LedMAC: More Efficient Variants of LightMAC

Yaobin Shen, Lei Wang, and Dawu Gu

Shanghai Jiao Tong University
yb_shen@sjtu.edu.cn, wanglei_hb@sjtu.edu.cn, dwgu@sjtu.edu.cn

Abstract. LightMAC is a lightweight MAC designed by Luykx et al. and recently standardized by ISO/IEC. In this paper, we refine LightMAC and suggest two simple variants called LedMAC1 and LedMAC2. Compared to LightMAC, our first scheme LedMAC1 avoids unnecessary padding without sacrificing the security. Our second scheme LedMAC2 further reduces the number of keys from two to one, and achieves the same level security as that of LightMAC.

Keywords: Lightweight cryptography · Message authentication code · Provable security

1 Introduction

This paper describes two simple variants of LightMAC. Our algorithms efficiently handle message of any bit length, and reduce the number of blockcipher keys. This makes them more suitable to be implemented in resource-constrained devices. We begin with some background.

Lightweight cryptography. Lightweight cryptography is a subfield of cryptography that aims to secure the communication of small devices, which are typically not powerful enough to use conventional cryptography. With the advent of Internet of Things, a myriad of small devices are being placed everywhere. These small devices such as embedded systems, RFID tags, and sensor networks are resource-constrained with limited processing and storage capacities. The cost of the cryptographic algorithms is important, especially for small devices.

LightMAC. LightMAC is an elegant lightweight MAC designed by Luykx et al. [10]. It aims to ensure the data authenticity for resource-constrained devices. It is proved to have a security bound $q^2/2^n$ that is independent of the message length, where q is the number of MAC queries and n is the block size of the underlying blockcipher. Let us recall how LightMAC works. Let $E : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ be a blockcipher. The LightMAC follows the Hash-then-PRF paradigm, and is built on top of a blockcipher E with two independent keys K_1 and K_2. Let i_s be the s-bit encoding of a counter i for $i \leq 2^s - 1$. Let 10^* be the padding that appending a one followed by as few zeros to make the total string

length a multiple of n. Let $M = M[1] \| \ldots \| M[\ell]$ be a message that we want to authenticate, where $|M[1]| = \ldots = |M[\ell-1]| = n-s$ and $0 \leq M[\ell] \leq n-s-1$. Then $\mathsf{LightMAC}_{K_1,K_2}(M)$, the tag of M under keys K_1 and K_2, is the value T where $T = E_{K_2}(S)$ and $S = M[\ell]10^* \oplus \left(\bigoplus_{i=1}^{\ell-1} E_{K_1}(i_s \| M[i])\right)$.

Let us recall the brief history of LightMAC. LightMAC evolved from two algorithms XOR MAC [1] and PCS [4]. The former XORed together finite-input-length pseudorandom functions (PRF) to create a stateful and randomized MAC, and the latter composes an XOR MAC with an independent PRF call to create a stateless and deterministic MAC. Yasuda [11] also mentioned the basic idea for LightMAC. Recently, LightMAC was standardized by ISO/IEC [8] for lightweight cryptography.

Our contribution. In this paper, we refine LightMAC and propose two simple variants called LedMAC1 and LedMAC2. Compared to LightMAC, our first algorithm LedMAC1 avoids unnecessary padding without sacrificing the security. Our second algorithm further reduces the number of keys to be one, and achieves the same level security as that of LightMAC.

One unfortunate feature of LightMAC is that there is some waste in the last block. First of all, the length of the last block is only up to $n-s-1$ bits, and at least $s+1$ redundant bits are always appended regardless of the length of the message. This results in the maximal length of a message being up to $2^s(n-s)-1$ bits.[1] Secondly, if $|M|$ is already a multiple of $n-s$, then an entire extra block of padding is appended, seemingly "wasting" an application of blockcipher E. Cryptographers have worked hard to optimize blockciphers, and it seems embarrassing to squander some of this efficiency with an unnecessary blockcipher call, especially for lightweight cryptography. Moreover, there are settings where one needs to authenticate short messages that are always or often a multiple of $n-s$. In such a case, saving one blockcipher call can be a significant performance improvement.

Our new scheme LedMAC1 avoids this waste: the length of the last block can be up to $n-1$ bits, therewith the maximal length of a message being up to $2^s(n-s)+s-1$ bits. Moreover, there is no extra padding when $|M|$ is a multiple of $n-s$. We simply use a one-bit counter in the finalization to make two cases. When the last block of message M is full of $n-1$ bits, a single 0 will be prefixed before the finalization. Otherwise a single 1 will be prefixed before the finalization. Note that this one-bit counter comes without any additional cost. Our proof shows that except for a small factor 2, LedMAC1 enjoys the same security bound as that of LightMAC.

Reducing the number of keys. Both LightMAC and LedMAC1 require two key schedulings of the underlying blockcipher E, which is undesired for resource-constrained devices. Our second scheme LedMAC2 further reduces the number

[1] It is claimed [8,10] that the maximal length of a message can be up to $2^s(n-s)$ bits. However, this claim is inaccurate since the maximal number of blocks of a message is at most 2^s and the maximal bit length of the last block is $n-s-1$ bits.

of blockcipher keys, and achieves the same level security as that of LightMAC. Here we use a two-bit counter: one bit for padding as is done in LedMAC1, one bit for domain separation between the hash phase and the finalization phase of LedMAC2. The cost to save one blockcipher key is that for a s-bit counter in the hash phase, the maximal length of a message now is up to $2^{s-1}(n-s)+s-2$ due to the one-bit domain separation.

We prove that LedMAC2 has a security bound $q\sigma/2^n$ where σ is the total number of blocks of these q queries. The security bound $q\sigma/2^n$ depends on the message length, and is slightly worse than the bound $q^2/2^n$ of LightMAC and LedMAC1. However, LedMAC2 only requires one blockcipher key, and thus may be more suitable for implementation in resource-constrained environment.

WHY NOT USE A MASK. It might be helpful to handle the padding via some masks, as is done in previous constructions such as PMAC [5], CMAC [7,9] and XCBC [6]. However, this requires either an extra blockcipher invocation plus field multiplication or a longer key, and is somewhat costly for small devices. Hence we choose to use a second counter, which is more natural for counter-based MACs. Moreover, the counter can also be used to reduce the number of blockcipher keys.

2 Preliminaries

NOTATION. Let ε denote the empty string. Let $\{0,1\}^*$ be the set of all finite bit strings including the empty string ε. For a finite set S, we let $x \xleftarrow{\$} S$ denote the uniform sampling from S and assigning the value to x. Let $|x|$ denote the length of the string x. Let $x[i:j]$ denote the substring from the i-th bit to the j-th bit (inclusive) of x. Concatenation of strings x and y is written as $x \,\|\, y$ or simply xy. For an integer $i \le 2^s - 1$, let i_s be the s-bit encoding of i. If A is an algorithm, we let $y \leftarrow A(x_1, \ldots; r)$ denote running A with randomness r on inputs $x_1, \ldots$ and assigning the output to y. We let $y \xleftarrow{\$} A(x_1, \ldots)$ be the result of picking r at random and letting $y \leftarrow A(x_1, \ldots; r)$. Let $\mathrm{Perm}(n)$ denote the set of all permutations over $\{0,1\}^n$, and let $\mathrm{Func}(*, n)$ denote the set of all functions from $\{0,1\}^*$ to $\{0,1\}^n$.

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 $E_K(\cdot) = E(K, \cdot)$ is a permutation over $\{0,1\}^n$ specified by the key $K \in \{0,1\}^k$. A MAC algorithm $F : \mathcal{K} \times \{0,1\}^* \to \{0,1\}^\tau$ is a function. It takes as input a key $K \in \mathcal{K}$ and a message $M \in \{0,1\}^*$, and outputs a fixed-length tag $T \in \{0,1\}^\tau$. The value τ is called the tag length, and we consider the case $\tau = n$.

SECURITY DEFINITIONS. An adversary A is an algorithm that always outputs a bit. We write $A^O = 1$ to denote the event that A outputs 1 when given access to oracle O. Let $\pi \xleftarrow{\$} \mathrm{Perm}(n)$ be a random permutation. The advantage of A against the PRP security of E is defined as

$$\mathsf{Adv}^{\mathrm{prp}}_E(A) = \Pr\left[\, A^{E_K} = 1 \,\right] - \Pr\left[\, A^{\pi} = 1 \,\right]$$

where K is chosen uniformly at random from $\{0,1\}^k$.

Let $F : \mathcal{K} \times \{0,1\}^* \to \{0,1\}^n$ be a MAC algorithm. Let $\mathcal{R} \leftarrow\!\!\$\, \mathrm{Func}(*, n)$ be a random function. The advantage of A against the PRF security of F is defined as

$$\mathsf{Adv}^{\mathrm{prf}}_{F}(A) = \Pr\left[\, A^{F_K} = 1 \,\right] - \Pr\left[\, A^{\mathcal{R}} = 1 \,\right]$$

where K is chosen uniformly at random from $\mathcal{K}$. We note that the above definition captures the security of a MAC as a pseudorandom function (PRF). It is well known that any PRF is a secure MAC [2].

3 Security of LedMAC1

In this section, we prove the PRF security of LedMAC1. Our result shows that compared with LightMAC, LedMAC1 avoids unnecessary padding without sacrificing the security.

The LedMAC1 construction. Let $E : \{0,1\}^k \times \{0,1\}^n$ be a blockcipher. The LedMAC1 is based on the Hash-then-PRF paradigm, and is built on top of a blockcipher E with two keys. The first key is used to hash the message into a fixed-length string, and the second blockcipher key is used to prf the fixed-length string to produce the tag. When the final block of message M is full of $n-1$ bits, the fixed-length string will be prefixed with a single 0 before the PRF phase. Otherwise it will be prefixed with a single 1.The specification of LedMAC1 is given in Fig. 1.

Security analysis of LedMAC1. The following result shows that LedMAC1 is a good PRF, provided that E_{K_1} and E_{K_2} are two good PRPs.

Theorem 1. *For any adversary A against the PRF security of* LedMAC1*, running in time t, making at most q queries of length at most $2^s(n-s)+s-1$ bits, we have*

$$\mathsf{Adv}^{\mathrm{prf}}_{\mathsf{LedMAC1}[E]}(A) \le \mathsf{Adv}^{\mathrm{prp}}_{E}(B_1) + \mathsf{Adv}^{\mathrm{prp}}_{E}(B_2) + \frac{5q^2}{2^{n+1}}$$

where B_1 and B_2 are two adversaries against the PRP security of the blockcipher E, the former running in time $t' = t + O(q \cdot (2^s - 1))$ and making at most $q \cdot (2^s - 1)$ queries, and the latter running in time $t'' = t + O(q)$ and making at most q queries.

Proof. Let A be an adversary against the PRF security of LedMAC1, running in time t, making at most q queries of length at most $2^s(n-s)+s-1$ bits. Without loss of the generality, we assume the adversary never repeats a prior query since otherwise it will receive the same response. Following the standard argument, we first replace the blockciphers E_{K_1} and E_{K_2} of $\mathsf{LedMAC1}[E]$ with two independent random permutations π_1 and π_2 to obtain $\mathsf{LedMAC1}[\pi]$. Let B_1 be an adversary against the PRP security of E_{K_1}. It picks up a key K_2 uniformly at random from $\{0,1\}^k$ to simulate E_{K_2}. It runs A and simulates A's oracle by using its own oracle and the simulated E_{K_2}. After the interaction, B uses A's

procedure $\mathsf{LedMAC1}_{K_1,K_2}(M)$
$M[1] \parallel \ldots \parallel M[\ell] \leftarrow M; S \leftarrow 0^n$
for $i \leftarrow 1$ **to** $\ell - 1$ **do**
  $S \leftarrow S \oplus E_{K_1}(i_s \parallel M[i])$
if $|M[\ell]| = n - 1$ **then**
  $V \leftarrow M[\ell] \oplus S[1 : n-1]$
  $T \leftarrow E_{K_2}(0 \parallel V)$
else $V \leftarrow M[\ell]10^* \oplus S[1 : n-1]$; $T \leftarrow E_{K_2}(1 \parallel V)$
return T

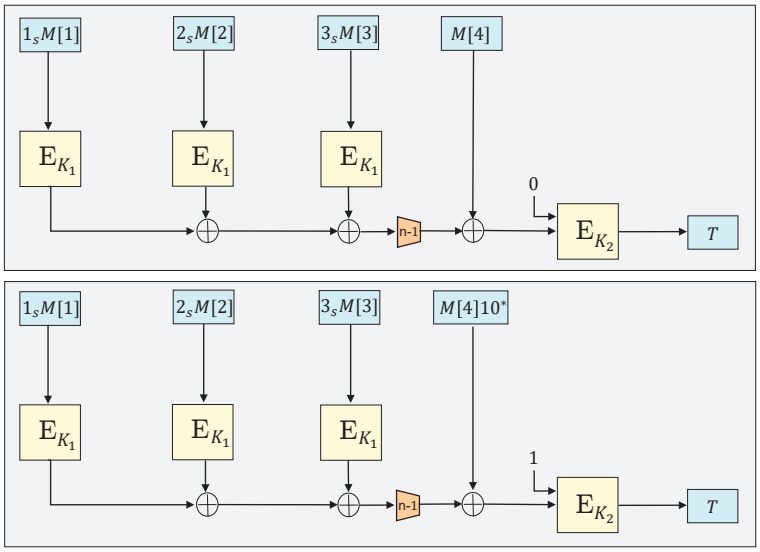

Fig. 1: **Top:** The detailed code description of $\mathsf{LedMAC1}$. Here $M[1] \parallel \ldots \parallel M[\ell] \leftarrow M$ denotes splitting M into $(n-s)$-bit blocks and stopping at the moment when it is left with a message of length less than or equal to $n-1$ bits. $M[\ell]10^*$ denotes right padded with a single 1 and as few 0 bits so that the length of string to be $(n-1)$-bit. **Middle:** The case when the final block of message M is full of $n-1$ bits, i.e., $|M[4]| = n-1$. **Bottom:** The case when the final block of message M is less than $n-1$ bits, i.e., $0 \leq |M[4]| < n-1$.

response as its own. Let B_2 be an another adversary against the PRP security of E_{K_2}. It follows a similar strategy as that of B_1. Then the PRF advantage of A against $\mathsf{LedMAC1}$ can be bounded by

$$\mathsf{Adv}^{\mathrm{prf}}_{\mathsf{LedMAC1}[E]}(A) \leq \mathsf{Adv}^{\mathrm{prp}}_{E}(B_1) + \mathsf{Adv}^{\mathrm{prp}}_{E}(B_2) + \mathsf{Adv}^{\mathrm{prf}}_{\mathsf{LedMAC1}[\pi]}(A) \ ,$$

where B_1 runs in time $t' = t + O(q \cdot (2^s - 1))$ and makes at most $q \cdot (2^s - 1)$ queries, and B_2 runs in time $t'' = t + O(q)$ and makes at most q queries.

We then focus on the last term on the right side of the inequality. For message M, let $H(M)$ denote the input to π_2 of LedMAC1, and

$$H(M) = 0 \,\|\, \left(M[\ell] \oplus \left(\bigoplus_{i=1}^{\ell-1} \pi_1(i_s \,\|\, M[i]) \right) [1 : n-1] \right)$$

if $|M[\ell]| = n-1$, and

$$H(M) = 1 \,\|\, \left(M[\ell]10^* \oplus \left(\bigoplus_{i=1}^{\ell-1} \pi_1(i_s \,\|\, M[i]) \right) [1 : n-1] \right)$$

if $0 \leq |M[\ell]| < n-1$. Denote by $\mathrm{Rng}(\pi_2)$ the set of output values of π_2 during the computation of LedMAC1. The size of $\mathrm{Rng}(\pi_2)$ will increase when A is interacting with LedMAC1. During the interaction between A and its oracle LedMAC1, we define two bad events as follows:

- bad1: at the i-th query, if $H(M_i)$ collides with previous $H(M_j)$ for $1 \leq j \leq i-1$, then set bad1 to be true.
- bad2: at the i-th query, after the computation of $H(M_i)$, sample a string $\widehat{T}_i$ independently and uniformly at random from $\{0,1\}^n$. If $\widehat{T}_i$ lies in $\mathrm{Rng}(\pi_2)$, then set bad2 to be true.

If neither of these two bad events happens, then we let $T_i = \widehat{T}_i$ and return T_i to adversary A. Otherwise we return $T_i = \pi_2(H(M_i))$. On the other hand, when A is interacting with a random function, we define the same bad events as above, but regardless of bad events happen or not, we always return $T_i = \widehat{T}_i$ to adversary A. If neither of these two bad events happens, then the output strings $T_1, \ldots, T_q$ from LedMAC1 are merely chosen independently and uniformly at random from $\{0,1\}^n$, which are exactly the same as the outputs from a random function. Denote by $\mathsf{bad} = \mathsf{bad1} \cup \mathsf{bad2}$. Then by the fundamental lemma of game-playing techniques [3], we have

$$\mathsf{Adv}^{\mathrm{prf}}_{\mathsf{LedMAC1}[\pi]}(A) \leq \Pr[\,\mathsf{bad}\,] \ .$$

We now bound the probability of bad events happen when A is interacting with a random function. Note that during this interaction, adversary A always receives random strings from its oracle, which are independent of its queries. In other words, the adaptive queries do not help here. For the event bad1,

$$\begin{aligned}
\Pr[\,\mathsf{bad1}\,] &\leq \sum_{i=2}^{q} \sum_{j=1}^{i-1} \Pr[\,H(M_i) = H(M_j)\,] \\
&\leq \sum_{i=2}^{q} \sum_{j=1}^{i-1} \frac{2}{2^n - 2 \cdot 2^s} \\
&\leq \sum_{i=2}^{q} \sum_{j=1}^{i-1} \frac{2}{2^{n-1}} \leq \frac{2q^2}{2^n} \ ,
\end{aligned}$$

where the second inequality is due to the following Lemma 1, and the third inequality is from the assumption that $s \leq n/2$. For the event bad2, since each T_i is chosen uniformly at random from $\{0,1\}^n$ and there are at most $i-1$ elements in the set $\mathrm{Rng}(\pi_2)$ at the i-th query,

$$\Pr[\,\mathsf{bad2}\,] \leq \sum_{i=1}^{q} \frac{i-1}{2^n} \leq \frac{q^2}{2^{n+1}} \ .$$

Thus by the union bound,

$$\mathsf{Adv}^{\mathrm{prf}}_{\mathsf{LedMAC1}[\pi]}(A) \leq \frac{2q^2}{2^n} + \frac{q^2}{2^{n+1}} = \frac{5q^2}{2^{n+1}} \ .$$

Summing up, the PRF advantage of A against LedMAC1 is at most

$$\mathsf{Adv}^{\mathrm{prf}}_{\mathsf{LedMAC1}[E]}(A) \leq \mathsf{Adv}^{\mathrm{prp}}_{E}(B_1) + \mathsf{Adv}^{\mathrm{prp}}_{E}(B_2) + \frac{5q^2}{2^{n+1}} \ .$$

□

Lemma 1. *For any message $M = M[1] \parallel \ldots \parallel M[\ell]$ where $|M[i]| = n-s$ for $1 \leq i \leq \ell-1$ and $0 \leq |M[\ell]| \leq n-1$, define $H(M)$ to be $H(M) = 0 \parallel \left(M[\ell] \oplus \left(\bigoplus_{i=1}^{\ell-1} \pi_1(i_s \parallel M[i])\right)[1:n-1]\right)$ if $|M[\ell]| = n-1$, and $H(M) = 1 \parallel \left(M[\ell]10^* \oplus \left(\bigoplus_{i=1}^{\ell-1} \pi_1(i_s \parallel M[i])\right)[1:n-1]\right)$ if $0 \leq |M[\ell]| < n-1$, where π is a random permutation over $\{0,1\}^n$. Then for any two distinct messages $M_1 = M_1[1] \parallel \ldots \parallel M_1[\ell_1]$ and $M_2 = M_2[1] \parallel \ldots \parallel M_2[\ell_2]$,*

$$\Pr[\,H(M_1) = H(M_2)\,] \leq \frac{2}{2^n - \ell_1 - \ell_2 + 3} \ .$$

Proof. Without loss of the generality, we assume $\ell_1 \leq \ell_2$. We first consider the case when $\ell_1 = \ell_2$. We analyze the probability of collision according to whether the final block of messages is full of $n-1$ bits or not.

– Both $M_1[\ell_1]$ and $M_2[\ell_1]$ have the length of $n-1$ bits. If $M_1[i] = M_2[i]$ for $1 \leq i \leq \ell_1 - 1$, and $M_1[\ell_1] \neq M_2[\ell_1]$, then obviously there is no collision between $H(M_1)$ and $H(M_2)$. If there exists some $1 \leq i \leq \ell_1 - 1$ such that $M_1[i] \neq M_2[i]$, then $H(M_1) = H(M_2)$ happens with probability at most $2/(2^n - \ell_1 - \ell_2 + 3)$ since $\pi(i_s \parallel M_1[i])$ is distributed uniformly at random in a set of size at least $2^n - \ell_1 - \ell_2 + 3$ and will not be canceled out by other variables in this equation, and

$$\begin{aligned}\Pr[\,\pi(i_s \parallel M_1[i])[1:n-1] = y\,] &\leq \Pr[\,\pi(i_s \parallel M_1[i]) \in \{y \parallel 0, y \parallel 1\}\,] \\ &\leq \frac{2}{2^n - \ell_1 - \ell_2 + 3}\end{aligned}$$

for any $y \in \{0,1\}^{n-1}$.

– Neither $M_1[\ell_1]$ nor $M_2[\ell_1]$ has the length of $n-1$ bits. Then following a similar argument as in the previous case, the collision $H(M_1) = H(M_2)$ occurs with probability at most $2/(2^n - \ell_1 - \ell_2 + 3)$.
– $M_1[\ell_1]$ has the length of $n-1$ bits while $M_2[\ell_1]$ does not. Then obviously there is no collision between $H(M_1)$ and $H(M_2)$ since they have different prefix.
– $M_2[\ell_2]$ has the length of $n-1$ bits while $M_1[\ell_1]$ does not. Then obviously there is no collision between $H(M_1)$ and $H(M_2)$ since they have different prefix.

Hence, when $\ell_1 = \ell_2$, the probability of $H(M_1) = H(M_2)$ occurs is at most $2/(2^n - \ell_1 - \ell_2 + 3)$. We then consider the case when $\ell_1 < \ell_2$. In this case, the variable $\pi((\ell_2 - 1)_s \,\|\, M_2[\ell_2 - 1])$ is distributed uniformly at random in a set of size at least $2^n - \ell_1 - \ell_2 + 3$, and will not be canceled out by other variables in the equation. Hence the equation $H(M_1) = H(M_2)$ holds with probability at most $2/(2^n - \ell_1 - \ell_2 + 3)$. □

4 Security of LedMAC2

Note that both LightMAC and LedMAC2 requires two key schedulings of the underlying blockcipher E. In this section, we propose LedMAC2 which reduces the number of keys from two to one. Our result shows that LedMAC2 achieves the same-level security as that of LedMAC2, while requires only one blockcipher key.

The LedMAC2 construction. Let $E : \{0,1\}^k \times \{0,1\}^n$ be a blockcipher. The LedMAC2 is based on the Hash-then-PRF paradigm, and is built from a blockcipher E with one key. The hash phase and the PRF phase use the same blockcipher key, while an additional counter bit is used to separate the input domain between these two phases. When the final block of message M is full of $n-2$ bits, the fixed-length string will be prefixed with a two-bit counter 10 before the PRF. Otherwise it will be prefixed with another two-bit counter 11. The specification of LedMAC2 is given in Fig. 2.

Discussion. The following security bound $q\sigma/2^n$ of LedMAC2 depends on the message length, which is slightly worse than the bound $q^2/2^n$ of LightMAC and LedMAC2. However, compared with LightMAC and LedMAC2, LedMAC2 requires only one blockcipher key, and thus may be more suitable for implementation in resource-constrained environment.

Security analysis of LedMAC2. The following result shows that LedMAC2 is a good PRF, provided that E_K is a good PRP.

Theorem 2. *For any adversary A against the PRF security of* LedMAC2*, running in time t, making at most q queries of length at most $2^{s-1}(n-s) + s - 2$*

procedure $\mathsf{LedMAC2}_K(M)$
$M[1] \parallel \ldots \parallel M[\ell] \leftarrow M$; $S \leftarrow 0^n$
for $i \leftarrow 1$ **to** $\ell - 1$ **do**
  $S \leftarrow S \oplus E_K(0 \parallel i_s \parallel M[i])$
if $|M[\ell]| = n - 2$ **then**
  $V \leftarrow M[\ell] \oplus S[1 : n-2]$
  $T \leftarrow E_K(10 \parallel V)$
else $V \leftarrow M[\ell]10^* \oplus S[1 : n-2]$; $T \leftarrow E_K(11 \parallel V)$
return T

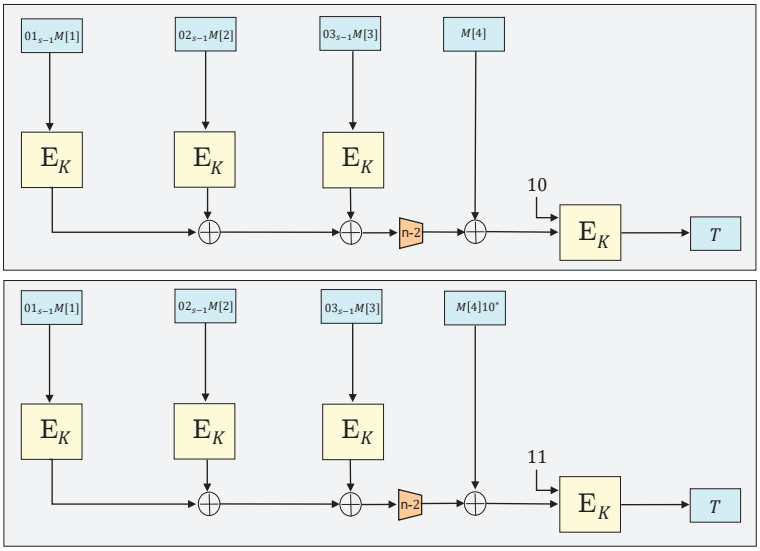

Fig. 2: **Top:** The detailed code description of $\mathsf{LedMAC2}$. Here $M[1] \parallel \ldots \parallel M[\ell] \leftarrow M$ denotes splitting M into $(n-s)$-bit blocks and stopping at the moment when it is left with a message of length less than or equal to $n-2$ bits. $M[\ell]10^*$ denotes right padded with a single 1 and as few 0 bits so that the length of string to be $(n-2)$-bit. **Middle:** The case when the final block of message M is full of $n-2$ bits, i.e., $|M[4]| = n-2$. **Bottom:** The case when the final block of message M is less than $n-2$ bits, i.e., $0 \leq |M[4]| < n-2$.

bits, and the total number of message blocks being at most σ, we have

$$\mathsf{Adv}^{\mathrm{prf}}_{\mathsf{LedMAC2}[E]}(A) \leq \mathsf{Adv}^{\mathrm{prp}}_{E}(B) + \frac{4q^2}{2^n} + \frac{q\sigma}{2^n}$$

where B is another adversary against the PRP security of blockcipher E, who runs in time $t' = t + O(q \cdot 2^{s-1})$ and makes at most $q \cdot 2^{s-1}$ queries.

Proof. Let A be an adversary against the PRF security of $\mathsf{LedMAC2}$, who runs in time t, makes at most q queries of length at most $2^{s-1}(n-s) + s - 2$ bits, and the total number of message blocks being at most σ. Without loss of the generality, we assume the adversary never repeats a prior query since otherwise

it will receive the same response. We first replace the underlying blockcipher E_K of LedMAC2 with a random permutation π. Thus,

$$\mathsf{Adv}^{\mathrm{prf}}_{\mathsf{LedMAC2}[E]}(A) \leq \mathsf{Adv}^{\mathrm{prp}}_{E}(B) + \mathsf{Adv}^{\mathrm{prf}}_{\mathsf{LedMAC2}[\pi]}(A) \ ,$$

where B is an adversary against the PRP security of blockcipher E, running in time $t' = t + O(q \cdot 2^{s-1})$ and making at most $q \cdot 2^{s-1}$ queries.

We then focus on the second term on the right side of the inequality. For message M, let $H(M)$ denote the last input to π of LedMAC2, and

$$H(M) = 10 \parallel \left(M[\ell] \oplus \left(\bigoplus_{i=1}^{\ell-1} \pi(0 \parallel i_{s-1} \parallel M[i]) \right) [1 : n-2] \right)$$

if $|M[\ell]| = n - 2$, and

$$H(M) = 11 \parallel \left(M[\ell]10^* \oplus \left(\bigoplus_{i=1}^{\ell-1} \pi(0 \parallel i_{s-1} \parallel M[i]) \right) [1 : n-2] \right)$$

if $0 \leq |M[\ell]| < n-2$. Denote by $\mathrm{Rng}(\pi)$ the set of output values of π during the computation of LedMAC2. The size of $\mathrm{Rng}(\pi)$ will increase when A is interacting with LedMAC2. During the interaction between A and its oracle LedMAC2, we define two bad events as follows:

- bad1: at the i-th query, if $H(M_i)$ collides with previous $H(M_j)$ for $1 \leq j \leq i-1$, then set bad1 to be true.
- bad2: at the i-th query, after the computation of $H(M_i)$, sample a string $\widehat{T}_i$ independently and uniformly at random from $\{0,1\}^n$. If $\widehat{T}_i$ lies in $\mathrm{Rng}(\pi)$, then set bad2 to be true.

If neither of these two bad events happens, then we let $T_i = \widehat{T}_i$ and return T_i to adversary A. Otherwise we return $T_i = \pi(H(M_i))$. On the other hand, when A is interacting with a random function, we define the same bad events as above, but regardless of bad events happen or not, we will always return $T_i = \widehat{T}_i$ to adversary A. If neither of these two bad events happens, then the output strings $T_1, \ldots, T_q$ from LedMAC2 are merely chosen independently and uniformly at random from $\{0,1\}^n$, which are exactly the same as the outputs from a random function. Denote by $\mathsf{bad} = \mathsf{bad1} \cup \mathsf{bad2}$. Then by the fundamental lemma of game-playing techniques [3], we have

$$\mathsf{Adv}^{\mathrm{prf}}_{\mathsf{LedMAC2}[\pi]}(A) \leq \Pr[\,\mathsf{bad}\,] \ .$$

We now bound the probability of bad events occur when A is interacting with a random function. Note that during this interaction, adversary A always receives random strings from its oracle, which are independent of its queries. In

other words, the adaptive queries do not help here. For the event bad1,

$$\begin{aligned}\Pr[\,\mathsf{bad1}\,] &\le \sum_{i=2}^{q}\sum_{j=1}^{i-1}\Pr[\,H(M_i)=H(M_j)\,] \\ &\le \sum_{i=2}^{q}\sum_{j=1}^{i-1}\frac{4}{2^n-2\cdot 2^s} \\ &\le \sum_{i=2}^{q}\sum_{j=1}^{i-1}\frac{4}{2^{n-1}} \le \frac{4q^2}{2^n}\ ,\end{aligned}$$

where the second inequality is due to the following Lemma 2, and the third inequality is from the assumption that $s \le n/2$. For the event bad2, since each T_i is chosen uniformly at random from $\{0,1\}^n$ and there are at most $\sum_{j=1}^{i}\ell_j$ elements in the set $\mathrm{Rng}(\pi)$ at the i-th query,

$$\begin{aligned}\Pr[\,\mathsf{bad2}\,] &\le \sum_{i=1}^{q}\frac{\sum_{j=1}^{i}\ell_i}{2^n} \\ &\le \frac{q\sigma}{2^n}\ .\end{aligned}$$

Thus by the union bound,

$$\mathsf{Adv}^{\mathrm{prf}}_{\mathsf{LedMAC2}[\pi]}(A) \le \frac{4q^2}{2^n}+\frac{q\sigma}{2^n}\ .$$

Wrapping up, the PRF advantage of A against LedMAC2 is at most

$$\mathsf{Adv}^{\mathrm{prf}}_{\mathsf{LedMAC2}[E]}(A) \le \mathsf{Adv}^{\mathrm{prp}}_{E}(B)+\frac{4q^2}{2^n}+\frac{q\sigma}{2^n}\ .$$

□

Lemma 2. *For any message $M = M[1] \| \ldots \| M[\ell]$ where $|M[i]| = n-s$ for $1 \le i \le \ell-1$ and $1 \le |M[\ell]| \le n-2$, define $H(M)$ to be $H(M) = 10\| \left(M[\ell] \oplus \left(\bigoplus_{i=1}^{\ell-1}\pi(0 \| i_{s-1} \| M[i])\right)[1:n-2]\right)$ if $|M[\ell]| = n-2$, and $H(M) = 11 \| \left(M[\ell]10^* \oplus \left(\bigoplus_{i=1}^{\ell-1}\pi(0 \| i_{s-1} \| M[i])\right)[1:n-2]\right)$ if $0 \le |M[\ell]| < n-2$, where π is a random permutation over $\{0,1\}^n$. Then for any two distinct messages $M_1 = M_1[1] \| \ldots \| M_1[\ell_1]$ and $M_2 = M_2[1] \| \ldots \| M_2[\ell_2]$,*

$$\Pr[\,H(M_1)=H(M_2)\,] \le \frac{4}{2^n-\ell_1-\ell_2+3}\ .$$

Proof. Following a similar analysis as that of Lemma 1, we can obtain the result. □

Acknowledgments

We thank the anonymous reviewers of Asiacrypt 2021 for their helpful comments.

References

1. M. Bellare, R. Guérin, and P. Rogaway. XOR MACs: New methods for message authentication using finite pseudorandom functions. In D. Coppersmith, editor, *CRYPTO'95*, volume 963 of *LNCS*, pages 15–28. Springer, Heidelberg, Aug. 1995.
2. M. Bellare, J. Kilian, and P. Rogaway. The security of the cipher block chaining message authentication code. *J. Comput. Syst. Sci.*, 61(3):362–399, 2000.
3. M. Bellare and P. Rogaway. The security of triple encryption and a framework for code-based game-playing proofs. In S. Vaudenay, editor, *EUROCRYPT 2006*, volume 4004 of *LNCS*, pages 409–426. Springer, Heidelberg, May / June 2006.
4. D. J. Bernstein. How to stretch random functions: The security of protected counter sums. *Journal of Cryptology*, 12(3):185–192, June 1999.
5. J. Black and P. Rogaway. A block-cipher mode of operation for parallelizable message authentication. In L. R. Knudsen, editor, *EUROCRYPT 2002*, volume 2332 of *LNCS*, pages 384–397. Springer, Heidelberg, Apr. / May 2002.
6. J. Black and P. Rogaway. CBC MACs for arbitrary-length messages: The three-key constructions. *Journal of Cryptology*, 18(2):111–131, Apr. 2005.
7. M. J. Dworkin. Recommendation for block cipher modes of operation: The cmac mode for authentication. 2016.
8. ISO/IEC 29192-6:2019 Information technology Lightweight cryptography Part 6: Message authentication codes (MACs). Iso/iec 29192-6:2019, 2019.
9. T. Iwata and K. Kurosawa. OMAC: One-key CBC MAC. In T. Johansson, editor, *FSE 2003*, volume 2887 of *LNCS*, pages 129–153. Springer, Heidelberg, Feb. 2003.
10. A. Luykx, B. Preneel, E. Tischhauser, and K. Yasuda. A MAC mode for lightweight block ciphers. In T. Peyrin, editor, *FSE 2016*, volume 9783 of *LNCS*, pages 43–59. Springer, Heidelberg, Mar. 2016.
11. K. Yasuda. PMAC with parity: Minimizing the query-length influence. In O. Dunkelman, editor, *CT-RSA 2012*, volume 7178 of *LNCS*, pages 203–214. Springer, Heidelberg, Feb. / Mar. 2012.