0 CBC-MAC

Write the basic construction of CBC-MAC.

1 Merkle-Damgård

Let $h : \{0,1\}^{n+t} \to \{0,1\}^n$ be a fixed-length compression function. Suppose we forgot a few features of Merkle-Damgård and construct H as follows:

- Value x is input.
- Split x into $y_0, x_1, \ldots, x_k$. Where y_0 is n bits and x_i (for $i = 1, \ldots, k$) is t bits.. The last piece x_k may be padded with zeroes.
- For $i = 1, \ldots, k$, set $y_i = h(y_{i-1}||x_i)$.
- Output y_k.

It's similar to Merkle-Damgård except no IV and the final padding block is missing.

1. Describe an easy way to find two messages that are broken up into the same number of pieces, which have the same hash value under H.
2. Describe an easy way to find two messages that are broken up into a different number of pieces, which have the same hash value under H. *Hint: Pick any string of length $n + 2t$, and find a shorter string that collides with it.*

Neither of your collisions above should involve finding a collision in h!

2 Hash Functions

I designed $H : \{0,1\}^* \to \{0,1\}^n$. I make $H(x) = x$ if x is n-bit string – but assume H's behavior is more complicated on strings of other lengths. This way we know there are no collisions among n-bit strings. Is this a good design decision?

3 MAC

Prove that the following modifications of basic CBC-MAC do not yield a secure MAC (even for fixed-length messages).

1. Mac outputs all blocks $t_1, \ldots, t_\ell$ rather than just t_ℓ. Verification only checks if t_ℓ is correct.

2. A random initial block is used each time a message is authenticated. That is, choose a uniform $t_0 \in \{0,1\}^n$, run basic CBC-MAC over the "message" $t_0, m_1, \ldots, m_\ell$ and output tag $\langle t_0, t_\ell \rangle$. Verification is done in a natural way.

4 Digital Signature

Let (G, S, V) be a secure signature scheme with message space $\{0,1\}^n$, and security parameter λ. Let $(pk_0, sk_0) \leftarrow_\$ G(1^\lambda)$ and $(pk_1, sk_1) \leftarrow_\$ G(1^\lambda)$ be two pairs of signing/verification keys.
Which of the following are secure signature schemes? Show an attack or prove security.

1. (S_1, V_1):

   - Sign. $S_1((sk_0, sk_1), m)$: Output $(S(sk_0, m), S(sk_1, m))$.
   - Verify. $V_1((pk_0, pk_1), m, (\sigma_0, \sigma_1))$: Output 1 if $(V(pk_0, m, \sigma_0) \vee V(pk_1, m, \sigma_1))$, 0 otherwise.

     I.e., the verification accepts if one of the two signatures accepts.

2. (S_2, V_2)

   - Sign. $S_2((sk_0, sk_1), (m_L, m_R))$: Output $(S(sk_0, m_L), S(sk_1, m_R))$.
   - Verify. $V_2((pk_0, pk_1), (m_L, m_R), (\sigma_0, \sigma_1))$: Output 1 if $(V(pk_0, m_L, \sigma_0) \wedge V(pk_1, m_R, \sigma_1))$, 0 otherwise. I.e., both verifications must accept.