T-79.4501
Cryptography and Data Security

Lecture 5:
5.1 MAC-functions
5.2 Hash-functions
Stallings: Ch 11, Ch 12

1

5.1.Message authentication codes (MAC)

(Secret key , Message) ⟶ MAC

- A MAC of a message P of arbitrary length is computed as a function $H_K(P)$ of P under the control of a secret key K. The MAC is appended to the message by the sender.
- Given a message P and its MAC value M, the MAC can be verified by anybody in possession of the secret key K and the MAC computation algorithm.
- The MAC length m is fixed.
- Security requirement: it must be infeasible, without the knowledge of the secret key, to determine the correct value of $H_K(P)$ with a success probability larger than $1/2^m$. This is the probability of simply guessing the MAC value correctly at random. It should not be possible to increase this probability even if a large number of correct pairs P and $H_K(P)$ is available to the attacker.

2

An Example: A Weak MAC

E_K is an encryption function of a block cipher

Given a message $P = P_1, P_2, \ldots, P_n$

a MAC is computed as

$$H_K(P) = E_K(P_1 \oplus P_2 \oplus \ldots \oplus P_n)$$

Then it is easy to produce a different message P' with an equal MAC:

$$P' = P_1', P_2', \ldots, P_{n-1}', \left(\bigoplus_{i=1}^{n-1} P_i' \right) \oplus \left(\bigoplus_{i=1}^{n} P_i \right)$$

Derived security requirements

The requirement: It must be infeasible, without the knowledge of the secret key, to determine the correct value of $H_K(P)$ with a success probability larger than $1/2^m$.

This means, in particular, that the following are satisfied

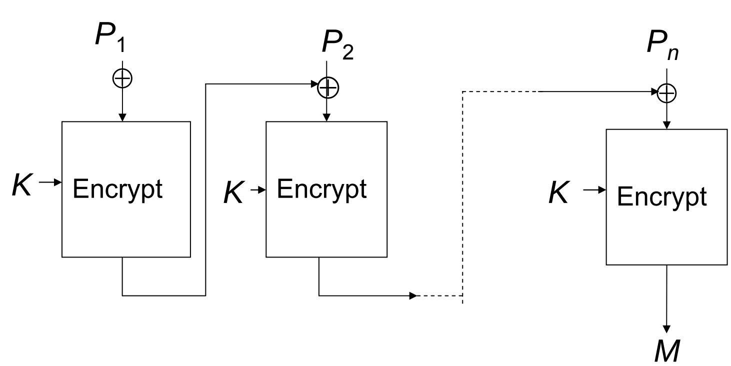
- Given a message *P* and $M = H_K(P)$ it should be infeasible to produce a modified message *P*' such that $H_K(P') = M$ without the knowledge of the key
- For each *K*, the function P → $H_K(P)$ is one-way
- Given known MACs for a number of known (or chosen or adaptively chosen) messages, it should be infeasible to derive the key.

MAC Designs

- Similarly as block ciphers, MAC algorithms operate on relatively large blocks of data.
- Most MACs are iterated constructions. The core function of the MAC algorithm is a compression function. At each round the compression function takes a new data block and compresses it together with the compression result from the previous rounds. Hence the length of the message to be authenticated determines how many iteration rounds are required to compute the MAC value.

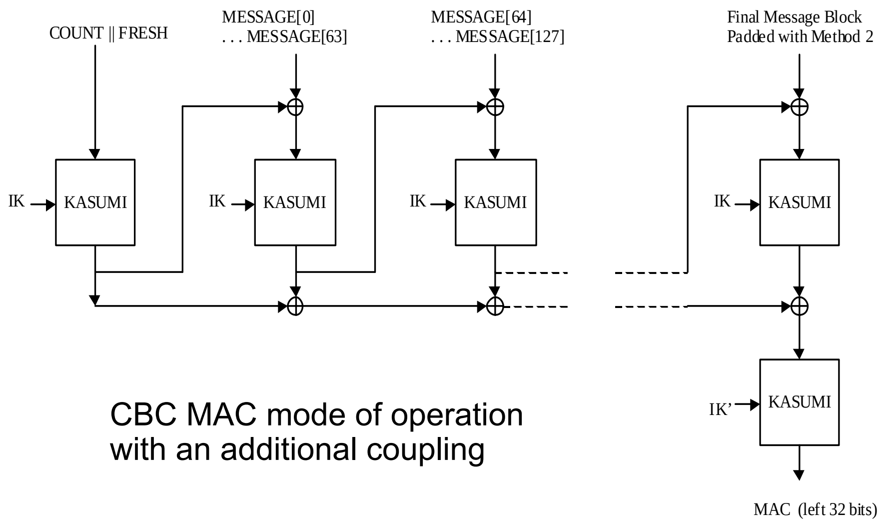
CBC MAC

A MAC mode of operation of any block cipher



- CBC encryption with fixed IV = `00…0`. The last ciphertext block (possibly truncated) is taken as the MAC.

Integrity function f9



CBC MAC mode of operation
with an additional coupling

7

CRC MAC

- A MAC for stream ciphers (see HAC 9.5.4.)
- Idea: A simple (cryptographically insecure) error detecting check sum is encrypted using non-repeating keystream (ideally, a one-time pad)

An n-bit message $P = p_0, p_1, \ldots, p_{n-1}$ is associated with the polynomial

$$P(x) = p_0 + p_1 x + p_2 x^2 + \ldots + p_{n-1} x^{n-1}$$

The secret key K consists of a polynomial $q(x)$ of degree m, and an m-bit one-time key stream string $(k_0, k_1, k_2, \ldots, k_{m-1})$.

First the remainder $c_0 + c_1 x + c_2 x^2 + \ldots + c_{m-1} x^{m-1}$ of the polynomial division $P(x)/q(x)$ is computed. The MAC is computed as the xor of the key stream string and the remainder string $(c_0, c_1, c_2, \ldots, c_{m-1})$ as

$$(c_0 \oplus k_0, c_1 \oplus k_1, c_2 \oplus k_2, \ldots, c_{m-1} \oplus k_{m-1})$$

8

Note: The polynomial $q(x)$ can be reused for different messages

Polynomial MAC

- Another MAC for stream ciphers
- Idea: An (cryptographically insecure) error detecting code is encrypted using non-repeating keystream (ideally, a one-time pad)

An n-block message $P = P_0, P_1, \ldots, P_{n-1}$ with block size m bits is associated with the polynomial with m-bit coefficients:

$$P(x) = P_0 + P_1 x + P_2 x^2 + \ldots + P_{n-1} x^{n-1}$$

Also the value of the polynomial is assumed to be expressed as an m-bit string.

The secret key K consists of a point $x = X$ and an m-bit one-time key stream string $(k_0, k_1, k_2, \ldots, k_{m-1})$.

First the message polynomial is evaluated at the point X. Let us denote the value by $(c_0, c_1, c_2, \ldots, c_{m-1})$. The MAC is computed as the xor of the key stream string and the value as

$$(c_0 \oplus k_0, c_1 \oplus k_1, c_2 \oplus k_2, \ldots, c_{m-1} \oplus k_{m-1})$$

Note: The point X can be reused for different messages

9

An Example

Poly1305-AES MAC

- By D J Bernstein, presented at FSE2005, http://cr.yp.to/mac.html
- Over finite fields: Carter-Wegman MAC anf Galois MAC (with Counter Mode key stream generator)

10

Combined modes of operation

Authenticated Encryption Modes

- CCM: Counter mode encryption and CBC MAC , see:

  1) IETF RFC 3610

  2) NIST Special Publication SP800-38C (with consideration to the IEEE 802.11i)

  (see Exercise 3.6)

- GCM: Counter mode encryption and a Polynomial-based MAC over Galois Field, see: http://csrc.nist.gov/CryptoToolkit/modes/proposedmodes/

11

5.2 Hash functions

Message → Hash code

- A hash code of a message *P* of arbitrary length is computed as a function *H*(*P*) of *P* . The hash length *m* is fixed.
- Hash function is public: Given a message *P* anybody can compute the hash code of *P*.
- Security requirements:
  1. *Preimage resistance:* Given *h* it is impossible to find *P* such that *H*(*P*) = *h*
  2. *Second preimage resistance:* Given *P* it is impossible to find *P'* such that *H*(*P'*) = *H*(*P*)
  3. *Collision resistance:* It is impossible to find *P* and *P'* such that *P* ≠ *P'* and *H*(*P'*) = *H*(*P*)

12

Design Principles

- Similarly as MAC algorithms, hash functions operate on relatively large blocks of data.
- Most hash functions are iterated constructions. The core function in a hash function is a compression function. At each round the compression function takes a new data block and compresses it together with the compression result from the previous rounds. Hence the length of the message to be authenticated determines how many iteration rounds are required to compute the MAC value.

13

SHA-1

- Designed by NSA
- FIPS 180-1 Standardi 1995 – www.itl.nist.gov/**fips**pubs/**fip**180-1.htm

February 2005:

Professor Xiaoyun Wang (Shandong University) announce an algorithm which finds collisions for SHA-1 with complexity 2^{69}

Recommendation: Use 256- or 512-bit versions of SHA:
csrc.nist.gov/publications/**fips**/**fips**180-2/**fips**180-2.pdf

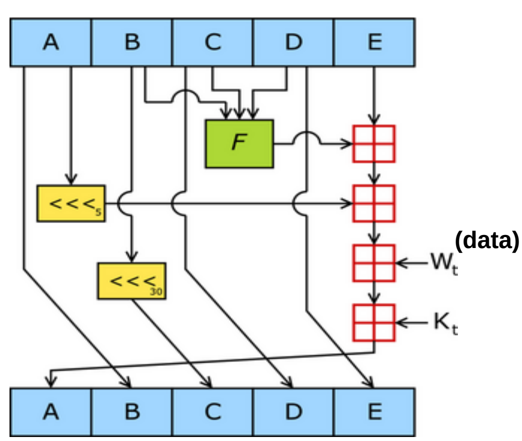
14

SHA-1

- Step 1: Append padding bits.
- Step 2: Append *length* of the message before padding (64 bits); *length* + 64 < 512 L
- Step 3: Initialise MD buffer composed of five 32-bit registers (A,B,C,D,E) with a constant IV (fixed in the spec).This is denoted by CV_0.
- Step 4 (repeated L times): Process message in 512-bit (16-word) blocks. It takes 80 rounds. At the end, the contents of the registers ABCDE are added to the input CV_q.The addition modulo 2^{32} is done for each word separately. The result is the output CV_{q+1} (input to the next round), $q = 0,1,\ldots,L\text{-}1$. The addition modulo 2^{32} is done for each word separately.
- Step 5: Output is CV_L

15

SHA-1 Compression function One round



512 bits of data – 80 rounds

Addition modulo 2^{32}

16

Function F and data expansion

$$q = 0,...,19: \quad F_q(B,C,D) = (B \wedge C) \vee (\overline{B} \wedge D)$$

$$q = 20,...,39: \quad F_q(B,C,D) = B \oplus C \oplus D$$

$$q = 40,...59: \quad F_q(B,C,D) = (B \wedge C) \vee (B \wedge D) \vee (C \wedge D)$$

$$q = 60,...79: \quad F_q(B,C,D) = B \oplus C \oplus D$$

Data expansion:

$(W_0, W_1, W_2, ..., W_{15}) =$ the 512-bit input data block

$$W_q = <<<_1 (W_{q-16} \oplus W_{q-14} \oplus W_{q-8} \oplus W_{q-3}),\ q = 16...79$$

17

Revised SHA Standard

csrc.nist.gov/publications/**fips**/**fips**180-2/**fips**180-2.pdf

	SHA-1	**SHA-256**	**SHA-384**	**SHA-512**
Hash size	160	256	384	512
Message size	$< 2^{64}$	$< 2^{64}$	$< 2^{128}$	$< 2^{128}$
Block size	512	512	1024	1024
Word size	32	32	64	64
Number of steps	80	80	80	80
Claimed security	80	128	192	256

18

HMAC- hash based MAC

- RFC 2104: the MAC for IP security
- To use available hash functions
- To allow hash function to be replaced easily
- To preserve the performance of a hash function
- Easy handling of keys
- Well understood cryptographic security

- Recent collision attacks against hash functions do not effect HMAC constructions

19

HMAC algorithm

H	hash function
M	message input to HMAC (after hash function specific padding added)
L	number of blocks in M
b	number of bits in a block
n	length of the hash code of H
K	secret key, recommended length ≥ n
K^+	a b-bit string formed by appending zeros to the end of K
ipad	= `00110110` repeated b/8 times
opad	= `01011100` repeated b/8 times

$$HMAC(K;M)= H[(K^+ \oplus opad) \,||\, H[(K^+ \oplus ipad)||M]]$$

20