1 Approach

In this article, we name the MAC scheme adopted in Cost-Effective Tag Design[] CETD-MAC.

1.1 Specification of the MAC Scheme in Cost-Effective Tag Design

In this section, we depict the design details of CETD-MAC scheme. At the beginning we express notations required to understand CETD-MAC scheme.

Notations Let $\{0,1\}^n$ be the set of all n-bit binary strings. The set of all binary string is expressed as $\{0,1\}^*$. For a string $X \in \{0,1\}^n$, —X— is its length in bits, and $\mid X \mid_l = \lceil \mid X \mid /l \rceil$ is the length of X in l-bit blocks. Let 0^l and 1^l denote bit strings of all zeros and all ones. For a bit string X and an integer l that $\mid X \mid \geq l$, $\text{msb}_l(X)$ denotes the most significant l bits(left most l bits) of X and $\text{lsb}_l(X)$ for least significant l bits(right most l bits) of X. For two bit string X and Y, we denote $X \| Y$ or XY as the their concatenation. For bit string X whose length in bits is multiple of integer l, we denote X parted into l-bit sub-strings as $X = (X[1]X[2]\ldots X[n])_l$, where X[1], X[2], ..., $X[n] \in \{0,1\}^l$. The number of bits in a string of X is denoted as len(X).

The block cipher encryption of a string X with a secret key K is denoted as $E_K(X)$. $E_K(X)$ expresses the String mapping of $\{0,1\}^n \rightarrow \{0,1\}^n$ where n is the len(X) and len(output).

CETD-MAC Scheme CETD-MAC scheme can be expressed as tag = CETD-MAC(M, nonce-input). The input arguments of CETD-MAC are message M and a tuple named nonce-input. The tuple nonce-input is the concatenation of the memory address of M, a counter and a random number, denoted as nonce-input=(address$\|$counter$\|$random). The length of nonce-input, len(nonce-input), is identical to the length of input to block cipher $E_K(X)$. The output of CETD-MAC is named tag, whose length is optional. We use Sub-BLK-No to express the value of len(M)/len(tag). One prelimary of CETD-MAC is that Sub-BLK-No should be no less than 2 to assure that swapping stage can work and 0s will be concatenated to the leftmost of input message when Sub-BLK-No is not integer.

We follow the defination of CETD-MAC in [**?**].

- Message is splitted to sub-blocks each of whose length is tag length
- The first stage is several rounds of bit-segment swapping. For each round , two sub-blocks are randomly chosen. Introduce shuffle paremeter V[i]
- The output sub-blocks from swapping stage, named X blocks, are sent to block-level rotate shifting stage. Each X block is rotate shifted for several bits. Introduce shift parameter S[i]

- The output sub-blocks from shifting stage, named Y blocks, are XORed to form the final tag

1.2 Tag Collision Under Replay Attack

In this section, we depicit the weaknesses we found in CETD-MAC under replay attack. These weaknesses in mechanism let the CETD-MAC to produce tags with high collision probability with some chosen inputs. The tags with highly collision probability ease the adversary‘s attack.

Security Notion of MAC Schemes under Replay Attacks We adopt the standard notion for the security of a MAC scheme in the presence of a replay attack, in which an adversary is given access to a tag generation black-box(named oracle) and a message-tag pair verification oracle. The encryption key is maintained unchanged while the nonce value is updated for each calling of tag generation oracle no matter whether the value of input message collide with a value in old timestamp. The adversary can record the message-tag pairs in old time stamp to copies and replace a pair in new timestamp with a piece of copies. The forgery advantage $\mathrm{F}_{CETD-MAC}$ under replay attack is the probability that the adversary can get the verification oracle to accept a copy with old timestamp at a new time point.

Assume the copy with old timestamp from the adversary is expressed as (M1,T1) where M1 is the messsage and T1 is the corresponding tag, and the correspoding nonce is denoted as N1. When applying M1 to the tag generation at a new time point, we mark the corresponding tag T2 and nonce N2. If the adversary can succeed a replay attack, then T1 is identical to T2. Then $\mathrm{F}_{CETD-MAC}$ under replay attack can be denoted as a probability Pr[T1 = T2 | Message value is M1 & N1,N2 are random].

1.2.1 The Block-level Pattern Collision

1.2.2 The Set-level Pattern Collision

1.3 A Solution to Elimitated the Collision

In this section we provide an modification of original CETD-MAC. Our approach does not require additional component or information to the original CETD-MAC. We then prove that the optimized CETD-MAC can fix the weakness and protect input message from replay attack.

2 Experiments and Results