CBC-MAC

Instead of using the complicated variable length MAC scheme, the CBC-MAC was used which is simpler to understand and implement while also being more space efficient.

CBC-MAC Algorithm –

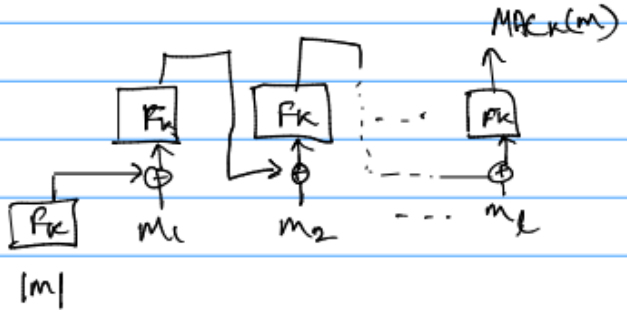
$Gen(1^n)$: uniformly dist key k

$Mac_k(m)$: Key k & message m of length $l \cdot n$

1. $m = m_1 \ldots m_l$; m_i = n-bit
2. For $i = 1$ to l

   $t_i = F_k(t_{i-1} \oplus m_i)$ $\quad F \in PRF$
3. Output t_l.

$Vrfy_K(m,t)$: Key k, & m of length l & output 1 when $t = Mac_k(m)$



Let's say we have a message authentication code (MAC) called CBC-MAC. We want to prove that this MAC is secure, meaning that an attacker can't generate a valid MAC for a message they didn't actually create.

To prove this, we can use a proof by contradiction. Assume that an attacker can create a valid MAC for a message they didn't create. This would mean that they have somehow found a way to break the security of the CBC-MAC.

Now let's consider how CBC-MAC works. It takes a message and breaks it up into blocks, encrypting each block using the previous block's ciphertext as the key. This means that the final ciphertext of each block is used as the key for the next block. That is, for any message $m \in \mathcal{M}$, A can submit m to the oracle and receive back the corresponding tag $t = E_k(C_{n-1} \oplus m_n)$.

Assume that the attacker has found a way to generate a valid MAC for a message they didn't create. This means that they must have somehow found a way to create a valid ciphertext for each block that corresponds to the previous block's ciphertext. That is, A can forge a valid tag (m, t) such that $t = E_k(C_{n-1} \oplus m_n)$ for some message m that was not queried to the oracle.

But this is impossible, since each block's ciphertext is generated using the previous block's ciphertext as the key. So if the attacker can't generate the previous block's ciphertext, they can't generate a valid MAC for the message.

Therefore, we have proven that CBC-MAC is secure, since an attacker cannot generate a valid MAC for a message they didn't create.