Message Authentication Code

Group 7

UM-SJTU Joint Institute

Ma Siyin 517370910003
Ming Xingyu 517370910224
Zhang Liqin 517370910123
Zhao Zhijie 517370910035

July 17, 2020

Overview

1. Introduction to Message Authentication Code(MAC)
   - Message Authentication
   - Message Authentication Code
2. MAC Classification
   - Hash based MAC
   - Cipher block chaining MAC
3. MAC Security
   - Brute-force attack on CBC-MAC
   - Brute-force attack on HMAC
4. MAC Application
   - Pseudo-random Number Generation Using Hash Functions and MACS
   - MAC used in Transport Layer Security protocol

WHY Message Authentication

Message authentication is a property that a message has not been modified while in transit (data integrity) and that the receiving party can verify the party claiming to send the message.

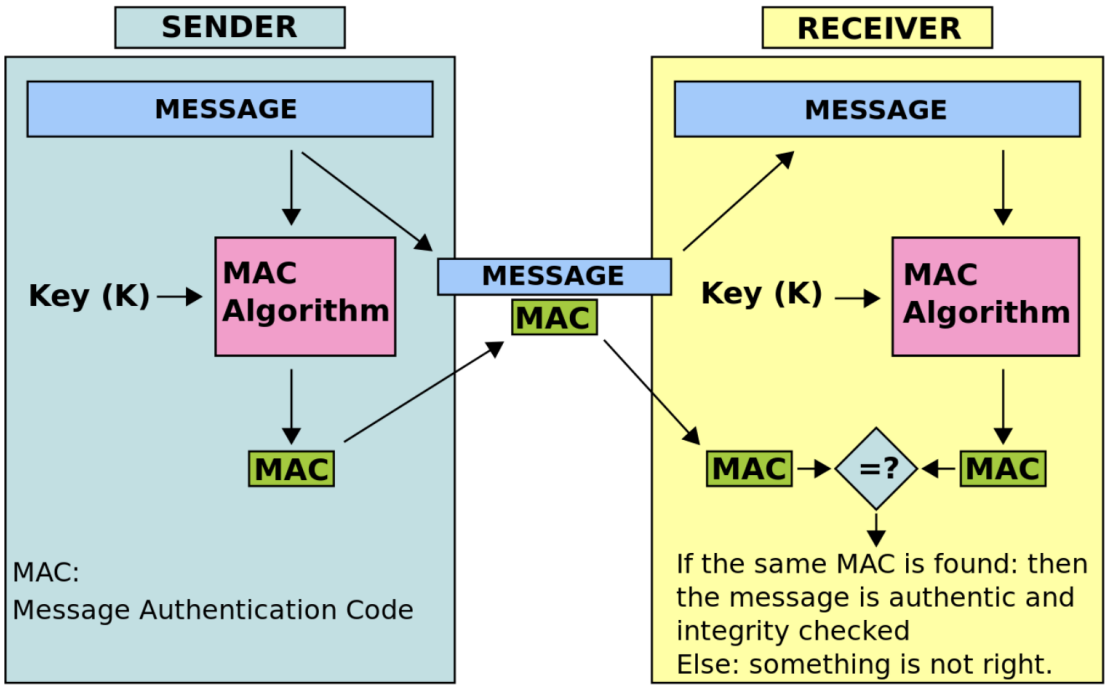


Figure: General Process of Message Authentication

Message Authentication Functions

Some example functions of authenticator generators:

- **Hash function**
- **Message encryption**
- **Message Authentication Code (MAC)**

Message Authenticaiton Code

A message authentication code consists of three probabilistic polynomial time algorithms $(Gen, Mac, Vrfy)$ such that:

Definition

1. Key generator Gen takes as input the security parameter 1^n and outputs a key k with $|k| \geq n$.
2. Tag-generator Mac takes as input a key k and a message $m \in \{0,1\}$, and outputs a tag t. Since this algorithm may be randomized, we write this as $t \longleftarrow \text{MAC}_k(m)$.
3. Deterministic verifier VRFY takes as input a key k, a message m, and a tag t. It outputs a bit b, with $b = 1$ meaning valid and $b = 0$ meaning invalid. We write this as $b := \text{VRFY}_k(m, t)$.

Message Authentication Codes Requirements

MAC Requirements

1. If M and $\mathrm{MAC}(K, M)$ are known to an attacker, it should be hard to construct a new M' that

$$\mathrm{MAC}\left(K, M'\right) = \mathrm{MAC}(K, M).$$

2. $\mathrm{MAC}(K, M)$ should be uniformly distributed, where the probability of two randomly selected messages are identical is 2^{-n}. n represents the number of bits in a tag.
3. Let $M' = \mathrm{f}(M)$, as a transformation on M,

$$\Pr\left[\mathrm{MAC}(K, M) = \mathrm{MAC}\left(K, M'\right)\right] = 2^{-n}.$$

Hash based MAC (HMAC)

HMAC, Keyed-hash message authentication code, defines a MAC that apply a hash function with a secret key K.

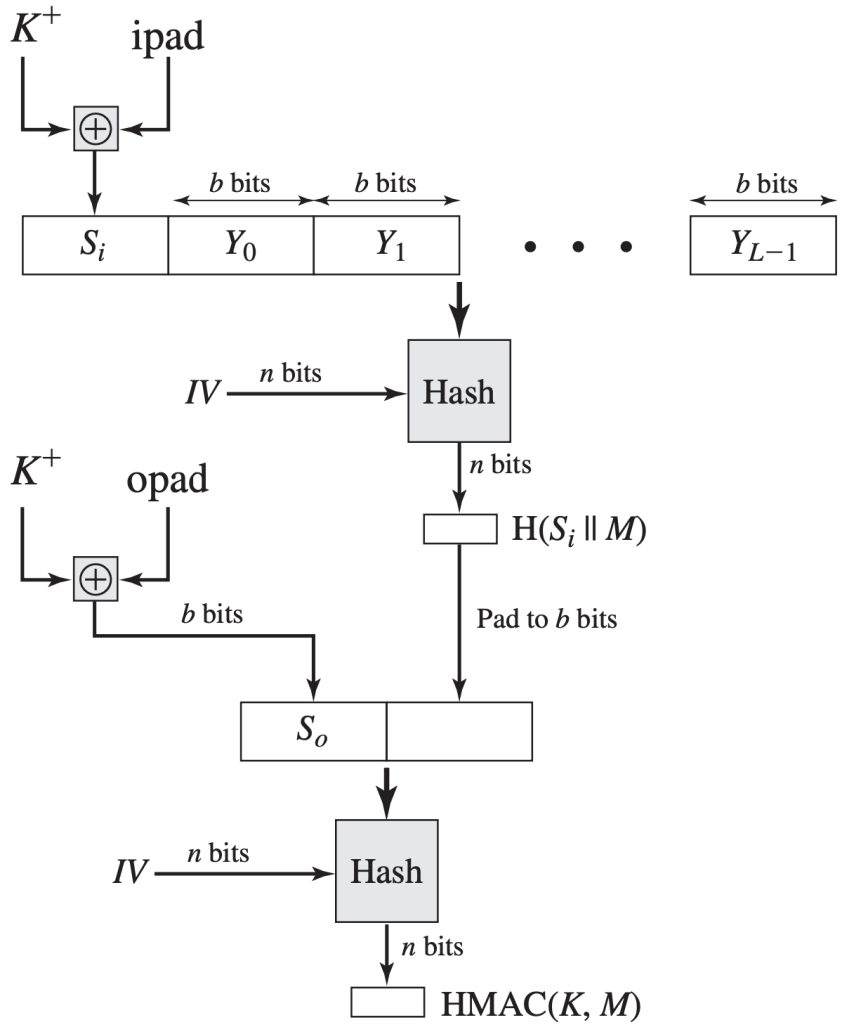
Definition

$$HMAC(K, text) = H\left((K_+ \oplus opad) \| H\left((K_+ \oplus ipad) \| text\right)\right)$$

The key K recommended length is $\geq n$, If key length is greater than b, the key is input to the hash function and produce an n-bit key. K^+ denotes K padded with zeros on the left so that the result is b bits in length.

Hash based MAC (HMAC)

- H = embedded hash function
- IV = initial value input to hash function
- M = message input to HMAC
- b = number of bits in a block
- n = length of hash result
- *ipad* = 00110110 (0x36) repeated $b/8$ times
- *opad* = 01011100 (0x5C) repeated $b/8$ times



Cipher block chaining MAC (CBC-MAC)

CBC-MAC is based on block ciphers to create a chain so that each block depends on the proper encryption of the previous block.

Definition

$$CBC(K, K_1, K_2) = \begin{cases} C_{n-1} = E(K, M_{n-1} \oplus C_{n-2}) \\ C_n = E(K, D_n \oplus C_{n-1} \oplus K_{1/2}) \end{cases}$$

Notice that when the message cannot be fully divided by the length of the cipher block, the final block is padded and replace K_1 by K_2.

Cipher block chaining MAC (CBC-MAC)

- M_i = message divided to each block
- K = encryption key with k bit
- K_1, K_2 = n bit constant generated by K
- b = number of bits in a block
- MSB = Take left most s bit as tag
- $Tlen$ = tag length

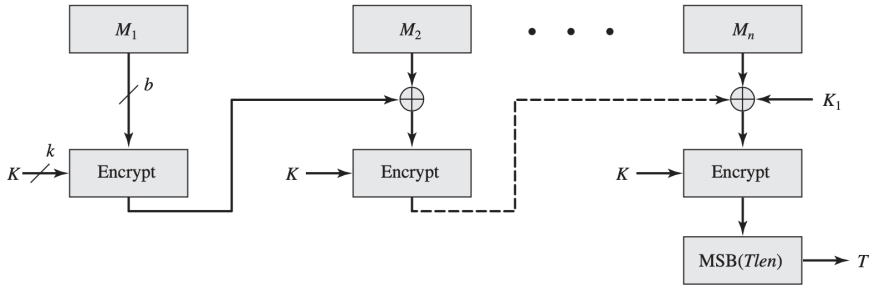


Figure: $|M| = kb, k \in \mathbb{Z}$

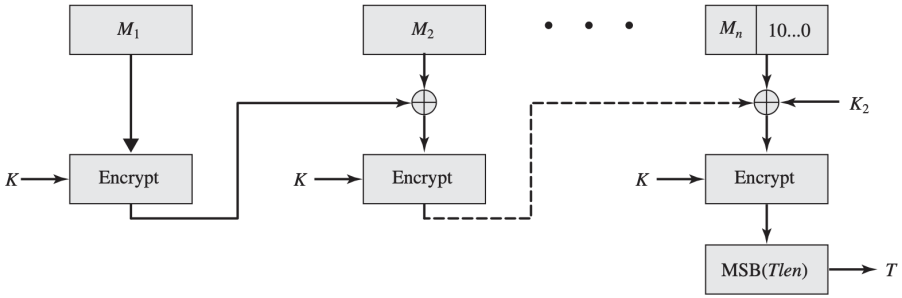


Figure: $|M| \neq kb, k \in \mathbb{Z}$

Brute-force attack on CBC-MAC

It's a kind of birthday attack.

Setup Form a message that are only different with the original message for the first two blocks. For the two first blocks, each we prepare $q \approx 1.17 \times 2^{t/2}$ different bit strings.

Attack As an assumption, the attacker can have the MAC of $x^1, x^2, ..., x^q$ in which y_n^i being the MAC of x^i. Since we have enough pairs of amount of message-tag pairs, let's say $y_n^i = y_n^j$, which means the MAC for x^i and x^j are the same. Noticing that $y_n^i = y_n^j$ if and only if $y_2^i = y_2^j$, which can further indicate that

$$y_1^i \oplus x_2^i = y_1^j \oplus x_2^j$$

They can have much more freedom on the content of x_2^i and x_2^j if collision found.

$$v = x_1^i || (x_2^i \oplus x_\delta) || ... || x_n^i$$

and

$$w = x_1^j || (x_2^j \oplus x_\delta) || ... || x_n^j$$

v and w are sharing the same tag.

Brute-force attack on HMAC

As stated before, the HMAC is consist of two Hash functions,

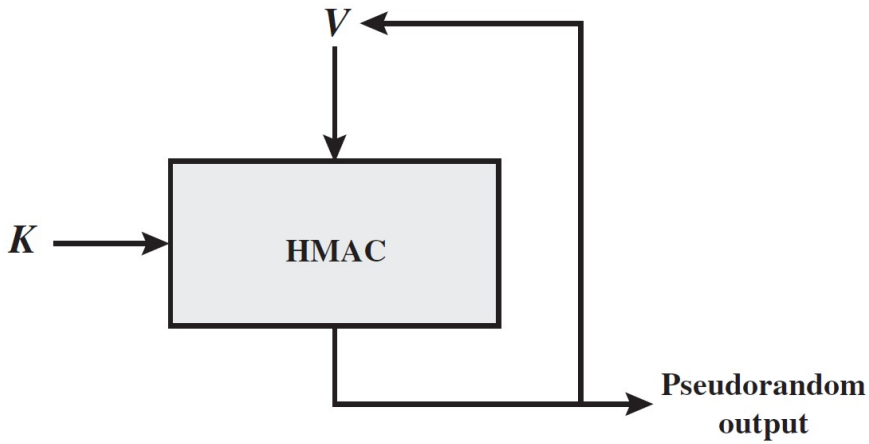
HMAC

$$\mathrm{HMAC}(K, M) = H_o((K \oplus opad)||H_i((K \oplus ipad)||M))$$

- **For the attack on HMAC**
  The attackers can choose $x = M$ and find $z = \mathrm{HMAC}(M)$ accordingly. Their goal is to generate some unchosen x' and z', such that $z' = \mathrm{HMAC}(x)$.
- **For the attack on the inside Hash function**
  The attackers can choose $x = M$ and find $y = H_i(M)$ accordingly. Their goal is to find some x' and x'', such that $x' \neq x''$ with $y' = y''$.
- **For the attack on the outside Hash function**
  The attackers can choose $y = M$ and find $z = H_o(M)$ accordingly. Their goal is to generate some unchosen y' and z', such that $z' = H_o(y)$.

Pseudo-random Number Generation Using MAC

- Recall the BBS generator (what is the goal of a block cipher)
- Repeatedly using MAC to generate a small block with a private key K
- Combine them together

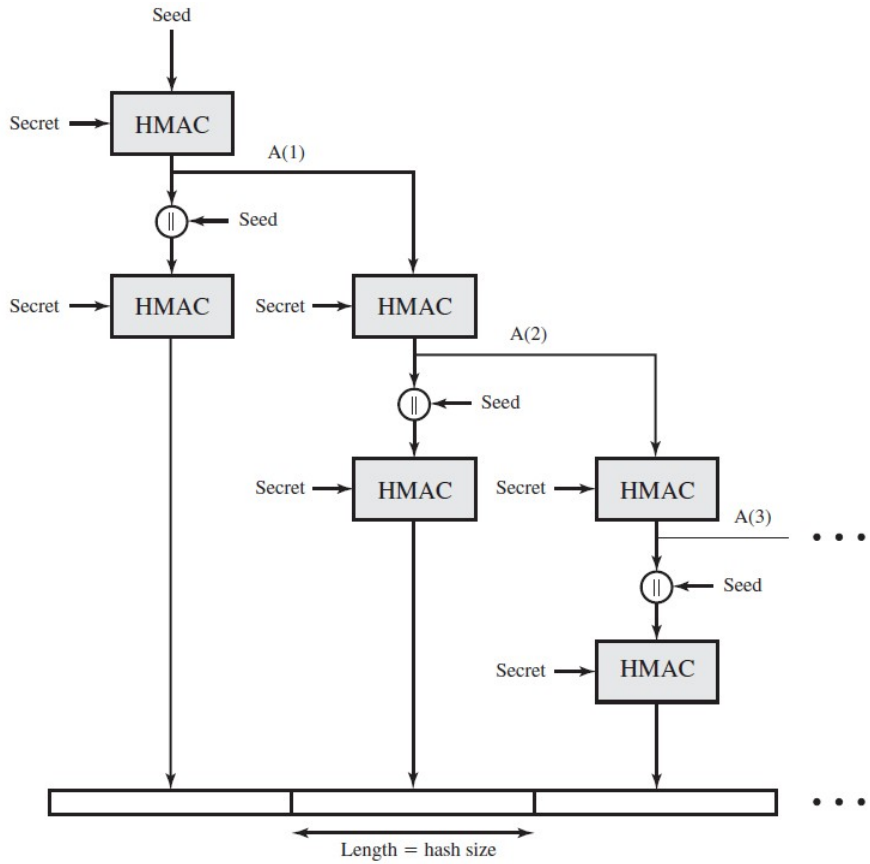


MAC used in Transport Layer Security protocol

Transport Layer Security protocol also uses HMAC as part of its pseudo-random function P_hash.

P_hash

P_hash$(S, V) = H(1)||H(2)||H(3)...$
where
$H(n) =$ HMAC_hash$(S, A(n)||V)$,
$A(n) =$ HMAC_hash$(S, A(n-1))$
and $A(0) = V$



Q&A

Thanks for listening!