Universal Forgery with Birthday Paradox: Application to Blockcipher-based Message Authentication Codes and Authenticated Encryptions

Fanbao Liu and Fengmei Liu

Science and Technology on Information Assurance Laboratory, Beijing, 10072, China
lfbjantie@163.com

Abstract. Universal forgery attack means that for any given message M, an adversary without the key can forge the corresponding Message Authentication Code (MAC) tag τ, and the pair (M, τ) can be verified with probability 1. For a secure MAC, the universal forgery attack should be infeasible to be implemented, and the complexity is believed to be $\min(2^n, 2^k)$ queries, where n is the tag length and k is the key length of the MAC, respectively.

In this paper, we propose a general universal forgery attack framework to some blockcipher-based MACs and authenticated encryptions (AEs) using birthday attack, whose complexity is about $O(2^{n/2})$ queries in the classic setting. The attack shows that such MACs and AEs are totally insecure. However, this attack is not applicable in the quantum model, since no inclusion of period in the input messages is guaranteed.

We also propose another some generic universal forgery attacks using collision finding with structural input messages, by birthday paradox in the classic setting. Since our attacks are based on the collision finding with fixed but unknown differences (or period), such attack can also be implemented with only $O(n)$ queries using *Simon's* algorithm in the quantum model, which shows that such MACs and AEs are completely broken in the quantum model.

Our attacks can be applied to CBC-MAC, XCBC, EMAC, OMAC, CMAC, PC-MAC, MT-MAC, XOR-MAC, PMAC, PMAC with parity, Light-MAC and some of their variants. Moreover, such attacks are also applicable to the authenticated encryptions of the third round CAESAR candidates: CLOC, SILC, OCB, AEZ, OTR, COLM (including COPA and ELmD) and Deoxys.

Keywords: Universal forgery, birthday attack, CBC-MAC, PMAC, CAESAR, quantum model.

1 Introduction

Message Authentication Code (MAC) [DK], frequently realized through blockcipher or hash function, aims to provide data integrity and authenticity, by allowing the verifier to detect any alternation to the message. A MAC algorithm takes a secret key and a message of arbitrary length as input, and outputs a short fixed length tag.

As an important cryptographic primitive, MAC has been widely used. Mainly, there are two kinds of blockcipher-based MACs, one is CBC-MAC [BKR94] like and the other is PMAC [BR02] like, where the latter provides parallel processing of messages.

Informally, a MAC consists of three algorithms: a key generation, a tag generation and a verifying algorithm. One of the most important security requirements for a MAC with n-bit output τ is that, given a message M and a k-bit secret key K, the computation of the MAC value $\tau = \mathrm{MAC}_K(M)$ should be easy, however, it should be computationally infeasible for an adversary to get $\mathrm{MAC}_K(M)$ without the knowledge of the key K. Mainly, there are three types of forgery: existential, selective and universal [1].

1. **Existential forgery**. An adversary can create at least one message/tag pair, (m, τ), where m has not been produced by the legitimate user. Moreover, the adversary need not have any control over the message m, hence, the message may be any meaningless information.
2. **Selective forgery**. An adversary can create the message/tag pair (m, τ), for a message m of his choice prior to the attack.
3. **Universal forgery**. An adversary can create the message/tag pair (m, τ), for any given message m.

It is widely believed that the computational complexity of existential forgery is about $O(2^{n/2})$ queries due to generic birthday attack on iterative MACs [PvO95]. However, the computational complexity of universal forgery is about $\min(2^n, 2^k)$ queries [Stab], where n is the tag length and k is the key length of the MAC, respectively. It means that the adversary has to launch an exclusive search attack on the tag τ or the key K, to forge the tag for the given message m, which implies that existential forgery will not immediately cause universal forgery attack without further complexity. In fact, we can consider an extreme situation that the complexity of existential forgery is $O(1)$, one can get an existential forgery with just one online computation. The complexity of universal forgery from existential forgery is about $O(2^n)$, for the input messages are random in the existential forgery, to find the forgery for any given message. The complexity of such attack will not be less than the exhaustive search of universal forgery.

CBC-MAC computes a MAC tag from a blockcipher through CBC mode, which can not be handled in parallel. The standardization of CBC-MAC like MACs are widely applied [(re86,X9.86,I99]. The security of CBC-MAC with fixed length messages was first formally analyzed in [BKR94,BM01,JPS03,BPR05]. Later, EMAC was proposed to provide the ability to process arbitrary varying

[1] The notion of security was first described in [GMR88], for the security of digital signature.

length message using two keys [PR00]. After then, many of variants of CBC-MAC were proposed, such as XCBC with three keys [BR05], TMAC with two keys [KI03], OMAC with one key [IK03] standardized as CMAC [fBCMoO05].

Unlike CBC-MAC, PMAC (Parallelizable MAC) [BR02] is a parallelizable blockcipher-based MAC, whose internal structure is suitable for parallelization. Parallelization is vital for the Internet of things.

Authenticated encryption (AE) or authenticated encryption with associated data (AEAD) is a cryptographic primitive [ae]. AE's encryption scheme simultaneously provides confidentiality, integrity assurances on the processed messages, and its decryption is combined with data integrity verification.

CAESAR competition (Competition for Authenticated Encryption: Security, Applicability, and Robustness), announced in 2013, aims at fulfilling the needs of secure, efficient and robust authenticated encryption schemes. In total, 57 candidates were submitted to the competition. To process the associated data, a MAC must be employed in authenticated encryption. CAESAR candidate CLOC and SILC uses CBC-MAC to authenticated the associated message [IMG$^+$]. The PMAC type MACs are widely used in the CAESAR competition, such as OCB [KR11], AEZ [HKR15,HKR], COPA [ABL$^+$13], OTR [Min14], POET [AFF$^+$15,Nan14], OMD [CMN$^+$14], ELmD [DN], COLM [ABL$^+$b], Deoxys [JNPS], and Minalpher [STA$^+$a].

Post-quantum cryptography focuses on providing cryptographic primitives resisting quantum adversary, under the pressure of quantum computing maturation. In [KLLNP16], a general existential forgery attack to CBC-MAC variants, PMAC variants MACs and authenticated encryptions with associated data was proposed, by utilizing the quantum period finding algorithm Simon's algorithm, the computational complexity is about $O(n)$ under the quantum computing setting which dramatically speeds up the classical setting of $O(2^{n/2})$ [PvO95]. Their attack really threats the security of such symmetric cryptosystems, such as authenticated encryption, however, we know that the existential forgery could not be easily used to launch meaningful message forgery, since the attacker can not control the message content.

We wonder if universal forgery attack can be launched with such low complexity $O(2^{n/2})$ in classic setting and $O(n)$ in quantum model, respectively, which eventually means that such symmetric cryptosystems are totally broken.

We recall that some iterated blockcipher-based Message Authentication Codes, for example PMAC [MT06], can be viewed as $\mathrm{PMAC}_K(M) = E_K(E_K(x_1 \oplus \Delta_1) \oplus E_K(x_2 \oplus \Delta_2) \oplus \cdots \oplus E_K(x_l \oplus \Delta_l)) = E_K(f_1(x_1) \oplus f_2(x_2) \oplus \cdots \oplus f_l(x_l))$, where E_K is a blockcipher with key K, Δ_i is secret offset, l represents the blocks of message M, and f_i is permutation with secret embedded key. If we focus on the collision of MAC, then only the inner part $f_1(x_1) \oplus f_2(x_2) \oplus \cdots \oplus f_l(x_l)$ should be concerned, since the outer part E_K is a permutation. For simplicity, if we assume that $\mathrm{PMAC}_K(x_1 || x_2) = \mathrm{PMAC}_K(x_1' || x_2')$, then the equation $f_1(x_1) \oplus f_2(x_2) = f_1(x_1') \oplus f_2(x_2')$ holds.

It is interesting that $a \oplus b = c \oplus d$ always implies $c \oplus b = a \oplus d$, where a, b, c, d are variables, for the $\oplus$ operation. We observe that this property can be utilized to transform an existential forgery to an universal one by embedding the given messages in the colliding messages with the complexity unchanged, for the iterated blockcipher-based MACs and AEs. For example, to forge the corresponding PMAC tag τ, for any given 2-block message $x||y$, we fix the first block message with x and randomly choose the second message y_i in the first group G_1, and fix the second message with y and randomly choose the first message x_j in the second group G_2. Then, there should exist a colliding pair $(x||y_i, x_j||y)$ satisfying $\tau_i = \mathrm{PMAC}_K(x||y_i) = \tau_j = \mathrm{PMAC}_K(x_j||y)$ for some i, j, by the generic birthday attack with two groups. We know it is true that $\mathrm{PMAC}_K(x||y) = \mathrm{PMAC}_K(x_j||y_i)$, which means that we get the very tag τ for the given 2-block message $x||y$, since the computation of PMAC utilizing the *XOR* operation to *sum* the encrypted messages. However, this attack is not applicable in the quantum model, since no period guarantee of messages is provided.

We note that once the secret difference Δ of PMAC variants is revealed, an universal forgery can be made directly, by carefully choosing the queried messages. Combined with the attack in [KLLNP16], an universal forgery attack for PMAC variants with complexity of $O(n)$ can be implemented in the quantum model. Moreover, the universal forgery for CBC-MAC variants can be handled in a similar way.

Our contributions: First, We propose a generic universal forgery attack to most of MACs and AEs, with birthday attack's complexity $O(2^{n/2})$, our attack shows that such schemes are insecure in the classical setting. This is the first generic universal forgery attack for such MACs and AEs to our knowledge. Our attack violates the instinctive thought of that the universal forgery attack is infeasible even under the condition that the existential forgery is available. However, we note that such attack can not be implemented with Simon's algorithm in the quantum model, since our attack provides no guarantee that the colliding input messages have any fixed but unknown difference, or period.

Second, based on the existential forgery attack in [KLLNP16], we propose another some generic universal forgery attacks, exploiting the inner structure of the messages, with complexity of $O(2^{n/2})$ in the classic setting. Fortunately, with fixed but unknown difference or period of the messages, we can mount universal forgery attacks to such MACs and AEs with complexity about $O(n)$ in the quantum model, which means that such schemes are completely broken in the quantum model.

Our attacks can be applied to CBC-MAC, XCBC, EMAC, OMAC, CMAC, PC-MAC, MT-MAC, XOR-MAC, PMAC, PMAC with parity, LightMAC and some of their variants. Moreover, such attacks ar also applicable to the authenticated encryptions of the third round CAESAR candidates: CLOC, SILC, OCB, AEZ, OTR, COLM (including COPA and ELmD) and Deoxys.

Organization of the Paper. The rest of this paper is organized as follows. We introduce some preliminaries in section 2. In section 3, we present a generic universal forgery attack to most of MACs and AEs, with complexity of about $O(2^{n/2})$, in the classic setting. We show another some generic universal forgery attacks with complexity of $O(2^{n/2})$ in the classic setting, which can be implemented in the quantum model with complexity of $O(n)$ in section 4. We summarize this paper in the last section.

2 Preliminaries

Notations Let E_K denote an n-bit blockcipher using k-bit key K, and E_K^T represent a tweakable blockcipher with a tweak T. Let $||$ denote the concatenation operation, and $pad10^*$ be the function that applies 10^* padding on n bits. Let Σ be the sum of XORed values.

2.1 CBC-MAC variants

CBC-MAC CBC-MAC is one of the most used MAC constructions, which is based on blockcipher in the CBC mode. However, the basic CBC-MAC is only secure when the queries are prefix-free. CBC-MAC is defined as follows.

$$\text{CBC-MAC}_K(M) \begin{cases} y_0 = 0, \\ y_i = E_K(y_{i-1} \oplus x_i), \text{ for } i \leq m, \\ \tau = f(y_m) \end{cases}$$

where f is a truncation function. The security of CBC-MAC was first analyzed in [BKR94].

EMAC An early strengthened version of CBC-MAC with two keys [PR00], aims to provide the ability of handling varying length messages. EMAC further encrypts the CBC-MAC value by a new key K_2 as follows

$$\tau = \text{EMAC}_{K_1,K_2}(M) = E_{K_2}(\text{CBC-MAC}_{K_1}(M))$$

XCBC XCBC was proposed to efficiently process messages with arbitrary lengths [GD02,BR05]. XCBC uses three keys, one key K_1 with k-bit for the blockcipher E, the other two n-bit keys K_2 and K_3 are used in the last message block processing with different conditions. If the message length is the multiple of the block length n, set $K = K_2$, and no padding is done. Otherwise, set $K = K_3$, and pad the last message with 10^*.

$$\text{XCBC}_{K_1,K_2,K_3}(M) \begin{cases} y_0 = 0, \\ y_i = E_{K_1}(y_{i-1} \oplus x_i), \text{ for } i \leq m-1, \\ \tau = E_{K_1}(y_{m-1} \oplus x_i \oplus K) \end{cases}$$

TMAC TMAC, a two key version of XCBC, was introduced by Kurosawa and Iwata in [KI03]. TMAC is obtained directly from XCBC by replacing (K_2, K_3) with $(K_2 \cdot u, K_2)$, where u is some non-zero constant.

OMAC and CMAC OMAC, a one key version of XCBC, was introduced by Iwata and Kurosawa in [IK03]. OMAC has two versions for different key generations. OMAC1 replaces $(K_2,\ K_3)$ with $(L \cdot u,\ L \cdot u^2)$, where u is some constant and L is computed by $L = E_K(0^n)$. OMAC2 replaces $(K_2,\ K_3)$ with $(L \cdot u,\ L \cdot u^{-1})$. OMAC1 was recommended by NIST as CMAC [fBCMoO05].

PC-MAC and MT-MAC PC-MAC and MT-MAC, both proposed by Minematsu and Tsunoo [MT06], aims to provide higher performance. PC-MAC is an efficient periodic CBC-like construction, and MT-MAC is another provable secure efficient MAC based on the modified tree hash.

2.2 PMAC variants

PMAC PMAC [BR02] is parallelizable blockcipher-based MAC, whose internal structure XE with secret offsets Δi derived from the secret key to turn the block cipher into a tweakable block cipher, is suitable for parallelization. The PMAC algorithm is defined as

$$\mathrm{PMAC}_K(M) \begin{cases} c_i = E_K(x_i \oplus \Delta_i), \\ \tau = E_K(x_m \oplus \Sigma c_i \oplus \Delta_f) \end{cases}$$

where Δ_f means that the last message block x_m is processed differently upon if the length of the whole message is multiple of the block length n. We omit the generation of the secret offsets because they are irrelevant to our attack.

PMAC with parity PMAC with parity is a variant of PMAC with four different keys, whose security bound is of the form $O(q^2/2^n + \sigma q/2^{2n})$ [Yas12]. PMAC with parity is shown in Fig. 1.

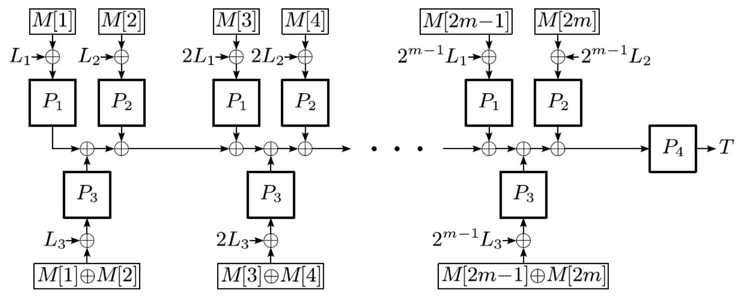

Fig. 1. PMAC with parity [Yas12]

LightMAC LightMAC, a MAC mode of operation specifically suited to lightweight applications [LPTY16]. Its security bound do not depend on the message length, allowing an order of magnitude more data to be processed per key. LightMAC is shown in Fig. 2.

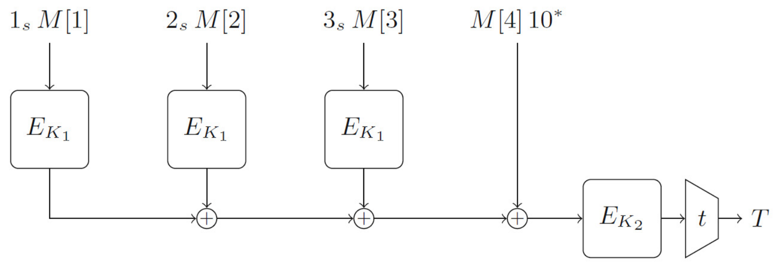

Fig. 2. LightMAC [LPTY16]

XOR-MAC XOR-MAC, proposed by Bellare et al. in 1995 [BRR95], is a parallelizable MAC, it is XORed together finite-input-length pseudorandom functions (PRF) to create stateful and randomized MACs. XOR-MAC is proven secure, and has bounds with no message length dependence. The lightMAC is similar to XOR-MAC [BRR95].

2.3 CAESAR candidates of the third round

CAESAR competition (Competition for Authenticated Encryption: Security, Applicability, and Robustness), announced in 2013, aims at fulfilling the needs of secure, efficient and robust authenticated encryption schemes. In total, 57 candidates were submitted to the competition. Now, there are 15 candidates remains in the third round. Mainly, there are two kinds of MACs used, one is based on CBC-MAC or PMAC, the other is based on sponge construction.

CLOC and SILC [IMG$^+$] both use CBC-MAC to authenticate the associated data. The PMAC type MACs are widely used in the third round of CAESAR competition, such as OCB [KR11], AEZ [HKR15], COLM [ABL$^+$b] (COPA [ABL$^+$13], ELmD [DN]), OTR [Min14], Deoxys [JNPS].

CLOC and SILC At abstract level CLOC is a straightforward combination of CFB and CBC MAC, where CBC MAC is called twice for processing associated data and a ciphertext, and CFB is called once to generate a ciphertext [IMG$^+$]. SILC is built upon CLOC, and inherits the overall structure. SILC is a combination of CFB and CBC MAC, where they use functions *fix1* and *zpp* to logically separate CFB and CBC MAC. The processing of associated data in CLOC and SILC are shown in Fig. 3 and Fig. 4, respectively.

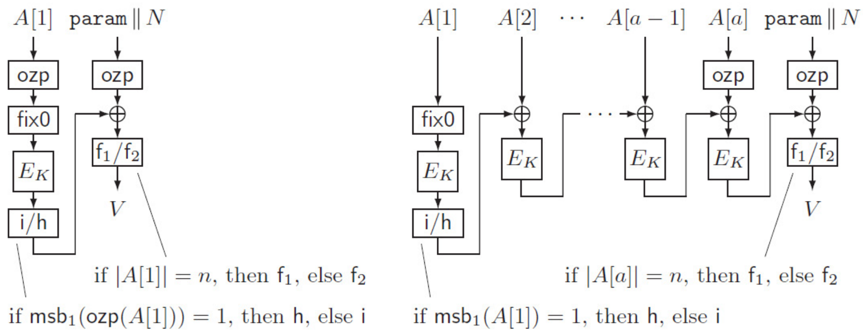
A[1]
param || N
ozp
fix0
E_K
i/h
f_1/f_2
V
if |A[1]| = n, then f_1, else f_2
if msb_1(ozp(A[1])) = 1, then h, else i
A[2]
A[a − 1]
A[a]
if |A[a]| = n, then f_1, else f_2
if msb_1(A[1]) = 1, then h, else i

Fig. 3. Processing of Associated data in CLOC authenticated encryption [IMG+]

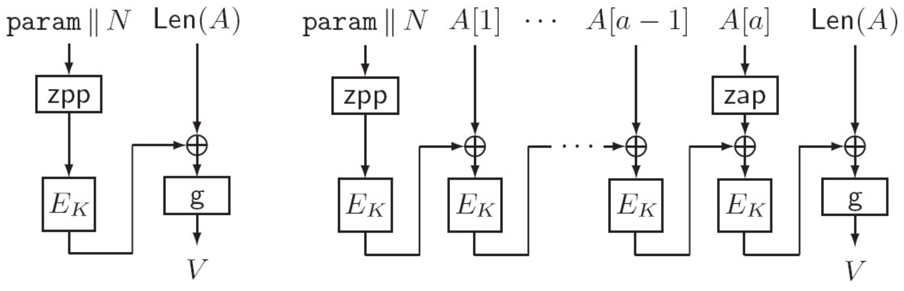

Fig. 4. Processing of Associated data in SILC authenticated encryption [IMG+]

AEZ One can view AEZ-prf as an approximation of an AES-based PMAC in which all but the final blockcipher call have the number of AES rounds reduced from 10 to 4 [HKR].

ELmD ELmD, designed by Datta and Nandi [DN], is fully parallelizable and online. It is an Encrypt-Linear-mix-Decrypt block cipher authentication mode accepting associated data, and its structure is similar to some other authenticated encryption schemes such as COPA, Marble, and SHELL. The processing of associated data in ELmD applies a variant of PMAC, which is shown as Fig. 5

COPA COPA has been designed to allow for high performance in parallel environments and to maintain security even if nonce is reused [ABL+a]. The processing of associated data in COPA applies a variant of PMAC, which is shown as Fig. 6

COLM At a high level, COLM can be seen as a block cipher based Encrypt-Linear mix-Encrypt mode, designed with the goal to achieve online misuse resistance, to be fully parallelizable, and to be secure against block-wise adaptive

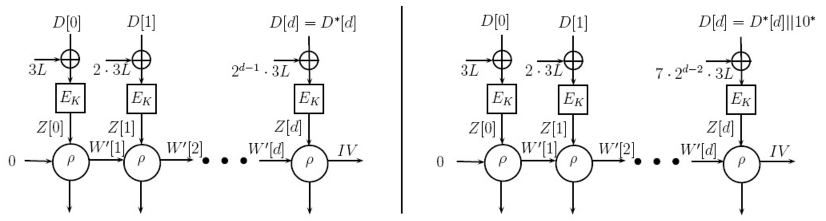

Fig. 5. Processing of Associated data in ELmD authenticated encryption [DN]

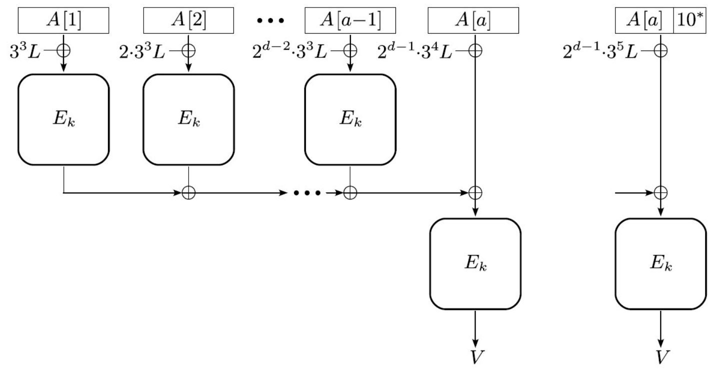

Fig. 6. Processing of Associated data in COPA Authenticated Encryption [ABL+a]

adversaries [ABL+b]. COLM resembles the best of both COPA [ABL+a] and ELmD [DN]. The processing of associated data in COLM applies a variant of PMAC, which is shown in Fig. 7.

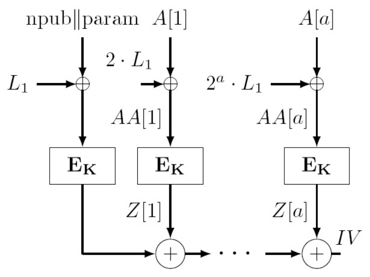

Fig. 7. Processing of Associated data in COLM authenticated encryption [ABL+b]

Deoxys Deoxys is designed to provide a sound ad-hoc tweakable block cipher based on AES [JNPS]. Indeed, the main idea heavily exploited in the design of Deoxys is the introduction of an efficient tweakable block cipher Deoxys-BC, belonging to the family of the well-known AES-based primitives. Speed benchmarks show that Deoxys achieves almost the same speed as OCB while offering much higher security than OCB. Deoxys has two versions, one Deoxys-I is nonce-respecting, the other Deoxys-II is nonce-reuse. The processing of associated data in Deoxys applies a variant of PMAC, which is shown in Fig. 8.

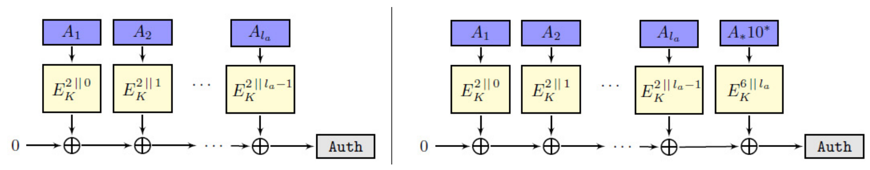

Fig. 8. Processing of Associated data in Deoxys authenticated encryption [JNPS]

OTR Nonce-respecting OTR has two versions depending on associated data processing, ADP [Min]. For parallel ADP version, the processing of AD is based on (a variant of) PMAC, and the computation can be done in parallel to the processing of plaintext/ciphertext. This is to maximize the parallel computation capability. For serial ADP version, the processing of AD is based on CMAC, hence is inherently serial. The processing of associated data in OTR with a CBC-MAC and PMAC variants are shown in Fig. 9 and Fig. 10, respectively.

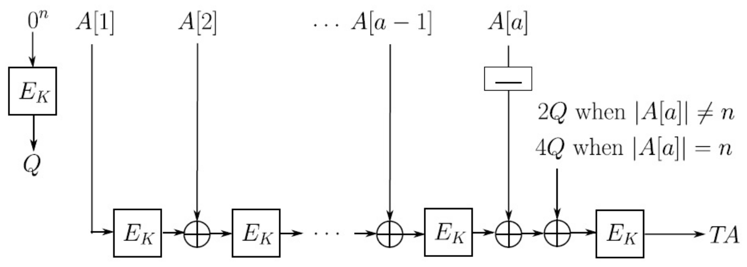

Fig. 9. Processing of Associated data with CBC-MAC variant in OTR authenticated encryption [Min]

2.4 Collision Searching in the quantum model

Simon's problem and algorithm Simon's problem says that: Given a boolean function $f : \{0,1\}^n \to \{0,1\}^n$ and the promise that there exists $s \in \{0,1\}^n$ such

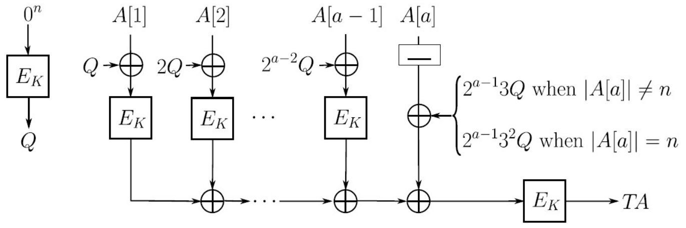
0^n
$A[1]$
$A[2]$
$A[a-1]$
$A[a]$
Q
$2Q$
$2^{a-2}Q$
E_K
Q
$2^{a-1}3Q$ when $|A[a]| \neq n$
$2^{a-1}3^2Q$ when $|A[a]| = n$
TA

Fig. 10. Processing of Associated data with PMAC variant in OTR authenticated encryption [Min]

that for any $(x, y) \in \{0,1\}^n$, $[f(x) = f(y)] \Leftrightarrow [x \oplus y \in \{0^n, s\}]$, the goal is to find s [Sim97].

This problem can be solved by searching for collision in the classic setting, where the input messages have a fixed but unknown difference, with complexity about $O(2^{n/2})$. However, this problem can be solved by Simon's algorithm with quantum complexity of $O(n)$ in the quantum model, which dramatically speed up the process.

The original formulation of Simon's algorithm is for functions whose collisions happen only at some hidden period, which also means a fixed but unknown difference. In [KLLNP16], the authors extended it to functions that have more collisions, which immediately leads to a better analysis of previous applications of Simon's algorithm in the quantum model.

How to apply Simon's algorithm, [KLLNP16]. The strategy is that: exhibit a new function f for the encryption oracle $E_K : \{0,1\}^n \to \{0,1\}^n$, that satisfies Simon's promise with two additional properties. First, The adversary can access the E_K in the quantum model, which means that he can query function f in superposition. Second, once the attacker get the information of s, it is sufficient to break the cryptographic scheme. In particular, the value s will usually be the difference in the internal state after processing a fixed pair of messages (α_0, α_1), i.e. $s = E_K(\alpha_0) \oplus E_K(\alpha_1)$. The input of f will be inserted into the state with the difference s so that $f(x) = f(x \oplus s)$.

For simplicity, we keep in mind that if the colliding input messages have fixed but unknown difference s, the complexity to find such a difference is about $O(2^{n/2})$ in the classic model, and about $O(n)$ in the quantum model, respectively. If find such a difference is critical for the cryptographic scheme, then it immediately means broken in the quantum model.

3 A generic universal forgery attack in the classic setting

In this paper, we mainly focus on the deterministic message authentication code and authenticated encryption with nonce-reuse model, there are two reasons, first, we think that the assumption of nonce-respecting is not immediately available in practice, for example, for the randomly generated l-bit nonce, there will be a collision after about $2^{l/2}$ accessing. Second, we note that the MACs of the associated data used in authenticated encryptions are basically deterministic, while the plaintext to be encrypted applying different nonces to be probability, and the problem of different nonces used in authentication encryption can be handled well in the quantum model in [KLLNP16].

3.1 Attack strategy

In general, the strategy of a generic existential forgery attack for function f is shown as follows. First, the attacker finds a colliding input message pair (x, x') satisfying $f(x) = f(x')$, by querying the oracle f with birthday attack complexity. Second, the attacker queries $\tau = f(x||y)$, for a message y. Finally, the tag τ is also valid for the un-queried message $x'||y$. However, this kind of attack is not suitable for the generic universal forgery, we must make some changes.

In the following, we show how to launch a generic universal forgery attack while the existential forgery attack is available with collision searching by birthday attack. We recall that the equation $f_1(x_1) \oplus f_2(x_2) = f_1(x'_1) \oplus f_2(x'_2)$ will always holds, if the equation $f_1(x'_1) \oplus f_2(x_2) = f_1(x_1) \oplus f_2(x'_2)$ holds, where f_1 and f_2 are both permutations [2].

For any given message $x_1||x_2$, we use the above property to launch a generic universal forgery attack, for colliding messages with 2-block width. For simplicity, we search a collision that $f(x_1||x'_2) = f(x'_1||x_2)$ with fixed x_1 and x_2, where message x'_i are randomly generated 1-block messages in collision searching, which also can be used for existential forgery directly.

Luckily, $f(x_1||x'_2) = f(x'_1||x_2)$ can be further handled as $f_1(x_1) \oplus f_2(x'_2) = f_1(x'_1) \oplus f_2(x_2)$ in most of blockcipher-based iterated MACs, like CBC-MAC or PMAC. Then, if we fix the 1-block message x_1 and message x_2, and search collisions with generic birthday attack in two groups $x_1||x_2^j$ and $x_1^i||x_2$ with $2^{n/2}$ elements, where $i, j \leq 2^{n/2}$. Finally, with high probability, we can find such a collision pair $(x_1||x'_2, x'_1||x_2)$ satisfying $f_1(x'_1) \oplus f_2(x_2) = f_1(x_1) \oplus f_2(x'_2)$, which will directly leads to that $f_1(x_1) \oplus f_2(x_2) = f_1(x'_1) \oplus f_2(x'_2)$, then finally, we have another collision pair $(x_1||x_2, x'_1||x'_2)$. If we query the oracle f for the tag τ of message $x'_1||x'_2$, then the corresponding tag τ is also valid with probability 1, for the given message $x_1||x_2$, the generic universal forgery succeeds.

3.2 How to launch universal forgery attack with birthday attack

We further use a generic birthday attack with two groups to implement a generic universal forgery for any given message $x_1||x_2|| \cdots ||x_l$, where $l \geq 2$ [3].

First, randomly generate $2^{n/2}$ 1-block messages x_2^i in group G_1, where the first message block is the fixed x_1 and $i \leq 2^{n/2}$, finally, query $x_1||x_2^i$ to the oracle MAC, there will be $2^{n/2}$ elements of corresponding MAC tags $\tau_i = \text{MAC}_K(x_1||x_2^i)$ returned in G_1;

[2] This property was first observed in [JWYX09] to launch the second pre-image attack.
[3] If $l = 1$, there will no collision happens, since f_1 and f_2 are both permutations.

Second, randomly generate $2^{n/2}$ 1-block messages x_1^j in group G_2, where the second message block is the fixed x_2 and $j \leq 2^{n/2}$, finally, query $x_1^j||x_2$ to the oracle MAC, there will be $2^{n/2}$ elements of corresponding MAC tags $\tau_j = \mathrm{MAC}_K(x_1^j||x_2)$ returned in G_2;

There should exist $\tau_i = \tau_j$ for some i, j with high probability, by the birthday paradox. So, we will get the key information that $f_1(x_1) \oplus f_2(x_2^i) = f_1(x_1^j) \oplus f_2(x_2)$, for the iterated blockcipher-based MACs.

Third, query the message $x_1^j||x_2^i||x_3||\cdots||x_l$ to the oracle MAC, a corresponding tag τ will be returned.

We note that the tag τ is also valid for the given message $x_1||x_2||x_3||\cdots||x_l$, which is never queried by the adversary to the oracle MAC.

Finally, the generic universal forgery attack succeeds.

Application to CBC-MAC variants We recall that all variants of CBC-MAC utilizing the CBC mode operation, here, we focus on the EMAC construction [PR00] for simplicity, however, this attack is suitable for all variants of CBC-MAC, including the MAC part of third round CAESAR candidate CLOC and SILC.

We note that EMAC further encrypts the CBC-MAC value by a new key K_2 as follows

$$\tau = \mathrm{EMAC}_{K_1,K_2}(M) = E_{K_2}(\text{CBC-MAC}_{K_1}(M))$$

For a colliding message pair (X, X') of EMAC, it should be satisfied that $\mathrm{EMAC}_{K_1,K_2}(X) = \mathrm{EMAC}_{K_1,K_2}(X')$, which eventually means that we have $\text{CBC-MAC}_{K_1}(X) = \text{CBC-MAC}_{K_1}(X')$, since E_{K_2} is a permutation. We point out that $f_1(x) = E_{K_1}(x \oplus 0)$ and $f_2(x) = x$ for the first two message blocks in CBC-MAC variants. If message length $l \geq 2$, we can fix the rest of messages as constant, for simplicity.

Hence, for a 2-block colliding message pair, we should just consider the situation that $E_{K_1}(x_1) \oplus x_2' = E_{K_1}(x_1') \oplus x_2$. To implement a generic universal forgery for any given message $x_1||x_2||\cdots||x_l$, we construct the first group G_1 with the fixed x_1, the group G_2 with the fixed x_2, and then we should get a collision that $\tau_i = \tau_j$ with high probability. We know that

$$E_{K_1}(x_1) \oplus x_2' = E_{K_1}(x_1') \oplus x_2 \Leftrightarrow E_{K_1}(x_1) \oplus x_2 = E_{K_1}(x_1') \oplus x_2'$$

Finally, we query EMAC for the tag τ of the message $x_1'||x_2'||x_3||\cdots||x_l$, where the returned tag τ is also valid for the un-queried given message $x_1||x_2||\cdots||x_l$ with probability 1, the universal forgery attack for EMAC succeeds.

Application to PMAC variants We recall that all variants of PMAC utilizing the notion of tweakable blockcipher with secret offsets, here, we focus on the PMAC construction [BR02] for simplicity. However, this attack is also suitable for all variants of PMAC, including the MAC part of the third round of CAESAR candidates, like AEZ, OCB, OTR, COLM (including COPA and ELmD) and Deoxys.

We note that PMAC further encrypts the XORed value generated in parallel by the same key K, here, we focus the 2-block message where the final message is ignored. We point out that $f_1(x) = E_K(x \oplus \Delta_1)$ and $f_2(x) = E_K(x \oplus \Delta_2)$ for the first two message blocks in PMAC and its variants.

$$\mathrm{PMAC}_K(M) \begin{cases} c_i = E_K(x_i \oplus \Delta_i), \\ \tau = E_K(c_1 \oplus c_2) \end{cases}$$

For a colliding message pair $(x_1||x_2', x_1'||x_2)$ with 2-block, it should be satisfied that $c_1 \oplus c_2' = c_1' \oplus c_2$, which eventually means that $E_K(x_1 \oplus \Delta_1) \oplus E_K(x_2 \oplus \Delta_2) = E_K(x_1' \oplus \Delta_1) \oplus E_K(x_2' \oplus \Delta_2)$, since E_K is a permutation.

To implement a generic universal forgery for any given message $x_1||x_2||\cdots||x_l$, we construct the first group G_1 with the fixed x_1, the second group G_2 with the fixed x_2, and then we should get a collision that $\tau_i = \tau_j$ with high probability, by the birthday paradox.

Finally, we query PMAC for the tag τ of the message $x_1'||x_2'||x_3||\cdots||x_l$, where the returned tag τ is also valid for the un-queried given message $x_1||x_2||\cdots||x_l$ with probability 1, the universal forgery attack for PMAC succeeds.

3.3 Discussions

Application to authenticated encryptions with nonce-reuse We know that the forgery of MAC is the forgery of authenticated encryption schemes, the above attack can be immediately implemented to the third round CAESAR candidates CLOC, SILC, AEZ, COLM (including COPA and ELmD) and Deoxys.

Complexity. We know that the queries made to the oracle MAC are about $O(2^{n/2})$ in the classic setting, by the birthday paradox, and the success probability 1 of such universal forgery can also be reached, through enlarging the capacity of the above two groups.

Drawbacks. First, the above attack learns nothing about the critical secret information like Δ_i used in PMAC variants, which can be used to launch further existential forgery attack, and sometimes key recovery attack [BEK16]. Second, such attack can not be implemented by *Simon's algorithm* in the quantum model, since the equation $\tau_i = \tau_j$, implying $f_1(x_1^j) \oplus f_2(x_2^i) = f_1(x_1) \oplus f_2(x_2) = \mathbf{Const}$, provides no guarantee of hidden periodicity for the collided messages.

3.4 Further application to AEs with TBC in the nonce-respecting model

Our attack is also applicable to the AEs with tweakable blockcipher in the nonce-respecting model in the classic setting, for example, OCB and Deoxys-I. We recall that OCB or Deoxys-I instantiated with a TBC is provable secure up to $O(2^n)$, however, our attack will show that the claim of security bound of such schemes is totally based on the ideal "nonce-respecting" model itself, which may not be realistic in practice, like "one-time pad". An universal forgery attack will happen with complexity only of $O(2^{n/2})$, even with one repeated nonce in the whole system.

Deoxys is designed to provide a sound ad-hoc tweakable block cipher based on AES [JNPS]. Deoxys-I is for adversaries that are assumed to be nonce-respecting, as OCB and OTR, meaning that the user must ensure that the nonce value N will never be used for encryption twice with the same key. The Message processing for the nonce-respecting mode in Deoxys-I is shown in Fig. 11.

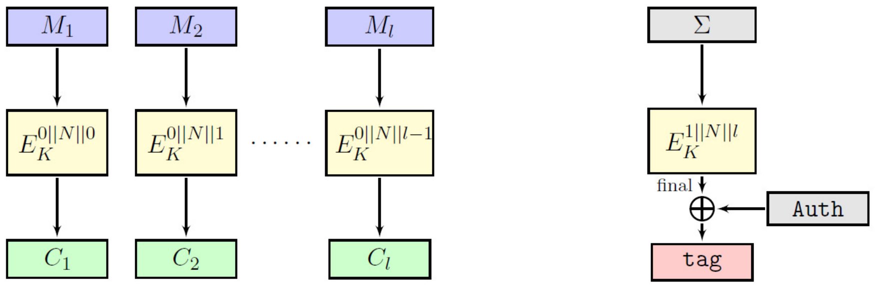

Fig. 11. Message processing for the nonce-respecting mode in Deoxys-I authenticated encryption [JNPS]

We know that auth $= \oplus_{i=1}^{l_a} E_K^{2||i}(A_i)$ in both Deoxys-I and Deoxys-II, where l_a is the blocks of the associated data. Here we just consider that the associated data is a multiples of block size n, the other condition can be processed similarly.

The tag generation with message processing that $E_K^{1||N||l}(\oplus_{i=1}^{l} E_K^{0||N||i}(N, M_i))$ can be viewed as a whole of $f_3(N, M)$ in Deoxys-I, for simplicity. So, the tag τ of Deoxys-I can be rewritten as $\tau = Auth \oplus f_3(N, M) = \oplus_{i=1}^{l_a} E_K^{2||i}(A_i) \oplus f_3(N, M)$, and finally, $\tau = E_K^{2||1}(A_1) \oplus E_K^{2||2}(A_2) \oplus f_3(N, M)$, while we just consider the 2-block associated data for simplicity.

To forge the corresponding tag τ for the given message $A_1||A_2||\cdots||A_{la},\ M$, where M can be anything fixed, we take actions as follows.

First, randomly generate $2^{n/2}$ message blocks A_2^i in group G_1, where $i \leq 2^{n/2}$ and A_1 is fixed, and query $A_1||A_2^i||M$ to the oracle Deoxys-I, there will be $2^{n/2}$ elements of corresponding Deoxys-I tags τ_i returned;

Randomly generate $2^{n/2}$ message blocks A_1^j in group G_2, where $j \leq 2^{n/2}$ and A_2 is fixed, and query $A_1^j||A_2||M$ to the oracle Deoxys-I, there will be $2^{n/2}$ elements of corresponding Deoxys-I tags τ_j returned;

There should exist $\tau_i = \tau_j$ for some i, j with high probability, by the birthday paradox. So, we will get a corresponding information that

$$E_K^{2||1}(A_1) \oplus E_K^{2||2}(A_2^i) \oplus f_3(N^i, M) = E_K^{2||1}(A_1^j) \oplus E_K^{2||2}(A_2) \oplus f_3(N^j, M)$$

From the above equation we can immediately know that

$$E_K^{2||1}(A_1) \oplus E_K^{2||2}(A_2) \oplus f_3(N^i, M) = E_K^{2||1}(A_1^j) \oplus E_K^{2||2}(A_2^i) \oplus f_3(N^j, M)$$

and we also know that

$$E_K^{2||1}(A_1) \oplus E_K^{2||2}(A_2) \oplus f_3(N^j, M) = E_K^{2||1}(A_1^j) \oplus E_K^{2||2}(A_2^i) \oplus f_3(N^i, M)$$

It means that we know the very tags of the message $A_1||A_2||\cdots||A_{la}$, M with different nonces N^i and N^j are equal to the tags of $A_1^j||A_2^i||A_3||\cdots||A_{la}$, M with corresponding nonces N^j and N^i, respectively. If we can query the latter message with a nonce only repeated once, where only one nonce is reused once in the whole system, the scheme will be totally broken with birthday attack complexity.

However, we can not query none of the messages $A_1^j||A_2^i||A_3||\cdots||A_{la}$, M with different nonces N^j and N^i to the oracle Deoxys, even we know that their tags are equal with probability 1, since the system assumes that the nonce will never be reused.

However, we know that the ideal nonce-respecting model is something like "one-time pad", which is impractical in reality.

We argue that the security claim of 2^n is meaningless, since the randomly generated nonce as long as n-bit will be reused in any case, by the birthday paradox. Moreover, it is obviously a bad idea to develop a dedicated synchronization protocol to support such AE schemes with nonce-respecting model, in case of network communication between two users.

3.5 Direct application to CLOC, SILC and Deoxys-*II* with nonce-reuse

The universal forgery attack can be also directly applied to the AEs of the third round of CAESAR candidates: CLOC, SILC and Deoxys-*II*, under the assumption of that the MAC part of such schemes are fixed or NULL. An universal forgery attack will happen with complexity of only $O(2^{n/2})$, in the classic setting. Here, we first discuss Deoxys-*II*, and leave the more complicated cases CLOC and SILC in the later.

Deoxys-*II* Deoxys-*II* is for adversaries that are assumed to be nonce-reusing. Unlike one pass AEs, the message processing is divided into two parts: the authentication part for tag generation and the encryption part for ciphertext. The Message processing in the authentication part and encryption part without padding in Deoxys-*II* are shown in Fig. 12 and Fig. 13, respectively. Here, we omit the case of messages with padding, which can be processed similarly.

To implement a generic universal forgery for any given message $m_1||m_2||\cdot\cdot\cdot||m_l$ in Deoxys-*II*, where we assume the MAC part is constant or NULL, we construct the first group G_1 with the fixed m_1, the second group G_2 with the

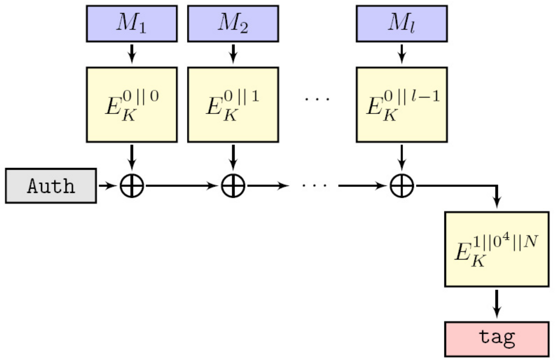

Fig. 12. Message processing in the authentication part of the nonce-misuse resisting mode without padding in Deoxys-II [JNPS]

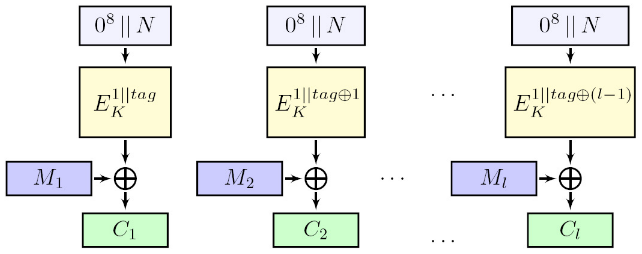

Fig. 13. Message processing in the encryption part of the nonce-misuse resisting mode without padding in Deoxys-II [JNPS]

fixed m_2, and then we should get a collision that $\tau_i = \tau_j$ with high probability, by the birthday paradox.

Then, we query Deoxys-II for the tag τ of the message N, $m'_1||m'_2||m_3||\cdots||m_l$, where the returned tag τ is also valid for the un-queried given message $m_1||m_2||\cdots||m_l$ with probability 1. We also get the corresponding ciphertexts $c'_1||c'_2||c_3||\cdots||c_l$ for $m'_1||m'_2||m_3||\cdots||m_l$.

However, the universal forgery attack is not completed, since the ciphertexts will also be changed for the altered messages $m_1||m_2||\cdots||m_l$. Luckily, we can directly gain the corresponding ciphertexts c_1 and c_2 as $c_1 = m_1 \oplus (m'_1 \oplus c'_1)$ and $c_2 = m_2 \oplus (m'_2 \oplus c'_2)$.

Finally, $c'_1||c'_2||c_3||\cdots||c_l||\tau$ is the universal forgery for the given messages $m_1||m_2||\cdots||m_l$.

The universal forgery attack for Deoxys-II succeeds.

CLOC and SILC CLOC and SILC are designed for adversaries that are assumed to be nonce-reusing. Sine CLOC and SILC are similar, here, we just describe the attack to CLOC. Like Deoxys-II, the message processing of CLOC is also divided into two parts: the authentication part for tag generation and the encryption part for ciphertext. The Message processing in the authentication part and encryption part without padding in CLOC are shown in Fig. 14 and Fig. 15, respectively. Here, we omit the case of messages with padding, which can be processed similarly.

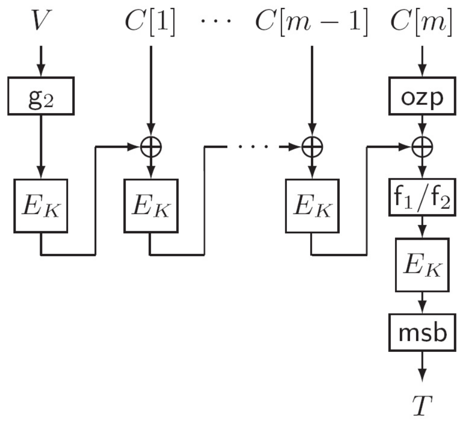

Fig. 14. Message processing in the authentication part of the nonce-misuse resisting mode without padding in CLOC [IMG$^+$]

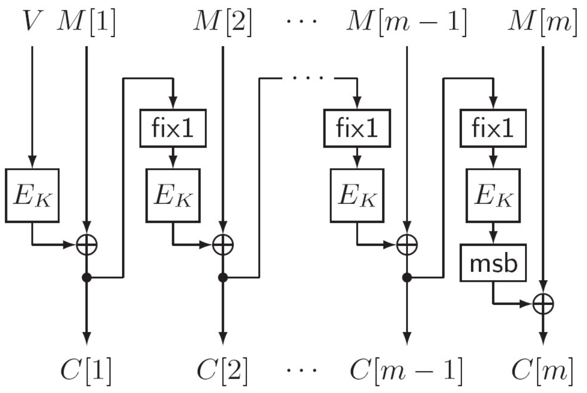

Fig. 15. Message processing in the encryption part of the nonce-misuse resisting mode without padding in CLOC [IMG$^+$]

We note that the input of the tag generation of CLOC, with CBC-MAC like structure shown in Fig. 14, is the ciphertext, instead of plaintext. To implement our universal forgery attack, we have to fix the corresponding values of $C[1]$ and $C[2]$ for the given message $m_1||m_2$. To fix the value of $C[1]$ is easy, we should just use the encrypted output of the given message m_1, by the encryption part for ciphertext shown in Fig. 15. There is more work to do, to fix the value of $C[2]$, even we can not query to the CLOC oracle for $m_1||m_2$'s encryption result. However, we can query $m_1||x$ for its output $C_1||C_2'$, where x can be anything but m_2. Then we can compute $C_2 = (m_2' \oplus C_2') \oplus m_2$, by utilizing the CFB mode.

To implement a generic universal forgery for any given message $m_1||m_2$ in CLOC, where we assume the MAC part is constant or NULL, we must construct two groups as usual. We construct the first group G_1 with the fixed m_1, which means the fixed $C[1]$ in the authentication part of CLOC. However, it is more complicated to construct the second group G_2 with fixed $C[2]$, since we can not directly control the content of the second ciphertext to be $C[2]$. We have to randomly generate a temp message block *temp* every time, and first query $m_1^j||temp$, with the corresponding $C[1]^j||C[2]^{temp}||\tau^{temp}$ returned, we compute the targeted message $m_2^j = C[2] \oplus (C[2]^{temp} \oplus temp)$, where $1 \geq j \geq 2^{n/2}$. Then, we query $m_1^j||m_2^j$, with the corresponding $C[1]^j||C[2]||\tau^j$ returned. So, the total on-line queries in the second group G_2 with fixed $C[2]$ is about $2 \times 2^{n/2}$.

We should get a collision that $\tau_i = \tau_j$ with high probability by the birthday paradox, where $C[1]||C[2]^i$ with τ_i and $C[1]|^j|C[2]$ with τ_j, respectively. Moreover, we note that the messages are $m_1||m_2^i$ with τ_i and $m_1^j||m_2^j$ with τ_j, respectively, and there is no targeted message m_2.

Then, we query CLOC for the message $m_1^j||m_2^i$, and get the corresponding $C[1]^j||C[2]^{ji}||\tau^{ji}$, where $C[2]^{ji} \neq C[2]^i$, and we must modify $m_2'' = C[2]^i \oplus (m_2^i \oplus C[2]^{ji})$ to keep $C[2]^i$ unchanged.

Finally, we query CLOC for the message $m_1^j||m_2''$, where the returned tag τ is also valid for the un-queried given message $m_1||m_2$ with probability 1. We also get the corresponding ciphertexts $C[1]^j||C[2]''$ for $m_1^j||m_2''$, where we should replace the ciphertexts $C[1]^j||C[2]''$ with $C[1]||C[2]$ for the forgery. Finally, $C[1]||C[2]||\tau$ is a valid forgery for the given message $m_1||m_2$.

4 Another generic universal forgery attacks can be applied in the quantum model

4.1 CBC-MAC variants

We recall that existential forgery attacks succeeds with only complexity of $O(n)$ in the quantum model [KLLNP16], here, we use the equation $f_1(x_1) \oplus f_2(x_2) = f_1(x_1') \oplus f_2(x_2')$ further with minor modification, to extend their attack to launch universal forgery attack.

In the following, we show how to use a generic birthday attack with two groups to implement an universal forgery attack for any given message $x_1||x_2||\cdot\cdot\cdot||x_l$, where $l \geq 2$. We recall that in CBC like MAC, $f_1(x) = E_{K_1}(x)$ and $f_2(x) = x$.

Randomly generate $2^{n/2}$ 1-block messages x_2^i in group G_1, where $i \leq 2^{n/2}$, and query $x_1||x_2^i$ to the oracle MAC, there will be $2^{n/2}$ elements of corresponding MAC tags τ_i returned;

Randomly generate $x_1' \neq x_1$, and generate $2^{n/2}$ message blocks x_2^j in group G_2, where $j \leq 2^{n/2}$, and query $x_1'||x_2^j$ to the oracle MAC, there will be $2^{n/2}$ elements of corresponding MAC tags τ_j returned;

There should exist $\tau_i = \tau_j$ for some i, j with high probability, by the birthday paradox. So, we will get the key information that $E_{K_1}(x_1) \oplus x_2^i = E_{K_1}(x_1') \oplus x_2^j$.

Query the message $x_1'||x_2 \oplus (E_{K_1}(x_1) \oplus E_{K_1}(x_1'))||x_3|| \cdots ||x_l$ to the oracle MAC, a corresponding tag τ will be returned.

We note that the tag τ is also valid for the given message $x_1||x_2|| \cdots ||x_l$ with probability 1, which is never queried by the adversary to the oracle MAC. The attack succeeds.

Here, we recall that the existential forgery attack in [KLLNP16] uses random block messages α_0 and α_1 to find the key information $x \oplus x' = f_1(\alpha_0) \oplus f_1(\alpha_1)$. However, we use the fixed x_1 directly to launch universal forgery attack, instead of existential forgery, with the same complexity.

Complexity. We know that the queries made to the oracle MAC are about $O(2^{n/2})$, by the birthday paradox, in the classic setting. The success probability 1 of such universal forgery can also be reached, through enlarging the capacity of the above two groups.

Advantage. Such attack can be implemented by *Simon's algorithm* in the quantum model with complexity $O(n)$, since the equation $\tau_i = \tau_j$ provides the guarantee of hidden periodicity for the collided messages, with $x_2^i \oplus x_2^j = E_{K_1}(x_1) \oplus E_{K_1}(x_1')$.

Drawback. This attack is not applicable to PMAC variants, no inclusion of hidden periodicity will be provided for the messages of PMAC variants.

Direct application to CLOC and SILC The above attack is also directly applicable to CLOC and SILC in the nonce-reuse model, instead of the MAC part, with constant associated data. The application to CLOC and SILC with fixed message $m_1||m_2$ is shown as follows.

Randomly generate $2^{n/2}$ 1-block messages m_2^i in group G_1, where $i \leq 2^{n/2}$, and query $m_1||m_2^i$ to the oracle CLOC or SILC, there will be $2^{n/2}$ elements of corresponding outputs $C[1]||C[2]^i||\tau_i$ returned;

First randomly generate and fix $m_1' \neq m_1$, and then randomly generate $2^{n/2}$ 1-block messages m_2^j in group G_2, where $j \leq 2^{n/2}$, and query $m_1'||m_2^j$ to the oracle, there will be $2^{n/2}$ elements of corresponding outputs $C[1]'||C[2]^j||\tau_j$ returned;

There should exist $\tau_i = \tau_j$ for some i, j with high probability, by the birthday paradox. So, we will get the key information that $\Delta = E_K(C[1]) \oplus E_K(C[1]') = m_2^i \oplus m_2^j$. We should also compute the corresponding ciphertext $C[2] = m_2 \oplus (m_i \oplus C[2]^i)$, to be used in the next step.

Randomly generate 1-block message m_2^{temp}, and query the message $m_1'||m_2^{temp}$ to the oracle, a corresponding $C[1]'||C[2]^{temp}||\tau_{temp}$ will be returned.

Compute $m_2' = C[2] \oplus \Delta \oplus (C[2]^{temp} \oplus m_2^{temp})$, and query the message $m_1'||m_2'$ to the oracle, a corresponding $C[1]'||C[2]'||\tau$ will be returned.

We note that $C[1]||C[2]||\tau$ is also valid for the given message $m_1||m_2$ with probability 1, which is never queried by the adversary. The universal forgery attack to CLOC or SILC succeeds.

4.2 PMAC variants

In the following, we show how to use birthday attack to implement an universal forgery attack for any given message $x_1||x_2||\cdots||x_l$, where $l \geq 2$.

Randomly generate $2^{n/2}$ message blocks $y^i||y^i$, where $i \leq 2^{n/2}$, and query $y^i||y^i$ to the oracle MAC, there will be $2^{n/2}$ elements of corresponding MAC tags τ_i returned, this is the same as in [KLLNP16];

There should exist $\tau_i = \tau_i'$ for some i, i' with high probability, by the birthday paradox. So, we will get the key information that $\Delta_0 \oplus \Delta_1 = y^i \oplus y^{i'}$, where $y^i \oplus y^{i'}$ is known.

Query the message $x_2 \oplus \Delta_0 \oplus \Delta_1||x_1 \oplus \Delta_0 \oplus \Delta_1||x_3||\cdots||x_l$ to the oracle MAC, a corresponding tag τ will be returned. Here, we recall that the existential forgery attack in [KLLNP16] queries the 2-block messages $m||m$ for the existential forgery. However, we use the 2-block message $x_2 \oplus \Delta_0 \oplus \Delta_1||x_1 \oplus \Delta_0 \oplus \Delta_1$ directly to launch universal forgery attack, instead of existential forgery, with the same complexity.

We note that the tag τ is also valid for the given message $x_1||x_2||\cdots||x_l$, which is never queried by the adversary to the oracle MAC. The attack succeeds.

The MAC part of the third round of CAESAR candidates, like AEZ, OCB, OTR, COLM (COPA) and Deoxys (except ELmD), suffer this attack.

Complexity. We know that the queries made to the oracle MAC are about $O(2^{n/2})$, by the birthday paradox, and the success probability 1 of such universal forgery can also be reached, through enlarging the capacity of the above two groups.

Advantage. Such attack can be implemented by *Simon's algorithm* in the quantum model with complexity $O(n)$, since the equation $\tau_i = \tau_j$ provides the guarantee of hidden periodicity for the collided messages, with $\Delta_0 \oplus \Delta_1 = y^i \oplus y^{i'}$. Moreover, this attack is much powerful, once the difference of the secret offsets is recovered, further universal forgery attack can be launched with just one oracle access, with probability 1.

Drawback. This attack can not be applied to such MACs that the computation is related to their position, for example, CAESAR candidate ELmD. This attack is applicable to CBC-MAC variants, neither, since CBC-MAC variants use no secret offset.

Why can not be applied to PMAC variant in ELmD. There should exist 2-block message pair $y_i||y_i$ and $y_j||y_j$ satisfying $y_i \oplus y_j = \Delta_1 \oplus \Delta_2$, which means that $E_K(y_i \oplus \Delta_1) = E_K(y_j \oplus \Delta_2)$ and $E_K(y_j \oplus \Delta_1) = E_K(y_i \oplus \Delta_2)$. However, the inner 2-block computation is $2 \cdot E_K(y_i \oplus \Delta_1) \oplus E_K(y_i \oplus \Delta_2)$ and $2 \cdot E_K(y_j \oplus \Delta_1) \oplus E_K(y_j \oplus \Delta_2)$ by using the ρ function further in ELmD, let

$a = E_K(y_i \oplus \Delta_1)$ and $b = E_K(y_i \oplus \Delta_2)$, the inner 2-block computation outputs will be $2 \cdot a \oplus b$ and $2 \cdot b \oplus a$, respectively. It means that even if such y_i, y_j exists, we can not observe that collision, for the corresponding outputs are totally different.

5 Discussion and Conclusion

This paper discusses how to use an existential forgery attack with birthday paradox to launch a generic universal forgery attack, for the blockcipher-based message authentication codes and authenticated encryptions. Our attacks are also applicable to the third round CAESAR candidates with associated data protected by CBC-MAC or PMAC variants, they are OCB, CLOC, SILC, AEZ, OTR, COLM (including COPA and ELmD) and Deoxys. Our results show immediately that an existential forgery means an universal forgery for so many of blockcipher-based MACs [4], why this happens? We already know that an existential forgery attack for a hash based MAC will not immediately cause an universal forgery attack. For example, the existential forgery attack of HMAC [BCK96] can be launched also by birthday attack, however, an universal forgery attack to such MAC is believed to be an exhaustive search with complexity of 2^n queries. We also note that a variant of HMAC with just one key, named H^2-MAC [Yas09], is too much susceptible to equivalent key recovery attack by using birthday attack with two groups [LXS12], however, the complexity of the universal forgery attack is still 2^n queries, for any given message.

We conclude that the use of XOR greatly decrease the security of such blockcipher-based MACs, and we suggest that if the XOR operation is replaced with modular 2^n, an existential forgery attack will not immediately cause an universal forgery one any more. Hence, we wonder that the design of blockcipher-based MACs and authenticated encryption should be reevaluated.

Acknowledgments

This work was partially supported by Foundation of Science and Technology on Information Assurance Laboratory under Grant 6142112010202.

References

[ABL$^+$a] Elena Andreeva, Andrey Bogdanov, Atul Luykx, Bart Mennink, Elmar Tischhauser, and Kan Yasuda. Aes-copa v.2. caesar submission, september 2016.

[ABL$^+$b] Elena Andreeva, Andrey Bogdanov, Atul Luykx, Bart Mennink, Elmar Tischhauser, and Kan Yasuda. Colm v1. caesar submission, september 2016.

[4] We wonder if such attack is also applicable to Sponge construction, which is similar to CBC mode to some extent.

[ABL$^+$13] Elena Andreeva, Andrey Bogdanov, Atul Luykx, Bart Mennink, Elmar Tischhauser, and Kan Yasuda. *Parallelizable and Authenticated Online Ciphers*, pages 424–443. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.

[ae] Authenticated encryption. retrived april, 2017 https://en.wijipedia.org/wiki/authenticaed_encryption.

[AFF$^+$15] Farzaneh Abed, Scott Fluhrer, Christian Forler, Eik List, Stefan Lucks, David McGrew, and Jakob Wenzel. *Pipelineable On-line Encryption*, pages 205–223. Springer Berlin Heidelberg, Berlin, Heidelberg, 2015.

[BCK96] Mihir Bellare, R. Canetti, and Hugo Krawczyk. *Keying hash functions for message authentication*, pages 1–15. Springer Berlin Heidelberg, Berlin, Heidelberg, 1996.

[BEK16] Aslı Bay, Oğuzhan Ersoy, and Ferhat Karakoç. *Universal Forgery and Key Recovery Attacks on ELmD Authenticated Encryption Algorithm*, pages 354–368. Springer Berlin Heidelberg, Berlin, Heidelberg, 2016.

[BKR94] Mihir Bellare, Joe Kilian, and Phillip Rogaway. *The Security of Cipher Block Chaining*, pages 341–358. Springer Berlin Heidelberg, Berlin, Heidelberg, 1994.

[BM01] Karl Brincat and Chris J. Mitchell. *New CBC-MAC Forgery Attacks*, pages 3–14. Springer Berlin Heidelberg, Berlin, Heidelberg, 2001.

[BPR05] Mihir Bellare, Krzysztof Pietrzak, and Phillip Rogaway. *Improved Security Analyses for CBC MACs*, pages 527–545. Springer Berlin Heidelberg, Berlin, Heidelberg, 2005.

[BR02] John Black and Phillip Rogaway. *A Block-Cipher Mode of Operation for Parallelizable Message Authentication*, pages 384–397. Springer Berlin Heidelberg, Berlin, Heidelberg, 2002.

[BR05] John Black and Phillip Rogaway. Cbc macs for arbitrary-length messages: The three-key constructions. *Journal of Cryptology*, 18(2):111–131, 2005.

[BRR95] Mihir Bellare, Guérin R., and Phillip Rogaway. *XOR MACs: new methods for message authentication using finite pseudorandom functions*, pages 15–28. Springer Berlin Heidelberg, Berlin, Heidelberg, 1995.

[CMN$^+$14] Simon Cogliani, Diana-Ştefania MaimuŢ, David Naccache, Rodrigo Portella do Canto, Reza Reyhanitabar, Serge Vaudenay, and Damian Vizár. *OMD: A Compression Function Mode of Operation for Authenticated Encryption*, pages 112–128. Springer International Publishing, Cham, 2014.

[DK] Hans Delfs and Helmut Knebl. *Introduction to Cryptography: Principles and Applications.*

[DN] Nilanjan Datta and Mridul Nandi. Elmd v2.0, submission to the caesar competition, august 2015.

[fBCMoO05] NIST. Recommendation for Block Cipher Modes of Operation. The cmac mode for authentication. *NIST Special Publication 800-38B*, 2005.

[GD02] Virgil D. Gligor and Pompiliu Donescu. *Fast Encryption and Authentication: XCBC Encryption and XECB Authentication Modes*, pages 92–108. Springer Berlin Heidelberg, Berlin, Heidelberg, 2002.

[GMR88] Shafi Goldwasser, Silvio Micali, and Ronald L. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM Journal on Computing*, 17(2):3281–308, 1988.

[HKR] Viet Tung Hoang, Ted Krovetz, and Phillip Rogaway. Aez v4.2: Authenticated encryption by enciphering. caesar submission, september 2016.

[HKR15] Viet Tung Hoang, Ted Krovetz, and Phillip Rogaway. *Robust Authenticated-Encryption AEZ and the Problem That It Solves*, pages 15–44. Springer Berlin Heidelberg, Berlin, Heidelberg, 2015.

[IK03] Tetsu Iwata and Kaoru Kurosawa. *OMAC: One-Key CBC MAC*, pages 129–153. Springer Berlin Heidelberg, Berlin, Heidelberg, 2003.

[IMG$^+$] Tetsu Iwata, Kazuhiko Minematsu, Jian Guo, Sumio Morioka, and Eita Kobayashi. Cloc and silc v3. caesar submission, september 2016.

[I99] Information technology Security techniques Message Authentication Codes (MACs) Part 1 ISO/IEC 9797C1. Mechanisms using a block cipher. *International Organization for Standardization, Genève, Switzerland*, 1999.

[JNPS] Jérémy Jean, Ivica Nikolić, Thomas Peyrin, and Yannick Seurin. Deoxys v1.41. caesar submission, september 2016.

[JPS03] Antoine Joux, Guillaume Poupard, and Jacques Stern. *New Attacks against Standardized MACs*, pages 170–181. Springer Berlin Heidelberg, Berlin, Heidelberg, 2003.

[JWYX09] Keting Jia, Xiaoyun Wang, Zheng Yuan, and Guangwu Xu. *Distinguishing and Second-Preimage Attacks on CBC-Like MACs*, pages 349–361. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.

[KI03] Kaoru Kurosawa and Tetsu Iwata. *TMAC: Two-Key CBC MAC*, pages 33–49. Springer Berlin Heidelberg, Berlin, Heidelberg, 2003.

[KLLNP16] Marc Kaplan, Gaëtan Leurent, Anthony Leverrier, and María Naya-Plasencia. *Breaking Symmetric Cryptosystems Using Quantum Period Finding*, pages 207–237. Springer Berlin Heidelberg, Berlin, Heidelberg, 2016.

[KR11] Ted Krovetz and Phillip Rogaway. *The Software Performance of Authenticated-Encryption Modes*, pages 306–327. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011.

[LPTY16] Atul Luykx, Bart Preneel, Elmar Tischhauser, and Kan Yasuda. *A MAC Mode for Lightweight Block Ciphers*, pages 43–59. Springer Berlin Heidelberg, Berlin, Heidelberg, 2016.

[LXS12] Fanbao Liu, Tao Xie, and Changxiang Shen. Equivalent key recovery attack to h^2-mac. *International Journal of Security and Its Applications*, 6(2):397–402, 2012.

[Min] Kazuhiko Minematsu. Aes-otr v3.1. caesar submission, september 2016.

[Min14] Kazuhiko Minematsu. *Parallelizable Rate-1 Authenticated Encryption from Pseudorandom Functions*, pages 275–292. Springer Berlin Heidelberg, Berlin, Heidelberg, 2014.

[MT06] Kazuhiko Minematsu and Yukiyasu Tsunoo. Provably secure macs from differentially-uniform permutations and aes-based implementations. *FSE 2006. LNCS 4047*, pages 226–241, 2006.

[Nan14] Mridul Nandi. *Forging Attacks on Two Authenticated Encryption Schemes COBRA and POET*, pages 126–140. Springer Berlin Heidelberg, Berlin, Heidelberg, 2014.

[PR00] Erez Petrank and Charles Rackoff. Cbc mac for real-time data sources. *Journal of Cryptology*, 13(3):315–338, 2000.

[PvO95] Bart Preneel and Paul C. van Oorschot. *MDx-MAC and Building Fast MACs from Hash Functions*, pages 1–14. Springer Berlin Heidelberg, Berlin, Heidelberg, 1995.

[(re86] ANSI X9.9 (revised). Financial institution messages authentication (wholesale), american bankers association. 1986.

[Sim97] D.R. Simon. On the power of quantum computation. *SIAM J. Comput*, 26(5):1474–1483, 1997.

[STA$^+$a] Yu Sasaki, Yosuke Todo, Kazumaro Aoki, Yusuke Naito, Takeshi Sugawara, Yumiko Murakami, Mitsuru Matsui, and Shoichi Hirose. Minalpher v1.1. caesar submission, august 2015.

[Stab] William Stallings. *Cryptography and Network security: principles and practice, fifth edition.*

[X9.86] ANSI X9.19. Financial institution retail messages authentication, american bankers association. 1986.

[Yas09] Kan Yasuda. *HMAC without the second key*, pages 443–458. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.

[Yas12] Kan Yasuda. *PMAC with Parity: Minimizing the Query-Length Influence*, pages 203–214. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.