1 Introduction

1.1 Motivation: CETD use new designed component, the formal security analysis is not provided with the design

Message authentication code(MAC) is a cryptographic system to protect the integrity of message transmitted between sender and receiver. If the content of message blocks are modified or external message blocks are injected, a secure MAC scheme should be eligible to examine the attacks. When using a MAC scheme to protect the integrity, message blocks are adopted to a MAC scheme as input and a short message block, called tag in majority of research works, is generated as output. The tag is concatenated to the input message blocks and this data-tag pair is used as unit in transmission. Early MAC schemes are symmetric-keyed, stateless(secrete key is the only resource of randomness in tag generation) and constructed with block cipher. As stateless MAC schemes suffer from replay attacks, MAC schemes using state information of message blocks protected as additional information in tag generation have been designed.

Hong, Guo and Hu[**?**] introduced a cost-effective tag design for integrity protection, specially for embedded systems. The MAC scheme adopted is an stateful design constructed with two components: bit segments swap and block-level rotate shift. When generating a tag, a nonce generated by encrypting a tuple(message address, counter, random number) is used as additional input in processing the input data for swap and shift components. A security analysis of the MAC scheme adopted was provided by Hong et al. indicating that tag generated is random under brute-force attack. This evaluation procedure contains the following weakness:

- The capability of the adversary is not depicted clearly
- The security result under each type of attack in the threat model is not provided
- The data processing components used in constructing the MAC scheme in cost-effective tag design are not cryptographic primitives, the security assumption of these components are unclear.
- The evaluation procedure does not belong to any common formal evaluation framework. The security result provided can not be used in comparison with the formal results from evaluation of other MAC schemes.

The tag design from Hong et al. showed reduction of on-chip, performance and memory overhead. This cost advantage attracted us to research its security with a formal approach.

1.2 Approach: Analysis the security of CETD with computational model

In this article, we will show that from the viewpoint of computational model based security analysis, the secure block cipher used in generating nonce can not ensure the security of Cost-Effective Tag Design as a message authentication system under several attacks. To make this statement meaningful, we follow the well recognized security notions of message authentication system in computational model.

1.3 Results: found security weakness, security optimization and results for optimized design

2 Preliminary: Formal Approaches of Security Analysis

Threat Model of Data Integrity We use processor-memory system to explain the threat and protection on the integrity of message blocks in transmission. In the scenario of processor-memory architecture, data integrity refers to maintaining the data on the transmission outside the chip and stored on off-chip memory untampered. If the data stored on off-chip memory , or some external data injected to the system S, we can say that the data integrity of S is attacked. If the system S can not examine the tampered data or injected data, the attack is assumed to be successful. According to our knowledge, there are two aspects of attacks on data integrity:

1. Introducing new content: The content of data in the system is modified, or data construct by the attacker is inserted to the system.
2. Replacing data with valid copy: The data D in transmission or on the memory is replace with the following two types of copy: A copy of D at an old time point or a copy of other data from different memory address.

When the processor writes a data block D to the memory, MAC scheme computes a tag and concatenate the tag with D forming a data-tag pair. The data-tag pair is sent to the frame on off-chip memory with address ADDR for storage. When the processor reads a data from the frame of ADDR, the data-pair is sent to the chip and to MAC scheme. Assume the data part in the pair read is D1 and tag part is T1. MAC scheme compute a tag T2 using D1 and T2 is compared with T1. If T1 is identical to T2, D1 is assumed to be untampered. Figure **??** expresses the functionality of MAC scheme in integrity protection. Assume the data-tag pair stored on the off-chip memory with address ADDR is marked as D-T, the possible attacks on MAC schemes includes:

1. Modify the content of D-T on ADDR
2. Insert new data-tag pair to a empty frame on the memory

3. Replace the content of D-T with a copy of data-tag pair from another off-chip memory frame

4. Replay attack: copy data pair D-T on ADDR wroten on early time stamp T1 and replace the data pair on ADDR on latter time stamp with D-T

If a MAC scheme is capable to defend the 1st and 2nd attacks in the list, it should ensure that for any two distinct data D1 and D2, the corresponding tag T1 and T2 should be two random message blocks. If this scheme is capable to defend the 3rd and 4th attacks too, it should ensure for any two identical data blocks D1 and D2 that are written to memory on different time or to different addresses, their tags T1 and T2 should be randomly generated.

Formal Approaches on Security Analysis Since 1980s, two formal approaches have been developed for analyzing the security protocols. "Formal" means means the security notions for a type of protocols is general to all designs and the result of each design in a kind of approach should be in the same format. One approach is named "computational model"in some research works. The computational model based approaches quantify the security of a protocol with probability and complexity. The adversary in computational model cannot acquire the secret information stored in trust area while has no other limitation in breaking the protocol. The security results acquired with computational model based approaches represent attack situations in real world. The weakness of computational model includes manually evaluation in most cases and error-prone. The other formal approach adopted in security analysis is named "symbolic model" in some research works. The cryptographic primitives used in the construction of protocol are assumed to be black box and the adversary can not predict the output of crypto-primitive without knowing the secret information. Each primitive is modeled to a term and represented as a string. The protocol can be represented with the terms to form a sequence of strings and the security result of a protocol has two options: security or insecure.

2.1 Symbolic Model based Security Analysis

Dolev and Yao introduced [?] a evaluation mechanism for analyzing the security of cryptographic protocol in symbolic model(called formal methods in some research works). The security result of a protocol in Dolev-Yao model has two possible value: secure with cryptographic primitive used behaves like black box or an attack breaking the protocol exist.

Symbolic Model Based Security Analysis on MAC Schemes In Dolev and Yao‘s work, only framework of modeling encryption protocols was introduced. The main weakness of Dolev-Yao model pointed by latter research articles is that security assumption on the components used in the protocol is too strong and unreal. The security result drawn with Dolev-Yao model can not clearly indicate the security of a protocol in real application. In [?], Backes, Pfitzmann and Waidner introduced an extension of Dolev-Yao framework to support analyzing authentication systems such as digital signature and MAC schemes. Another progress made by this extension is a bridge from symbolic model to a result in computational model which represent the security of authentication protocol in real world application. Backes et al. analyzed the security of an authentication protocol Kerberos with the framework in [**?**].

As pointed by Boldyreva and Kumar in [**?**], the weakness of the symbolic model based security analysis of Kerberos from Backes et al. is the strong assumption of cryptographic primitives used in Kerberos. This weakness represents the limitation of existing symbolic model based security analysis approaches on authentication systems: the authentication systems analyzed should be constructed with cryptographic primitives meeting strong security assumptions. If the authentication system analyzed is constructed with original designed components, symbolic model based security analysis approaches can not be applied. Detailed survey of symbolic model based security analysis approaches can be referred from [**?**].

2.2 Computational Model based Security Analysis

2.2.1 Security Notion of MAC Schemes in Computational Model

From the viewpoint of computational model, the security of a MAC scheme can be broken if the resources, such as computation time, is unlimited for the attacker. Research works on security analysis on MAC schemes try quantify the evaluation result under the assumption that the attack has limitation on computation resources.

The forgery attacks on MAC Schemes When attacking a MAC Scheme, the adversary try to send a pair(M,T) to the receiver to make VF_k(M,T)=1 while M did not originate with the legal sender. The fake pair(M_f,T_f) that makes VF_k(M_f,T_f)=1 is called a forgery from the adversary. A successful forgery attack indicates that the adversary has made a forgery. The purpose of a message authentication system is preventing the receiver to accept the message from unauthorized senders, such as an adversary. The quantitative property of a secure message authentication system is the low probability for an adversary to make a successful forgery attack with the limited resource.

Chosen-message attacks A strong type of attack that an adversary can conduct on the message authentication system is the adaptive chosen-message attack, marked as uf-cma. When doing uf-cma, the adversary chooses its own input message M and acquires the relative tag T. The adversary try to find the weakness in the design of message authentication system by analyzing the pairs(M,T) of his choice. The uf-cma provides the adversary with the most capability to succeed in the forgery attack. The probability that an adversary conducts a successful forgery attack after limited times of uf-cma is adopted as the basic quantitative security property of a message authentication in cryptography. This fact was also mentioned in [?].

The Security Notions of MAC schemes The formalised quantitative notion of the security of a MAC scheme was introduced by Bellare et al. in [?]. This notion follows the security notion of digital signature introduced in [?]. The successful forgery on a MAC scheme from an adversary A is measured by a experiment called Forgery(MAC,A). In Forgery(MAC,A), the adversary A is provided a black-box access to the tag generation system $\mathrm{TG}_K()$. When $\mathrm{TG}_K()$ takes an input message M_i, it returns tag T_i to A. A conducts uf-cma by keep sending the message queries M_i and observes the relative tag T_i for limited times. On the other hand, A is provided a black-box access to the verification system $\mathrm{VF}_K()$. When A sends a pair(M_j,T_j) to $\mathrm{VF}_K()$, the $\mathrm{VF}_K()$ computes the tag T of M_j and compares T with T_j. If T=T_j then $\mathrm{VF}_K()$=1 otherwise 0. If A sends a pair(M,T) that makes $\mathrm{VF}_K()$ outputs 1 while M has not appeared in the previous queries of uf-cma, then A succeeds a forgery attack and Forgery(MAC,A)=1.

The quantitative security notion of a MAC scheme is forgery probability, expressed as $\mathrm{Forgery}_{MAC}$=Pr[Forgery(MAC,A)=1].

The Correlation between Security and Randomness As ideal random value generator is impossible to implement in real world, research works on constructing high quality random function in real world has been introduced since 1980s. Goldreich, Goldwasser, and Micali introduced the concept of pseudorandom function(PRF) in [?] indicating that a PRF is a function whose behaviour cannot be distinguish from a ideal random value generator by an adversary with some resource limitation. They also asserted that any good PRF is a secure MAC scheme under the quantitative security notion. Bellare, Kilian and Rogaway proved this assertion in [?] saying that if a system behave like a PRF, this system is a secure MAC scheme. Based on this converted security notion of MAC schemes, latter researches on computational model based security evaluation of MAC schemes posted their focuses on analyzing whether the MAC scheme evaluated behaves like a PRF.

The Randomness of a MAC scheme The definition of PRF was introduced in [?] indicating that PRF could not be distinguished from a ideal random function each bit of whose output was a coin flip. To define how closely a MAC scheme behaves like a PRF, Bellare et al. provided a quantitative notion in [?] named $\mathrm{Adv}^{PRF}_{MAC}()$, which was based on the concept of distinguisher introduced in[?].

Let F_0 and F_1 be two function with a common domain D and a common range R. A distinguisher A for F_0 versus F_1 is an adversary A that has access to a black box named oracle f:D$\longrightarrow$R. After accessing the oracle f, A computes a bit. Assume the function stored in the oracle f is X and A guesses that X is in the oracle, then A computes 1 otherwise 0. The advantage of A in distinguishing F_0 from F_1 is expressed as Adv_{F1}^{F0}=Pr[f$\xleftarrow{R}$F0:A^{F0}=1]-Pr[f$\xleftarrow{R}$F1:A^{F1}=1]. Pr[f$\xleftarrow{R}$F0:A^{F0}=1] means when the content of oracle f is F0, A guesses that F0 is in oracle then output 1.

We can see that if F0 behaves much like F1, it is hard for A to distinguish between F0 and F1 then Adv_{F1}^{F0} is very small. This case is adopted by Bellare et al. in the quantitative notion of randomness of a MAC scheme. If the randomness of a MAC scheme is good, then the MAC scheme behaves like a PRF and $\mathrm{Adv}_{MAC}^{[PRF]}$ is small.

Code-based Game-playing Proofs In cryptography, a viewpoint of game-playing proofs represents the technique that abstracts the interaction between the adversary and environment to a program named game. Computing the probability of adversary becomes the stepwise refinement of a sequences of games. The application of this viewpoint of game-playing proofs in security analysis began with the article[**?**] from Goldwasser and Micali, and[**?**] from Yao and then adopted in the security analysis of various encryption systems like[**?**] and authentication systems like [**?**].

In [**?**], Kilian and Rogaway modeled a game with disciplines to a piece of code and introduced the idea of code-base game-playing proofs. The application of code-based game-playing proof on analyzing authentication systems origins from CBC-MAC, provided by Bellare and Rogaway in [**?**]. Researchers then adopt code-based game-playing proofs to analyzing various MAC schemes and authenticated-encryption schemes, like [**?**].

In the concepts of game-playing proofs from [**?**], a game is program. When using code-based game-playing proofs to analyze the security of a MAC scheme, the MAC scheme is conceptualized to a game program and each step represent an operation in the MAC scheme. This game representing the MAC scheme is marked as G0. The PRF with same range and domain to the MAC scheme is conceptualized to a game G1. Based on the depiction in [**?**], some steps in G0 are can set the signal named "bad" from initialized status(false) to "true". The codes expressing G0 and G1 are exactly identical except the steps following the operation of setting "bad" signal to true. That means the adversary observing G0 and G1 cannot make distinguish until bad signal is set, which will effect the behaviour of outputs from two games. After G0 and G1 are modeled, the security notion of MAC scheme, the distinguishing probability, is computed by quantify the probability that bad signal is set to true in some attack conditions. The procedure in game-playing proofs to compute Pr[bad is set to true] is conducted by modifying the game each time to form a new game with less steps setting bad signal. The end of this reduction of games is the game in which pr[bad is true] can be easily computed manually.

The obvious benefit brought by code-based game-playing proofs is the simplification of computational model based security analysis. Computing the bound of distinguishing probability of an adversary for a MAC scheme and an ideal random function is converted to the computation of probability that "bad" signal is set. The procedure of security analysis in computational model is formalized and can be conducted automatically with programing language.

Based on our analysis, the prerequisite of adopting code-based game-playing proofs to analyze the security of a MAC scheme is that the scheme analyzed is constructed with components whose behaviour have been analyzed in cryptographic level. There should be at lease one kind of component in the MAC scheme has strong randomness, such as ideal block cipher or one-way permutation. This prerequisite ensure the properties of game-playing technique:

- If the bad signal is not set to true, the behaviour of outputs from G0(MAC Scheme) should be exactly identical to the outputs from G1(ideal random value generator).
- It is concrete to define the steps setting bad signal
- When bad signal is set to true, the behaviour of next step operation in G0 the affection on final output should be predictable.

If the MAC scheme is constructed the components that be behaviour of at least one of them is not systematically analyzed , at least one of the above properties can not be met, then code-based game-playing proofs technique is not capable to work.

Automated security analysis for authentication system origin from Blanchet and Pointcheval[**?**]. Their automated security analysis system is designed based on game-playing proof technique and games are modeled with programming language. As game-playing technique cannot be applied on the MAC schemes constructed without cryptographic primitives, to our knowledge the existing automated security analysis tools have be used only on stateless MAC schemes as they are constructed with ideal block cipher.

Analysis approach for the MAC scheme in Cost-Effective Tag Design As the MAC scheme in Cost-Effective Tag Design is constructed with bit segment swap and block-level rotate shifting,nonce of which is a cryptographic primitive and has not been formal analyzed, the symbolic model based security analysis approach and code-based game-playing proofs can not be applied. We adopt the security notion of MAC schemes in computational model to do manually analysis.

3 Related Works

3.1 MAC Schemes and Security Analysis

Stateless and Stateful MAC schemes To our knowledge, existing MAC schemes can be classified to two categories: stateless MAC schemes, which use only a key as secret information to process the input and generate the tag; and stateful MAC schemes, which adopt other secret information blocks(such as nonce in some designs) besides the key in input message processing and tag generation. In some research works, stateless MAC schemes are denoted as deterministic MAC schemes as for a fixed key, each input message is mapped to a tag, which for stateful MAC schemes the message can be mapped to different tags with variant values of nonce. The category of stateless and stateful MAC schemes was also mentioned in [**?**].

Deterministic MAC schemes cannot defend against replay attack. The reason is that the secrete key used in generating the tag will not be refreshed for each message. If two message are identical, their tags are identical. To address this issue, nonce is introduced to the tag generation in a MAC scheme. A nonce is a short message block whose value will be updated by each message in tag generation. To update the value of nonce, the address of message, a random number, a counter or the combination of these three elements are adopted in the nonce generation for each message. With the nonce introduced, a well designed stately MAC scheme should be capable in defending type 2 attack in threat model.

Iterated and Parallel MAC Schemes In iterated MAC scheme, the process of a block in message M relies on the output of the process of its previous block. On the other hand, a block in message M can be processed only after its previous block finishes processing. The research on iterated MAC schemes started from

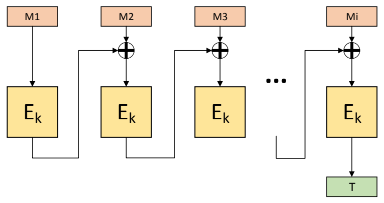


Figure 1: The Original CBC-MAC

the original CBC-MAC expressed in Figure 1.

One performance weakness of iterated MAC schemes is that the processing of each input block relys on the complete of the processing of the previous block. This weakness leads to the latency of tag generation when the number of input blocks is large and becomes more and more obvious with fast development of parallel compting and multi-processor technology. Facing this weakness, attempts have been made to design parallel MAC schemes. In parallel MAC scheme, all the blocks in input message M are processed in parallel. The output blocks of M processing are transformed to a single block B, then processed to generate tag T.

3.1.1 Iterated MAC Schemes

Original CBC-MAC The concept of CBC-MAC was proposed in [] and revised in 1986. The original CBC-MAC is a secure MAC if every input message has same number of blocks. If the number of blocks for two distinct inputs are different, CBC-MAC may produce two identical tags. One known attack on original CBC-MAC is that the first input message contains a single block M, while the second input message contains two blocks M and (M⊕T). Assume the tag T of message M from CBC-MAC can be expressed as T = CBC-MAC$_k$(M), for another input (M, (M⊕T)), the tag T1=CBC-MAC(M, (M⊕T))=T. Another weakness of original CBC-MAC is that only the input whose length is dividible to a block length n can be adopted to generate a tag. The original CBC-MAC did not provide a rule to deal with the last input block if its length is less than n.

Besides the vulnerability when the input length is not fixed, the original CBC-MAC scheme suffers birthday attack which means the adversary needs only $2^{n/2}$ input queries to succeed a forgery.

Bellare et al. provided the first security evaluation on original CBC-MAC[**?**]. The conclusion is that if the adversary A is allowed to conduct arbitrary fixed length queries for q time, the probability that A succeeds in a forgery after the queries can be expressed with the number of queries q and the upper bound of this probability is tight if the block cipher used in CBC-MAC show good randomness.

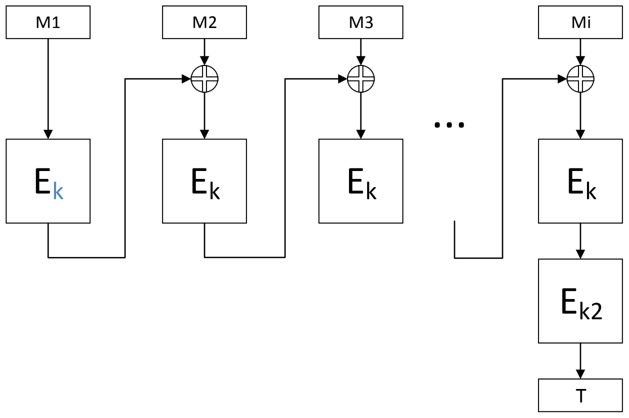


Figure 2: EMAC

EMAC: CBC-MAC for Arbitrary Number of Input Blocks EMAC is a optimized version of original CBC-MAC providing security for arbitrary number of input blocks. EMAC was proposed in the report of RACE project in []. In the EMAC scheme the original CBC-MAC is applied to process input blocks using key K1 and produces a temporary output named T$_{tmp}$. T$_{tmp}$ is encrypted with a different key K2 and the output of encryption is the final tag T. EMAC does not limited the number of blocks for each input, while the num of bits of each input should divisible by the block block length n. Like original

CBC-MAC, EMAC did not provide rule to deal with the last input block if its length is less than n. The concept of EMAC is expressed in Figure 2. EMAC can also be expressed as E_k2(CBC-MAC_k1(M)).

Petrank and Rackof provided the original security evaluation of EMAC in [**?**] showing that the EMAC is secure for arbitrary number of input blocks if the block cipher shows good randomness. However, the EMAC still suffers the birthday attack.

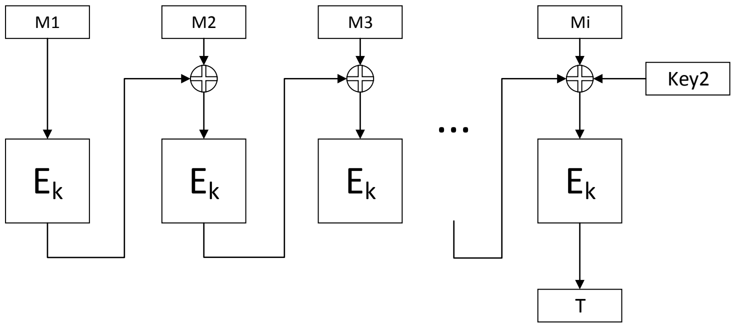


Figure 3: XCBC-MAC no padding

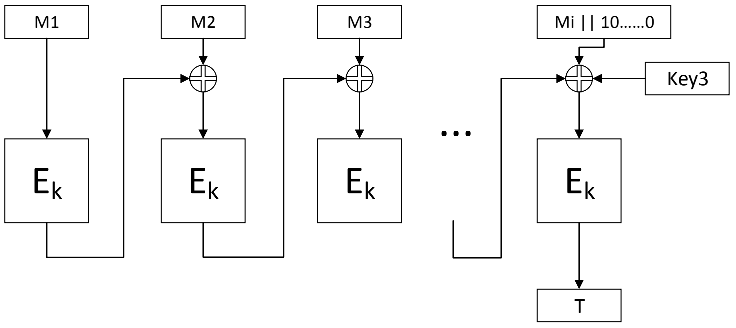


Figure 4: XCBC-MAC padding

CMAC(Cipher based MAC) is another variant of original CBC-MAC. The motivation of CMAC origins from two weaknesses existing in EMAC:

- The length of input must be dividible to block length n
- Assume there m input blocks, there are totally m+1 block cipher callings to generate a tag, including m blocks to compute CBC-MAC_k1(M) and another E_{k2} to generate the final tag.

The structure of CMAC schemes can be expressed as xoring a special value to the input of last block cipher calling in original CBC-MAC. Figure **??** expresses the concept of CMAC schemes.

XCBC: Three Key Version of CMAC Black and Rogaway introduced the first version of CMAC called XCBC in [?], which required three distinct keys to cover all types of inputs. There are two motivations of XCBC scheme: providing security for arbitrary length input; eliminate unnecessary padding operation on the input blocks in EMAC. The concept of XCBC scheme is expressed in Figure 4.

The contributions of XCBC scheme compared with the EMAC are: the length of input message is not limited to the integer times of the length of block cipher input; the block ciphers used in processing m input blocks is m other than m+1 in EMAC; all the block cipher in XCBC share a same key, the additional two keys are used to do XOR operation with the final input block. As one block cipher calling is eliminated, XCBC has performace advantage to EMAC.

Black and Rogaway provided a original security analysis based on computational approach of XCBC scheme in [?]. Their quantitative result was improved by Minematsu and Matsushima in [?].

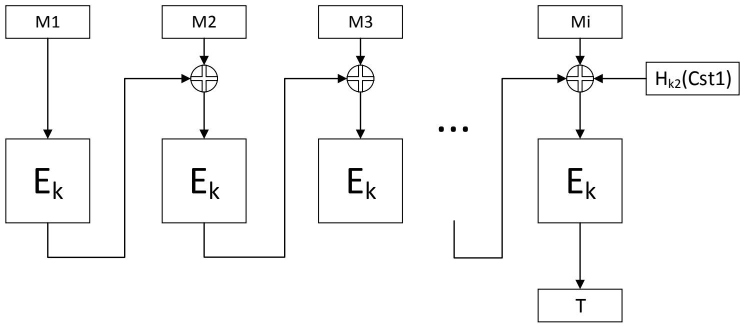


Figure 5: TMAC: no padding

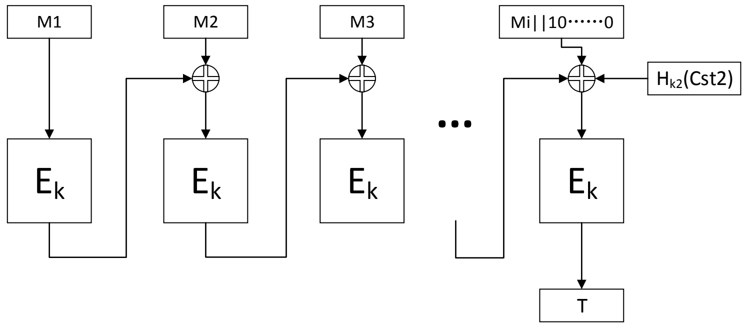


Figure 6: TMAC: padding

TMAC: Two Key Version of CMAC Kurosawa and Iwata designed a optimized version of CMAC named TMAC in [?] requiring two different keys. In TMAC, the value used to XOR with the input of final block cipher calling is replaced from the secrete key k2 and k3 themselves to a hash value generated with k2. If the last input block has n bits, constant number c1 is used to generate the hash value, otherwise c2 is adopted. This optimization aims to reduce number of the keys in XCBC. As mentioned by the authors, generating a new key usually requires an additional calling of block cipher and additional temporary storage. This fact means reducing the number of distinct keys used in tag generation can reduced the cost and enhance the performance in distinct key preparation.

Kurosawa and Iwata provided an original security analysis on TMAC in [?] showing that the same level of security was achieved by TMAC compared with XCBC. The original quantitative security conclusion of TMAC was improved by Minematsu and Matsushima in [?].

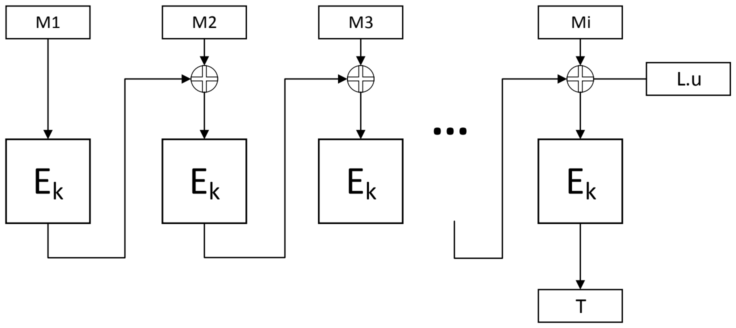


Figure 7: OMAC: no padding

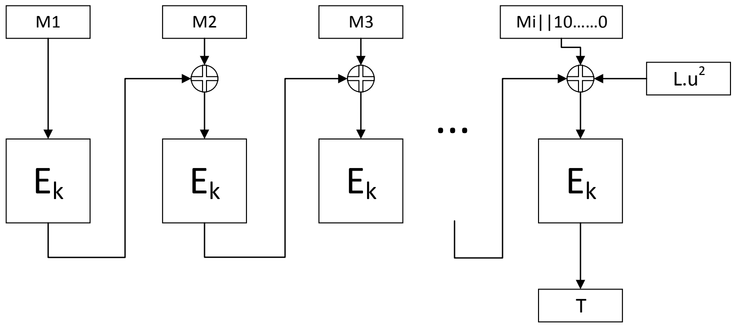


Figure 8: OMAC: padding

OMAC: Single Key Version of CMAC Iwata and Kurosawa introduced a optimization of original XCBC scheme requiring only one key, named OMAC in [**?**]. This key is used by the block cipher in the OMAC and the keyed hash function used in TMAC is replaced by a result of Galois field multiplication. OMAC is the variant of CMAC using the least number of keys. If the length of last input block is n bits, the second key K2 in XCBC is replaced with L·u. L is the result of encrypting constant value 0 with block cipher and key K. L can be expressed as $\mathrm{E}_K(0)$, and u is a non-zero constant. If the length of last input block is less than n, the third key K3 in XCBC is replaced with $\mathrm{L}\cdot\mathrm{u}^2$. The reason to reducing the num of distinct keys from two in TMAC to one in OMAC is reducing the cost and latency in new key generation.

Iwata and Kurosawa provided an original security analysis on OMAC in [**?**]. This result have no optimized version yet.

GMAC McGrew and Viega introduced the Galois/Counter Mode authenticated encryption design and the original security analysis is provided in [**?**]. The message authentication system in GCM(named GMAC) is a iterated scheme based on the concept of universal hashing. The block processing component used in GMAC is the Galois field multiplication other than the block cipher in deterministic MAC schemes. The motivation of GCM is to design a scheme combining counter mode of operation(CTR) with a message authentication code(the GMAC) to form a efficient and secure authenticated encryption system. One advantage of adopting the Galois field multiplication in GMAC is that it can be made easily in hardware and has efficient performance in software.

In the security analysis of GCM from McGrew and Viega, the GMAC was not discussed alone. The security of GMAC as a message authentication code is based on the fact that the encryption part in GCM is secure and the collision probability of ciphertext blocks is low. In GCM design, the nonce is called initialization vector (IV). For each invocation of GCM, the IV should be distinct.

In [**?**], Iwata, Ohashi and Minematsu provided an optimised analysis for the security of GCM. They pointed out the weakness in the lemma used in forming the bound of the quantitative result of security for GCM with a counter example. This counter example was developed to a distinguishing attack on GCM. Iwata et al. then provided a approach fixing the problem of original lemma and provided the new quantitative security result of GCM.

In [**?**], Rogaway pointed the weakness of GMAC as a MAC scheme when used alone that the security under adaptive chosen-message attack is not as good as the security of deterministic MAC schemes. On the other hand, GMAC requires the nonce for tag generation and this nonce should be maintained and refreshed by the user of GMAC. The potential issue of this design pointed by Rogaway is the reuse of nonce, which may lead to the collision of the tag.

Handschuh and Preneel introduced key-recovery attacks on universal hash function based MAC algorithms in [**?**]. They introduced two types of attacks: weak key finding and partial information leaks.

Other Iterated MAC Designs Daemen and Rijmen introduced a iterated MAC design named ALRED-MAC in [?]. The motivation of of this work is to provide a design with tighter upper bound in birthday paradox. Yuan et al. analyzed the security of ALRED-MAC and proved its weakness under birthday attack in [].

3.1.2 Parallel MAC Schemes

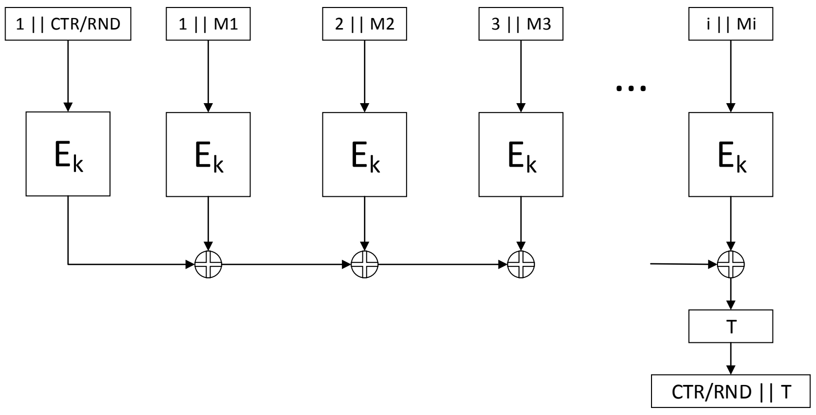


Figure 9: XOR MAC

XOR MAC scheme Bellare, Guerin and Rogaway introduced a parallel MAC scheme named XOR MAC in [?]. The parallel input processing makes the xor MAC faster the CBC-MAC. In [?], the authors provided a quantitative security conclusion of xor mac that it is security under the security notion of message authentication system if the pseudorandom function used in the design is secure. The security bound of xor mac from [?] expressed a tighter upper bound compared with the result of original CBC-MAC.

The weaknesses of XOR MAC are categorized as below:

- Only half of bits in a input of block cipher come from a input data block. That means when generating a tag for a m n-bit input data blocks, the number of block cipher calling to processing input data is 2m, which is more expansive compared with iterated MAC schemes
- Besides the 2m block cipher calling for input data blocks processing, an additoinal block cipher calling is adopted in encrypting a value concatenated by constant 1 and a incrementing counter or random number. That means in XOR MAC, the total number of block cipher calling to generate a tag is 2m+1.

- The tag of XOR MAC is a concatenation of the counter or the random number and the output of the MAC. This design increases the cost of tag storage.

The main weakness of XOR mac is the cost of storage for additional input parameters. According the design of XOR mac, assume the input message is divided into m blocks and the length of each block is half of the length of block cipher input. Each input block has a index with domain as [1,m]. A input block is the concatenated with the encoder of its index(whose length is same as the input block) to form the input to a block cipher, for example (i||M[i]). Besides the m concatenated input blocks, a nonce formed with random number or counter is used a input to the block cipher. Then the m+1 output blocks from block cipher is xored and the output block of XOR operation is concatenated with a random number or counter to form the final tag. The concept of XOR mac is expressed in Figure **??**.

We can see that the length of tag from XOR-mac is the sum of the length of a output block from block cipher and the length of a random number or counter. This long-length MAC needs additional storage compared with the tag from iterated MAC schemes. On the other hand, the nonce maintained by the user of XOR mac is regarded a disadvantage by some researchers.

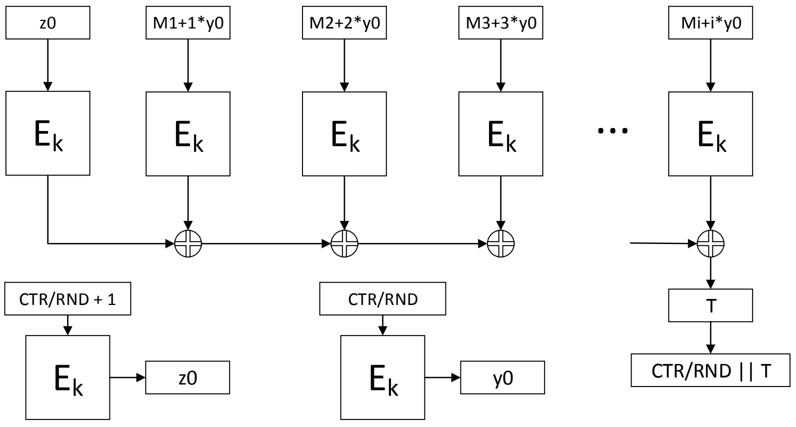


Figure 10: XECB MAC

XECB-MAC: optimized XOR MAC The main weakness of XOR MAC is that each block cipher calling encrypts only n/2 bits of input. This property doubled the num of block cipher calling in tag generation compared with iterated MAC schemes. To address this problem, Gligor and Donescu introduced an optimization named XECB-MAC in []. In XECB-MAC design, each block cipher calling will encrypt the output of the equation M_i + i * nonce, where i is the input block index and M_i is the ith input block. Nonce is the encryption of a nonce input, such as a random number of a incrementing counter. Marking the output of encrypting M$_i$ + i * Nonce, the tag is generated as T = E$_k$(z0) $\oplus sum_{i=1}^{m}$($\oplus$E$_k$(M$_i$ + i * Nonce)) , where $sum_{i=1}^{m}\oplus$ represent the result of xoring m blocks and z0 is the encryption of a simple function of nonce input. The tag and the nonce input are concatenated as the final tag and transfer with the input data.

The main contribution of XECB-MAC is reducing the num of block cipher calling to processing input blocks from 2m to m. The weaknesses of XECB-MAC are categorized below:

- There are m+3 block cipher calling to generate a tag: m+1 for processing input blocks and an additional random block z0; one for nonce generation and one for z0 generation. This design requires 3 additional block cipher calling compared with CMAC variants.
- The final tag is the concatenation of nonce input and the output of XOR. This format of tag is same as the one of XOR MAC and is not efficient in cost.
- In XECB MAC, the length of input should be dividiable to n and there is no rule of padding the final input block when its length is less than n. That means XECB MAC is not applicable to arbitrary length input.

Gligor and Donescu provided the original security analysis of XECB-MAC based on computational model in [].

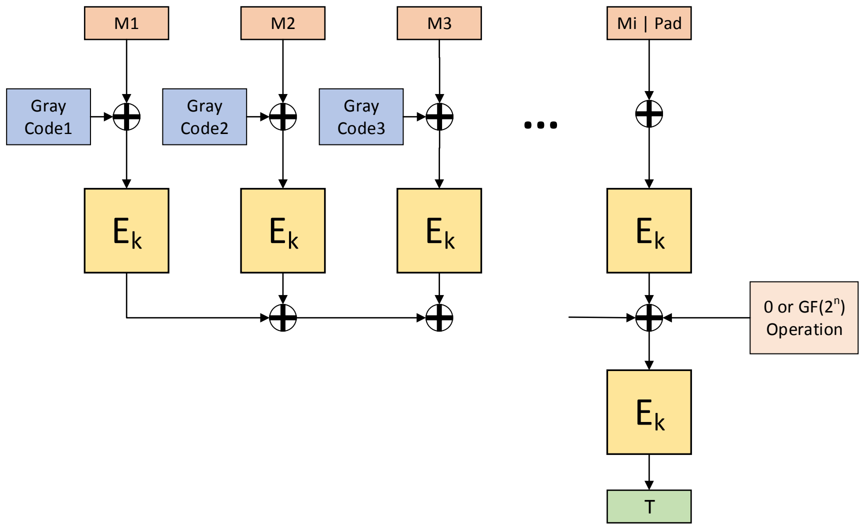


Figure 11: The Original PMAC

The Original PMAC As XECB-MAC is not efficient in both cost and performace and does not support arbitrary length input, Black and Rogaway introduced a parallel MAC scheme named PMAC in [?] to provide an optimization parallel MAC scheme. In PMAC, there are m-1 block cipher calling to process the first m-1 input blocks. Before encryption, each input block is xored with $\mathrm{r}_i\cdot\mathrm{L}$, where r_i is an incrementing grey code and L is $\mathrm{E}_k(0)$. The m-1 output blocks of block cipher are xored to one block, then xored with the padded final input block M[m] to generat a single block marked as T_{tmp}. M[m] is maintained unmodified if its length is n and padded with bit segment 10*, where the num of 0s is n-1-len(M[m]). T_{tmp} is xored with n-bit all-zero block if len(M[m]) or $\mathrm{L}\cdot\mathrm{x}^{-1}$. The output of xor is encrypted to generate the final tag T. The concept of original PMAC scheme is expressed in Figure 11.

The main advantages of original PMAC compared with previous parallel MAC schemes include reducing the number of block cipher calling in PMAC to m, the elimination of the nonce input concatenated with the tag and the padding rule to support arbitrary length input.

The quantitative security result of PMAC was acquired by Black and Rogaway in [?]. They analysed the probability of internal collision in the arbitrary types of inputs and asserted that the PMAC design is a secure MAC if the block cipher used behaves like a pseudorandom permutation. In the proof from Black and Rogaway, the probability that an adversary can distinguish the PMAC from pseudorandom function is the probability that collision of internal blocks(the outputs of block cipher) plus the probability of tag collision.

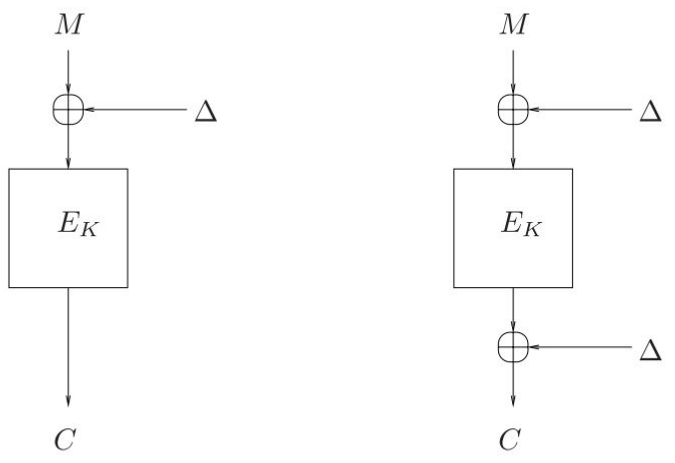
Lee et al. posted attacks on the original PMAC in [?]. Their research indicated that the original PMAC design did not bring advantage in security compared with iterated MACs. The original PMAC suffers from their birthday like attack and the key recover attack.

Tweakable block cipher and Optimized PMAC The concept of tweakable block cipher was formally defined by Liskov, Rivest, and Wagner in [?], converting the block cipher from using only one secret information key to two blocks Key and a tweak. The tweak is supplied by the caller of block cipher.

Rogaway introduced a efficient implementation of tweakable block cipher in [?]and adopted this implementation to replace the block cipher used in the original PMAC and OCB authenticated encryption mode. The concept of Rogaway's tweakable block cipher is expressed in Figure **??**.

The advantage introduced by constructing PMAC with tweakable block cipher include easier implementation and simpler security analysis. In tweakable based PMAC, input block processing is packaged into a tweakable block cipher. This packaged processing simplify the structure of PMAC compared with the original one in which a input block is processed by xoring with $\mathrm{r}_i\cdot\mathrm{L}$ then encryption. This simplication of structure also reduces the complexity of computational based security analysis on PMAC. The concept of PMAC based on tweakable block cipher is expressed in Figure 13.

In [?], Rogaway provided an assertion that the tweakable block cipher XE



$$\Delta = \alpha_1^{i_1}\alpha_2^{i_2}\cdots\alpha_k^{i_k}\ \mathsf{N}$$
$$\mathsf{N} = E_K(N)$$

Figure 12: Tweakable Block Cipher

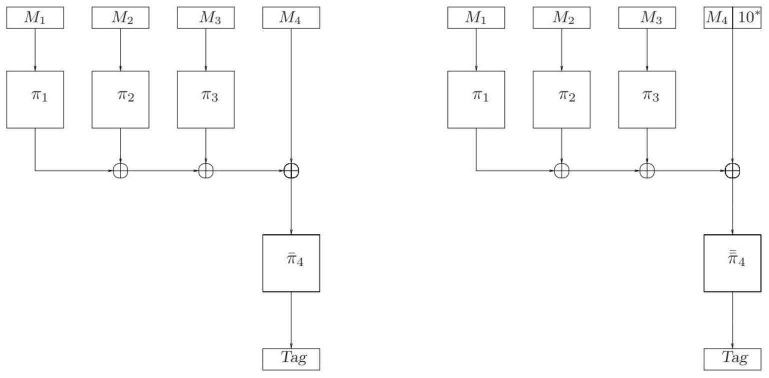


Figure 13: Tweakable Block Cipher

and XEX behave like a pseudorandom permutation if the block cipher inside is a pseudorandom permutation. Based on this assertion about security of tweakable block cipher, Rogaway gave an quantitative conclusion that the new PMAC based on tweakable block cipher is security under the security notion of message authentication system. This quantitative security result of tweakable block cipher based PMAC was provided in [?]. Minematsu and Matsushima optimized this result in [?].

The evaluation procedure in [?] is questioned by the latter researchers hence the correlation between collision probability and distinguishing probability was not explained . This issue was also pointed in Nandi's improved analysis on PMAC in [?]. Nandi also provided a improved quantitative security result of PMAC which is smaller in all the cases than the one in [?] and [?].

Other PMAC variants Sarkar introduce an optimisation on original PMAC in [?]. The galois field multiplication used in the tweakable block cipher introduced is replaced by a technique named tower field representation of Galois field. This replacement enhance the processing speed in software. Hence the structure of iPMAC is same as the original PMAC except the input masking stage, the security evaluation by the author in [?] followed the approach in [?].

3.2 Authentication-Encryption Schemes and Security Analysis

After various kind of MAC schemes were evaluated and claimed to be secure, a new cryptosystem, the Authenticated Encryption(AE) was introduced The Authenticated to combine the encryption of plaintext blocks and generation of the MAC in a single scheme to provide both confidentiality and integrity. According to the analysis of by Bellare and Namprempre in [?], AE schemes can be categorized as Encrypt-then-MAC, Encrypt-and-MAC and MAC-then-Encrypt three structures. Encrypt-then-MAC was proved to be most secure structure and can defend most kinds of attacks. In Rogawa's works [?] a systematical analysis about AE schemes using associate data was expressed.

Based the notion of security evaluation for AE systems introduced in [?], the security of several AE schemes in Encrypt-then-MAC mode have been analyzed systematically, including CCM [?]based on CBC-MAC,EAX[?] based on OMAC, GCM [?] based on universal hashing(new bound revised in [?]), and OCB[?] based on PMAC(new bound revised in [?, ?]). Kasper and Schwabe introduced a implementation of AES that run faster than previous AES block cipher and adopted this implementation to construct a faster AES-GCM AE scheme[?].

Attacking the Authentication Systems There were research works on attacks to existing authentication system or AE systems, such as [?, ?, ?]. Some of these works recover the security weakness in the design works while some other works depict attacks that are out of the security bound of system analyzed.

3.3 Memory Integrity Protection Designs

Concepts of Memory Integrity Protection After various kinds of MAC schemes and authenticated-encryption schemes are evaluated systematically, researchers have paid effort on designing memory integrity protection(MA) systems to protect data integrity from or to the cache on-chip. In the design of MA systems, the on-chip data is assumed to be inaccessible and secure, and the chip is called trusted area; while the data in the transmission and stored in off-chip memory is considered to be accessible to attackers and vulnerable. Transmission path(such as bus) and off-chip memory are called malicious area. Figure 14 expresses the concept of MA system for chip-memory communication.

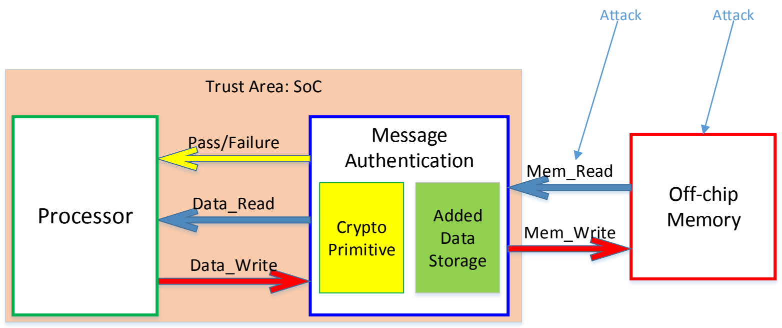


Figure 14: The Concept of MA System

When processor writes a data block, marked as D, to and address, marked as addr, on the memory, processor sends D to MA system and output a message block, marked as message(D, add$_B$), where add$_B$ represents an additional data block. The message block message(D,add$_B$) can be the data block D only, D concatenated with a short block(commonly called tag) or other format of D. The content of message(D,add$_B$) is determined by the cryptographic primitive adopted by the MA system. When the processor wants to read the data D from address addr, the message(D,add$_B$) is read to SoC and sent to MA system first. After verified by MA, if the data D is not tampered, MA sends a signal indicating pass to processor and send the D to the processor, where D is extracted from message(D,add$_B$) block read from memory. If the MA system finds that the D is tampered, MA sends a failure signal to processor, and invoke the exception handling procedure.

Security Evaluation for MA Systems Early research works on authentication systems have indicated that encryption can only protect the confidentiality of data from and to the cache. Authentication Primitives(AP) should be adopted on the chip to process the data with additional information block, such as secret key or nonce.

According to our knowledge, existing theoretical authentication model(AP designs) includes cryptographic hash function, MAC scheme and Added Redundancy Explicit Authentication[**?**]. Message from cache is processed with an AP design on chip and the format of data sent to memory varies according to the AP design adopted. For example, the data to memory is ciphertext blocks if the on-chip AP design is hash function, and the data format is [ciphertext | tag] for MAC scheme, and E_k[plaintext | nonce] for AREA. According to our analysis, the procedure of evaluating a MA design should be the following steps: The first step to evaluate the security of a MA system is figure out the AP system adopted and additional information needed then check whether the MA system meets the needs. For each kind of AP design, there are requirements on the inputs to the AP system. For example, a MAC scheme accepting a nonce requires the nonce to be distinct to each data that invoke the MAC scheme. This requirement ensure that the scheme can defend two types of attacks in the threat model. The MA system design based on such a MAC scheme should ensure that for each data write to the off-chip area from cache, the related nonce should be distinctive. Otherwise, the MA system design is not secure.

If the MA system design meets the requirements of inputs for the AP system adopted, it is necessary to figure out what information is stored in off-chip area and what else are on-chip. Using MAC scheme as instance, the nonce and secrete key in tag generation should be inaccessible to the attacker. The second step to evaluation the security of an MA system is examining the whether the secret information can be acquired by the adversary. If there is some information that can be adopted by the attacker to break the authentication system, the MA design can be assume not secure.

The third step to evaluate the security of a MA system is analyzing the security of AP adopted. If the AP system is a novel design without previous systematically analysis, it is necessary to do the analysis with a evaluation mechanism, such as computational model based on provable security theory.

Some MA systems are constructed based on authenticated-encryption schemes to provide protection on both confidentiality and integrity of data. The security analysis of integrity should also following the above three steps.

Uni-processor MA Systems In [**?**], Yan et al. proposed a MA design based on a variant of Galois/Counter Mode(GCM) authenticated encryption scheme proposed by McGrew and Viega in [**?**]. A data block to be protected is spliced to several chunks each of whose length is same as the length of input to block cipher. For each chunk, a tuple(address, counter, random value(EIV)) is generated as chunk nonce for encryption. For the whole data block, a tuple (address of chunk 1, counter, random value 2(AIV)) is generated for tag generation. In the design from Yan et al., the counter for a data block is formed by two parts: a long-length major counter(64 bits) concatenated with a short-length minor counter(no more than 8 bits). When the minor counter overflow, the major counter updates. For each data block, the counter is distinct. For each chunk in a data block, the address is distinct and the counter is same. The frequency of updating EIV depends on the security level required by the whole system. In this way, the nonce for each chunk and the whole data block is update for each invocation of GCM scheme. As the security of GCM scheme has been systematically analyzed in [?] and the security result was optimized by Iwata, Ohashi and Minematsu in [?], the design from Yan et al. is a secure design as the first two steps had been adopted.

In [?], Elbaz et al. introduced a hardware design aiming to protect both the confidentiality and integrity of data. The structure adopted in this design is similar to MAC-then-Encrypt. The tag for read-only data is the address of the data, while the tag for read-write data is the address of data concatenated with a random number. Data protected is concatenated with the tag and encrypted with block cipher. The ciphertext block is sent to off-chip area. One weakness of this design is that MAC-then-Encrypt is less secure compared with Encrypt-then-MAC structure as MAC-then-Encrypt can not defend NM-CPA attack. This proposition was proved by Bellare and Namprempre in [?]. Another weakness of security is that the bit-length of random number in the tag design is short, which is easy to collide if the amount of data to protect is large. The easy-to-collide tag can enhance the probability of succeed for the attacker.

In [?] Rogers and Milenkovic proposed an AE system design aiming to offer protection on confidentiality and integrity of both code and data. In this design, the protection mechanism for static data is same as the one for code while the mechanism for dynamic data is different. The structure of AE scheme adopted in the design from Rogers and Milenkovic is Encrypt-and-MAC. The MAC scheme adopted is a variant of PMAC. When using this variant to generate a tag for the data protected, the addressed and a unique sequence number are introduced as additional information aiming to defend type 2 attack in the threat model. The sequence number is unique to each data that invoke the tag generation. One weakness of the design from Rogers et al. is that Encrypt-and-MAC structure is less secure according to the analysis from Bellare and Namprempre in [?]. On the other hand, there is two distinct secret keys adopted in the tag generation, which introduce additional on-chip cost.

In [?], Vaslin et al. designed an AE system using one-time pad(OTP) for encryption and CRC checksum module for integrity checking. For each data protected, a tuple(address, timestamp(TS), padding value(PV)) is encrypted to form a OTP. A checksum is computed with CRC using plaintext data as input. This checksum is stored on-chip. The timestamp for each data is also recored on the chip. The OTP is xored with plaintext data to form the ciphertext. The ciphertext is sent to off-chip area. When cache reads data, ciphertext block is read to chip, xored with the OTP computed with tuple(address, timestamp(TS), PV). The output of xor is used to compute a checksum(CS2) with CRC and compared to the checksum(CS1) on-chip. The weakness of this design is that CRC is less secure compared with hash functions such as MD5 or SHA-1. This weakness was pointed by the authors in this paper and by Elbaz et al. in [?].

References