A MAC Mode for Lightweight Block Ciphers

Atul Luykx[1,2(✉)], Bart Preneel[1,2], Elmar Tischhauser[3], and Kan Yasuda[4]

[1] Department of Electrical Engineering, ESAT/COSIC, KU Leuven, Leuven, Belgium
{atul.luykx,bart.preneel}@esat.kuleuven.be
[2] iMinds, Ghent, Belgium
[3] Department of Applied Mathematics and Computer Science, Technical University of Denmark, Lyngby, Denmark
ewti@dtu.dk
[4] NTT Secure Platform Laboratories, NTT Corporation, Tokyo, Japan
yasuda.kan@lab.ntt.co.jp

Abstract. Lightweight cryptography strives to protect communication in constrained environments without sacrificing security. However, security often conflicts with efficiency, shown by the fact that many new lightweight block cipher designs have block sizes as low as 64 or 32 bits. Such low block sizes lead to impractical limits on how much data a mode of operation can process per key. MAC (message authentication code) modes of operation frequently have bounds which degrade with both the number of messages queried and the message length. We present a MAC mode of operation, LightMAC, where the message length has *no effect* on the security bound, allowing an order of magnitude more data to be processed per key. Furthermore, LightMAC is incredibly simple, has almost no overhead over the block cipher, and is parallelizable. As a result, LightMAC not only offers compact authentication for resource-constrained platforms, but also allows high-performance parallel implementations. We highlight this in a comprehensive implementation study, instantiating LightMAC with PRESENT and the AES. Moreover, LightMAC allows flexible trade-offs between rate and maximum message length. Unlike PMAC and its many derivatives, LightMAC is not covered by patents. Altogether, this makes it a promising authentication primitive for a wide range of platforms and use cases.

Keywords: Lightweight · MAC · LightMAC · Message length · Birthday bound · Integrity · Verification

1 Introduction

With the rise of the Internet of Things, connected devices are being placed everywhere, resulting in a wide variety of efficiency, robustness, and feature requirements for communication. Securing the communcation remains important, and as a result, many block ciphers have been created to work efficiently in constrained environments. These block ciphers offer a range of block and key sizes, from 128 to 32 bits; see Table 1 for a sample.

© International Association for Cryptologic Research 2016
T. Peyrin (Ed.): FSE 2016, LNCS 9783, pp. 43–59, 2016.
DOI: 10.1007/978-3-662-52993-5_3

The key size is often chosen carefully to ensure a sufficiently high security level, resulting in the block size becoming the dominant factor in determining security. As is well known, reducing block size can increase the chance of an inner state collision when block ciphers are used in so-called *modes of operation*: constructions which repeatedly apply a block cipher to achieve functionality beyond what a block cipher offers.

Consider MAC (Message Authentication Code) modes of operation, which aim to provide data authenticity for long messages. Common MAC modes, such as CBC-MAC [5], OMAC [24], and PMAC [10] have security bounds which degrade relative to both the number of messages tagged, q, and the length of the messages measured in blocks, ℓ; see Table 2 for a list of modes with their dependence on ℓ. For many modes, an adversary which is able to tag q messages of length ℓ blocks will have a success probability of roughly

$$\frac{q^2\ell}{2^n}, \tag{1}$$

where n is the block size of the underlying block cipher. With a 32 bit block size and a guarantee that adversaries do not forge with probability more than one in a million, one gets a restriction of the form

$$\frac{q^2\ell}{2^{32}} \leq \frac{1}{2^{20}} \quad \text{or} \quad q^2\ell \leq 2^{12}, \tag{2}$$

meaning 64 one-block messages can be tagged under the same key. But what if the messages are longer than one block? With conventional MACs only 32 four-block messages can be tagged, corresponding to $32 \cdot 2^2 \cdot 32 = 2^{12}$ bits, or 512 Bytes of data per key. If the messages are sixteen blocks long, only 16 messages can be tagged, which is $16 \cdot 2^4 \cdot 32 = 2^{13}$ bits, or 1 KiB of data per key. Figure 1 displays how much data the various modes from Table 2 can process per key, when the threshold success probability is set to $1/2^{20}$.

1.1 Contributions

We present a MAC mode, LightMAC, which enables one to tag much longer messages than typically possible. LightMAC is depicted in Fig. 2 and Algorithm 1.

The security upper bound for LightMAC is

$$(1+\epsilon)\cdot\frac{q^2}{2^n}, \quad \text{where}\, \epsilon \in O\left(\frac{1}{2^{n/2}-1}\right), \tag{3}$$

which is independent of the message length (see Sect. 4). In other words, with a 32 bit block size, and setting the message-length parameter s to 16, roughly 64 messages can be tagged with length up to 2^{15} blocks. Note that keys are used most efficiently when the messages are as long as possible: up to $64\cdot 2^{15}\cdot 32 = 2^{26}$ bits, or *8 MiB* of data can be tagged per key. LightMAC uses two independent keys, but even after normalizing by the number of keys, the amount of data processed per key is still 4 MiB, a significant improvement over 1 KiB.

Table 1. Supported block sizes are often small, and can be as low as 32 bits.

Block size (bits)	32	48	64	80	96	128	256
AES [15]						×	
CLEFIA [38]						×	
DESLX [27]			×				
Fantomas [19]						×	
HIGHT [23]			×				
ITUbee [26]				×			
KLEIN [18]			×				
KATAN [13]	×	×	×				
LBlock [42]			×				
LED [21]			×				
LEA [22]						×	
mCrypton [28]			×				
Mysterion [25]						×	×
Noekeon [14]						×	
Piccolo [37]			×				
PRESENT [11]			×				
PRIDE [1]			×				
PRINCE [12]			×				
RC5 [36]	×		×			×	
Rectangle [48]			×				
RoadRunneR [2]			×				
Robin [19]						×	
SEA [39]					×		
SIMECK [43]	×	×	×				
Simon [3]	×	×	×		×	×	
Speck [3]	×	×	×		×	×	
TWINE [40]			×				
XTEA [33]			×				
Zorro [17]						×	

Figure 1 compares LightMAC to the other published modes from Table 2. The figure shows that LightMAC starts with a factor 2^4 improvement over many of the modes, which grows to roughly 2^{10} as the number of queries increases. Modes such as PMAC with Parity and PMACX were designed to handle long message lengths and offer competitive bounds, at the cost of increased design complexity. LightMAC's advantage over these modes is its simplicity and low overhead.

Table 2. The table below contains the coefficients of the powers of ℓ contained in the security bounds for adversaries making q queries of length ℓ, with block size n bits. References are to papers proving the bounds. In the bound for EMAC, the function $d'(\ell)$ has been replaced by ℓ.

Mode	1	ℓ	ℓ^2	ℓ^3	ℓ^4
3kf9 [47]	$\frac{4q}{2^n} + \frac{4q^3}{2^{2n}}$	$\frac{4q}{2^n} + \frac{4q^3}{2^{2n}}$	$\frac{2q^3}{2^{2n}}$	$\frac{4q^3}{2^{2n}}$	
CBC-MAC [6]		$\frac{12q^2}{2^n}$			$\frac{64q^2}{2^{2n}}$
EMAC [6]		$\frac{q^2}{2^n}$			$\frac{32q^2}{2^{2n}}$
OMAC [31]		$\frac{5q^2}{2^n}$			$\frac{8q^2}{2^{2n}}$
PMAC [32]	$\frac{-3.5q^2}{2^n}$	$\frac{5q^2}{2^n}$			
PMAC_Plus [45]		$\frac{3q}{2^n}$		$\frac{27q^3}{2^{2n}}$	
$\underset{(m=14,l=12)}{\text{PMACX}}$ [49]	$\frac{72+1.5q^2}{2^n} + \frac{576q^2}{2^{2n}}$	$\frac{576q^2}{2^{2n}}$	$\frac{144q^2}{2^{2n}}$		
PMAC with Parity [46]	$\frac{q^2}{2^n}$		$\frac{q^2}{2^{2n}}$		
Sum of CBCs [44]					$\frac{12q^3}{2^{2n}}$

Like PMAC [10], LightMAC allows block cipher calls to be made in parallel, but unlike PMAC, LightMAC is based on Bernstein's *protected counter sum* [8], and hence should not suffer from patent issues.

A disadvantage of LightMAC is that its rate is low. In order to tag messages of length up to $2^{n/2-1}$ blocks, $n/2$ bits of the block must be sacrificed for a counter, hence two block cipher calls must be called per block of data. However, the rate can be improved: if the maximum message length that will be communicated is known to be less than $2^s(n-s)$ bits, then the rate can be set to $(n-s)/n$ blocks per block cipher call. For example, using a 32 bit block cipher, if the message lengths are less than 2^9 blocks, then the rate can be set to 2/3 blocks per call. Therefore, unlike other modes, LightMAC can be optimized according to the application: the shorter the messages, the more efficient LightMAC is, while allowing the same number of message to be queried. Section 5 presents implementation results for LightMAC instantiated with the AES [15] and PRESENT [11], and discusses LightMAC's efficiency in more detail.

1.2 Related Work

In 1995, Bellare et al. [4] described the *XOR MACs*, which XORed together finite-input-length pseudorandom functions (PRF) to create stateful and randomized MACs. In 1999, Bernstein [8] introduced the protected counter sum, which composes an XOR MAC with an independent PRF call to create a stateless, deterministic MAC. In 2012, Yasuda [46] explained the basic idea for LightMAC in his paper's introduction, which can be viewed as an adaptation of Bernstein's protected counter sum using block ciphers.

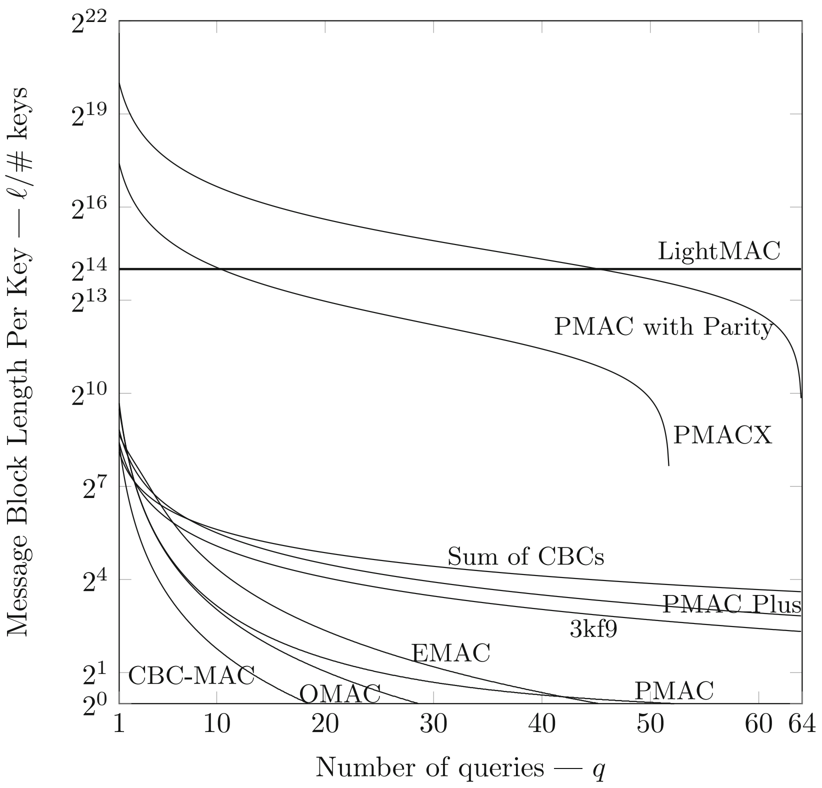

Fig. 1. A plot of message block lengths per key versus the number of queries that can be made in order to achieve the threshold success probability of 2^{-20}. In other words, if (x, y) is a point on the graph, then $x \cdot y$ represents the number of blocks that can be processed per key. The blocksize is set to 32 bits.

Another MAC algorithm designed for lightweight use is Chaskey [30]. The Chaskey paper includes a block cipher and a permutation mode, but both have bounds which deteriorate quadratically with respect to message length.

In certain cases the bounds in Table 2 can be improved. For example, for $\ell \leq 2^{n/8}$ and $q \geq \ell^2$, EMAC's bound becomes $\frac{16q^2}{2^n} + \frac{128q^2\ell^8}{2^{2n}}$ as shown by Pietrzak [34]. For the sum of CBCs, Yasuda [44] also showed that if $\ell \leq 2^{2n/5}$, the advantage becomes $\frac{40\ell^3q^3}{2^{2n}}$.

2 Preliminaries

The set $\{0,1\}^n$ represents all bit-strings of length n; the set $\{0,1\}^{\leq n}$ is all bit-strings of length less than or equal to n. For two bit-strings A and B, we write $A \| B$ and AB interchangeably for the concatenation of A and B. Let r be an integer, then $M[1]M[2]\cdots M[\ell] \xleftarrow{r} M$ represents splitting M into r-bit blocks with the length of the last block, $M[\ell]$, being anywhere from zero to $r-1$ bits.

A block cipher is a function $E : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ where $E(K,\cdot)$ defines a permutation for all $K \in \{0,1\}^k$. The integer n is the *block length* of E and we write $E_K(X)$ to mean $E(K,X)$. Given a block length n, concatenation of 10^* to a string means appending a one followed by the minimum number of zeros to make the total string length a multiple of n bits.

The symbol 0^n represents the n-bit string consisting of only zeros. Given a string A of length n, and an integer $t \le n$, then $\lfloor A \rfloor_t$ denotes the t least significant bits of A.

For an integer $1 \le i \le 2^s$, i_s represents some s-bit constant with the property that if $1 \le i < j \le 2^s$ then $i_s \ne j_s$. For example, i_s could be an s-bit representation of the integer i, or the ith s-bit Gray code.

3 LightMAC

Let $E : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ be a block cipher. Let s and t be integers not greater than $n/2$ and n, respectively, and fix some representation for i_s (see Sect. 2). LightMAC accepts two independent and uniformly generated keys K_1 and K_2 from $\{0,1\}^k$, and a message M of length at most $2^s(n-s)$ bits. LightMAC produces an output of length t bits. Figure 2 and Algorithm 1 depict how the output is produced.

LightMAC can be used as either a pseudorandom function (PRF) or a MAC (see Sects. 4.2 and 4.3 for definitions). When used as a PRF, LightMAC is fully described by Algorithm 1. When used as a MAC, tags are generated using Algorithm 1, and verification of a message-tag pair (M,T) is done by comparing LightMAC (M) with T: if the two are equal, verification succeeds, otherwise not.

The parameters of LightMAC are the integers s and t, the representation of i_s, and the block cipher E, which implicity fixes k and n. The parameters must be agreed upon before a session starts, and remain constant during.

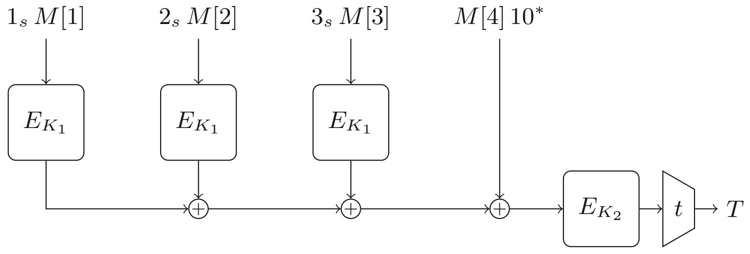

Fig. 2. LightMAC evaluated on a message $M[1]\,M[2]\,M[3]\,M[4] \xleftarrow{n-s} M$. The rounded squares represent block cipher calls and the trapezium is truncation to t bits.

Algorithm 1. LightMAC $_{K_1,K_2}(M)$

```
   Input: $K_1, K_2 \in \{0,1\}^k$, $M \in \{0,1\}^{\leq 2^s(n-s)}$
   Output: $T \in \{0,1\}^t$
1  $V \leftarrow 0^n \in \{0,1\}^n$
2  $M[1]M[2]\cdots M[\ell] \xleftarrow{n-s} M$
3  for $i = 1$ to $\ell - 1$ do
4      $V \leftarrow V \oplus E_{K_1}\big(i_s\, M[i]\big)$
5  end
6  $V \leftarrow V \oplus (M[\ell]\, 10^*)$
7  $T \leftarrow \lfloor E_{K_2}(V) \rfloor_t$
8  return $T$
```

4 Security

Although Bellare, Guérin, and Rogaway [4] describe how to instantiate an XOR MAC using the Data Encryption Standard, they only provide proofs for pseudorandom functions, not pseudorandom permutations. Hence, even though the XOR MACs were proven to have bounds with no message length dependence, subsequent application of the PRP-PRF switching lemma would establish quadratic message length dependence. A similar explanation applies to the protected counter sum's security bound. Therefore a direct security proof is necessary for LightMAC.

The XOR MACs and protected counter sum did not exhibit any message length dependence because the XOR of independent, uniformly distributed random variables is still uniformly distributed. In this section we use the fact that roughly the same applies to the XOR of distinct block cipher outputs to achieve message length independence for LightMAC.

4.1 Block Cipher Security

The security of LightMAC is reduced to that of its underlying block cipher, that is, if an attack is found against LightMAC, then the attack can be reduced to an attack against the block cipher. The quality of the reduction is measured by the security bounds computed in Theorems 1 and 2.

The statements of the theorems include terms describing the quality of the underlying block cipher, which is measured as follows.

Definition 1. *Let $E : \mathsf{K} \times \mathsf{X} \to \mathsf{X}$ be a block cipher, and let π be a uniformly distributed random permutation over the set of permutations on X. Then the PRP-advantage against E of adversaries $\mathcal{A}$ making q queries and running in time τ is*

$$\mathsf{PRP}(q,\tau) := \sup_{A\in\mathcal{A}} \left|\mathbf{P}\left[A^{E_K} = 1\right] - \mathbf{P}\left[A^{\pi} = 1\right]\right|, \tag{4}$$

where $A^O = 1$ is the event that A outputs 1 when given access to oracle O, and K is uniformly distributed over K.

4.2 LightMAC as a PRF

A PRF $\Phi : \mathsf{K} \times \mathsf{M} \rightarrow \mathsf{T}$ is a construction which should be computationally indistinguishable from a uniformly distributed random function (URF), that is, a uniformly distributed random variable over the set of all functions from M to T. The quality of the PRF is measured via the PRF-advantage of adversaries.

Definition 2. *The PRF-advantage of an adversary A in distinguishing the PRF $\Phi : \mathsf{K} \times \mathsf{M} \rightarrow \mathsf{T}$ from the URF $\$: \mathsf{M} \rightarrow \mathsf{T}$ is*

$$\left| \mathbf{P}\left[A^{\Phi_K} = 1\right] - \mathbf{P}\left[A^{\$} = 1\right] \right|, \tag{5}$$

where $A^O = 1$ is the event that A outputs 1 when given access to oracle O, and K is uniformly distributed over K.

Theorem 1. *The PRF-advantage against LightMAC of any adversary running in time τ and making at most q queries of length at most $2^s(n-s)$ bits is bounded above by*

$$\left(1 + \frac{1}{2^{n/2}-1} + \frac{1}{2(2^{n/2}-1)^2}\right) \cdot \frac{q^2}{2^n} + \mathsf{PRP}(q \cdot (2^s - 1), \tau_1) + \mathsf{PRP}(q, \tau_2), \tag{6}$$

where n is the block size in bits, $\tau_1 \in \tau + O(q \cdot (2^s - 1))$, and $\tau_2 \in \tau + O(q)$.

Proof. Let A be a PRF-adversary against LightMAC running in time τ and making at most q queries of length at most $2^s(n-s)$ bits. Construct the PRP adversary B_1 against E_{K_1} as follows: B_1 simulates E_{K_2} by uniformly randomly choosing key K_2, runs A, and responds to A's queries using a combination of its own oracle and the simulated E_{K_2}; B_1 forwards A's response as its own. Construct the PRP adversary B_2 against E_{K_2} similarly. Then A's PRF-advantage against LightMAC is bounded above by

$$\alpha + \mathsf{PRP}(q \cdot (2^s - 1), \tau_1) + \mathsf{PRP}(q, \tau_2), \tag{7}$$

where α is A's PRF-advantage against LightMAC with its E_{K_1} and E_{K_2} calls replaced with π_1 and π_2 calls, respectively, where π_1 and π_2 are independent, uniformly distributed random permutations.

We replace π_2 with a uniformly distributed random function ϕ using the PRP-PRF switching lemma, at a cost of $q^2/2^{n+1}$ in advantage. The PRF we are left with is

$$\Phi(M) = \phi\left(M[\ell]10^* \oplus \bigoplus_{i=1}^{\ell-1} \pi_1(i_s M[i])\right), \tag{8}$$

which is LightMAC instantiated with π_1 and ϕ, and

$$\alpha \leq \alpha' + \frac{q^2}{2^{n+1}}, \tag{9}$$

where α' is A's PRF-advantage against Φ.

Let F denote the function contained in the call to ϕ in Eq. 8. Then, as long as F's outputs are distinct, each input to ϕ is unique, meaning Φ will be indistinguishable from \$. In other words,

$$\alpha' \leq \sum_{i<j} \mathbf{P}\left[F(M_i) = F(M_j)\right] \leq \frac{q^2}{2} \max_{M_i \neq M_j} \mathbf{P}\left[F(M_i) = F(M_j)\right], \tag{10}$$

where M_i for $i = 1, \ldots, q$ are the messages queried by A. The maximum on the right hand side is computed in Sect. 4.4, resulting in the bound

$$\alpha' \leq \frac{q^2}{2} \cdot \frac{1}{2^n - 2^{s+1} + 1}. \tag{11}$$

Therefore, using the fact that $s \leq n/2$, we have

$$\alpha \leq \frac{q^2}{2^{n+1}} + \frac{q^2}{2} \cdot \frac{1}{2^n - 2^{s+1} + 1} \tag{12}$$

$$\leq \frac{q^2}{2^n}\left(1 + \frac{1}{2^{n/2} - 1} + \frac{1}{2(2^{n/2} - 1)^2}\right), \tag{13}$$

giving us our desired bound. □

4.3 LightMAC as a MAC

A MAC consists of a tagging and a verification algorithm. The tagging algorithm accepts messages from some message set M and produces tags from a tag set T. The verification algorithm receives message-tag pairs (M, T) as input, and outputs 1 if the pair (M, T) is valid, and 0 otherwise. The insecurity of a MAC is measured as follows.

Definition 3. *Let A be an adversary with access to a MAC. The advantage of A in breaking the MAC is the probability that A is able to produce a message-tag pair (M, T) for which the verification algorithm outputs 1, where M has not been previously queried to the tagging algorithm.*

Theorem 2. *The MAC-advantage against LightMAC of any adversary running in time τ and making at most q tagging queries and v verification queries of length at most $2^s(n-s)$ bits, is bounded above by*

$$\left(1 + \frac{2}{2^{n/2} - 1} + \frac{1}{(2^{n/2} - 1)^2}\right) \cdot \left(\frac{q^2}{2^n} + \frac{v}{2^t}\right) + \\ \mathsf{PRP}(q \cdot (2^s - 1), \tau_1) + \mathsf{PRP}(q, \tau_2) + \mathsf{PRP}(v2^s, \tau_3), \tag{14}$$

where n is the block size in bits, $\tau_1 \in \tau + O(q \cdot (2^s - 1))$, $\tau_2 \in \tau + O(q)$, and $\tau_3 \in \tau + O(v2^s)$.

Proof. We apply the same reduction as in the proof of Theorem 1 to replace LightMAC's E_{K_1} and E_{K_2} calls with π_1 and π_2 calls, respectively. As a MAC, LightMAC follows the hash-then-encrypt paradigm as described by Dodis and Pietrzak [16], with the function F from Sect. 4.4 as the "hash" part, hence applying Proposition 1 from their paper we get an upper bound of

$$\left(1 + \frac{2}{2^{n/2}-1} + \frac{1}{(2^{n/2}-1)^2}\right) \cdot \left(\frac{q^2}{2^n} + \frac{v}{2^t}\right). \tag{15}$$

□

4.4 Collision Probability of *F*

Proposition 1. *Let* $m = 2^s(n-s)$. *Let* $M[1]M[2]\cdots M[\ell] \xleftarrow{n-s} M$ *for* $M \in \{0,1\}^{\leq m}$, *and define* F *to be*

$$F(M) = M[\ell]10^* \oplus \bigoplus_{i=1}^{\ell-1} \pi(i_s\, M[i])\,, \tag{16}$$

where π *is a uniformly distributed random permutation over* $\{0,1\}^n$, *then the probability that two distinct messages* $M_1, M_2 \in \{0,1\}^{\leq m}$ *collide is*

$$\mathbf{P}\left[F(M_1) = F(M_2)\right] \leq \frac{1}{2^n - \ell_1 - \ell_2 + 1}\,, \tag{17}$$

where ℓ_i *is the length of* M_i *in* $(n-s)$*-bit blocks rounded up.*

Proof. The equation $F(M_1) = F(M_2)$ can be rewritten as

$$\bigoplus_{i=1}^{\ell_1} \pi(i_s M_1[i]) \oplus \bigoplus_{i=1}^{\ell_2} \pi(i_s M_2[i]) = M_1[\ell_1]10^* \oplus M_2[\ell_2]10^*. \tag{18}$$

Since $M_1 \neq M_2$ there are two cases:

1. $\ell_1 = \ell_2$, $M_1[\ell_1]10^* \neq M_2[\ell_2]10^*$, and $M_1[i] = M_2[i]$ for all i, or
2. either $\ell_1 \neq \ell_2$ or there exists an i such that $M_1[i] \neq M_2[i]$.

In the first case there is no collision, hence we focus on the second case. Without loss of generality we can assume that $M_1[i] \neq M_2[i]$ for all i, and we can simplify the problem to calculating the probability that

$$\bigoplus_{i=1}^{\ell} \pi(x_i) = c, \tag{19}$$

where $\ell = \ell_1 + \ell_2$, $c = M_1[\ell_1]10^* \oplus M_2[\ell_2]10^*$, and $x_i \neq x_j$ for $i \neq j$.

Let $N = 2^n$, then $\mathbf{P}\left[\bigoplus_{i=1}^{\ell} \pi(x_i) = c\right]$ equals

$$\frac{1}{N!}\left|\left\{ y_1, \ldots, y_N \;\middle|\; \bigoplus_{i=1}^{\ell} y_i = c \text{ and } y_i \neq y_j \text{ for } i \neq j \right\}\right|. \tag{20}$$

By Lemma 1 we have that the probability is bounded above by $N!/(N-\ell+1)$, giving us our desired result. □

Lemma 1. *Let $c \in \{0,1\}^n$ and let $N = 2^n$. The number of sequences $(y_1, y_2, \ldots, y_N) \in (\{0,1\}^n)^N$ with $y_i \neq y_j$ for $i \neq j$ such that*

$$\bigoplus_{i=1}^{\ell} y_i = c, \tag{21}$$

is not greater than $N!/(N-\ell+1)$.

Proof. We start by fixing y_1, for which there are N possibilities. Since y_2 cannot equal y_1, there are $N-1$ possibilities for y_2. Continuing this way, we have that there are $N-i$ possibilities for y_{i+1}, with $i \leq \ell - 2$. For y_ℓ there is at most one possibility, namely $c \oplus y_1 \oplus y_2 \oplus \cdots y_{\ell-1}$. All y_j for $j > \ell$ must be distinct from all preceding y_i, hence in total there are at most

$$N \cdot (N-1) \cdot \cdots \cdot (N-\ell+2) \cdot (N-\ell)! = \frac{N!}{N-\ell+1} \tag{22}$$

possible sequences. □

5 Implementation

In this section, we discuss the implementation characteristics of LightMAC and compare it to the serial two-key CBC-MAC with last block encryption, EMAC [6], and to PMAC with Parity (PMAC/P) [46], which provides a parallelizable rate 2/3 construction and can be considered its main competitor.

5.1 Implementation Characteristics of LightMAC

LightMAC is a mode with very low overhead: besides the block cipher calls, it only requires an s-bit counter generator and one additional n-bit state for summing the block cipher outputs.

This means that the code size (for embedded software or microcontrollers) and area requirements (for hardware implementations) of LightMAC can be estimated as roughly equivalent to CBC-MAC with encryption of the last block by a second key. Compared to PMAC with Parity, LightMAC uses only two keys instead of four. In comparison to all PMAC variants, the absence of finite field doubling further improves its implementation characteristics on embedded platforms or hardware.

In terms of throughput, a compact serial implementation of LightMAC will give a performance of about $n/(n-s)$ block cipher call equivalents per message block of $n-s$ bits, which means that the serial performance of LightMAC on a given platform can readily be evaluated based on the performance of the best available implementation of the chosen underlying block cipher. Except for very short messages, the overhead imposed by the final block cipher call is negligible.

Like PMAC and its derivatives, LightMAC has the advantage that the individual block cipher calls can be parallelized. While this is typically less important on lightweight platforms, where compactness and power/energy consumption are the prime concerns, this property enables high-performance implementations for the server side: since exactly the same lightweight algorithms used on small devices will also have to be used by the servers communicating with them, they should ideally also have good implementation characteristics in high-performance software environments. The importance of this was for instance pointed out in [29]. Many lightweight algorithms and modes of operation are inherently serial in nature and therefore inefficient in software. Our implementation study therefore focuses on this scenario.

5.2 The Setting

We explore the high-performance parallel software implementation possibilities for LightMAC, with the following choices regarding platform and instantiation parameters:

Underlying Block Ciphers. We use the block ciphers PRESENT [11] and AES [15] for our implementations. PRESENT is a lightweight 64-bit block cipher that was recently standardised by ISO, and AES serves as a baseline.

Choice of *s* and *t*. We always use full tag lengths $t = n$, meaning 64-bit tags for PRESENT and 128-bit tags for AES. We furthermore instantiate LightMAC with the following values of s:

1. $s = n/2$ for the maximum supported message length (and correspondingly lowest rate 1/2);
2. $s = n/3$, rounded to the nearest multiple of 8, for a mode with rate 2/3;
3. $s = 8$, for a short maximum message length with the highest rate $(1 - 8/n)$.

Altogether, these parameter choices illustrate a wide spectrum of use cases.

Platform. We implement LightMAC on Intel's recent Skylake microarchitecture, using the 256-bit AVX2 instruction set. PRESENT was implemented in a bitsliced fashion processing 8 blocks in parallel. Other implementation strategies are known to yield a significantly lower performance, see [7] for a comprehensive study. For the AES, the AES-NI instruction set [20] was used. The key scheduling was precomputed for both ciphers. Since byte-aligned s-bit addition is inexpensive on this platform, the counters i_s are implemented as the s-bit representation of the integer i.

Message Lengths. We provide performance data for all message lengths of $\ell = 2^b$ bytes, with $7 \leq b \leq 13$, wherever $8\ell \leq 2^s(n-s)$.

5.3 Performance Measurements

All measurements were taken on a single core of an Intel Core i7-6700 CPU at 3.4 GHz with Turbo Boost disabled, and averaged over 200000 repetitions. The performance of the block ciphers AES and PRESENT, both in serial and parallel implementations, is provided as a reference point in Table 3. Our findings on the performance of LightMAC and related MACs are summarised in Table 4. All performance numbers are given in cycles per byte (cpb).

Table 3. Baseline performance of ciphers PRESENT and AES on Skylake (AVX2, AES-NI).

Block cipher	Encryption [cycles/byte]	Key schedule [cycles]
PRESENT (table-based)	57.83	353
PRESENT (8 blocks bitsliced)	11.23	790
AES (AES-NI, serial)	2.57	116
AES (AES-NI, pipelined)	0.63	116

Table 4. Software performance of LightMAC, EMAC and PMAC with Parity (PMAC/P), instantiated with PRESENT and AES on the Intel Skylake platform (AVX2, AES-NI). All numbers are given in cycles per byte (cpb). Data is provided for message lengths smaller than $2^s(n-s)$ bits.

			Message length (bytes)						
Algorithm	s	Rate	128	256	512	1024	2048	4096	8192
EMAC-PRESENT	–	1	63.02	61.21	60.28	59.80	59.57	59.41	59.32
PMAC/P-PRESENT	–	2/3	39.62	32.44	28.82	27.07	26.48	26.14	26.00
LightMAC-PRESENT	32	1/2	25.50	23.67	22.75	22.32	22.08	21.97	21.92
LightMAC-PRESENT	24	2/3	25.70	21.21	20.17	19.03	18.09	17.80	17.80
LightMAC-PRESENT	8	7/8	20.31	18.34	14.65	13.48	–	–	–
EMAC-AES	–	1	3.42	3.19	3.03	2.91	2.74	2.68	2.67
PMAC/P-AES	–	2/3	1.53	1.48	1.33	1.24	1.17	1.15	1.14
LightMAC-AES	64	1/2	1.33	1.29	1.27	1.26	1.26	1.26	1.25
LightMAC-AES	40	2/3	1.37	1.31	1.12	1.04	0.95	0.95	0.92
LightMAC-AES	8	15/16	1.38	1.00	0.82	0.80	0.72	–	–

Discussion. One can observe that with both PRESENT and the AES as the underlying block ciphers, LightMAC provides a performance of about the inverse of its rate times the baseline block cipher speed. This confirms that LightMAC imposes very low overhead in addition to the block cipher invocations.

In contrast to the serial EMAC, LightMAC provides significantly greater performance despite featuring a smaller rate. This demonstrates the advantage of parallelisability over a sequential algorithm.

Comparing the LightMAC instantiations with rate 2/3 to PMAC with Parity (PMAC/P), we note that the use of the same key throughout the message processing (as opposed to three different keys in PMAC/P) significantly improves the performance for the PRESENT-based implementation: LightMAC is consistently around 50 % faster. This is largely due to the fact that the parts of each subkey of PMAC/P's three bitsliced keys have to be interleaved in an appropriate way. The effect is less pronounced for the AES where no conversion to bitsliced format is needed, and due to the AES-NI instructions which freely accept both registers and memory locations for the subkeys. Still, LightMAC is about 20 % faster, while additionally providing a flexible range of trade-offs between rate and maximum message length.

6 Conclusions

We proposed LightMAC, a new MAC mode of operation specifically suited to lightweight applications. Its security bound was shown in Sect. 4 to not depend on the message length, allowing an order of magnitude more data to be processed per key.

Featuring a simple design with very low overhead over the block cipher, it not only offers compact authentication for resource-constrained platforms, but also allows high-performance parallel implementations, as demonstrated by the implementation study of LightMAC instantiated with PRESENT and the AES in Sect. 5. Furthermore, the implementation results show how the s-parameter translates directly to a trade-off between rate and maximum message length.

Unlike PMAC and its many derivatives, LightMAC is not covered by patents. Altogether, this makes it a promising authentication solution for a wide range of platforms and use cases.

Acknowledgments. We would like to thank the various anonymous reviewers for providing useful comments. This work was supported in part by the Research Council KU Leuven: GOA TENSE (GOA/11/007). In addition, this work was supported by the Research Fund KU Leuven, OT/13/071. Atul Luykx is supported by a Ph.D. Fellowship from the Institute for the Promotion of Innovation through Science and Technology in Flanders (IWT-Vlaanderen).

References

1. Albrecht, M.R., Driessen, B., Kavun, E.B., Leander, G., Paar, C., Yalçın, T.: Block ciphers – focus on the linear layer (feat. PRIDE). In: Garay, J.A., Gennaro, R. (eds.) CRYPTO 2014, Part I. LNCS, vol. 8616, pp. 57–76. Springer, Heidelberg (2014). doi:10.1007/978-3-662-44371-2_4

2. Baysal, A., Sahin, S.: RoadRunneR: a small and fast bitslice block cipher for low cost 8-bit processors. In: Güneysu, T., Leander, G., Moradi, A. (eds.) LightSec 2015. LNCS, vol. 9542, pp. 58–76. Springer, Heidelberg (2016). doi:10.1007/978-3-319-29078-2_4
3. Beaulieu, R., Shors, D., Smith, J., Treatman-Clark, S., Weeks, B., Wingers, L.: The SIMON and SPECK families of lightweight block ciphers. Cryptology ePrint Archive, Report 2013/404 (2013). http://eprint.iacr.org/
4. Bellare, M., Guérin, R., Rogaway, P.: XOR MACs: new methods for message authentication using finite pseudorandom functions. In: Coppersmith, D. (ed.) CRYPTO 1995. LNCS, vol. 963, pp. 15–28. Springer, Heidelberg (1995). doi:10.1007/3-540-44750-4_2
5. Bellare, M., Kilian, J., Rogaway, P.: The security of cipher block chaining. In: Desmedt, Y.G. (ed.) CRYPTO 1994. LNCS, vol. 839, pp. 341–358. Springer, Heidelberg (1994). doi:10.1007/3-540-48658-5_32
6. Bellare, M., Pietrzak, K., Rogaway, P.: Improved security analyses for CBC MACs. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 527–545. Springer, Heidelberg (2005). doi:10.1007/11535218_32
7. Benadjila, R., Guo, J., Lomné, V., Peyrin, T.: Implementing lightweight block ciphers on x86 architectures. In: Lange, T., Lauter, K., Lisoněk, P. (eds.) SAC 2013. LNCS, vol. 8282, pp. 324–352. Springer, Heidelberg (2014). doi:10.1007/978-3-662-43414-7_17
8. Bernstein, D.J.: How to stretch random functions: the security of protected counter sums. J. Cryptology **12**(3), 185–192 (1999). doi:10.1007/s001459900051
9. Biryukov, A. (ed.): FSE 2007. LNCS, vol. 4593. Springer, Heidelberg (2007)
10. Black, J.A., Rogaway, P.: A block-cipher mode of operation for parallelizable message authentication. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 384–397. Springer, Heidelberg (2002). doi:10.1007/3-540-46035-7_25
11. Bogdanov, A.A., Knudsen, L.R., Leander, G., Paar, C., Poschmann, A., Robshaw, M., Seurin, Y., Vikkelsoe, C.: PRESENT: an ultra-lightweight block cipher. In: Paillier, P., Verbauwhede, I. (eds.) CHES 2007. LNCS, vol. 4727, pp. 450–466. Springer, Heidelberg (2007). doi:10.1007/978-3-540-74735-2_31
12. Borghoff, J., Canteaut, A., Güneysu, T., Kavun, E.B., Knezevic, M., Knudsen, L.R., Leander, G., Nikov, V., Paar, C., Rechberger, C., Rombouts, P., Thomsen, S.S., Yalçin, T.: PRINCE - a low-latency block cipher for pervasive computing applications - extended abstract. In: Wang, X., Sako, K. (eds.) [41], pp. 208–225. http://dx.org/10.1007/978-3-642-34961-4_14
13. De Cannière, C., Dunkelman, O., Knežević, M.: KATAN and KTANTAN — a family of small and efficient hardware-oriented block ciphers. In: Clavier, C., Gaj, K. (eds.) CHES 2009. LNCS, vol. 5747, pp. 272–288. Springer, Heidelberg (2009). doi:10.1007/978-3-642-04138-9_20
14. Daemen, J., Peeters, M., Van Assche, G., Rijmen, V.: Nessie proposal: Noekeon. In: First Open Nessie Workshop (2000)
15. Daemen, J., Rijmen, V.: AES proposal: Rijndael. In: First Advanced Encryption Standard (AES) Conference (1998)
16. Dodis, Y., Pietrzak, K.: Improving the security of MACs via randomized message preprocessing. In: Biryukov, A. (ed.) [9], pp. 414–433. http://dx.org/10.1007/978-3-540-74619-5_26
17. Gérard, B., Grosso, V., Naya-Plasencia, M., Standaert, F.-X.: Block ciphers that are easier to mask: how far can we go? In: Bertoni, G., Coron, J.-S. (eds.) CHES 2013. LNCS, vol. 8086, pp. 383–399. Springer, Heidelberg (2013). doi:10.1007/978-3-642-40349-1_22

18. Gong, Z., Nikova, S., Law, Y.W.: KLEIN: a new family of lightweight block ciphers. In: Juels, A., Paar, C. (eds.) RFIDSec 2011. LNCS, vol. 7055, pp. 1–18. Springer, Heidelberg (2012). doi:10.1007/978-3-642-25286-0_1
19. Grosso, V., Leurent, G., Standaert, F.-X., Varıcı, K.: LS-designs: bitslice encryption for efficient masked software implementations. In: Cid, C., Rechberger, C. (eds.) FSE 2014. LNCS, vol. 8540, pp. 18–37. Springer, Heidelberg (2015). doi:10.1007/978-3-662-46706-0_2
20. Gueron, S.: Intel Advanced Encryption Standard (AES) Instructions Set. Intel White paper, September 2012
21. Guo, J., Peyrin, T., Poschmann, A., Robshaw, M.J.B.: The LED block cipher. In: Preneel, B., Takagi, T. (eds.) [35], pp. 326–341. http://dx.org/10.1007/978-3-642-23951-9_22
22. Hong, D., Lee, J.-K., Kim, D.-C., Kwon, D., Ryu, K.H., Lee, D.-G.: LEA: a 128-bit block cipher for fast encryption on common processors. In: Kim, Y., Lee, H., Perrig, A. (eds.) WISA 2013. LNCS, vol. 8267, pp. 1–24. Springer, Heidelberg (2014). doi:10.1007/978-3-319-05149-9_1
23. Hong, D., Sung, J., Hong, S.H., Lim, J.-I., Lee, S.-J., Koo, B.-S., Lee, C.-H., Chang, D., Lee, J., Jeong, K., Kim, H., Kim, J.-S., Chee, S.: HIGHT: a new block cipher suitable for low-resource device. In: Goubin, L., Matsui, M. (eds.) CHES 2006. LNCS, vol. 4249, pp. 46–59. Springer, Heidelberg (2006). doi:10.1007/11894063_4
24. Iwata, T., Kurosawa, K.: Stronger security bounds for OMAC, TMAC, and XCBC. In: Johansson, T., Maitra, S. (eds.) INDOCRYPT 2003. LNCS, vol. 2904, pp. 402–415. Springer, Heidelberg (2003). doi:10.1007/978-3-540-24582-7_30
25. Journault, A., Standaert, F.X., Varici, K.: Improving the security and efficiency of block ciphers based on LS-designs. In: Proceedings of the 9th International Workshop on Coding and Cryptography, WCC 2015 (2015)
26. Karakoç, F., Demirci, H., Harmancı, A.E.: ITUbee: a software oriented lightweight block cipher. In: Avoine, G., Kara, O. (eds.) LightSec 2013. LNCS, vol. 8162, pp. 16–27. Springer, Heidelberg (2013). doi:10.1007/978-3-642-40392-7_2
27. Leander, G., Paar, C., Poschmann, A., Schramm, K.: New lightweight DES variants. In: Biryukov, A. (ed.) [9], pp. 196–210. http://dx.org/10.1007/978-3-540-74619-5_13
28. Lim, C.H., Korkishko, T.: mCrypton – a lightweight block cipher for security of low-cost RFID tags and sensors. In: Song, J.-S., Kwon, T., Yung, M. (eds.) WISA 2005. LNCS, vol. 3786, pp. 243–258. Springer, Heidelberg (2006). doi:10.1007/11604938_19
29. Matsuda, S., Moriai, S.: Lightweight cryptography for the cloud: exploit the power of bitslice implementation. In: Prouff, E., Schaumont, P. (eds.) CHES 2012. LNCS, vol. 7428, pp. 408–425. Springer, Heidelberg (2012). doi:10.1007/978-3-642-33027-8_24
30. Mouha, N., Mennink, B., Van Herrewege, A., Watanabe, D., Preneel, B., Verbauwhede, I.: Chaskey: an efficient MAC algorithm for 32-bit microcontrollers. In: Joux, A., Youssef, A. (eds.) SAC 2014. LNCS, vol. 8781, pp. 306–323. Springer, Heidelberg (2014). doi:10.1007/978-3-319-13051-4_19
31. Nandi, M.: Improved security analysis for OMAC as a pseudorandom function. J. Math. Cryptology **3**(2), 133–148 (2009)
32. Nandi, M., Mandal, A.: Improved security analysis of PMAC. J. Math. Cryptology **2**(2), 149–162 (2008)
33. Needham, R.M., Wheeler, D.J.: Tea extensions. Computer Laboratory, University of Cambridge (Technical report), October 1997. http://www.cix.co.uk/~klockstone/xtea.pdf.

34. Pietrzak, K.: A tight bound for EMAC. In: Bugliesi, M., Preneel, B., Sassone, V., Wegener, I. (eds.) ICALP 2006. LNCS, vol. 4052, pp. 168–179. Springer, Heidelberg (2006). doi:10.1007/11787006_15
35. Preneel, B., Takagi, T. (eds.): CHES 2011. LNCS, vol. 6917. Springer, Heidelberg (2011). doi:10.1007/978-3-642-23951-9
36. Rivest, R.L.: The RC5 encryption algorithm. In: Preneel, B. (ed.) FSE 1994. LNCS, vol. 1008, pp. 86–96. Springer, Heidelberg (1995). doi:10.1007/3-540-60590-8_7
37. Shibutani, K., Isobe, T., Hiwatari, H., Mitsuda, A., Akishita, T., Shirai, T.: Piccolo: an ultra-lightweight blockcipher. In: Preneel, B., Takagi, T. (eds.) [35], pp. 342–357. http://dx.org/10.1007/978-3-642-23951-9_23
38. Shirai, T., Shibutani, K., Akishita, T., Moriai, S., Iwata, T.: The 128-bit blockcipher CLEFIA (extended abstract). In: Biryukov, A. (ed.) [9], pp. 181–195. http://dx.org/10.1007/978-3-540-74619-5_12
39. Standaert, F.-X., Piret, G., Gershenfeld, N., Quisquater, J.-J.: SEA: a scalable encryption algorithm for small embedded applications. In: Domingo-Ferrer, J., Posegga, J., Schreckling, D. (eds.) CARDIS 2006. LNCS, vol. 3928, pp. 222–236. Springer, Heidelberg (2006). doi:10.1007/11733447_16
40. Suzaki, T., Minematsu, K., Morioka, S., Kobayashi, E.: TWINE: a lightweight block cipher for multiple platforms. In: Knudsen, L.R., Wu, H. (eds.) SAC 2012. LNCS, vol. 7707, pp. 339–354. Springer, Heidelberg (2013). doi:10.1007/978-3-642-35999-6_22
41. Wang, X., Sako, K. (eds.): ASIACRYPT 2012. LNCS, vol. 7658. Springer, Heidelberg (2012). doi:10.1007/978-3-642-34961-4
42. Wu, W., Zhang, L.: LBlock: a lightweight block cipher. In: Lopez, J., Tsudik, G. (eds.) ACNS 2011. LNCS, vol. 6715, pp. 327–344. Springer, Heidelberg (2011). doi:10.1007/978-3-642-21554-4_19
43. Yang, G., Zhu, B., Suder, V., Aagaard, M.D., Gong, G.: The simeck family of lightweight block ciphers. In: Güneysu, T., Handschuh, H. (eds.) CHES 2015. LNCS, vol. 9293, pp. 307–329. Springer, Heidelberg (2015). doi:10.1007/978-3-662-48324-4_16
44. Yasuda, K.: The sum of CBC MACs is a secure PRF. In: Pieprzyk, J. (ed.) CT-RSA 2010. LNCS, vol. 5985, pp. 366–381. Springer, Heidelberg (2010). doi:10.1007/978-3-642-11925-5_25
45. Yasuda, K.: A new variant of PMAC: beyond the birthday bound. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 596–609. Springer, Heidelberg (2011). doi:10.1007/978-3-642-22792-9_34
46. Yasuda, K.: PMAC with parity: minimizing the query-length influence. In: Dunkelman, O. (ed.) CT-RSA 2012. LNCS, vol. 7178, pp. 203–214. Springer, Heidelberg (2012). doi:10.1007/978-3-642-27954-6_13
47. Zhang, L., Wu, W., Sui, H., Wang, P.: 3kf9: enhancing 3GPP-MAC beyond the birthday bound. In: Wang, X., Sako, K. (eds.) [41], pp. 296–312. http://dx.org/10.1007/978-3-642-34961-4_19
48. Zhang, W., Bao, Z., Lin, D., Rijmen, V., Yang, B., Verbauwhede, I.: RECTANGLE: A Bit-slice Lightweight Block Cipher Suitable for Multiple Platforms. Cryptology ePrint Archive, Report 2014/084 (2014). http://eprint.iacr.org/
49. Zhang, Y.: Using an error-correction code for fast, beyond-birthday-bound authentication. In: Nyberg, K. (ed.) CT-RSA 2015. LNCS, vol. 9048, pp. 291–307. Springer, Heidelberg (2015). doi:10.1007/978-3-319-16715-2_16