On the Security of the COPA and Marble Authenticated Encryption Algorithms against (Almost) Universal Forgery Attack

Jiqiang Lu

Infocomm Security Department, Institute for Infocomm Research,
Agency for Science, Technology and Research,
1 Fusionopolis Way, Singapore 138632
jlu@i2r.a-star.edu.sg,lvjiqiang@hotmail.com

Abstract. COPA is a block-cipher-based authenticated encryption mode with a provable birthday-bound security under the assumption that the underlying block cipher is a strong pseudorandom permutation, and its instantiation with the AES block cipher is called AES-COPA. Marble is an AES-based COPA-like authenticated encryption algorithm with a full security. In this paper, we analyse the security of COPA and Marble against universal forgery attacks. We present beyond-birthday-bound (almost) universal forgery attacks on the COPA when used with constant or variable associate data, and present (almost) universal forgery attacks on the Marble when used without associated data or with (variable) associate data. Our attacks on the COPA with variable associate data have a complexity very near the birthday bound, and their applications to AES-COPA show that the security claim of AES-COPA against tag guessing may be not correct; and our attacks on the (newest as well as initial version of) Marble with associate data show that Marble does not provide a full security that the designer claimed. Like many recently published cryptanalytic results on message authentication algorithms with a provable birthday-bound security, our attacks on COPA do not violate its security proofs, but provide a comprehensive understanding of its security against universal forgery attack, show that the success probability of a universal forgery on the COPA is larger than the ideal bound 2^{-n} of the standard forgery-resistance, and boil down to an existing open question: Should a message authentication algorithm with a weaker security claim than the standard forgery-resistance be regarded as a sound design?

Key words: Authenticated encryption algorithm, COPA, Marble, Universal forgery attack.

1 Introduction

A block cipher is an algorithm that transforms a fixed-length data block, called a plaintext (block), into another data block of the same length, called a ciphertext

(block), under the control of a secret key. The main purpose of a block cipher is to provide data confidentiality. A basic requirement on the security of a block cipher is that: Given all the plaintext-ciphertext pairs (i.e. the entire codebook) generated under some key, it should be not possible to recover the key faster than exhaustive key search (without the knowledge of the key). An example of such attacks is Ferguson et al.'s square attack [12] on 7 rounds of the Rijndael [7] block cipher[1] with a 128-bit key, that requires the entire codebook (i.e. 2^{128} chosen/known plaintexts) and a memory complexity of 2^{64} bits and has a time complexity of 2^{120} encryptions;[2] and another similar example is Lucks' saturation attack [24] on 7 rounds of the Twofish [34] block cipher[3] with a 128-bit key, that requires half of the entire codebook (i.e. 2^{127} chosen plaintexts) and has a time complexity of 2^{126} encryptions.

A message authentication code (MAC) is an algorithm that transforms an arbitrary-length data stream (below an upper bound generally), called a message, into a fixed-length data block, called an (authentication) tag, under the control of a secret key. The main purpose of a MAC is to provide data authenticity and integrity; authenticity protects from impersonation, and integrity protects data from being modified (or at least enables modifications to be detected). There are two types of forgery attacks on MACs [31], one is the so-called existential forgery attack, which is to produce the correct tag for an unspecified message whose tag is not given (under the secret key and some public nonce if any), and the other is the so-called universal (or selective) forgery attack, which is to produce the correct tag for any specified message whose tag is not given. In 1996, Menezes, van Oorschot and Vanstone [25] defined forgery-resistance in terms of computation-resistance, as follows:

Definition 1 (Forgery-Resistance interpreted from pages 325 and 335 of [25]). *Given zero or more message-tag pairs, it is computationally infeasible to compute the tag for any new message, in other words, computing the tag for any new message should have a success probability no more than the bigger one of* 2^{-n} *and* 2^{-k}*, where* n *is the tag bit length and* k *is the key bit length.*

Observe that Definition 1 requires a uniform probability of 2^{-n} or 2^{-k} even when there are a number of message-tag pairs available. In this sense, it is very analogous to the aforementioned basic requirement on the security of a block cipher. Most recently, Dunkelman, Keller and Shamir [10] stressed this requirement from a different perspective by writing that even after choosing a large number of messages and obtaining their corresponding tags under some key, the adversary should not be able to compute with a high success probability the tag for a new message in time which is substantially smaller than the time of

[1] Rijndael was selected finally as the Advanced Encryption Standard (AES) [29].

[2] Typically in block cipher cryptanalysis like [12, 24], encrypting chosen plaintexts is assumed to be done by some "challenger" who holds the user key (i.e. the challenger's running time) and is associated with the data complexity of an attack, and is not counted as part of the time complexity of the attack.

[3] Twofish was one of the five finalists of the AES selection process.

exhaustive key search (or 2^n verification queries).[4] Besides, note that Definition 1 indicates that the time complexity for encrypting given messages should not be counted as part of the time complexity of an attack.

An authenticated encryption algorithm is an algorithm that transforms an arbitrary-length data stream (below an upper bound generally), called a message or plaintext, into another data stream of the same length, called a ciphertext, and generates an (authentication) tag for the message at the same time, under the control of a secret key. It combines the functionalities of a symmetric cipher and a MAC, and achieves data confidentiality, integrity and authenticity at one pass. Since an authenticated encryption algorithm outputs ciphertexts besides tags, the target of forgery attacks on MACs has to be revised when such attacks apply to an authenticated encryption algorithm. It seems that there is no formal definitions for such attacks on an authenticated encryption algorithm; anyway, similarly to those on MAC, we can define their targets as follows:

- An existential forgery attack on an authenticated encryption algorithm is to produce the correct ciphertext and tag for an unspecified message whose ciphertext and tag are not given (under the secret key and some public nonce if any). (Thus, during the decryption and tag verification phase, the message resulted from decrypting the forged ciphertext can result in the forged tag (under the same key and nonce if any).)
- A universal forgery attack on an authenticated encryption algorithm is to produce the correct ciphertext and tag for any specified message whose ciphertext and tag are not given (under the secret key and public nonce if any). (Thus, during the decryption and tag verification phase, the specified message will be generated from decrypting the forged ciphertext, and result in the forged tag (under the same key and nonce if any).)

Note that here the target of a universal forgery attack is much stronger than the target of its counterpart on MAC, where the target is to find the correct tag only. Thus, the seemingly only way to perform such a universal forgery attack on an authenticated encryption algorithm is by recovering the key or some critical internal state (if any), since the other way which guesses the ciphertext and tag is usually less efficient because the ciphertext and tag is usually longer than the key. Besides, Dunkelman et al. [10] introduced the notion of almost universal forgery attack on MAC, which works for almost any specified message although not for any.

COPA [2, 3] is a block-cipher-based authenticated encryption mode, which was proposed at ASIACRYPT '13 for parallel architectures such as general-purpose Central Processing Units and dedicated hardware. COPA was proved

[4] We note that there may be a subtle difference between Dunkelman et al.'s statement and Menezes et al.'s definition: While Menezes et al.'s definition implies Dunkelman et al.'s statement, Dunkelman et al.'s statement may have another meaning that the adversary should produce the tag with a high probability at a time complexity less than exhaustive key search, but Menezes et al.'s definition only requires that there is no more advantageous forgery attack than exhaustive key search. It is unclear whether or not this subtle difference is Dunkelman et al.'s intention.

by the designers to have a birthday-bound security for its privacy and integrity, as long as the underlying block cipher is a strong pseudorandom permutation. Marble (v1.0) [15] is an AES-based COPA-like authenticated encryption algorithm, which was claimed to achieve a full security by setting its internal state twice as long as the key or tag. The key length is equal to the tag length for both COPA and Marble, that is $n = k$. In March 2014, Marble and the COPA instantiated with the AES block cipher under 128 key bits [1] (AES-COPA for short below) were submitted to the CAESAR competition [5] on authenticated encryption, and shortly later the Marble designer made a revision (v1.1) [16] to Marble. Most recently, Fuhr et al. [14] presented universal forgery and key recovery attacks on the revised version of Marble, and then the Marble designer made another revision (v1.2) [17] to it. The newest version of Marble is identical to the initially submitted version, except when the last block of associated data is not full.

In this paper, we analyse the security of COPA and Marble against universal forgery attacks as defined above, and show that there exist (almost) universal forgery attacks on some versions of COPA and Marble, which are more advantageous than exhaustive key search (that is, the attacks are obtained under the standard definition of forgery-resistance or unforgeability owing to Menezes et al., namely Definition 1; more specifically, the time complexity for chosen queries is associated with the data complexity of an attack and is not counted as part of the time complexity of the attack. Thus, as typically in block cipher cryptanalysis, when checking whether an attack is more advantageous than exhaustive key search, we compare the success probability of the attack to the success probability of exhaustive key search with the same time complexity, rather than to the success probability of exhaustive key search with the same number of queries). The attacks require only chosen queries to the message encryption and tag generation oracle of COPA or Marble, and our main attack results are:

- We present a beyond-birthday-bound (almost) universal forgery attack on COPA when used with constant associate data (including the case without associate data). When applied to AES-COPA, it requires about 2^{124} encryption queries and a memory of 2^{121} bytes and has a computational complexity of about 2^{124} simple operations and a success probability of about 32%.
- A slight variant of the aforementioned attack requires $(2^{\phi} + 2)$ encryption queries and a negligible memory and has a computational complexity of about 2^{ϕ} simple operations and a success probability of about $2^{\phi-n}$, where n is the block bit length of the underlying block cipher as well as the tag bit length ($1 \leq \phi < \frac{n}{2}$). This attack variant applies similarly to Marble when used without associated data, with $1 \leq \phi < n$.
- We present beyond-birthday-bound (almost) universal forgery attacks on COPA when used with variable associate data. Each attack has a complexity that is very near the birthday bound. When applied to AES-COPA, each attack requires nearly 2^{63} encryption queries with the total (associated data, message) pairs having a length of nearly 2^{64} blocks (which is very close to the approximate maximum length 2^{64} that AES-COPA can process with a

Table 1. Main (almost) universal forgery attacks on COPA and Marble

Algorithm	Data	Memory	Time	Success Prob.	Source
COPA	$2^{\theta}+2^{\phi}$ Queries	$3n \cdot 2^{\phi}$Bits	2^{ϕ}Simple operations $(\frac{n}{2} < \theta, \phi < n)$	See Sect. 3.1	Sect. 3.1
	2^{ϕ} Queries	negligible	2^{ϕ}Simple operations $(1 \le \phi < \frac{n}{2})$	$2^{\phi-n}$	Sect. 3.2
	$2^{\sigma}+2^{\varphi}$ Queries	$n \cdot 2^{\sigma}$Bits	2^{φ}Memory accesses $(1 \le \sigma, \phi < \frac{n}{2})$	$1-\mathsf{e}^{-2^{\sigma+\varphi-n}}$	Sect. 3.3
AES-COPA (v.1)	2^{124}Queries	$2^{120.6}$Bytes	2^{124}Simple operations	32%	Sect. 3.2
	2^{63}Queries (2^{64}Blocks)	2^{66}Bytes	2^{62}Memory accesses	6%	Sect. 3.3
Marble (v1.1)	2^{65}Queries	2^{68}Bytes	2^{65}Memory accesses	63%	[14]
Marble (v1.0/1/2)	2^{ϕ} Queries	negligible	2^{ϕ}Simple operations $(1 \le \phi < n)$	$\frac{1-\mathsf{e}^{-2^{\phi-127}}}{2}$	Sect. 4.1
Marble (v1.0/2)	2^{65}Queries ($2^{66.6}$Blocks)	2^{68}Bytes	2^{65}Memory accesses	63%	Sect. 4.2

single key), and a memory of about 2^{66} bytes, and has a time complexity of about 2^{62} memory accesses and a success probability of about 6%. (Note that the COPA designers proved its integrity as well as privacy security to be (slightly below) the birthday bound in [2, 3]; for AES-COPA, they claimed its integrity security to be the birthday bound, claimed its security against key recovery to be 128-bit, and also claimed its security against tag guessing to be 128-bit without giving a detailed explanation. We are not clear about their security definition on tag guessing; from a general understanding, our attacks show that this security claim on tag guessing for AES-COPA may be not correct.)

- We present (almost) universal forgery attacks on the newest [17] (as well as initial [15]) version of Marble that uses variable associated data, following Fuhr et al.'s attack [14] on the second version of Marble. Each attack has a data/memory/time complexity of about 2^{65}. (Likewise, there exists a key recovery attack which is similar to Fuhr et al.'s key recovery attack.) Thus, the versions of Marble do not provide a full 128-bit security that the designer claimed for its authentication as well as confidentiality property; that is, the newest (as well as initial) version of Marble is still not secure.

Table 1 summarises previously published and our main (almost) universal forgery attacks on COPA and Marble, where e is the base of the natural logarithm. Besides, our attacks on COPA have the following meanings:

1. Recently, many cryptanalytic results with a data complexity beyond the birthday bound have been published on some MAC algorithms with a provable birthday-bound security, for instance, those [6, 9, 10, 13, 18–20, 22, 23, 28,

30, 33, 37–39] on the ALRED [8], HMAC and NMAC [4] constructions. Theoretically speaking, such attacks are of little significance, for their data complexities are beyond the birthday bounds on the maximum numbers of the data that the MAC algorithms can process with a single key; but nevertheless, they are of academic or practical interest as mentioned in [10, 19, 23, 28], namely, they show that the general belief of a full security against universal forgery attack does not hold for the concerned MAC algorithms, show the gaps between the proved security levels and the real security levels, and remind the user not to misuse the algorithms for a full security on their forgery-resistance. (Despite of the academic and practical interest, we would like to express that we personally think that for a provable scheme such attacks go too far beyond theory, unless they show that its security proof or claim is not correct; if being argued like that, any result deserves being published.) Like these recently published cryptanalytic results, our attacks do not violate the security proofs of COPA, and give us a comprehensive understanding of its security against universal forgery attacks, but nevertheless, their applications to AES-COPA show that the 128-bit security claim on tag guessing of AES-COPA may be not correct.

2. Anyway, the attacks show that the success probability of a universal forgery on the COPA is larger than the bound 2^{-n} of the standard unforgeability owing to Menezes et al. (i.e. Definition 1), and they pose a question (that has been posed by previous work) that needs to be discussed extensively among the cryptology community by taking the opportunity of the ongoing CAESAR competition: Those security claims are weaker than the standard definition of unforgeability (owing to Menezes et al.), should we continue regarding an authenticated encryption algorithm with such a security claim as a sound design? We note that this problem exists in many authenticated encryption algorithms with a provable birthday-bound security, including a few well-known ones of early days, but there are also some authenticated encryption algorithms with/without a provable security that do not suffer from this problem. It seems that the answers to this question are not unified among the cryptology community, which can be illustrated by some responses to early Ferguson's existential forgery attack [11] and recent Huang and Wu's existential forgery attack [21]. Although it does not mean a falw to these existing authenticated encryption algorithms with a provable birthday-bound security, for a long-term perspective it seems preferable to design an authenticated encryption algorithm with the standard unforgeability from now on.

The remainder of the paper is organised as follows. In the next section, we give the notation used throughout this paper, and describe the COPA and Marble algorithms. We present our (almost) universal forgery attacks on COPA and Marble in Sections 3 and 4, respectively. We give some discussions on the forgery-resistance security of authenticated encryption algorithms in Section 5. Section 6 concludes this paper.

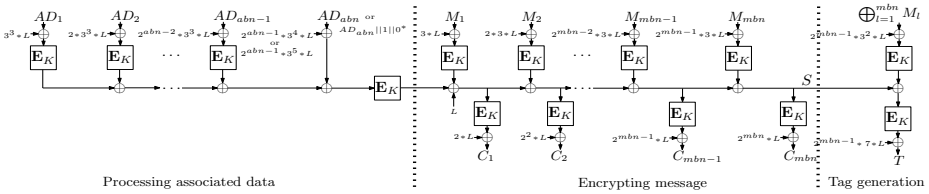


Fig. 1. Message encryption and tag generation of COPA

2 Preliminaries

In this section, we give the notation used throughout this paper and briefly describe the COPA and Marble authenticated encryption algorithms.

2.1 Notation

We use the following notation.

$\oplus$	bitwise logical exclusive OR (XOR) operation
$*$	polynomial multiplication modulo the polynomial $x^{128} \oplus x^7 \oplus x^2 \oplus x \oplus 1$ in GF(2^{128})
e	the base of the natural logarithm ($\mathsf{e} = 2.71828\cdots$)

2.2 The COPA Authenticated Encryption Mode

The COPA [2, 3] authenticated encryption mode was published in 2013. Its internal state, key and tag have the same length. It has three phases: processing associate data, message encryption, and tag generation. Fig. 1 illustrates the message encryption and tag generation phase of COPA, where

- $\mathbf{E}_K$ is an n-bit block cipher with a k-bit user key K;
- $L = \mathbf{E}_K(0)$ is an n-bit secret internal parameter, which is called subkey sometimes [1];
- S is an n-bit internal state;
- $(AD_1, AD_2, \cdots, AD_{abn})$ is an associated data of abn n-bit blocks;
- $(M_1, M_2, \cdots, M_{mbn})$ is a message of mbn n-bit blocks;
- $(C_1, C_2, \cdots, C_{mbn})$ is the ciphertext for $(M_1, M_2, \cdots, M_{mbn})$; and
- T is the tag for $(M_1, M_2, \cdots, M_{mbn})$.

Decryption is the inverse of encryption, and tag verification is identical to tag generation. COPA can take no associate data, by setting the output of the processing associated data phase to zero. Please refer to [2,3] for the specification of COPA.

In 2014, an instantiation [1] of COPA that uses AES with 128 key bits [29] (i.e. AES-COPA) was submitted to the CAESAR competition [5], where a nonce

of 128 bits long is used and is appended to associate data, and the resulting value is treated as the associate data in the COPA mode.

We noted that the COPA designers did not distinguish between a existential forgery and a universal forgery in [2, 3]; both were referred to be a forgery simply. But for AES-COPA [1], they claimed a 64-bit security on its integrity and privacy according to the proved integrity and privacy security from [2, 3], and also claimed a 128-bit security against key recovery, and a 128-bit security against tag guessing. There is no proof or explanation on the security against tag guessing.

2.3 The Marble Authenticated Encryption Algorithm

The Marble [15] authenticated encryption algorithm is like COPA. Its internal state is twice as long as its key or tag to achieve a full security. Marble has four phases: initialization, processing associate data, message encryption, and tag generation. Fig. 2 illustrates the message encryption and tag generation phase of the newest version (i.e. v1.2) of Marble, where

- each of the operations $\mathbf{E_1}, \mathbf{E_2}$ and $\mathbf{E_3}$ is a 4-round reduced version of the AES [29] block cipher, with four fixed round subkeys chosen from the eleven round subkeys of the AES with 128 key bits;
- the TRANS operation is defined as $\text{TRANS}(x, y) = (x \oplus y, 3 * x \oplus y)$, where x and y are 128-bit inputs;
- $Const_0, Const_1$ and $Const_2$ are three 128-bit constants;
- S_1 and S_2 are two 128-bit internal states;
- $(AD_1, AD_2, \cdots, AD_{abn})$ is an associated data of abn 128-bit blocks;
- L and τ are 128-bit secret parameters;
- $(M_1, M_2, \cdots, M_{mbn})$ is a message of mbn 128-bit blocks;
- $(C_1, C_2, \cdots, C_{mbn})$ is the ciphertext for $(M_1, M_2, \cdots, M_{mbn})$; and
- T is the tag for $(M_1, M_2, \cdots, M_{mbn})$.

(Padding is required if the bit length of the associated data or message is not a multiple of 128.) No nonce is used in Marble, (we note that in the last two versions [16,17] of the Marble specification the designer mentioned that one can opt to replace $Const_0$ with a nonce, but this option is not recommended by the designer). Marble is allowed to take no associated data, and in this scenario $\tau = 0$ (and an empty message is not allowed). Decryption is the inverse of encryption, and tag verification is identical to tag generation. Please refer to [17] for the specification of Marble.

The main differences between the newest version of Marble and the other two versions are as follows: In the second version (i.e. v1.1), the mask parameter before $\mathbf{E_1}$ is $2^{abn-1} * 3^2 * L$ for the last block of associate data; and in the initial version (i.e. v1.0), the mask parameter before $\mathbf{E_1}$ is $2^{abn-1} * 3^3 * L$ for the last block of associate data if it is full, and is $2^{abn-1} * 3^4 * L$ if it is not full. Thus, the newest version of Marble is identical to the initial version when the last block of associated data is full.

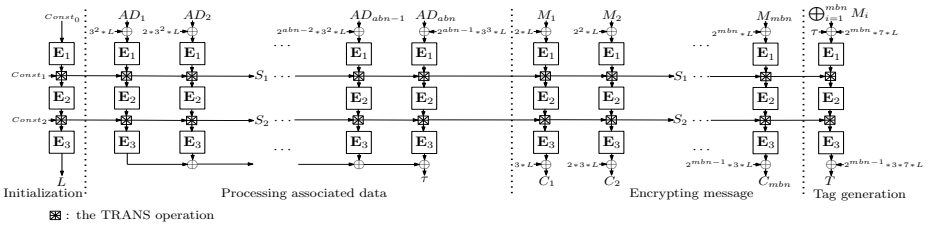


Fig. 2. Message encryption and tag generation of Marble

3 Beyond-Birthday-Bound (Almost) Universal Forgery Attacks on COPA

In this section, we first present a beyond-birthday-bound (almost) universal forgery attack on the COPA that uses constant associated data (including the case that does not use any associate data) to process messages, then give an attack variant when there is a birthday bound on the maximum number of the data that can be processed under a single key, and finally present a beyond-birthday-bound (almost) universal forgery attack on the COPA that uses variable associated data to process messages and apply it to AES-COPA. Each attack is more advantageous than exhaustive key search, and consists of two phases: recovering the secret parameter L, followed by a forgery if L is recovered.

3.1 Beyond-Birthday-Bound (Almost) Universal Forgery Attack on the COPA When Used with Constant Associated Data

We first show how to recover the secret parameter L of the COPA under the constant associated data, in a more advantageous way than exhaustive key search.

3.1.1 Recovering the Secret Parameter L

The procedure is as follows, which is illustrated in Fig. 3-(a). Since the same associated data is used, we will omit it in the attack description.

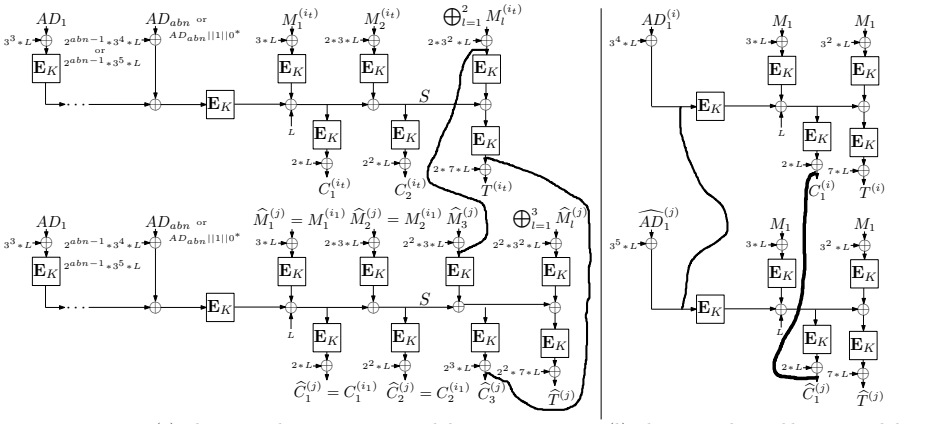
1. Choose randomly at uniform 2^θ messages $M^{(i)} = (M_1^{(i)}, M_2^{(i)})$ of two n-bit blocks long (a specific value of θ will be given below, and $i = 1, 2, \cdots, 2^\theta$). Query the COPA encryption and tag generation oracle, and obtain all the ciphertexts and tags for the 2^θ messages; we denote by $C^{(i)} = (C_1^{(i)}, C_2^{(i)})$ and $T^{(i)}$ the ciphertext and tag for message $M^{(i)}$, respectively.
2. Select a tuple of δ messages $(M^{(i_1)}, M^{(i_2)}, \cdots, M^{(i_\delta)})$ such that
$$C_2^{(i_1)} = C_2^{(i_2)} = \cdots = C_2^{(i_\delta)}. \quad (1)$$
(A specific value of δ will be given below.) This can be done efficiently by storing $(M^{(i)}, C^{(i)}, T^{(i)})$ into a table indexed by $C_2^{(i)}$. Go to Step 1 if there does not exist such a δ-tuple.

3. Choose two n-bit constants α and β such that
$$\alpha * (2 * 3^2 * L \oplus 2^2 * 3 * L) = \beta * (2^3 * L \oplus 2 * 7 * L). \tag{2}$$
Observe that the secret parameter L cancels out in Eq. (2).
4. Choose randomly at uniform 2^ϕ messages $\widehat{M}^{(j)} = (\widehat{M}_1^{(j)}, \widehat{M}_2^{(j)}, \widehat{M}_3^{(j)})$ of three n-bit blocks long (a specific value of ϕ will be given below, and $j = 1, 2, \cdots, 2^\phi$), such that $\widehat{M}_l^{(j)} = M_l^{(i_1)}$ for $1 \le l \le 2$; that is, $\widehat{M}^{(j)} = (M_1^{(i_1)}, M_2^{(i_1)}, \widehat{M}_3^{(j)})$. Query the COPA encryption and tag generation oracle, and obtain all the ciphertexts and tags for the 2^ϕ messages; we denote by $\widehat{C}^{(j)} = (\widehat{C}_1^{(j)}, \widehat{C}_2^{(j)}, \widehat{C}_3^{(j)})$ and $\widehat{T}^{(j)}$ the ciphertext and tag for message $\widehat{M}^{(j)}$, respectively.[5] Since the same user key is used, clearly $\widehat{C}_l^{(j)} = C_l^{(i_1)}$ for $1 \le l \le 2$; i.e., $\widehat{C}^{(j)} = (C_1^{(i_1)}, C_2^{(i_1)}, \widehat{C}_3^{(j)})$.
5. Select the message-ciphertext pair $(\widehat{M}^{(j)}, \widehat{C}^{(j)})$ such that the following two equations hold for some t, here $1 \le t \le \delta$:
$$\widehat{M}_3^{(j)} \oplus 2^2 * 3 * L = \bigoplus_{l=1}^{2} M_l^{(i_t)} \oplus 2 * 3^2 * L; \tag{3}$$
$$\widehat{C}_3^{(j)} \oplus 2^3 * L = T^{(i_t)} \oplus 2 * 7 * L. \tag{4}$$
This can be partially done efficiently by checking whether
$$\alpha * \widehat{M}_3^{(j)} \oplus \beta * \widehat{C}_3^{(j)} = \alpha * \bigoplus_{l=1}^{2} M_l^{(i_t)} \oplus \beta * T^{(i_t)}; \tag{5}$$
we denote the qualified message-ciphertext pair(s) by $(\widehat{M}^{(\omega)}, \widehat{C}^{(\omega)})$ (if any), where $1 \le \omega \le 2^\phi$.
6. Recover L from Eq. (3) with respect to $\widehat{M}^{(\omega)}$, that is $\widehat{M}_3^{(\omega)} \oplus 2^2 * 3 * L = \bigoplus_{l=1}^{2} M_l^{(i_t)} \oplus 2 * 3^2 * L$, and output the recovered L.

Step 1 requires a memory of about $5n \cdot 2^\theta$ bits, which can be reduced to $3n \cdot 2^\theta$ bits by storing only $(\bigoplus_{l=1}^{2} M_l^{(i)}, C_2^{(i)}, T^{(i)})$. It is expected that Eq. (1) holds with probability $(2^{-n})^{\delta-1} = 2^{-n(\delta-1)}$, and the probability that there is at lease one δ-tuple satisfying Eq. (1) is approximately $1 - (1 - 2^{-n(\delta-1)})^{\binom{2^\theta}{\delta}} \approx 1 - \mathsf{e}^{-\binom{2^\theta}{\delta} \cdot 2^{-n(\delta-1)}}$. Eq. (1) guarantees that messages $M^{(i_1)}, M^{(i_2)}, \cdots, M^{(i_\delta)}$ have the same internal state immediately before the tag generation phase.

Observe that for the correct value for L, Eq. (4) holds once Eq. (3) holds, and vice versa. If both Eqs. (3) and (4) hold, then Eq. (5) always holds, because from Eqs. (3) and (4) we have
$$\widehat{M}_3^{(j)} \oplus \bigoplus_{l=1}^{2} M_l^{(i_t)} = 2 * 3^2 * L \oplus 2^2 * 3 * L;$$
$$\widehat{C}_3^{(j)} \oplus T^{(i_t)} = 2^3 * L \oplus 2 * 7 * L.$$

[5] The tags for the 2^ϕ chosen messages are not required in this attack.



(a): the case with constant associated data (b): the case with variable associated data

Fig. 3. State recovery attacks on COPA

Then, we can obtain Eq. (5) after applying α and β to the above two equations and XORing the resulting two equations.

Note that once we obtain the ciphertext-tag pair for a message in Step 4, we can discard it if it does not meet Eq. (5), and thus we only need to store the qualified message-ciphertext-tag tuples in Step 4. Particularly, if we choose $\alpha = 1$ or $\beta = 1$, then Eq. (5) can be checked with one $*$ operation and one $\oplus$ operation (which is negligible compared with one $*$ operation) for a message-ciphertext pair, since the right-hand side of Eq. (5) is one-off.

For a random message-ciphertext pair $(\widehat{M}^{(j)}, \widehat{C}^{(j)})$, it is expected that Eq. (5) holds for a given i_t with a probability of $2^{-n} \times 1 + (1 - 2^{-n}) \times 2^{-n} \approx 2^{1-n}$, assuming that Eq. (5) holds randomly at uniform when at least one of Eqs. (3) and (4) does not hold. On the other hand, for a given i_t the probability that both Eqs. (3) and (4) hold when Eq. (5) holds is

$$
\begin{aligned}
&\Pr.(\text{Eqs. (3) and (4) hold when Eq. (5) holds}) \\
&= \frac{\Pr.(\text{Eq. (5) holds when Eqs. (3) and (4) hold}) \times \Pr.(\text{Eqs. (3) and (4) hold})}{\Pr.(\text{Eq. (5) holds})} \\
&= \frac{1 \times 2^{-n}}{2^{1-n}} \\
&= \frac{1}{2}.
\end{aligned}
$$

Since there are 2^{ϕ} message-ciphertext pairs $(\widehat{M}^{(j)}, \widehat{C}^{(j)})$, the expected number of qualified message-ciphertext pairs satisfying Eq. (5) for an i_t is approximately $2^{\phi} \times 2^{1-n} \times \delta = \delta \cdot 2^{\phi-n+1}$. The probability that there is at least one message-ciphertext pair satisfying Eq. (5) for an i_t is approximately $1 - (1 - \delta \cdot$

$2^{1-n})^{2^\phi} \approx 1 - \mathsf{e}^{-\delta \cdot 2^{\phi-n+1}}$, and the probability that the recovered L is correct is $\frac{1}{2} \cdot (1 - \mathsf{e}^{-\delta \cdot 2^{\phi-n+1}})$.

Therefore, the state recovery attack requires $2^\theta + 2^\phi$ encryption queries (the tags for the 2^ϕ chosen messages are not required) and a memory of approximately $3n \cdot 2^\theta$ bits, and has a computational complexity of about 2^ϕ simple $*$ operations,[6] with a success probability of approximately $\frac{1}{2} \cdot (1 - \mathsf{e}^{-\binom{2^\theta}{\delta} \cdot 2^{-n(\delta-1)}}) \cdot (1 - \mathsf{e}^{-\delta \cdot 2^{\phi-n+1}})$. (However, if one would treat the time complexity for encrypting chosen messages as part of the time complexity of the attack, the resulting time complexity would be about $\lambda \cdot (2^\theta + 2^\phi) + 2^{\phi+1}$ block cipher encryptions, (2^ϕ simple $*$ operations are negligible compared with the block cipher encryptions), where λ is the number of block cipher encryptions for one of the 2^θ messages.)

3.1.2 Making an (Almost) Universal Forgery

Once the correct n-bit secret parameter L is recovered by the above state recovery attack, we can make a universal forgery attack on the COPA with a single query at a one-hundred-percent success probability. Below we assume a target (associated data of abn n-bit blocks long, message of mbn n-bit blocks long) pair $(AD, M) = (AD_1, AD_2, \cdots, AD_{abn}, M_1, M_2, \cdots, M_{mbn})$, where $abn \geq 0$ and $mbn > 0$.

1. Query the COPA encryption and tag generation oracle with the (the same associated data, message of $(mbn + 1)$ n-bit blocks long) pair $(AD, \widetilde{M}) = (AD_1, AD_2, \cdots, AD_{abn}, M_1, M_2, \cdots, M_{mbn}, 2^{mbn} * 3 * L \oplus 2^{mbn-1} * 3^2 * L \oplus \bigoplus_{i=1}^{mbn} M_i)$, and obtain its ciphertext $\widetilde{C} = (C_1, C_2, \cdots, C_{mbn}, \widetilde{C}_{mbn+1})$.
2. The ciphertext for (AD, M) is $C = (C_1, C_2, \cdots, C_{mbn})$, and the tag for (AD, M) is $\widetilde{C}_{mbn+1} \oplus 2^{mbn+1} * L \oplus 2^{mbn-1} * 7 * L$.

In summary, the universal forgery attack that includes the phase of recovering L requires approximately $2^\theta + 2^\phi$ encryption queries (the tags for 2^ϕ chosen messages are not required actually) and a memory of approximately $3n \cdot 2^\theta$ bits, and has a computational complexity of about 2^ϕ simple $*$ operations, with a success probability of approximately $\frac{1}{2} \cdot (1 - \mathsf{e}^{-\binom{2^\theta}{\delta} \cdot 2^{-n(\delta-1)}}) \cdot (1 - \mathsf{e}^{-\delta \cdot 2^{\phi-n}})$. (Note that if one would treat the time complexity for encrypting chosen messages as part of the time complexity of the attack, the resulting time complexity would be about $\lambda \cdot (2^\theta + 2^\phi) + 2^{\phi+1}$ block cipher encryptions, where λ is the number of block cipher encryptions for one of the 2^θ messages.)

In particular, for AES-COPA [1], (which has $n = k = 128$), when we set $\theta = 115, \delta = 8$ and $\phi = 124$, the attack requires about 2^{124} encryption queries and a memory of approximately $2^{120.6}$ bytes, and has a computational complexity of about 2^{124} simple $*$ operations, with a success probability of about 32%.

[6] Here, as typically in block cipher cryptanalysis as well as indicated by Definition 1, encrypting chosen messages is associated with the data complexity of an attack and is not counted as part of the time complexity of the attack. The same statement applies to subsequent attacks, although we do not make any further explicit statements.

3.2 A Variant of the Above Attack on the COPA

When there is a birthday bound on the number of data available, the above attack becomes somewhat less advantageous than exhaustive key search, as $\theta < \frac{n}{2}$. Anyway, we can revise it slightly to make it still more advantageous than exhaustive key search by modifying the first two steps of the above state recovery attack as follows:

1. Choose a message $M^{(1)} = (M_1^{(1)}, M_2^{(1)})$ of two n-bit blocks long. Query the COPA encryption and tag generation oracle, and obtain its ciphertext and tag, and we denote them by $C^{(1)} = (C_1^{(1)}, C_2^{(1)})$ and $T^{(1)}$, respectively.
2. Subsequently, treat $M^{(1)}$ as the $M^{(i_1)}$ in the above beyond-birthday-bound attack (and thus $\delta = t = 1$).

A similar analysis reveals that recovering L requires $(2^\phi + 1)$ encryption queries and a negligible memory, and has a computational complexity of about 2^ϕ simple $*$ operations, with a success probability of approximately $\frac{1-\mathsf{e}^{-2^{\phi-n+1}}}{2} \approx 2^{\phi-n}$, where $1 \leq \phi < \frac{n}{2}$.

Once L is recovered, it is identical to the above attack to make a universal forgery. As a result, when it is possible to obtain a sufficient number of chosen message-ciphertext pairs, one may trade messages for time and memory to perform a universal forgery attack on the COPA when used with constant associated data, at a negligible cost of memory.

3.3 Beyond-Birthday-Bound (Almost) Universal Forgery Attack on the COPA When Used with Variable Associated Data

In this subsection, we first describe how to attack the COPA that uses variable associated data, and then apply it to AES-COPA. The attack is based on Fuhr et al.'s universal forgery attack [14] on Marble.

3.3.1 Recovering the Secret Parameter L

The procedure is as follows, which is illustrated in Fig. 3-(b).

1. Choose 2^σ (associated data of one n-bit block long, fixed message of one n-bit block long) pairs $(AD_1^{(i)}, M_1) = (i, M_1)$, where $0 < \sigma \leq \frac{n}{2}$ and $i = 0, 1, \cdots, 2^\sigma - 1$. Query the COPA encryption and tag generation oracle, and obtain all the ciphertexts and tags for the 2^σ (associated data, message) pairs; we denote by $C_1^{(i)}$ and $T^{(i)}$ the ciphertext and tag under associated data $AD_1^{(i)}$, respectively. Store $C_1^{(i)}$ into a table indexed by $C_1^{(i)}$.
2. Choose $(2^\varphi - \rho)$ (associated data of less than n bits long, the same fixed message of one n-bit block long) pairs such that the (padded associated data, message) pairs $(\widehat{AD}_1^{(j)}, M_1) = (j \times 2^{\frac{n}{2}}, M_1)$, where $0 < \varphi \leq \frac{n}{2}$, $j = 1, 2, \cdots, 2^\varphi - 1$; if $\varphi = \frac{n}{2}$, then $j \neq 2^{\frac{n}{2}-1}$ and $\rho = 2$; and if $\varphi \neq \frac{n}{2}$, then $\rho = 1$. (The padded associated data are possible by the padding rule for associated

data of COPA, namely, first a one then as many zeros as required to reach a multiple of the block size n. ρ represents the number of impossible values for the last block of padded associated data, that is 0 or 2^{n-1}.) Query the COPA encryption and tag generation oracle, and obtain all the ciphertexts and tags for the $(2^{\varphi}-\rho)$ (associated data, message) pairs; we denote by $\widehat{C}_1^{(j)}$ and $\widehat{T}^{(j)}$ the ciphertext and tag under associated data $\widehat{AD}_1^{(j)}$, respectively.

3. Check whether $\widehat{C}_1^{(j)}$ matches one of the set $\{C_1^{(i)}|i=0,1,\cdots,2^{\eta}-1\}$ for $j=1,2,\cdots,2^{\varphi}-1, j \neq 2^{\frac{n}{2}-1}$. We denote the match(es) by $(\widehat{C}_1^{(\omega)}, C_1^{(\mu)})$ if any, that is $\widehat{C}_1^{(\omega)} = C_1^{(\mu)}$.
4. For the match $(\widehat{C}_1^{(\omega)}, C_1^{(\mu)})$, we have $AD_1^{(\mu)} \oplus 3^4 * L = \widehat{AD}_1^{(\omega)} \oplus 3^5 * L$ by the structure of COPA. Thus, we can recover L from this equation.

The reason that we use padded associated data in Step 2 is that an input mask (i.e. $3^5 * L$) different from the one (i.e. $3^4 * L$) used in Step 1 will be introduced for the first block of (padded) associated data. This state recovery attack requires approximately $2^{\sigma} + 2^{\varphi}$ encryption queries, a memory of approximately $n \cdot 2^{\sigma}$ bits (as we do not need to store $\widehat{C}_1^{(j)}$), and has a time complexity of about 2^{φ} memory accesses (from Step 3) and a success probability of approximately $1 - \binom{2^{\eta}\cdot(2^{\varphi}-\rho)}{0} \cdot (2^{-n})^0 \cdot (1-2^{-n})^{2^{\eta}\cdot(2^{\varphi}-\rho)} \approx 1 - \mathsf{e}^{-2^{\eta+\varphi-n}}$.

3.3.2 Making an (Almost) Universal Forgery

If the secret parameter L is recovered by the above state recovery attack, we have two ways to make a universal forgery attack on COPA with a single query at a one-hundred-percent success probability. One way is based on modifying message and is similar to that described in Section 3.1.2, and the other way is based on modifying associated data and is similar to Fuhr et al.'s universal forgery attack [14] on Marble, which we briefly describe below. Assume a target (associated data of abn n-bit blocks long, message of mbn n-bit blocks long) pair $(AD, M) = (AD_1, AD_2, \cdots, AD_{abn}, M_1,\ M_2, \cdots, M_{mbn})$, where $abn > 0$ and $mbn \geq 0$.

1. Query the COPA encryption and tag generation oracle with the (associated data of $(abn+2)$ blocks long, the same message) pair $(\widetilde{AD}, M) = (AD_1, AD_2, \cdots, AD_{abn-1}, \widetilde{AD}_{abn}, \widetilde{AD}_{abn} \oplus 2^{abn} * 3^3 * L \oplus 2^{abn-1} * 3^3 * L, AD_{abn} \oplus 2^{abn-1} * 3^4 * L \oplus 2^{abn+1} * 3^4 * L, M_1, M_2, \cdots, M_{mbn})$, where $\widetilde{AD}_{abn}$ is an arbitrary block. Obtain its ciphertext and tag, denoted respectively by $\widetilde{C} = (\widetilde{C}_1, \widetilde{C}_2, \cdots, \widetilde{C}_{mbn})$ and $\widetilde{T}$.
2. The ciphertext for (AD, M) is $C = (\widetilde{C}_1, \widetilde{C}_2, \cdots, \widetilde{C}_{mbn})$, and the tag for (AD, M) is $\widetilde{T}$.

Particularly, when $\sigma = \varphi = 64$ and $n = 128$, each universal forgery attack that includes the phase of recovering L requires approximately 2^{65} encryption queries, a memory of approximately 2^{68} bytes, and has a time complexity of 2^{64} memory accesses and a success probability of about 63%. (Note that if one

would treat the time complexity for encrypting chosen messages as part of the time complexity of the attack, the resulting time complexity would be about $2^{65} \times 5 \approx 2^{67.4}$ block cipher encryptions.)

3.3.3 Application to AES-COPA

AES-COPA has an additional (public) input parameter called nonce, which has a constant length of 128 bits. It is appended to associated data (if any), and then the resulting value is treated as associated data in COPA. As a consequence, when applying the state recovery attack described in Section 3.3.1 to AES-COPA, we should obtain associated data satisfying Steps 1 and 2; this can be easily done, for example, we choose (associated data of one 128-bit block long, nonce of one 128-bit long) pairs $(AD, N^{(i)})$ in Step 1, and in Step 2 we choose the (associated data of less than 128 bits long, nonce of one 128-bit long) pairs such that the padded (associated data, nonce) pairs are $(AD, X^{(j)})$, where $N^{(i)} = AD_1^{(i)}$ and $X^{(j)} = \widehat{AD}_1^{(j)}$; and a value of AD can be $(1, \cdots, 1, 0)$ in binary form. Then, the first blocks for all the $(2^\eta + 2^\varphi - \rho)$ (padded) (associated data, nonce) pairs are identical, and the first block cipher encryption operations produce the same output, and we only need to modify the above state recovery attack slightly. As a result, the nonces used are different one another, and the state recovery attack works in the nonce-respecting scenario. Of course, it can also work in the nonce-misuse scenario.

For AES-COPA, when we set $\eta = \varphi \approx 62$ extremely, the attack requires nearly 2^{63} queries with the total (associated data, message) pairs having a length of nearly 2^{64} blocks (which is very close to the approximate maximum length 2^{64} that AES-COPA can process with a single key), and a memory of about $2^{62} \times 16 = 2^{66}$ bytes, and has a time complexity of about 2^{62} memory accesses and a success probability of about 6%. (For a longer (associated data, nonce, message) tuple, we need to reduce the values of η and φ accordingly.)

3.4 Notes

Since there is no proof or explanation on the security claim of tag guessing for AES-COPA, we are not clear about how their security against tag guessing for AES-COPA is defined; anyway, from a general understanding of security against tag guessing, it seems that the above attacks invalidate the 128-bit security claim and show that the security of AES-COPA against tag guessing is of the same level as its security on integrity, that is the birthday bound roughly.

Observe that if there is a constraint on the maximum number of the blocks of an associated data or a message in COPA, the attack described in Section 3.1.2 as well as Section 3.2 does not work for a message with the maximum number of blocks, and the attack described in Section 3.3.2 does not work for an associate data with the number of blocks being equal to or one smaller than the maximum number. Thus, the attacks are almost universal forgery attacks. In Section 3.1.2, if associated data is not fixed, we can conduct another attack similarly to that described in Section 3.3.2. Of course, we can combine the two ways together, so

that the attacks can apply to a wider range of (associated data, message) pairs. Besides, when there is a birthday bound on the number of data available, the attacks described in Section 3.3 can be readily revised to be more advantageous than exhaustive key search. (The attacks may apply to a message with the last block being an incomplete block.)

4 (Almost) Universal Forgery Attacks on Marble

In this section, we present two (almost) universal forgery attacks on Marble. Each attack consists of two phases: recovering the secret parameter L, followed by a forgery if L is recovered.

4.1 (Almost) Universal Forgery Attack on the Three Versions of Marble When Used without Associated Data

The procedure for recovering L is basically the same as that described for COPA in Section 3.2, except some minor distinctions due to the differences between COPA and Marble. Different from COPA, the phase of processing associated data of Marble produces a secret parameter τ, which is later used in the phase of tag generation. Thus, the state recovery attack works on the Marble that does not use any associated data (and that uses constant $Const_0$, otherwise, our attack works in the nonce-reuse model, for which the Marble designer claims it to have a full security level as well), and applies to all the three versions of Marble, since the three versions are identical when there is no associated data.

The state recovery attack is illustrated in Fig. 4-(a), without any further text explanation. Its complexity is the same as that given in Section 3.2, except $1 \leq \phi < 128$. (Recall that Marble has a full security, and thus the range of ϕ for Marble is different from that for COPA.)

Once the 128-bit secret parameter L is recovered by the abovementioned state recovery attack, we can then make a universal forgery attack on the Marble (when used without associated data) with a single query at a one-hundred-percent success probability, as follows. Assume a target message $M = (M_1, M_2, \cdots, M_{mbn})$ of mbn 128-bit blocks long ($mbn \geq 1$).

1. Query the Marble encryption and tag generation oracle with the $(mbn+1)$-block message $\widetilde{M} = (M_1, M_2, \cdots, M_{mbn}, 2^{mbn+1} * L \oplus 2^{mbn} * 7 * L \oplus \bigoplus_{i=1}^{mbn} M_i)$, and obtain its ciphertext $\widetilde{C} = (C_1, C_2, \cdots, C_{mbn}, \widetilde{C}_{mbn+1})$.
2. The ciphertext for M is $C = (C_1, C_2, \cdots, C_{mbn})$, and the tag for M is $\widetilde{C}_{mbn+1} \oplus 2^{mbn} * 3 * L \oplus 2^{mbn-1} * 3 * 7 * L$.

The universal forgery attack, including the phase of recovering L, requires $2^{\phi} + 2$ encryption queries (the tags for $2^{\phi} + 1$ chosen messages are not required actually) and a negligible memory, and has a computational complexity of about 2^{ϕ} simple $*$ operations, with a success probability of approximately $\frac{1-\mathsf{e}^{-2^{\phi-127}}}{2}$. Particularly, it has a success probability of about 32% when $\phi = 127$.

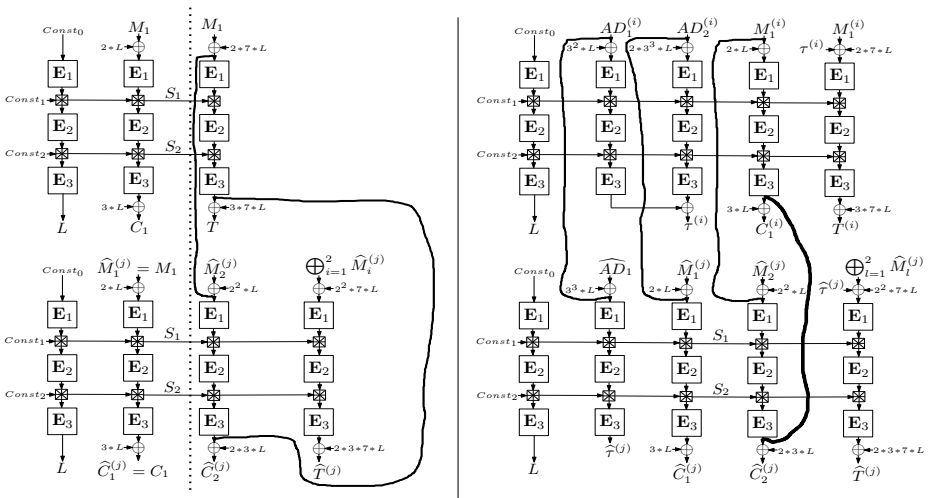


Fig. 4. State recovery attacks on Marble

4.2 (Almost) Universal Forgery Attacks on the Newest and Initial Versions of Marble When Used with (Variable) Associated Data

In this subsection, we show that the newest version of Marble still suffers from (almost) universal forgery attacks that are based on Fuhr et al.'s (almost) universal forgery attack [14] on the second version of Marble. Our attacks also apply to the initial version of Marble, since the last blocks of associated data used in our attacks are full and the newest version of Marble is identical to the initial version when the last block of associated data is full. First, we describe how to recover the secret parameter L.

4.2.1 Recovering the Secret Parameter L

1. Choose 2^{64} (associated data of two blocks long, message of one block long) pairs $(AD_1^{(i)}, AD_2^{(i)}, M_1^{(i)}) = ((3^2 \oplus 3^3) * i, (2 * 3^3 \oplus 2) * i, (2 \oplus 2^2) * i)$, and obtain their ciphertexts (and tags), where $i = 0, 1, \cdots, 2^{64} - 1$; we denote by $C_1^{(i)}$ the ciphertext for message $M_1^{(i)}$. Store $C_1^{(i)}$ into a table indexed by $C_1^{(i)} \oplus (3 \oplus 2 * 3) * i$ (i.e. $C_1^{(i)} \oplus 5 * i$).
2. Choose 2^{64} (associated data of one block long, message of two blocks long) pairs $(\widehat{AD}_1^{(j)}, \widehat{M}_1^{(j)}, \widehat{M}_2^{(j)}) = ((3^2 \oplus 3^3) * (j \times 2^{64}), (2 * 3^3 \oplus 2) * (j \times 2^{64}), (2 \oplus 2^2) * (j \times 2^{64}))$, and obtain their ciphertexts (and tags), where $j = 0, 1, \cdots, 2^{64} - 1$; we denote by $(\widehat{C}_1^{(j)}, \widehat{C}_2^{(j)})$ the ciphertext for message $(\widehat{M}_1^{(j)}, \widehat{M}_2^{(j)})$.
3. Check whether $\widehat{C}_2^{(j)} \oplus (3 \oplus 2 * 3) * (j \times 2^{64})$ (i.e. $\widehat{C}_2^{(j)} \oplus 5 * (j \times 2^{64})$) matches one of the set $\{C_1^{(i)} \oplus 5 * i | i = 0, 1, \cdots, 2^{64} - 1\}$ for $j = 0, 1, \cdots, 2^{64} - 1$. We

denote the match(es) by $(\widehat{C}_2^{(\omega)} \oplus 5 * (\omega \times 2^{64}), C_1^{(\mu)} \oplus 5 * \mu)$ if any, that is, $\widehat{C}_2^{(\omega)} \oplus 5 * (\omega \times 2^{64}) = C_1^{(\mu)} \oplus 5 * \mu$.

4. Recover L from $\widehat{C}_2^{(\omega)} \oplus C_1^{(\mu)} = 5 * (\omega \times 2^{64}) \oplus 5 * \mu = 5 * L$.

For $(AD_1^{(i)}, AD_2^{(i)}, M_1^{(i)})$, the immediate inputs to the three $\mathbf{E_1}$ operations are $(3^2 \oplus 3^3) * i \oplus 3^2 * L, (2 * 3^3 \oplus 2) * i \oplus 2 * 3^3 * L, (2 \oplus 2^2) * i \oplus 2 * L$, respectively; for $(\widehat{AD}_1^{(j)}, \widehat{M}_1^{(j)}, \widehat{M}_2^{(j)})$, the immediate inputs to the three $\mathbf{E_1}$ operations are $(3^2 \oplus 3^3) * (j \times 2^{64}) \oplus 3^3 * L, (2 * 3^3 \oplus 2) * (j \times 2^{64}) \oplus 2 * L, (2 \oplus 2^2) * (j \times 2^{64}) \oplus 2^2 * L$, respectively. Thus, the input difference to the three $\mathbf{E_1}$ operations under $(AD_1^{(i)}, AD_2^{(i)}, M_1^{(i)})$ and $(\widehat{AD}_1^{(j)}, \widehat{M}_1^{(j)}, \widehat{M}_2^{(j)})$ are $(3^2 \oplus 3^3) * [i \oplus (j \times 2^{64}) \oplus L], (2 * 3^3 \oplus 2) * [i \oplus (j \times 2^{64}) \oplus L], (2 \oplus 2^2) * [i \oplus (j \times 2^{64}) \oplus L]$, respectively. Now, if $i \oplus (j \times 2^{64}) = L$, the input difference to the corresponding three $\mathbf{E_1}$ operations will be zero.

As a result, the state recovery attack requires about 2^{65} encryption queries and a memory of about $2^{64} \times 16 = 2^{68}$ bytes, and has a time complexity of about 2^{65} memory accesses and a success probability of about $1 - \binom{2^{128}}{0} \cdot (2^{-128})^0 \cdot (1 - 2^{-128})^{2^{128}} \approx 63\%$.

4.2.2 Making an (Almost) Universal Forgery

Once L is recovered, we have two different ways to make an (almost) universal forgery attack on Marble; one way is similar to that described for COPA in Section 3.1.2, working on the Marble that does not use associated data, and the other way is similar to Fuhr et al.'s attack [14] on the second version of Marble.

In summary, each universal forgery attack that includes the phase of recovering L requires about 2^{65} encryption queries and a memory of about 2^{68} bytes, and has a time complexity of about 2^{65} memory accesses and a success probability of about 63%. (Note that if one would treat the time complexity of encrypting chosen messages as part of the time complexity of the attack, the resulting time complexity would be about $2^{65} \times 5 \approx 2^{67.4}$ AES encryptions.)

4.3 A Note

We note that the Marble designer distinguished the cases whether the last block of associated data is full or not in the initial version [15], (by using different input mask parameters for them), but seemed not to distinguish the cases in the second and newest versions [16,17]. If this was the intention of the designer, the second and newest versions of Marble are also vulnerable to other universal forgery attacks obtained simply by changing the length of the last block of associated data as long as there is an associated data with the last block being not the 128-bit string $(0, 0, \cdots, 0)$ or $(1, 0, \cdots, 0)$; specifically, when the last block of the original associated data is not full, query with the padded associated data (as well as the original message), or when the last block of the original associated data is full, query with the truncated associated data such that the padded associated data is the original associated data. The ciphertexts as well as tags for a message are identical under such a pair of associated data.

5 Discussions on Forgery-Resistance of Authenticated Encryption

We notice that there exist many open problems in the field of authenticated encryption. First, the standard definition of forgery-resistance, namely Definition 1, requires that the security should always have a uniform probability of 2^{-n} even when there are a number of message-tag pairs available. In other words, Definition 1 requires that there should be no more advantageous forgery attack than exhaustive key search (or other generic attacks), until the number of available messages reaches the maximum number of messages that can be processed under a key. However, we note that some cryptographers and cryptanalysts may have a different understanding of the forgery-resistance security of an authentication algorithm with a provable birthday-bound security. This is partially due to the various security claims that are not consistent with the standard definition of forgery-resistance. Sometimes, the standard definition of forgery-resistance is deliberately weakened, for example, Daemen and Rijmen [8] claimed that (given known message-tag pairs) the success probability of any forgery attack not involving key recovery or internal collisions is 2^{-n} for the ALRED construction (that has an iterated structure with a provable birthday-bound security), by adding restrictions "not involving key recovery or internal collisions".

Recently, Huang and Wu [21] described an existential forgery attack on AES-OTR [27] — the OTR [26] authenticated encryption mode instantiated with AES. Huang and Wu's attack requires one query of an maximum-length message of 2^{60} blocks and has a time complexity of about 2^{60} simple operations and a success probability of 2^{-11}. By Definition 1, this attack should be regarded as valid; and Wu thought AES-OTR has an 11-bit security only, rather than the claimed 128-bit security, however, it does not violate the security proof or claim of AES-OTR and is not considered to be a valid attack by the OTR designer. A similar example from early days is Ferguson's existential forgery attack [11] on the OCB [32] authenticated encryption mode; the attack is valid by Definition 1, but does not contradict with the security proof of OCB.

A less serious but somewhat similar problem also exists in some other candidates submitted to the CAESAR competition. For example, for CLOC [35, 36], which is also a block-cipher-based authenticated encryption mode with a provable birthday-bound security and works in a nonce-respecting model, if we obtain about $2^{\frac{n+1}{2}}$ encryption queries under random associated data and nonces, then it is expected that there is a pair of queries colliding on the XOR value of the first message block and the first ciphertext block, which means the internal states immediately before the first message block are identical for the colliding query pair, and thus we can make an existential forgery by exchanging the (associated data, nonce) pairs for the pair of colliding queries. Likewise, it does not violate the security proof or claim of CLOC.

For these examples and the versions of COPA and Marble that we have cryptanalysed above as well as some others, their security margins become smaller and smaller when there are more and more message-tag pairs available, (even in the nonce-respecting model for some algorithms). This weak property does

not conform to the standard definition of forgery-resistance, which requires a uniform probability of 2^{-n} until the number of available messages reaches the maximum number of messages that can be processed under the same key. On the other hand, this weak property does not exist in some authenticated encryption algorithms with/without a provable security. In this sense, this weak property is due to the designs of the specific algorithms.

Now we face a question: should we continue regarding an authentication algorithm with a weaker security claim than the standard forgery-resistance as a sound design? The ongoing CAESAR competition may be a good opportunity for the cryptology community to unify the answers to this controversial question, by discussing extensively and defining explicitly what the forgery-resistance security is. A design may have an advantage if it has a provable security, however, its motivation is doubtable if the proof is at the compromise of the standard security. From a future perspective, it seems preferable to design an authenticated encryption algorithm with the standard unforgeability security from now on.

There are also some other questions that seem to have no unified answers and need to be discussed as well. For example, should we distinguish between existential and universal forgeries by defining different security levels for them? If yes, how to define a universal forgery? Our definition on a universal forgery attack given in Section 1 may have connection with privacy, thus, considering that an authenticated encryption algorithm provides more functionalities than a MAC, should we introduce a new security notion that defines integrity and privacy simultaneously? Should we treat the time complexity for encrypting chosen messages (queries) as part of the time complexity of an attack, or simply associate it with the data complexity as typically in block cipher cryptanalysis? As indicated in Definition 1, it seems preferable to associate it with the data complexity of an attack; otherwise, an attack may be very easy in practice after the attacker collects sufficient data. If so, when checking whether an attack is more advantageous than exhaustive key search, we should compare the success probability of the attack to the success probability of exhaustive key search with the same time complexity, rather than to the success probability of exhaustive key search with the same number of queries, as typically in block cipher cryptanalysis.

It would be another major contribution of the CAESAR competition if the competition could unify these controversial problems and give formal security definitions, so that a future design or cryptanalysis can follow. We think this would be not less (if not more) significant than selecting a specific algorithm.

6 Concluding Remarks

In this paper, we have presented beyond-birthday-bound (almost) universal forgery attacks on the COPA that uses the same associated data or variable associate data, and have presented (almost) universal forgery attacks on the Marble that does not use any associated data or uses variable associated data. Our attacks on the COPA that uses variable associate data have a complexity

that is very close to the birthday bound, and their applications to AES-COPA may show that the security claim of AES-COPA against tag guessing may be not correct; and our attacks on the newest (as well as initial) version of Marble shows that Marble does not provide a full 128-bit security as the designer claimed. The attacks are mainly based on the structures of COPA and Marble, and thus may apply to other authenticated encryption algorithms with similar structures. Like many recently published cryptanalytic results on message authentication algorithms with a provable birthday-bound security, our attacks on COPA do not violate its security proofs, but give us a comprehensive understanding of its security against universal forgery attacks, show that the success probability of a universal forgery on the COPA is larger than the ideal bound 2^{-n} of the standard forgery-resistance, and boil down to an existing open question: Should a message authentication algorithm with a weaker security claim than the standard forgery-resistance be regarded as a sound design?

Acknowledgments

The author is grateful to Jian Guo and Kan Yasuda for their discussions on some of the attacks and to Hongjun Wu for his discussions on forgery-resistance.

References

1. Andreeva, E., Bogdanov, A., Luykx, A., Mennink, B., Tischhauser, E., Yasuda, K.: AES-COPA v.1, Submission to the CAESAR competition, March 2014. http://competitions.cr.yp.to/round1/aescopav1.pdf
2. Andreeva, E., Bogdanov, A., Luykx, A., Mennink, B., Tischhauser, E., Yasuda, K.: Parallelizable and Authenticated Online Ciphers. In: Sako, K., Sarkar, P. (eds.) ASIACRYPT 2013. LNCS, vol. 8269, pp. 424–443. Springer, Heidelberg (2013)
3. Andreeva, E., Bogdanov, A., Luykx, A., Mennink, B., Tischhauser, E., Yasuda, K.: Parallelizable and Authenticated Online Ciphers. Cryptology ePrint Archive, Report 2013/790 (2013). http://eprint.iacr.org/2013/790
4. Bellare, M., Canetti, R., Krawczyk, H.: Keying Hash Functions for Message Authentication. In: Koblitz, N. (ed.) CRYPTO 1996. LNCS, vol. 1109, pp. 1–15. Springer, Heidelberg (1996)
5. CAESAR — Competition for Authenticated Encryption: Security, Applicability, and Robustness. http://competitions.cr.yp.to/caesar.html
6. Contini, S., Yin, Y.L.: Forgery and Partial Key-Recovery Attacks on HMAC and NAMC using Hash Collisions. In: Lai, X., Chen, K. (eds.) ASIACRYPT 2006. LNCS, vol. 4284, pp. 37–53. Springer, Heidelberg (2006)
7. Daemen, J., Rijmen, V.: AES proposal: Rijndael. Presented at the First AES Candidate Conference. NIST, 1998.
8. Daemen, J., Rijmen, V.: Refinmements of the ALRED construction and MAC security claims. IET Information Security 4(3), 149–157 (2010)
9. Dinur, I., Leurent, G.: Improved Generic Attacks against Hash-Based MACs and HAIFA. In: Garay, J.A., Gennaro, R. (eds.) CRYPTO 2014. LNCS, vol. 8616, pp. 149–168. Springer, Heidelberg (2014)

10. Dunkelman, O., Keller, N., Shamir, A.: Almost universal forgery attacks on AES-based MAC's. Designs, Codes and Cryptography, avaialble as Online First. DOI: 10.1007/s10623-014-9969-x
11. Ferguson, N.: Collision attacks on OCB, 2002. http://csrc.nist.gov/groups/ST/toolkit/BCM/documents/comments/General_Comments/papers/Ferguson.pdf
12. Ferguson, N., Kelsey, J., Lucks, S., Schneier, S., Stay, M., Wagner, D., Whiting, D.: Improved Cryptanalysis of Rijndael. In: Schneier, B. (ed.) FSE 2000. LNCS, vol. 1978, pp. 213–230. Springer, Heidelberg (2001)
13. Fouque, P.-A., Leurent, G., Nguyen, P.Q.: Full Key-Recovery Attacks on HMAC/NMAC-MD4 and NMAC-MD5. In: Menezes, A. (ed.) CRYPTO 2007. LNCS, vol. 4622, pp. 13–30. Springer, Heidelberg (2007)
14. Fuhr, T., Leurent, G., Suder, V.: Forgery and Key-Recovery Attacks on CAESAR Candidate Marble. HAL archive hal-01102031, 13 January 2015. http://hal.inria.fr/hal-01102031v2
15. Guo, J.: Marble Specification Version 1.0, Submission to the CAESAR competition, 15 March 2014. http://competitions.cr.yp.to/round1/marblev10.pdf
16. Guo, J.: Marble Specification Version 1.1, Submission to the CAESAR competition, 26 March 2014. http://competitions.cr.yp.to/round1/marblev11.pdf
17. Guo, J.: Marble Specification Version 1.2, 16 January 2015.
18. Guo, J., Peyrin, T., Sasaki, Y., Wang, L.: Updates on Generic Attacks against HMAC and NMAC. In: Garay, J.A., Gennaro, R. (eds.) CRYPTO 2014. LNCS, vol. 8616, pp. 131–148. Springer, Heidelberg (2014)
19. Guo, J., Sasaki, Y., Wang, L., Wu, S.: Cryptanalysis of HMAC/NMAC-Whirlpool. In: Sako, K., Sarkar, P. (eds.) ASIACRYPT 2013. LNCS, vol. 8270, pp. 21–40. Springer, Heidelberg (2013)
20. Guo, J., Sasaki, Y., Wang, L., Wang, M., Wen, L.: Equivalent Key Recovery Attacks against HMAC and NMAC with Whirlpool Reduced to 7 Rounds. In: Cid, C., Rechberger, C. (eds.) FSE 2014. To appear.
21. Huang, T., Wu. H.: Attack on AES-OTR, 2014. https://groups.google.com/forum/#!forum/crypto-competitions
22. Kim, J., Biryukov, A., Preneel, B., Hong, S.: On the Security of HMAC and NMAC Based on HAVAL, MD4, MD5, SHA-0 and SHA-1 (Extended Abstract). In: De Prisco, R., Yung, M. (eds.) SCN 2006. LNCS, vol. 4116, pp. 242–256. Springer, Heidelberg (2006)
23. Leurent, G., Peyrin, T., Wang, L.: New Generic Attacks against Hash-Based MACs. In: Sako, K., Sarkar, P. (eds.) ASIACRYPT 2013. LNCS, vol. 8270, pp. 1–20. Springer, Heidelberg (2013)
24. Lucks, S.: The Saturation Attack – A Bait for Twofish. In: Matsui, M. (ed.) FSE 2001. LNCS, vol. 2355, pp. 1–15. Springer, Heidelberg (2002)
25. Menezes, A., van Oorschot, P., Vanstone, S.: Handbook of Applied Cryptography. CRC Press, 1996.
26. Minematsu, K.: Parallelizable Rate-1 Authenticated Encryption from Pseudorandom Functions. In: Nguyen, P.Q., Oswald, E. (eds.) EUROCRYPT 2014. LNCS, vol. 8441, pp. 275–292. Springer, Heidelberg (2014)
27. Minematsu, K.: AES-OTR v1, 14 March 2014. http://competitions.cr.yp.to/round1/aesotrv1.pdf
28. Naito, Y., Sasaki, Y., Wang, L., Yasuda, K.: Generic State-Recovery and Forgery Attacks on ChopMD-MAC and on NMAC/HMAC. In: Sakiyama, K., Terada, M. (eds.) IWSEC 2013. LNCS, vol. 8231, pp. 83–98. Springer, Heidelberg (2013)
29. National Institute of Standards and Technology (NIST). Advanced Encryption Standard (AES), FIPS-197 (2001)

30. Peyrin, T., Wang, L.: Generic Universal Forgery Attack on Iterative Hash-Based MACs. In: Nguyen, P.Q., Oswald, E. (eds.) EUROCRYPT 2014. LNCS, vol. 8441, pp. 147–164. Springer, Heidelberg (2014)
31. Preneel, B., van Oorschot, P.C.: On the Security of Iterated Message Authentication Codes. IEEE Transcations on Information Theory 45(1), 188–199 (1999)
32. Rogaway, P., Bellare, M., Black, J.: OCB: A block-cipher mode of operation for efficient authenticated encryption. ACM Transactions on Information and System Security 6 (3), 365–403 (2003)
33. Rechberger, C., Rijmen, V.: New Results on NMAC/HMAC when Instantiated with Popular Hash Functions. Journal of Universal Computer Science 14(3), 347–376 (2008)
34. Schneier, S., Kelsey, J., Whiting, D., Wagner, D., Hall, C., Ferguson, N.: The Twofish Encryption Algorithm. Presented at the First AES Candidate Conference. NIST, 1998.
35. Iwata, T., Minematsu, K., Guo, J., Morioka, S.: CLOC: Authenticated Encryption for Short Input. In: Cid, C., Rechberger, C. (eds.) FSE 2014. To appear.
36. Iwata, T., Minematsu, K., Guo, J., Morioka, S.: CLOC: Compact Low-Overhead CFB, 15 March 2014. `http://competitions.cr.yp.to/round1/clocv1.pdf`
37. Wang, L., Ohta, K., Kunihiro, N.: New Key-Recovery Attacks on HMAC/NMAC-MD4 and NMAC-MD5. In: Smart, N. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 237–253. Springer, Heidelberg (2008)
38. Wang, X., Yu, H., Wang, L., Zhang, H., Zhan, T.: Cryptanalysis on HMAC/NMAC-MD5 and MD5-MAC. In: Joux, A. (ed.) EUROCRYPT 2009. LNCS, vol. 5479, pp. 121–133. Springer, Heidelberg (2009)
39. Yuan, Z., Wang, W., Jia, K., Xu, G., Wang, X.: New Birthday Attacks on Some MACs Based on Block Ciphers. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 209–230. Springer, Heidelberg (2009)