1 Experiments and Results

This section represents the simulation of two types of integrity attacks on the original CETD-MAC design and its improved versions. We use short ciphertext-tag pair to demonstrate our security analysis results.

1.1 Integrity Attacks Simulation

Our simulations of the content modification and copying-then-replaying attack follow the attack definitions depicted in Section 2. We developed the simulator of original CETD-MAC scheme and its improved variants with C programming language. We designed a simulation to evaluation the behaviour of tags from each simulator under integrity attacks and demestracte our theoretical analysis with the records of tag behaviours.

Simulation Platform and Parameters Our simulator of original CETD-MAC and its improved version are developed with C programming language under C99 standard. The simulations were conducted on Linux x64 platforms. Due to the limitation of computation resources, we choose short cihpertext and tag. A ciphertext is consist of two 8-bit blocks. The tag length is 8 bits. We use 128-bits key to generate the 128-bits nonce. The block cipher we choose is the Rijndael AES implemented in PolarSSL [].

1.1.1 Content Modification Attack Simulation

When conducting a content modification attack, the adversary modifies the content of a memory frame. The verification stage compute the tag T for the ciphertext $C_f$on the modified frame $Frame_f$ using the same nonce generating the original ciphertext and tag. This procesure can be modeled as querying the MAC scheme with distinct inputs while maintain the nonce unchanged. A secure MAC scheme should ensure that for any two distint input in the domain, the probability that the two tags collide when the nonce is identical should be low.

In the simulation of content modification attack, totally 17 input sets are created and each input set is consist of all possible 16-bit two-block inputs with same number of 1s. The size of a input set is computed with the equation: $\binom{n}{k}$, where n is the Len(input) which is 16 in our simulation and k is the number of 1s in a input. We divided all 16-bits binary numbers to 17 distinct sets and each set consist of the number contains k 1s and 16-k 0s, where k $\in$ [0,16]. For each set, we use its elements as inputs to tag generations. A base nonce is generated before the test with AES and the nonce for each input in the same set is a copy of the base nonce.

Table 1 expressed the relationship between the number of 1s in a 16-bits input and the set size. In our simulation of content modification attack, each round is expressed in the following steps:

Table 1: The relationship between the proportion of 1s in 16-bits message and the the set size

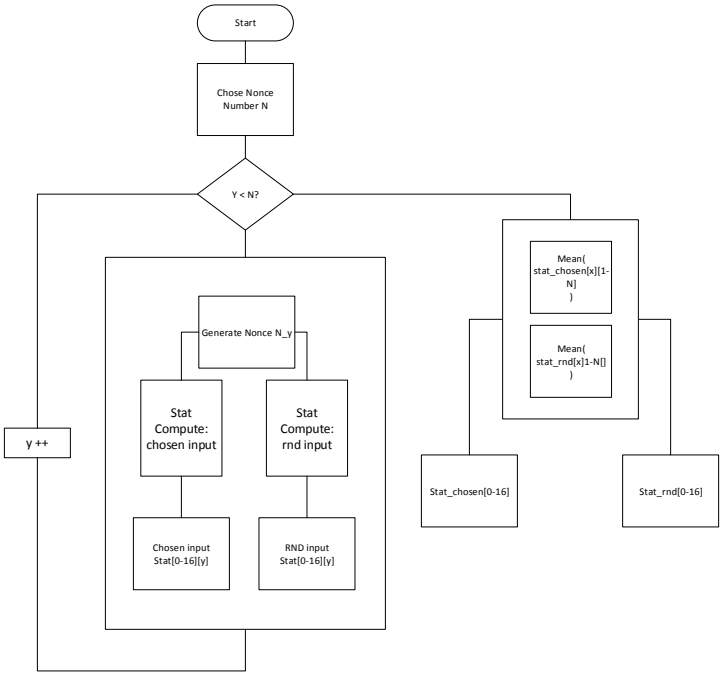
No. of 1s	Set Size
0	1
1	16
2	120
3	560
4	1820
5	4368
6	8008
7	11440
8	12870
9	11440
10	8008
11	4368
12	1820
13	560
14	120
15	16
16	1

- Step 1: Generate a base nonce N_{base} with a tuple(address, counter, random number) for a set S_i of inputs. Any element in S_i contains i 1s and 16-i 0s.
- Step 2: Generate two tags with a copy of the base nonce N_{base}. The first tag T1 uses a element from set S_i and the second tag T2 uses a 16-bits random number as input.
- Step 3: Repeat step 2 for all the elements in set S_i, compute the number of distinct T1s and T2s, marked as $Round_r\_num\_T1\_S_i$ and $Round_r\_num\_T2\_S_i$. $Round_r$ is the index of current test round.
- Step 4: Repeat step 3 for all the sets where i $\in$ [0,16]

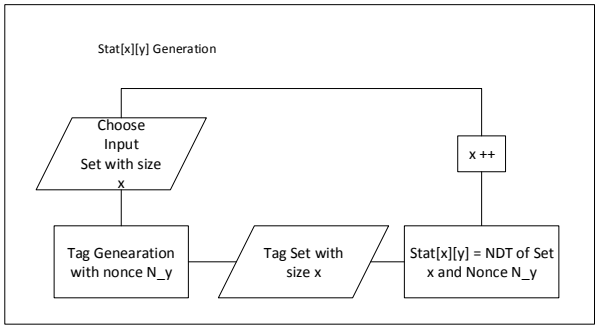
We repeat the simulation 1000 rounds and compute the average of $Round_r_num_T1_S_i$s and $Round_r_num_T2_S_i$s seperately. The average results $Aver\_num\_T1\_S_i$ and $Aver\_num\_T2\_S_i$ for each set S_i serves as the indicator of the security of a tested scheme under content modification attack. The concept of our simulation of content modification attack is expressed in Figure 1.

1.1.2 Copying-then-Replaying Attack Simulation

If the adversary conducts a copy-then-replay attack, he will maintain a copy of valid fram written in an older time point of on different memory address and



(a) Simulation flowchart of content modification attack



(b) Statistic Array Stat

Figure 1: Concept of Content Modification Attack Simulation

then replace the vailid frame with this copy. This procedure can be modeled as querying the MAC scheme with a fixed chosen input while the input of nonce generation is distinct for each query. It is possible that two tags collide when they are generated with two distinct nonce but one identical input. The high probability of this collision indicates that the MAC scheme is unsecure under copy-then-replay attack.

Our simulation of copy-then-replay attack is modeled into the following steps:

- Step 1: Choose 0x0000 as the ciphertext
- Step 2: Generate k tags with k nonces. The input to generation each nonce is distinct
- Step 3: Compute the No. of distinct tags among the k tags
- Step 4: If the ciphertext is not bigger than 0xFFFF, add it with one then repeat step 2 and step 3.

Table 2 expresses the original CETD-MAC and its improved variants. For each

Table 2: MAC Designs

Name	Description
CETD	Original CETD
D1	With GF-Mult of each input block with a random number
D2	With GF-Mult of each input block with a random number or a constant
D3	With GF-Mult of selected number of input blocks with random numbers
D4	Same as D1
D5	Block flipping plus D3
D6	Block flipping plus D4

ciphertext, we simulated with two different sample size: 1000 rounds and 100000 rounds. In each round a randomly generated nonce will be adopted to generate the tag for the fixed input. The concept of our simulation of copy-then-replay attack is also expressed in Figure 2.

Figure 3 shows the No. of distinct tags of the first 100 input ciphertexts in 1000 rounds cases. The statistic information of the No. of distinct tags for all the inputs in 1000 rounds case is expressed in Table 3. Figure 3 shows the No. of distinct tags of the first 100 input ciphertexts in 1000 rounds cases. The statistic information of the No. of distinct tags for all the inputs in 1000 rounds case is expressed in Table 3.

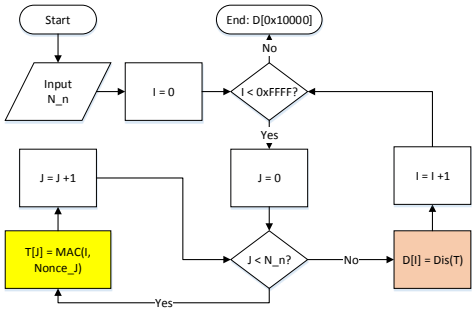
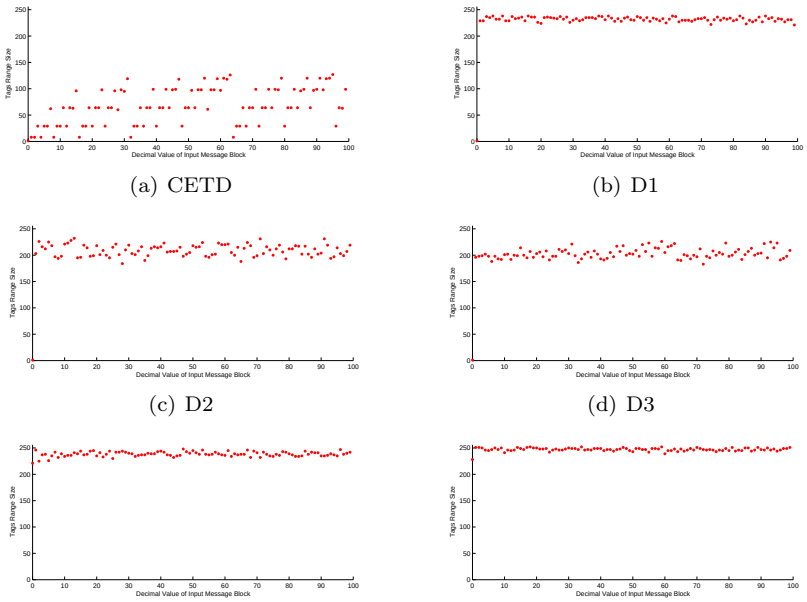


Figure 2: Simulation Flowchart of Copy-then-replay Attack: Each element in D[0x10000] is No. of distinct tags of ciphertext i with 1000 different nonce under copy-then-replay attack, where i is the decimal value of a 16-bits ciphertext



(a) CETD (b) D1

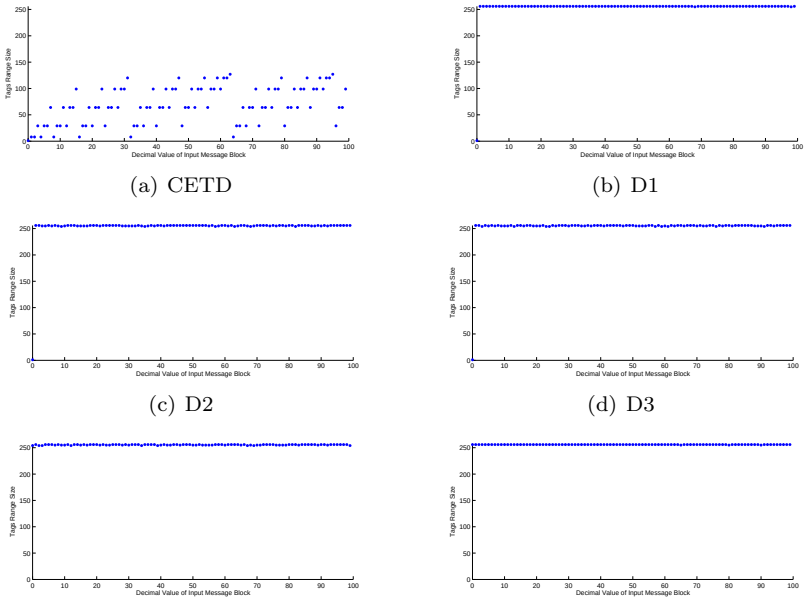
(c) D2 (d) D3

(e) D5 (f) D6

Figure 3: The No. of Distinct Tags: The input domain is [0,99], for each input replay 1000 times; the nonce for each replay is randomly generated

Table 3: Statistic on the No. of distinct tags: 1000 times case

	CETD	D1	D2	D3	D5	D6
max	128	256	256	256	256	256
min	1	1	1	1	221	228
var	148.91	8.2537	21.484	20.408	7.5639	4.8166
average	123.2	250.74	246.31	249.94	250.12	250.85



(a) CETD (b) D1

(c) D2 (d) D3

(e) D5 (f) D6

Figure 4: The No. of Distinct Tags: The input domain is [0,99], for each input replay 10000 times; the nonce for each replay is randomly generated

Table 4: Statistic on the No. of distinct tags: 10000 rounds case

	CETD	D1	D2	D3	D5	D6
max	128	256	256	256	256	256
min	1	1	1	1	253	255
var	149.22	0.99235	0.99682	0.99671	0.002621	0.000107
average	123.56	256	255.99	255.99	256	256