cryptology 3(2), 133–148 (2009)
25. Petrank, E., Rackoff, C.: CBC MAC for Real-Time Data Sources. J. Cryptology 13(3), 315–338 (2000)
26. Pietrzak, K.: A Tight Bound for EMAC. In: Bugliesi, M., Preneel, B., Sassone, V., Wegener, I. (eds.) ICALP (2). Lecture Notes in Computer Science, vol. 4052, pp. 168–179. Springer (2006)
27. Preneel, B., van Oorschot, P.C.: On the Security of Iterated Message Authentication Codes. IEEE Transactions on Information 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. Lecture Notes in Computer Science, vol. 3329, pp. 16–31. Springer (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 $n/2^{n/2}$.

C Proof of Theorem 1 (2)

C.1 Proof of MAC1.2 (Theorem 1 (2))

We let $s = n$. We first replace the blockcipher E_K in MAC1.2[E] with a random permutation $P : \{0,1\}^n \to \{0,1\}^n$. We write $\mathrm{MAC1.2}_P$ for the resulting algorithm and MAC1.2[Perm(n)] for the MAC, where the key check value is now $\mathsf{KCV} = P(0^n)$. Let $\mathcal{A}$ be an adversary, where the time complexity, the number of queries, and the query complexity of $\mathcal{A}$ are at most t, q, and σ, respectively. Now we can construct an adversary $\mathcal{B}$ that uses $\mathcal{A}$ and distinguishes between the blockcipher E_K and the random permutation P. The simulation requires at most $\sigma + 1$ calls to E_K or P, where one call is used to compute KCV. We have

$$\mathbf{Adv}^{\text{prf-kcv}}_{\text{MAC1.2}[E]}(t, q, \sigma) \leq \mathbf{Adv}^{\text{prp}}_{E}(t', \sigma + 1) + \mathbf{Adv}^{\text{prf-kcv}}_{\text{MAC1.2[Perm}(n)]}(q, \sigma),$$

where $t' = t + O(\sigma + 1)$. Note that the time complexity is omitted from the last term, since it is irrelevant to the security of MAC1.2[Perm(n)].

Then we replace P with a random function $F : \{0,1\}^n \to \{0,1\}^n$. We write $\mathrm{MAC1.2}_F$ for the algorithm and MAC1.2[Rand(n)] for the MAC. Using the PRP/PRF switching lemma [7], we obtain

$$\mathbf{Adv}^{\text{prf-kcv}}_{\text{MAC1.2[Perm}(n)]}(q, \sigma) \leq \frac{0.5(\sigma + 1)^2}{2^n} + \mathbf{Adv}^{\text{prf-kcv}}_{\text{MAC1.2[Rand}(n)]}(q, \sigma).$$

Now for an adversary $\mathcal{A}$, we evaluate $\mathbf{Adv}^{\text{prf-kcv}}_{\text{MAC1.2[Rand}(n)]}(\mathcal{A})$, which is

$$\Pr\left[\mathcal{A} \leftarrow \mathsf{KCV}, \mathcal{A}^{\mathrm{MAC1.2}_F(\cdot)} \Rightarrow 1\right] - \Pr\left[\mathcal{A} \leftarrow \mathsf{KCV}, \mathcal{A}^{\mathcal{R}(\cdot)} \Rightarrow 1\right],$$

where $\mathsf{KCV} = F(0^n)$ is the key check value given to $\mathcal{A}$.

Algorithm $\mathrm{MAC1.2}_{\mathcal{Q}}(M)$

1. $\ell \leftarrow |M|$
2. **if** $|M| = 0$ **then**
3. $\quad T \leftarrow Q^{(0)}(Q^{(0)}(0^n))$
4. $\quad$ **return** T
5. $M[1] \cdots M[m] \xleftarrow{n} \mathsf{pad2}(M)$
6. $Y[0] \leftarrow 0^n$
7. **For** $j = 1$ **to** m
8. $\quad Y[j] = Q^{(\ell,j)}(Y[j-1] \oplus M[j])$
9. $T \leftarrow Y[m]$
10. **return** T

Fig. 4. Definition of $\mathrm{MAC1.2}_{\mathcal{Q}}(M)$. KCV is defined as $\mathsf{KCV} \leftarrow Q^{(0)}(0^n)$.

Without loss of generality, assume that $\mathcal{A}$ makes exactly q queries, and suppose that $\mathcal{A}$ has access to the $\mathrm{MAC1.2}_F$ oracle. We next define a number of oracles. Let $\ell_{\max}$ be a constant, which is defined as the maximum integer satisfying $\lceil \ell_{\max}/n \rceil + 1 \le 2^{n/2}$. Our oracles are defined as follows, where $1 \le \ell \le \ell_{\max}$ and $m = \lceil \ell/n \rceil + 1$.

$$\begin{cases} Q^{(0)}(X) = F(X) \\ Q^{(\ell,1)}(X) = F(X) \oplus \mathsf{rand}^{(\ell,1)} \\ Q^{(\ell,j)}(X) = F(X \oplus \mathsf{rand}^{(\ell,j-1)}) \oplus \mathsf{rand}^{(\ell,j)} \text{ for } 2 \le j \le m-1 \\ Q^{(\ell,m)}(X) = F(X \oplus \mathsf{rand}^{(\ell,m-1)}) \end{cases}$$

Let $\mathcal{Q}$ be the set of these oracles. In the above definition, $\mathsf{rand}^{(\ell,j)} \xleftarrow{\$} \{0,1\}^n$ are all random strings of n bits. These oracles do not take arbitrarily strings as input. $Q^{(0)}(X)$ takes two inputs, 0^n and $Q^{(0)}(0^n)$, and $Q^{(\ell,1)}(X)$ takes one input, $\mathsf{bin}_n(\ell)$. We note that we have $\mathsf{bin}_n(\ell) \neq 0^n$ from the range of ℓ. The domain of other oracles is $\{0,1\}^n$.

We see that these oracles are sufficient to simulate KCV and the $\mathrm{MAC1.2}_F$ oracle. KCV is obtained as $Q^{(0)}(0^n)$. For a query M, where $|M| \neq 0$, we let $\ell \leftarrow |M|$, and then let $M[1] \cdots M[m] \xleftarrow{n} \mathsf{pad2}(M)$, where $m \leftarrow \lceil \ell/n \rceil + 1$. We proceed as $Y[1] \leftarrow Q^{(\ell,1)}(M[1])$, $Y[j] \leftarrow Q^{(\ell,j)}(Y[j-1] \oplus M[j])$ for $2 \le j \le m-1$, and the tag is obtained by $T \leftarrow Q^{(\ell,m)}(Y[m-1] \oplus M[m])$. We see that the simulation is correct since $\mathsf{rand}^{(\ell,j)}$'s are all canceled. The tag for the empty string is an exception. We obtain this by $T \leftarrow Q^{(0)}(\mathsf{KCV})$. The value of $\ell_{\max}$ and the range of ℓ are chosen to handle M such that $\mathsf{pad2}(M)$ has at most $2^{n/2}$ blocks, which is sufficient for our purpose. See Fig. 4 for the simulation, which we call $\mathrm{MAC1.2}_{\mathcal{Q}}$. We have

$$\mathbf{Adv}^{\text{prf-kcv}}_{\mathrm{MAC1.2[Rand}(n)]}(q,\sigma) = \mathbf{Adv}^{\text{prf-kcv}}_{\mathrm{MAC1.2}_{\mathcal{Q}}}(q,\sigma).$$

We next introduce a set of random functions. Let $G^{(0)}(X)$, $G^{(\ell,1)}(X)$, $G^{(\ell,j)}(X)$ for $2 \le j \le m-1$, and $G^{(\ell,m)}(X)$ be random functions, where $1 \le \ell \le \ell_{\max}$ and $m = \lceil \ell/n \rceil + 1$. These random functions have the same domain and range as $Q^{(0)}(X)$, $Q^{(\ell,1)}(X)$, $Q^{(\ell,j)}(X)$, and $Q^{(\ell,m)}(X)$, respectively. Let $\mathcal{G}$ be the set of these random functions. We claim that $\mathcal{Q}$ is indistinguishable from $\mathcal{G}$. For an adversary $\mathcal{B}$, define

$$\mathbf{Adv}^{\text{prf}}_{\mathcal{Q}}(\mathcal{B}) \stackrel{\text{def}}{=} \Pr\left[\mathcal{B}^{\mathcal{Q}} \Rightarrow 1\right] - \Pr\left[\mathcal{B}^{\mathcal{G}} \Rightarrow 1\right],$$

where $\mathcal{B}^{\mathcal{Q}}$ indicates $\mathcal{B}$ that has access to the oracles in $\mathcal{Q}$, and the same for $\mathcal{B}^{\mathcal{G}}$. In the above definition, the first probability is taken over F, $\mathsf{rand}^{(\ell,j)}$'s, and $\mathcal{B}$'s coin, and the last one is over random functions in $\mathcal{G}$ and $\mathcal{B}$'s coin. $\mathcal{B}$ makes queries of the form $(0,X) \in \{0\} \times \{0,1\}^n$, and receives $Q^{(0)}(X)$ or $G^{(0)}(X)$, or $\mathcal{B}$ makes queries of the form $(\ell,j,X) \in \{1,\ldots,\ell_{\max}\} \times$

$\{1,\ldots,\lceil \ell_{\max}/n\rceil+1\}\times\{0,1\}^n$, and receives $Q^{(\ell,j)}(X)$ or $G^{(\ell,j)}(X)$, where $1\le j\le \lceil \ell/n\rceil+1$. We only consider $\mathcal{B}$ that first makes a query $(0,0^n)$, and then makes the rest of queries to other oracles. We have the following lemma. A proof is in Appendix C.2.

Lemma 3. *Let $\mathcal{B}$ be an adversary that makes at most q queries. Then we have* $\mathbf{Adv}^{\mathrm{prf}}_{\mathcal{Q}}(\mathcal{B}) \le n/2^{n/2}+q^2/2^n$.

We next consider a variant of MAC1.2$_{\mathcal{Q}}$ where we replace $\mathcal{Q}$ in Fig. 4 with random functions in $\mathcal{G}$, which we write MAC1.2$_{\mathcal{G}}$. KCV is defined as $\mathsf{KCV}=G^{(0)}(0^n)$. We have

$$\mathbf{Adv}^{\text{prf-kcv}}_{\text{MAC1.2}_{\mathcal{Q}}}(q,\sigma) \le \mathbf{Adv}^{\text{prf-kcv}}_{\text{MAC1.2}_{\mathcal{G}}}(q,\sigma) + \frac{n}{2^{n/2}} + \frac{(\sigma+1)^2}{2^n}$$

from Lemma 3. Let $\mathcal{A}$ be an adversary and consider $\mathbf{Adv}^{\text{prf-kcv}}_{\text{MAC1.2}_{\mathcal{G}}}(\mathcal{A})$. Suppose $\mathcal{A}$ has access to the MAC1.2$_{\mathcal{G}}$ oracle, where $\mathsf{KCV}=G^{(0)}(0^n)$. Without loss of generality, we assume that $\mathcal{A}$ makes q queries, and let $M_1,\ldots,M_q$ be the queries.

We see that the only case where KCV is relevant is when $|M_i|=0$, since we use independent random functions for $|M_j|\ge 1$. Notice for non-empty queries with different bit lengths, M_i and M_j such that $|M_i|\ne|M_j|$, MAC1.2$_{\mathcal{G}}$ uses independent random functions. Thus we only need to evaluate the queries with the same bit length. We say that a bad event occurs and write $\mathcal{A}^{\text{MAC1.2}_{\mathcal{G}}}$ sets bad if $\mathsf{KCV}=0^n$, or for some $1\le i<j\le q$, we have $S_i=S_j$ for non-empty M_i and M_j such that $|M_i|=|M_j|$. Here S_i and S_j are the input values of the last random function $G^{(\ell,m)}$. We see that the absence of the bad event implies that the responses that $\mathcal{A}$ receives from the oracle are uniform and independent random bit strings of n bits. Therefore, we have

$$\mathbf{Adv}^{\mathrm{prf}}_{\text{MAC1.2}_{\mathcal{G}}}(\mathcal{A}) \le \Pr\left[\mathcal{A}^{\text{MAC1.2}_{\mathcal{G}}} \text{ sets } \mathsf{bad}\right].$$

We also see that, from the above argument, even if $\mathcal{A}$ is given KCV and then prepares all the queries $M_1,\ldots,M_q$ simultaneously, the probability that $\mathcal{A}$ produces collisions is not made smaller.

Now it is easy to see that $\Pr[\mathsf{KCV}=0^n]=1/2^n$. We see that, if $|M_i|=|M_j|=\ell\ge 1$, we have $\Pr[S_i=S_j]\le m/2^n$, where $m=\lceil \ell/n\rceil+1$. To see this, suppose that $\mathsf{pad2}(M_i)$ and $\mathsf{pad2}(M_j)$ are identical until the first k blocks. We arbitrarily fix $G^{(\ell,1)},\ldots,G^{(\ell,k)}$, and then the probability that we have a collision at the input of $G^{(\ell,k+1)}$ is 0. Under the assumption that $G^{(\ell,k+1)}$ has distinct strings as input, the probability that we have a collision at the input of $G^{(\ell,k+2)}$ is $1/2^n$. We continue this until the final block to see that $\Pr[S_i=S_j]\le m/2^n$.

Now for a set $\{M_1,\ldots,M_q\}$, we parse them into t subsets $\mathcal{M}_1,\ldots,\mathcal{M}_t$, where $|M_j|=\ell_i$ for $M_j\in\mathcal{M}_i$, that is, the strings in $\mathcal{M}_i$ share the same bit length ℓ_i. Let $m_i=\lceil \ell_i/n\rceil+1$, and suppose that $\mathcal{M}_i$ has q_i strings. From the sum bound, $\Pr[\mathcal{A}^{\text{MAC1.2}_{\mathcal{G}}} \text{ sets } \mathsf{bad}]$ is at most

$$\frac{1}{2^n}+\sum_{1\le i\le t}\frac{0.5m_iq_i^2}{2^n} \le \frac{1}{2^n}+\frac{0.5\sigma^2}{2^n}.$$

Finally we obtain the claimed bound as

$$\frac{0.5(\sigma+1)^2}{2^n}+\frac{n}{2^{n/2}}+\frac{(\sigma+1)^2}{2^n}+\frac{1}{2^n}+\frac{0.5\sigma^2}{2^n} \le \frac{n}{2^{n/2}}+\frac{7.5\sigma^2}{2^n},$$

and this completes the proof. □

C.2 Proof of Lemma 3

Instead of $\mathcal{B}$, we consider an adversary $\mathcal{B}'$ that has access to $\mathcal{Q}$, and is given values of $Q^{(0)}(0^n)$ and $Q^{(0)}(Q^{(0)}(0^n))$ as the input without making a query, and then makes $q-1$ queries. $\mathcal{B}'$ corresponds to $\mathcal{B}$ that first obtains $Q^{(0)}(0^n)$, and then $\mathcal{B}$ is forced to make a query $Q^{(0)}(Q^{(0)}(0^n))$, but $\mathcal{B}$ can make additional $q-1$ queries. That is, $\mathcal{B}'$ corresponds to $\mathcal{B}$ that obtains $Q^{(0)}(Q^{(0)}(0^n))$ without making a query, and hence the distinguishing probability of $\mathcal{B}'$ is not smaller than that of $\mathcal{B}$.

Now $\mathcal{B}'$ is given $Q^{(0)}(0^n)$ and $Q^{(0)}(Q^{(0)}(0^n))$ before making queries, and without loss of generality, we assume that $\mathcal{B}'$ makes exactly $q-1$ queries, and let $(\ell_1, j_1, X_1), \ldots, (\ell_{q-1}, j_{q-1}, X_{q-1})$ be the queries. For a query (ℓ, j, X), let $I^{(\ell,j)}(X)$ be an input value of F, i.e., we have $I^{(\ell,1)}(X) = X$, and $I^{(\ell,j)}(X) = X \oplus \mathsf{rand}^{(\ell,j-1)}$ for $2 \le j \le m$. We also let $\mathcal{I}$ be a set of all these input values. We say that a bad event occurs and write $\mathcal{B}'^{\mathcal{Q}}$ sets bad if $\mathsf{int}(Q^{(0)}(0^n)) \le \ell_{\max}$, or we have a collision in $\{0^n\} \cup \mathcal{I}$, or we have a collision in $\{Q^{(0)}(0^n)\} \cup \mathcal{I}$. Recall that $\mathsf{int}(X)$ is an integer representation of X.

The absence of the bad event implies that all the input values of F during making queries are distinct, and are different from the ones used during preparing the input of $\mathcal{B}'$. Therefore, the responses that $\mathcal{B}'$ receives from the oracles are uniform and independent random bit strings. We have

$$\mathbf{Adv}^{\mathrm{prf}}_{\mathcal{Q}}(\mathcal{B}') \le \Pr\left[\mathcal{B}'^{\mathcal{Q}} \text{ sets } \mathsf{bad}\right]. \tag{11}$$

Since the adaptivity does not help in increasing the probability of the bad event, we may now fix all queries $(\ell_1, j_1, X_1), \ldots, (\ell_{q-1}, j_{q-1}, X_{q-1})$ made by $\mathcal{B}'$. We first consider the event $\mathsf{int}(Q^{(0)}(0^n)) \le \ell_{\max}$, and the probability is at most $n/2^{n/2}$ from $\ell_{\max} \le n2^{n/2}$. Next, suppose that $\mathsf{int}(Q^{(0)}(0^n)) > \ell_{\max}$, and consider queries (ℓ, j, X) with $j = 1$. With these queries, $\mathcal{B}'$ directly chooses the input value of F. We see that, for these queries, we have $X \notin \{0^n\} \cup \{Q^{(0)}(0^n)\}$ since $X \neq 0^n$ and $X \neq Q^{(0)}(0^n)$, where the former follows from the restriction on the domain and the latter follows from $\mathsf{int}(Q^{(0)}(0^n)) > \ell_{\max}$. For the rest of the collisions, the probability is at most $1/2^n$ from the randomness of $\mathsf{rand}^{(\ell,j-1)}$'s.

Therefore, we obtain the upper bound on the probability of the bad event as

$$\Pr\left[\mathcal{B}'^{\mathcal{Q}} \text{ sets } \mathsf{bad}\right] \le \frac{n}{2^{n/2}} + \sum_{1 \le i < i' \le q+1} \frac{1}{2^n} \le \frac{n}{2^{n/2}} + \frac{q^2}{2^n},$$

and we have the lemma. □

D Proof of Theorem 1 (3), (4), (9), and (10)

D.1 Proof of MAC2.1 (Theorem 1 (3))

We first replace the blockciphers E_K and $E_{K'}$ in MAC2.1$_K$ of MAC2.1[E] with random permutations $P, P' : \{0,1\}^n \to \{0,1\}^n$. We write MAC2.1$_{P,P'}$ for the resulting algorithm and MAC2.1[Perm(n)] for the MAC, where the key check value is now $\mathsf{KCV} = \mathsf{msb}_s(P(0^n))$. Then we have

$$\mathbf{Adv}^{\text{prf-kcv}}_{\text{MAC2.1}[E]}(t, q, \sigma) \le \mathbf{Adv}^{\text{prp-rka}}_{E}(t', q + \sigma + 1) + \mathbf{Adv}^{\text{prf-kcv}}_{\text{MAC2.1}[\text{Perm}(n)]}(q, \sigma),$$

where $t' = t + O(q + \sigma + 1)$. We then replace P and P' with random functions $F, F' : \{0,1\}^n \to \{0,1\}^n$. We write MAC2.1$_{F,F'}$ for the algorithm and MAC2.1[Rand(n)] for the MAC. Using the PRP/PRF switching lemma [7], we obtain

$$\mathbf{Adv}^{\text{prf-kcv}}_{\text{MAC2.1}[\text{Perm}(n)]}(q, \sigma) \le \frac{0.5(\sigma+1)^2}{2^n} + \frac{0.5q^2}{2^n} + \mathbf{Adv}^{\text{prf-kcv}}_{\text{MAC2.1}[\text{Rand}(n)]}(q, \sigma).$$

Now for an adversary $\mathcal{A}$, we evaluate $\mathbf{Adv}^{\text{prf-kcv}}_{\text{MAC2.1[Rand}(n)\text{]}}(\mathcal{A})$, which is

$$\Pr\left[\mathcal{A} \leftarrow \mathsf{KCV}, \mathcal{A}^{\text{MAC2.1}_{F,F'}(\cdot)} \Rightarrow 1\right] - \Pr\left[\mathcal{A} \leftarrow \mathsf{KCV}, \mathcal{A}^{\mathcal{R}(\cdot)} \Rightarrow 1\right],$$

where $\mathsf{KCV} = \mathsf{msb}_s(F(0^n))$ is the key check value given to $\mathcal{A}$.

Without loss of generality, assume that $\mathcal{A}$ makes exactly q queries, and let $M_1, \ldots, M_q$ be the queries, and suppose that $\mathcal{A}$ has access to the $\text{MAC2.1}_{F,F'}$ oracle. We say that a bad event occurs and write $\mathcal{A}^{\text{MAC2.1}_{F,F'}(\cdot)}$ sets bad if $\text{CBC}_F(M_i) = \text{CBC}_F(M_j)$ holds for some $1 \le i < j \le q$.

We see that the absence of the bad event implies that the responses that $\mathcal{A}$ receives from the oracle are uniform and independent random bit strings of n bits that are unrelated to $\mathcal{A}$'s queries or F. Therefore, we have

$$\mathbf{Adv}^{\text{prf-kcv}}_{\text{MAC2.1[Rand}(n)\text{]}}(\mathcal{A}) \le \Pr\left[\mathcal{A}^{\text{MAC2.1}_{F,F'}(\cdot)} \text{ sets } \mathsf{bad}\right].$$

We also see that, from the above argument, even if A is given KCV and then prepares all the queries $M_1, \ldots, M_q$ simultaneously, the probability that $\mathcal{A}$ produces collisions in CBC_F is not made smaller.

Therefore, by the sum bound, $\Pr[\mathcal{A}^{\text{MAC2.1}_{F,F'}(\cdot)} \text{ sets } \mathsf{bad}]$ is at most

$$\sum_{1 \le i < j \le q} \Pr\left[\text{CBC}_F(M_i) = \text{CBC}_F(M_j) \mid \mathsf{msb}_s(F(0^n)) = \mathsf{KCV}\right]. \tag{12}$$

By writing $M_i[1] \cdots M_i[m_i] \xleftarrow{n} \mathsf{pad1}(M_i)$ for $1 \le i \le q$ and using Lemma 1, (12) is at most

$$\sum_{1 \le i < j \le q} \frac{m_i m_j + \max\{m_i, m_j\}}{2^{n-s}}, \tag{13}$$

and by simplifying (13) as was done in [9], we obtain $\mathbf{Adv}^{\text{prf-kcv}}_{\text{MAC2.1[Rand}(n)\text{]}}(A) \le \sigma^2/2^{n-s}$. Finally, we obtain the bound as

$$\begin{aligned}\mathbf{Adv}^{\text{prf-kcv}}_{\text{MAC2.1}[E]}(t,q,\sigma) &\le \mathbf{Adv}^{\text{prp-rka}}_{E}(t', q+\sigma+1) + \frac{0.5(\sigma+1)^2}{2^n} + \frac{0.5q^2}{2^n} + \frac{\sigma^2}{2^{n-s}} \\ &\le \mathbf{Adv}^{\text{prp-rka}}_{E}(t', q+\sigma+1) + \frac{3.5\sigma^2}{2^{n-s}},\end{aligned}$$

and this completes the proof. □

D.2 Proof of MAC2.2 (Theorem 1 (4))

The proof of MAC2.2[E] is similar to that of MAC2.1[E] in Appendix D.1. Here we show the proof of MAC2.2[E] by illustrating the differences in the proofs. We use similar notations as in Appendix D.1.

We first recall the difference between the two MACs. MAC2.2[E] uses two independent keys K and K', while MAC2.1[E] uses single key K and derives K' by $K \oplus \texttt{0xf0f0}\cdots\texttt{f0}$. Accordingly KCV is $(\mathsf{msb}_s(E_K(0^n)), \mathsf{msb}_s(E_{K'}(0^n)))$ in MAC2.2[E], while KCV is $\mathsf{msb}_s(E_K(0^n))$ in MAC2.1[E]. These differences have three impacts in the proof of MAC2.2[E].

- We may assume that E_K and $E_{K'}$ are PRPs. More precisely, when we replace them by random permutations P and P', we have

$$\mathbf{Adv}^{\text{prf-kcv}}_{\text{MAC2.2}[E]}(t,q,\sigma) \le 2\mathbf{Adv}^{\text{prp}}_{E}(t', \sigma+1) + \mathbf{Adv}^{\text{prf-kcv}}_{\text{MAC2.2[Perm}(n)\text{]}}(q,\sigma).$$

- When we replace $E_{K'}$ by P' and then replace P' by a random function F', the number of queries in the simulation becomes $q+1$, where the increased one query is to compute KCV. Then $\mathbf{Adv}^{\text{prf-kcv}}_{\text{MAC2.2[Perm}(n)]}(q,\sigma)$ is at most

$$\frac{0.5(\sigma+1)^2}{2^n}+\frac{0.5(q+1)^2}{2^n}+\mathbf{Adv}^{\text{prf-kcv}}_{\text{MAC2.2[Rand}(n)]}(q,\sigma).$$

- When we evaluate the upper bound on $\mathbf{Adv}^{\text{prf-kcv}}_{\text{MAC2.2[Rand}(n)]}(\mathcal{A})$ for an adversary $\mathcal{A}$ that has access to the $\text{MAC2.2}_{F,F'}$ oracle, we need to take into account the event $\text{CBC}_F(M_i)=0^n$, since $\mathsf{msb}_s(F'(0^n))$ is provided to $\mathcal{A}$ as a part of KCV. Therefore, in addition to have collisions at the output of CBC_F, we have a bad event if $\text{CBC}_F(M_i)=0^n$ holds for some $1\le i\le q$. This implies that $\Pr[A^{\text{MAC2.2}_{F,F'}(\cdot)}$ sets $\mathsf{bad}]$ is *enlarged* by the added bad event by at most

$$\sum_{1\le i\le q}\Pr[\text{CBC}_F(M_i)=0^n \mid \mathsf{msb}_s(F(0^n))=\mathsf{msb}_s(\mathsf{KCV})]. \tag{14}$$

  From Lemma 2, (14) is at most

$$\sum_{1\le i\le q}\frac{2(m_i+1)}{2^{n-s}}\le\frac{2(\sigma+q)}{2^{n-s}}.$$

Finally, we obtain the claimed bound from

$$\frac{0.5(\sigma+1)^2}{2^n}+\frac{0.5(q+1)^2}{2^n}+\frac{\sigma^2}{2^{n-s}}+\frac{2(\sigma+q)}{2^{n-s}}\le\frac{8\sigma^2}{2^{n-s}},$$

and this completes the proof. □

D.3 Proof of MAC6.1 (Theorem 1 (9))

We first replace E_K in MAC6.1_K of MAC6.1[E] with a random permutations P, and then replace P by a random function F. We write the resulting algorithm as MAC6.1_F and the MAC as MAC6.1[Rand(n), E], where $\mathsf{KCV}=\mathsf{msb}_s(F(0^n))$. Note that we still use $E_{K'}$ in the main CBC computation and $E_{K''}$ for the last blocks, where K' and K'' are derived from F. Then $\mathbf{Adv}^{\text{prf-kcv}}_{\text{MAC6.1}[E]}(t,q,\sigma)$ is at most

$$\mathbf{Adv}^{\text{prp}}_{E}(t'',2\ell+1)+\frac{0.5(2\ell+1)^2}{2^n}+\mathbf{Adv}^{\text{prf-kcv}}_{\text{MAC6.1[Rand}(n),E]}(t,q,\sigma), \tag{15}$$

where $t''=t+O(2\ell+\sigma+1)$.

Now we see that $\mathsf{KCV}=\mathsf{msb}_s(F(0^n))$ and $\text{MAC6.1}_F(M)$ are independent, since the computation of $\text{MAC6.1}_F(M)$ is processed with K' and K'', and these values are independent from KCV. We thus have $\mathbf{Adv}^{\text{prf-kcv}}_{\text{MAC6.1[Rand}(n),E]}(t,q,\sigma)=\mathbf{Adv}^{\text{prf}}_{\text{MAC6.2}[E]}(t,q,\sigma)$, i.e., its security is equivalent to MAC6.2[E] without KCV. We obtain the claimed bound from (15) and by setting $s=0$ in (10). □

D.4 Proof of MAC6.2 (Theorem 1 (10))

Due to structural similarity, the security proof of MAC6.2[E] closely follows that of MAC2.2[E] in Appendix D.2. We highlight the differences in the proofs by using similar notations as in Appendix D.2.

We first recall their differences. MAC6.2[E] uses $E_{K'}(\cdot)$ as the final iteration to process the last block of a padded message and to produce a tag, while MAC2.2[E] uses $E_{K'}(E_K(\cdot))$. Let $M[1]\cdots M[m] \xleftarrow{n} \mathsf{pad1}(M)$ be a partition of a message M. In MAC6.2[E], if $m = 1$, we use $M[1]$ as the input value of $E_{K'}$, and if $m \geq 2$, we use $\mathrm{CBC}_K(M[1]\cdots M[m-1]) \oplus M[m]$. On the other hand, in MAC2.2[E], we always use $\mathrm{CBC}_K(M[1]\cdots M[m])$ as the input value of $E_{K'}$. It has an impact in the proof of MAC6.2[E] *only* when we evaluate $\mathbf{Adv}^{\text{prf-kcv}}_{\text{MAC6.2[Rand}(n)]}(\mathcal{A})$ for an adversary $\mathcal{A}$ that has access to the $\text{MAC6.2}_{F,F'}$ oracle.

More precisely, the bad events are defined on the input values of F' for the adversary's queries. Recall that F' is the random function to replace $E_{K'}$. Let S_i be the input value of F' for a query M_i. Then $S_i = M_i[1]$ if $m_i = 1$, and $S_i = \mathrm{CBC}_F(M_i[1]\cdots M_i[m_i-1]) \oplus M_i[m_i]$ if $m_i \geq 2$. Two bad events in MAC6.2[Rand(n)] are defined as $S_i = S_j$ for some $1 \leq i < j \leq q$, or $S_i = 0^n$ for some $1 \leq i \leq q$.

For the former bad event, $S_i = S_j$, in fact implies $\mathrm{CBC}_F(M_i) = \mathrm{CBC}_F(M_j)$. For the latter, we always have $M_i[1] \neq 0^n$ for $m_i = 1$ from the definition of $\mathsf{pad1}$, and for $m_i \geq 2$, Lemma 2 can be used to obtain the upper bound of

$$\Pr\left[\mathrm{CBC}_F(M_i[1]\cdots M_i[m_i-1]) = M_i[m] \mid \mathsf{msb}_s(F(0^n)) = \mathsf{msb}_s(\mathsf{KCV})\right].$$

Then we see that $\Pr[A^{\text{MAC6.2}_{F,F'}(\cdot)} \text{ sets } \mathsf{bad}]$ is not larger than $\Pr[A^{\text{MAC2.2}_{F,F'}(\cdot)} \text{ sets } \mathsf{bad}]$ in Appendix D.2. Thus we have the same security bound of MAC6.2[E] as MAC2.2[E]. □

E Proof of MAC3 (Theorem 1 (5))

We replace E_K and $E_{K'}$ in $\text{MAC3}_{K,K'}$ with random permutations P and P'. Let P'^{-1} be the inverse of P'. We write $\text{MAC3}_{P,P'}$ for the resulting algorithm and MAC3[Perm(n)] for the MAC, where KCV is now $(\mathsf{msb}_s(P(0^n)), \mathsf{msb}_s(P'(0^n)))$. Then we have

$$\mathbf{Adv}^{\text{prf-kcv}}_{\text{MAC3}[E]}(t,q,\sigma) \leq 2\mathbf{Adv}^{\text{sprp}}_{E}(t',\sigma+q+1) + \mathbf{Adv}^{\text{prf-kcv}}_{\text{MAC3[Perm}(n)]}(q,\sigma),$$

where $t' = t + O(\sigma + 2q + 2)$.

Then we replace P with a random function F, but we leave P'. We write $\text{MAC3}_{F,P'}$ for the resulting algorithm and MAC3[Rand(n)] for the MAC, where $\mathsf{KCV} = (\mathsf{msb}_s(F(0^n)), \mathsf{msb}_s(P'(0^n)))$. Using the PRP/PRF switching lemma [7], we obtain

$$\mathbf{Adv}^{\text{prf-kcv}}_{\text{MAC3[Perm}(n)]}(q,\sigma) \leq \frac{0.5(\sigma+q+1)^2}{2^n} + \mathbf{Adv}^{\text{prf-kcv}}_{\text{MAC3[Rand}(n)]}(q,\sigma).$$

Let $Q(X) = F(P'^{-1}(F(X)))$ and $G : \{0,1\}^n \to \{0,1\}^n$ be a random function. We claim that three oracles $\mathcal{Q} = (P'(\cdot), F(\cdot), Q(\cdot))$ are indistinguishable from three oracles $\mathcal{G} = (P'(\cdot), F(\cdot), G(\cdot))$, where the domain of the first oracle, P', is $\{0^n\}$, i.e., it takes only 0^n as the input, and other oracles, F, Q, and G, have the domain $\{0,1\}^n$.

For an adversary $\mathcal{B}$, define

$$\mathbf{Adv}^{\text{prf}}_{\mathcal{Q}}(\mathcal{B}) \stackrel{\text{def}}{=} \Pr\left[\mathcal{B}^{\mathcal{Q}} \Rightarrow 1\right] - \Pr\left[\mathcal{B}^{\mathcal{G}} \Rightarrow 1\right],$$

where $\mathcal{B}^{\mathcal{Q}}$ is a shorthand for $\mathcal{B}^{P'(\cdot),F(\cdot),Q(\cdot)}$, the same notation is used for $\mathcal{B}^{\mathcal{G}}$, the first probability is taken over F, P', and $\mathcal{B}$'s coin, and the second probability is taken over F, P', G, and $\mathcal{B}$'s coin. We only consider $\mathcal{B}$ that first makes a query 0^n to P', and makes the rest of queries to other oracles. We have the following lemma.

Lemma 4. *Let $\mathcal{B}$ be an adversary that makes at most q queries. Then we have $\mathbf{Adv}^{\text{prf}}_{\mathcal{Q}}(\mathcal{B}) \leq 2q^2/2^n$.*

Algorithm $\mathrm{MAC3}_{\mathcal{Q}}(M)$

1. $M[1]\cdots M[m] \xleftarrow{n} \mathsf{pad1}(M)$
2. $S \leftarrow 0^n$
3. **if** $m \geq 2$ **then**
4. $\quad S \leftarrow \mathrm{CBC}_F(M[1]\cdots M[m-1])$
5. $T \leftarrow Q(S \oplus M[m])$
6. **return** T

Fig. 5. Definition of $\mathrm{MAC3}_{\mathcal{Q}}(M)$

Proof. Suppose that $\mathcal{B}$ has access to $\mathcal{Q} = (P'(\cdot), F(\cdot), Q(\cdot))$. During interacting with the oracles, we define $\mathcal{X}$ as a set of input values of the last oracle Q, and $\mathcal{X}'$ as a set of input values of the second oracle F. For a query X to Q, let $Y = F(X)$ and $Z = P'^{-1}(Y)$. We let $\mathcal{Y}$ be the set of Y's and $\mathcal{Z}$ be the set of Z's.

We say that a bad event occurs and write $\mathcal{B}^{\mathcal{Q}}$ sets bad if $\mathcal{X}' \cap \mathcal{Z} \neq \emptyset$, or $\mathcal{Z}$ has a collision. We see that, without the bad event, the responses that $\mathcal{B}$ receives from Q are uniform and independent random bit strings, and we therefore have

$$\mathbf{Adv}^{\mathrm{prf}}_{\mathcal{Q}}(\mathcal{B}) \leq \Pr\left[\mathcal{B}^{\mathcal{Q}} \text{ sets } \mathsf{bad}\right],$$

and we can now fix the queries made by $\mathcal{B}$. Let $\mathcal{X} = \{X_1, \ldots, X_{q_1}\}$ and $\mathcal{X}' = \{X'_1, \ldots, X'_{q_2}\}$ be the queries. We evaluate $\Pr[\mathcal{B}^{\mathcal{Q}} \text{ sets } \mathsf{bad}]$. Let E_1 denote the bad event $\mathcal{X}' \cap \mathcal{Z} \neq \emptyset$, or $\mathcal{Z}$ has a collision. To evaluate $\Pr[\mathsf{E}_1]$, we define two additional events, E_2 and E_3. We have E_2 if we have a collision in $\mathcal{Y}$, i.e., if $\mathcal{Y}$ has less than q_1 distinct elements. We have E_3 if we have $W \in \{Y_1, \ldots, Y_{q_1}\}$, where W is the response from the P' oracle, i.e., $W = P'(0^n)$. First, since $Y_1, \ldots, Y_{q_1}$ are all random strings, we have $\Pr[\mathsf{E}_3] \leq q_1/2^n$. Next, we have E_2 if and only if $F(X_i) = F(X_j)$ holds for some $1 \leq i < j \leq q_1$, and hence we have $\Pr[\mathsf{E}_2] \leq 0.5q_1^2/2^n$. We evaluate $\Pr[\mathsf{E}_1]$ by

$$\Pr[\mathsf{E}_1] \leq \Pr[\mathsf{E}_1 \mid \neg\mathsf{E}_2 \wedge \neg\mathsf{E}_3] + \Pr[\mathsf{E}_2] + \Pr[\mathsf{E}_3],$$

and it remains to evaluate $\Pr[\mathsf{E}_1 \mid \neg\mathsf{E}_2 \wedge \neg\mathsf{E}_3]$. Now since $Y_1, \ldots, Y_{q_1}$ are all distinct, we do not have a collision in $\mathcal{Z}$. We also see that since $Y_1, \ldots, Y_{q_1}$ are all distinct and are different from W, for each Z_i, we have at least $2^n - q_1 - 1$ choices. Therefore, $\Pr[\mathsf{E}_1 \mid \neg\mathsf{E}_2 \wedge \neg\mathsf{E}_3]$ is at most $q_1q_2/(2^n - q_1 - 1) \leq 2q_1q_2/2^n$ from $q_1 \leq 2^{n-1} - 1$. Then we have

$$\Pr\left[\mathcal{B}^{\mathcal{Q}} \text{ sets } \mathsf{bad}\right] \leq \frac{2q_1q_2}{2^n} + \frac{0.5q_1^2}{2^n} + \frac{q_1}{2^n} \leq \frac{2q^2}{2^n},$$

and this completes the proof. □

Next, we consider a MAC, called $\mathrm{MAC3}_{\mathcal{Q}}$, that uses P', F and Q, and KCV is defined as $\mathsf{KCV} = (\mathsf{msb}_s(F(0^n)), \mathsf{msb}_s(P'(0^n)))$. The definition is presented in Fig. 5. We see that $\mathrm{MAC3}_{\mathcal{Q}}(M)$ is identical to $\mathrm{MAC3}_{F,P'}(M)$. So we have

$$\mathbf{Adv}^{\text{prf-kcv}}_{\mathrm{MAC3[Rand}(n)]}(q, \sigma) = \mathbf{Adv}^{\text{prf-kcv}}_{\mathrm{MAC3}_{\mathcal{Q}}}(q, \sigma).$$

Next we replace the function Q in $\mathrm{MAC3}_{\mathcal{Q}}$ with the random function G. We write the MAC as $\mathrm{MAC3}_{\mathcal{G}}$, where $\mathsf{KCV} = (\mathsf{msb}_s(F(0^n)), \mathsf{msb}_s(P'(0^n)))$. From Lemma 4, we have

$$\mathbf{Adv}^{\text{prf-kcv}}_{\mathrm{MAC3}_{\mathcal{Q}}}(q, \sigma) \leq \mathbf{Adv}^{\text{prf-kcv}}_{\mathrm{MAC3}_{\mathcal{G}}}(q, \sigma) + \frac{2(\sigma+2)^2}{2^n}.$$

Now we see that P', and hence $\mathsf{msb}_s(P'(0^n))$, is irrelevant to $\mathbf{Adv}^{\text{prf-kcv}}_{\text{MAC3}_{\mathcal{G}}}(q,\sigma)$ since the computation of $\text{MAC3}_{\mathcal{G}}(M)$ does not depend on P'. Thus we may simply remove P' and $\mathsf{msb}_s(P'(0^n))$. We then see that the resulting algorithm is identical to MAC6.2[Rand(n)] where the last s bits of the key check value is removed. That is, MAC6.2[Rand(n)] has $(\mathsf{msb}_s(F(0^n)), \mathsf{msb}_s(F'(0^n)))$ as the key check value, but the resulting algorithm corresponds to MAC6.2[Rand(n)] with $\mathsf{msb}_s(F(0^n))$ as the key check value. Its security proof follows from that of MAC2.2[Rand(n)] by removing the evaluation of (14), and we have

$$\mathbf{Adv}^{\text{prf-kcv}}_{\text{MAC3}_{\mathcal{G}}}(q,\sigma) \le \frac{\sigma^2}{2^{n-s}}. \tag{16}$$

Finally, putting everything together, we obtain the claimed bound from

$$\frac{0.5(\sigma+q+1)^2}{2^n} + \frac{2(\sigma+2)^2}{2^n} + \frac{\sigma^2}{2^{n-s}} \le \frac{23.5\sigma^2}{2^{n-s}},$$

and this completes the proof. □

F Proof of Theorem 1 (6) and (7)

F.1 Proof of MAC4.1 (Theorem 1 (6))

We replace E_K, $E_{K'}$, and $E_{K''}$ in $\text{MAC4.1}_{K,K'}$ with random permutations P, P', and P'', respectively. Then we replace P, P' and P'' by random functions F, F', and F'', respectively. We write the resulting algorithm as $\text{MAC4.1}_{F,F',F''}$ and the MAC as MAC4.1[Rand(n)], where $\mathsf{KCV} = (\mathsf{msb}_s(F(0^n)), \mathsf{msb}_s(F'(0^n)))$. Then we obtain the upper bound on $\mathbf{Adv}^{\text{prf-kcv}}_{\text{MAC4.1}[E]}(t,q,\sigma)$ as

$$2\mathbf{Adv}^{\text{prp-rka}}_{E}(t', 2\sigma+1) + \frac{1.5(\sigma+1)^2}{2^n} + \mathbf{Adv}^{\text{prf-kcv}}_{\text{MAC4.1[Rand}(n)]}(q,\sigma),$$

where $t' = t + O(\sigma + 2q + 2)$.

We define the following five functions from F, F', F'' and rand, where $\mathsf{rand} \xleftarrow{\$} \{0,1\}^n$. We write $\mathcal{Q}$ for the set of these functions.

$$\begin{cases} Q_1(X) = \mathsf{msb}_s(F(X)) \\ Q_2(X) = \mathsf{msb}_s(F'(X)) \\ Q_3(X) = F''(F(X)) \oplus \mathsf{rand} \\ Q_4(X) = F(X \oplus \mathsf{rand}) \oplus \mathsf{rand} \\ Q_5(X) = F'(F(X \oplus \mathsf{rand})) \end{cases}$$

The domain of Q_1 and Q_2 is $\{0^n\}$, i.e., they take only 0^n as the input. Other functions have the domain of $\{0,1\}^n$. Let $G_1, \ldots, G_5$ be five independent random functions with the same domain and range as $Q_1, \ldots, Q_5$, respectively. We write $\mathcal{G}$ for the set of these functions. We will prove that $\mathcal{Q}$ is indistinguishable from $\mathcal{G}$.

For an adversary $\mathcal{B}$, define

$$\mathbf{Adv}^{\text{prf}}_{\mathcal{Q}}(\mathcal{B}) \stackrel{\text{def}}{=} \Pr\left[\mathcal{B}^{\mathcal{Q}} \Rightarrow 1\right] - \Pr\left[\mathcal{B}^{\mathcal{G}} \Rightarrow 1\right],$$

where the first probability is taken over F, F', F'', rand, and $\mathcal{B}$'s coin, and the last one is over $G_1, \ldots, G_5$ and $\mathcal{B}$'s coin. The adversary makes queries of the form $(j, X) \in \{1, \ldots, 5\} \times \{0,1\}^n$, and receives $Q_j(X)$ or $G_j(X)$. We only consider $\mathcal{B}$ that first makes a query $(1, 0^n)$ and then $(2, 0^n)$, and makes the rest of queries to other oracles. We have the following lemma.

Algorithm $\mathrm{MAC4.1}_{\mathcal{Q}}(M)$

1. $M[1]\cdots M[m] \xleftarrow{n} \mathsf{pad1}(M)$
2. $M[2] \leftarrow Q_3(M[1]) \oplus M[2]$
3. $S \leftarrow 0^n$
4. **if** $m \geq 3$ **then**
5. $\quad S \leftarrow \mathrm{CBC}_{Q_4}(M[2]\cdots M[m-1])$
6. $T \leftarrow Q_5(S \oplus M[m])$
7. **return** T

Fig. 6. Definition of $\mathrm{MAC4.1}_{\mathcal{Q}}(M)$. KCV is defined as $\mathsf{KCV} \leftarrow (Q_1(0^n), Q_2(0^n))$.

Lemma 5. *Let $\mathcal{B}$ be an adversary that makes at most q queries. Then we have $\mathbf{Adv}^{\mathrm{prf}}_{\mathcal{Q}}(\mathcal{B}) \leq 0.5q^2/2^n$.*

Proof. Suppose that $\mathcal{B}$ has access to $\mathcal{Q}$. Consider $\mathcal{B}'$ that is given $Q_1(0^n)$ and $Q_2(0^n)$ as the input, and makes $q-2$ queries to Q_3, Q_4, and Q_5. During interacting with the oracles, we define three sets. $\mathcal{I}_1$ is the set of input values of F'' in Q_3, $\mathcal{I}_2$ is the set of input values of F in Q_4, and $\mathcal{I}_3$ is the set of input values of F' in Q_5. We say that a bad event occurs and write $\mathcal{B}'^{\mathcal{Q}}$ sets bad if $\mathcal{I}_1$ has a collision, or $\{0^n\} \cup \mathcal{I}_2$ has a collision, or $\{0^n\} \cup \mathcal{I}_3$ has a collision.

Without the bad event, the responses that $\mathcal{B}'$ receives from the oracles are uniform and independent random bit strings. We have

$$\mathbf{Adv}^{\mathrm{prf}}_{\mathcal{Q}}(\mathcal{B}') \leq \Pr\left[\mathcal{B}'^{\mathcal{Q}} \text{ sets } \mathsf{bad}\right].$$

Since the adaptivity does not help in increasing the probability of the bad event, we may now fix all queries made by $\mathcal{B}'$. Suppose that $\mathcal{B}'$ makes q_3 queries to Q_3, q_4 queries to Q_4, and q_5 queries to Q_5. Then it is not hard to see $\Pr[\mathcal{I}_1 \text{ has a collision}] \leq 0.5q_3^2/2^n$, $\Pr[\{0^n\} \cup \mathcal{I}_2 \text{ has a collision}] \leq q_4/2^n$, and $\Pr[\{0^n\} \cup \mathcal{I}_3 \text{ has a collision}] \leq q_5/2^n + 0.5q_5^2/2^n$. Therefore, we obtain the upper bound on the probability of the bad event as

$$\Pr\left[\mathcal{B}'^{\mathcal{Q}} \text{ sets } \mathsf{bad}\right] \leq \frac{0.5q_3^2}{2^n} + \frac{q_4}{2^n} + \frac{q_5}{2^n} + \frac{0.5q_5^2}{2^n} \leq \frac{0.5q^2}{2^n},$$

and we have the lemma. □

We consider a MAC, which we call $\mathrm{MAC4.1}_{\mathcal{Q}}$, that uses $Q_1, \ldots, Q_5$ as oracles. Its definition is presented in Fig. 6. It is easy to verify that $\mathrm{MAC4.1}_{\mathcal{Q}}(M)$ is identical to $\mathrm{MAC4.1}_{F,F',F''}(M)$, as rand is canceled during the MAC computation. So we have

$$\mathbf{Adv}^{\text{prf-kcv}}_{\mathrm{MAC4.1[Rand}(n)]}(q,\sigma) = \mathbf{Adv}^{\text{prf-kcv}}_{\mathrm{MAC4.1}_{\mathcal{Q}}}(q,\sigma).$$

Next we replace $Q_1, \ldots, Q_5$ with $G_1, \ldots, G_5$, and write it as $\mathrm{MAC4.1}_{\mathcal{G}}$. We have

$$\mathbf{Adv}^{\text{prf-kcv}}_{\mathrm{MAC4.1}_{\mathcal{Q}}}(q,\sigma) \leq \mathbf{Adv}^{\text{prf-kcv}}_{\mathrm{MAC4.1}_{\mathcal{G}}}(q,\sigma) + \frac{0.5(\sigma+2)^2}{2^n}$$

from Lemma 5. Let $\mathcal{A}$ be an adversary and consider $\mathbf{Adv}^{\text{prf-kcv}}_{\mathrm{MAC4.1}_{\mathcal{G}}}(\mathcal{A})$. Note that KCV is completely irrelevant since the randomness used in $\mathrm{MAC4.1}_{\mathcal{G}}(M)$ does not depend on G_1 nor G_2. Thus we just need to evaluate the upper bound of $\mathbf{Adv}^{\mathrm{prf}}_{\mathrm{MAC4.1}_{\mathcal{G}}}(\mathcal{A})$, which is $\Pr[\mathcal{A}^{\mathrm{MAC4.1}_{\mathcal{G}}(\cdot)} \Rightarrow 1] - \Pr[\mathcal{A}^{\mathcal{R}(\cdot)} \Rightarrow 1]$. Now it is not hard to show that

$$\mathbf{Adv}^{\mathrm{prf}}_{\mathrm{MAC4.1}_{\mathcal{G}}}(\mathcal{A}) \leq \frac{\sigma^2}{2^n}$$

by considering the collision probability at the input of G_5. Finally we obtain the claimed bound from

$$\frac{1.5(\sigma+1)^2}{2^n} + \frac{0.5(\sigma+2)^2}{2^n} + \frac{\sigma^2}{2^n} \le \frac{11.5\sigma^2}{2^n},$$

and this completes the proof. □

F.2 Proof of MAC4.2 (Theorem 1 (7))

The proof of MAC4.2[E] is almost the same as that of MAC4.1[E]. The only difference between them is the padding algorithm, and we see that the proof of MAC4.1[E] works as long as the padding function, pad, is *injective*, namely $\mathsf{pad}(M) \neq \mathsf{pad}(M')$ holds if $M \neq M'$ holds, and $|\mathsf{pad}(M)| > n$ holds for any M. This is indeed the case for $\mathsf{pad2}$ in MAC4.2[E], and the same proof of MAC4.1[E] can be extended to MAC4.2[E]. □

G Proof of Theorem 1 (8)

G.1 Proof of MAC5 (Theorem 1 (8))

We replace E_K in MAC5$_K$ of MAC5[E] with a random permutation P, and then replace P by a random function F. We write the resulting algorithm as MAC5$_F$ and the MAC as MAC5[Rand(n)], where $\mathsf{KCV} = \mathsf{msb}_s(F(0^n))$. We have the upper bound on $\mathbf{Adv}^{\text{prf-kcv}}_{\text{MAC5}[E]}(t,q,\sigma)$ as

$$\mathbf{Adv}^{\text{prp}}_{E}(t',\sigma+1) + \frac{0.5(\sigma+1)^2}{2^n} + \mathbf{Adv}^{\text{prf-kcv}}_{\text{MAC5[Rand}(n)]}(q,\sigma),$$

where $t' = t + O(\sigma+1)$.

We proceed by defining the following seven oracles from F and rand, where $\mathsf{rand} \xleftarrow{\$} \{0,1\}^n$. Let $L = \mathsf{double}(F(0^n))$.

$$\begin{cases} Q_1(X) = \mathsf{msb}_s(F(X)) \\ Q_2(X) = F(X) \oplus \mathsf{rand} \\ Q_3(X) = F(X \oplus \mathsf{rand}) \oplus \mathsf{rand} \\ Q_4(X) = F(X \oplus \mathsf{rand} \oplus L) \\ Q_5(X) = F(X \oplus \mathsf{rand} \oplus \mathsf{double}(L)) \\ Q_6(X) = F(X \oplus L) \\ Q_7(X) = F(X \oplus \mathsf{double}(L)) \end{cases}$$

The domain of Q_1 is $\{0^n\}$, i.e., it takes only 0^n as the input, and other oracles have the domain $\{0,1\}^n$. Let $\mathcal{Q}$ be the set of these oracles. Now let $G_1,\ldots,G_7$ be seven independent random functions with the same domain and range as $Q_1,\ldots,Q_7$, respectively. Let $\mathcal{G}$ be the set of these functions. We claim that $\mathcal{Q}$ is indistinguishable from $\mathcal{G}$.

For an adversary $\mathcal{B}$, define

$$\mathbf{Adv}^{\text{prf}}_{\mathcal{Q}}(\mathcal{B}) \stackrel{\text{def}}{=} \Pr\left[\mathcal{B}^{\mathcal{Q}} \Rightarrow 1\right] - \Pr\left[\mathcal{B}^{\mathcal{G}} \Rightarrow 1\right],$$

where $\mathcal{B}^{\mathcal{Q}}$ is a shorthand for $\mathcal{B}^{Q_1(\cdot),\ldots,Q_7(\cdot)}$ and the same for $\mathcal{B}^{\mathcal{G}}$. In the above definition, the first probability is taken over F, rand, and $\mathcal{B}$'s coin, and the last one is over $G_1,\ldots,G_7$ and $\mathcal{B}$'s coin. $\mathcal{B}$ makes queries of the form $(j,X) \in \{1,\ldots,7\} \times \{0,1\}^n$, and receives $Q_j(X)$ or $G_j(X)$. We only consider $\mathcal{B}$ that first makes a query $(1,0^n)$, and then makes the rest of queries to other oracles. We have the following lemma. A proof is in Appendix G.2.

Algorithm $\mathrm{MAC5}_{\mathcal{Q}}(M)$

1. **if** $|M| = n$ **then**
2. $\quad T \leftarrow Q_6(\mathsf{pad1}(M))$
3. $\quad$ **return** T
4. **if** $0 \le |M| \le n-1$ **then**
5. $\quad T \leftarrow Q_7(M)$
6. $\quad$ **return** T
7. **if** $(|M| \ge n+1) \wedge (|M| \bmod n = 0)$ **then**
8. $\quad M[1] \cdots M[m] \xleftarrow{n} M$
9. **if** $(|M| \ge n+1) \wedge (|M| \bmod n > 0)$ **then**
10. $\quad M[1] \cdots M[m] \xleftarrow{n} \mathsf{pad1}(M)$
11. $Y[1] \leftarrow Q_2(M[1])$
12. **for** $i \leftarrow 2$ **to** $m-1$ **do**
13. $\quad X[i] \leftarrow Y[i-1] \oplus M[i]$
14. $\quad Y[i] \leftarrow Q_3(X[i])$
15. **if** $(|M| \ge n+1) \wedge (|M| \bmod n = 0)$ **then**
16. $\quad T \leftarrow Q_4(Y[m-1] \oplus M[m])$
17. **if** $(|M| \ge n+1) \wedge (|M| \bmod n > 0)$ **then**
18. $\quad T \leftarrow Q_5(Y[m-1] \oplus M[m])$
19. **return** T

Fig. 7. Definition of $\mathrm{MAC5}_{\mathcal{Q}}(M)$. KCV is defined as $\mathsf{KCV} \leftarrow Q_1(0^n)$.

Lemma 6. *Let $\mathcal{B}$ be an adversary that makes at most q queries. Then we have* $\mathbf{Adv}^{\mathrm{prf}}_{\mathcal{Q}}(\mathcal{B}) \le 0.5q^2/2^{n-s}$.

We next consider a MAC, which we call $\mathrm{MAC5}_{\mathcal{Q}}$ that uses $\mathcal{Q}$. The definition is presented in Fig. 7. We claim that $\mathrm{MAC5}_{\mathcal{Q}}(M)$ and $\mathrm{MAC5}_F(M)$ are identical, as rand is canceled during the computation of $\mathrm{MAC5}_{\mathcal{Q}}(M)$. We also see that the definition of KCV is identical in both cases. Therefore,

$$\mathbf{Adv}^{\text{prf-kcv}}_{\mathrm{MAC5[Rand}(n)]}(q,\sigma) = \mathbf{Adv}^{\text{prf-kcv}}_{\mathrm{MAC5}_{\mathcal{Q}}}(q,\sigma).$$

We next introduce another MAC, which we call $\mathrm{MAC5}_{\mathcal{G}}$ that uses $\mathcal{G}$. It is obtained from $\mathrm{MAC5}_{\mathcal{Q}}$ by replacing $Q_1, \ldots, Q_7$ with $G_1, \ldots, G_7$. From Lemma 6, we obtain

$$\mathbf{Adv}^{\text{prf-kcv}}_{\mathrm{MAC5}_{\mathcal{Q}}}(q,\sigma) \le \frac{0.5(\sigma+1)^2}{2^{n-s}} + \mathbf{Adv}^{\text{prf-kcv}}_{\mathrm{MAC5}_{\mathcal{G}}}(q,\sigma).$$

Now in $\mathrm{MAC5}_{\mathcal{G}}$, KCV is completely irrelevant since the randomness used in the computation of $\mathrm{MAC5}_{\mathcal{G}}(M)$ does not depend on G_1. It is thus left to obtain the upper bound on $\mathbf{Adv}^{\mathrm{prf}}_{\mathrm{MAC5}_{\mathcal{G}}}(\mathcal{A})$, which is $\Pr[\mathcal{A}^{\mathrm{MAC5}_{\mathcal{G}}(\cdot)} \Rightarrow 1] - \Pr[\mathcal{A}^{\mathcal{R}(\cdot)} \Rightarrow 1]$. $\mathrm{MAC5}_{\mathcal{G}}$ is almost equivalent to a MAC called MOMAC in [16]. Then we can show

$$\mathbf{Adv}^{\mathrm{prf}}_{\mathrm{MAC5}_{\mathcal{G}}}(A) \le \frac{\sigma^2}{2^n},$$

and we obtain the claimed bound from

$$\frac{0.5(\sigma+1)^2}{2^n} + \frac{0.5(\sigma+1)^2}{2^{n-s}} + \frac{\sigma^2}{2^n} \le \frac{5\sigma^2}{2^{n-s}},$$

and this completes the proof. □

G.2 Proof of Lemma 6

We consider an adversary $\mathcal{B}'$ that is given values of $Q_1(0^n)$ and $Q_2(0^n)$ as the input, and then makes $q-1$ queries. $\mathcal{B}'$ corresponds to $\mathcal{B}$ that obtains $Q_2(0^n)$ without making a query, and hence the distinguishing probability of $\mathcal{B}'$ is not smaller than that of $\mathcal{B}$.

Now $\mathcal{B}'$ is given $Q_1(0^n)$ and $Q_2(0^n)$, and without loss of generality, we assume that $\mathcal{B}'$ makes exactly $q-1$ queries, and let $(j_1, X_1), \ldots, (j_{q-1}, X_{q-1})$ be the queries. We have $1 \notin \{j_1, \ldots, j_{q-1}\}$ and $(2, 0^n) \notin \{(j_1, X_1), \ldots, (j_{q-1}, X_{q-1})\}$. Consider the case where $\mathcal{B}'$ has access to $Q_2, \ldots, Q_7$ oracles (note that we omit Q_1 as it is irrelevant). For $2 \le j \le 7$, let $I_j(X)$ be an input value of F for a query (j, X), i.e., we have $I_2(X) = X$, $I_3(X) = X \oplus \mathsf{rand}$, $I_4(X) = X \oplus \mathsf{rand} \oplus L$, $I_5(X) = X \oplus \mathsf{rand} \oplus \mathsf{double}(L)$, $I_6(X) = X \oplus L$, and $I_7(X) = X \oplus \mathsf{double}(L)$. We also let, for $2 \le j \le 7$, $\mathcal{I}_j$ be a set of all input values of F in Q_j used during making queries. Note that 0^n is not included in $\mathcal{I}_2$. We write $\mathcal{B}'^{\mathcal{Q}}$ sets bad if we have a collision in $\{0^n\} \cup \mathcal{I}_2 \cup \cdots \cup \mathcal{I}_7$. That is, we have the bad event if $I_j(X) = I_{j'}(X')$ holds for some distinct $(j, X), (j', X') \in \{(j_1, X_1), \ldots, (j_{q-1}, X_{q-1})\}$, or $I_j(X) = 0^n$ holds for some $(j, X) \in \{(j_1, X_1), \ldots, (j_{q-1}, X_{q-1})\}$.

We see that the absence of the bad event implies that the responses that $\mathcal{B}'$ receives from the oracles are uniform and independent random bit strings. Therefore, we have

$$\mathbf{Adv}^{\mathrm{prf}}_{\mathcal{Q}}(\mathcal{B}') \le \Pr\left[\mathcal{B}'^{\mathcal{Q}} \text{ sets } \mathsf{bad}\right]. \tag{17}$$

Since the adaptivity does not help in increasing the probability of the bad event, we may fix all queries $(j_1, X_1), \ldots, (j_{q-1}, X_{q-1})$ made by $\mathcal{B}'$, and evaluate the right hand side of (17) based on the randomness of $F(0^n)$ and rand under the assumption that $\mathsf{msb}_s(F(0^n))$ and $F(0^n) \oplus \mathsf{rand}$ are fixed to constants, i.e., $\mathsf{msb}_s(F(0^n)) = \mathsf{constant}_1$ and $F(0^n) \oplus \mathsf{rand} = \mathsf{constant}_2$. To simplify the notation, let E denote this event. Note that $\mathsf{msb}_s(\mathsf{rand})$ is fixed to $\mathsf{constant}_1 \oplus \mathsf{msb}_s(\mathsf{constant}_2)$, and we write $\mathsf{constant}_3$ for this fixed value.

Now if $j = j'$, then we have $X \neq X'$, and hence we never have $I_j(X) = I_{j'}(X')$. So it remains to evaluate $\Pr[I_j(X) = I_{j'}(X') \mid \mathsf{E}]$ for $2 \le j < j' \le 7$, and $\Pr[I_j(X) = 0^n \mid \mathsf{E}]$ for $2 \le j \le 7$.

We first consider $\Pr[I_j(X) = I_{j'}(X') \mid \mathsf{E}]$. Using the relation of $\mathsf{rand} = \mathsf{constant}_2 \oplus F(0^n)$ and $L = \mathsf{double}(F(0^n))$, we have $I_2(X) = X$, $I_3(X) = X \oplus \mathsf{constant}_2 \oplus F(0^n)$, $I_4(X) = X \oplus \mathsf{constant}_2 \oplus F(0^n) \oplus \mathsf{double}(F(0^n))$, $I_5(X) = X \oplus \mathsf{constant}_2 \oplus F(0^n) \oplus \mathsf{double}(\mathsf{double}(F(0^n)))$, $I_6(X) = X \oplus \mathsf{double}(F(0^n))$, and $I_7(X) = X \oplus \mathsf{double}(\mathsf{double}(F(0^n)))$. Now it is easy to check that, for any $2 \le j < j' \le 7$, the event $I_j(X) = I_{j'}(X')$ can be simplified as $F(0^n) = Y$ for some constant Y. Therefore $\Pr[I_j(X) = I_{j'}(X') \mid \mathsf{E}]$ is 0 if $\mathsf{msb}_s(Y) \neq \mathsf{constant}_3$, and otherwise it is $1/2^{n-s}$.

We next consider $\Pr[I_j(X) = 0^n \mid \mathsf{E}]$. Similarly to the above, the event $I_j(X) = 0^n$ can be written as $F(0^n) = Y$ for some constant Y when $3 \le j \le 7$, and this event never occurs when $j = 2$. Therefore, $\Pr[I_j(X) = 0^n \mid \mathsf{E}]$ is at most $1/2^{n-s}$.

Finally we evaluate the probability of the bad event. We have

$$\Pr\left[\mathcal{B}'^{\mathcal{Q}} \text{ sets } \mathsf{bad}\right] \le \sum_{1 \le i < i' \le q-1} \frac{1}{2^{n-s}} + \sum_{1 \le i \le q-1} \frac{1}{2^{n-s}} \le \frac{0.5q^2}{2^{n-s}}$$

as claimed. □