Network information security

MAC (消息认证码) and HASH function

By Rachel yuan

2019.12

1

Outline

- Message integrity: MAC
- How to build MAC?
  – Secure block ciphers
  – Hash function

2

Message Integrity

Goal: **integrity**, no confidentiality.

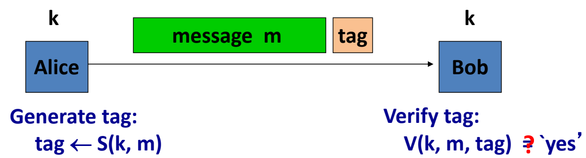
Examples:

– Protecting system files on disk.

– Protecting banner ads(广告) on web pages.

3

Message integrity: MACs



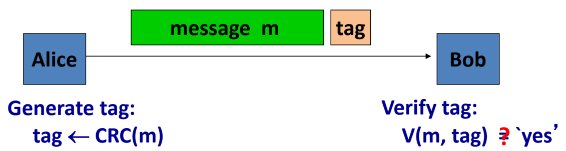
Def: MAC I = (S,V) defined over (K,M,T) is a pair of algs:

– S(k,m) outputs t in T

– V(k,m,t) outputs `yes` or `no`

4

Integrity requires a secret key



- Attacker can easily modify message m and re-compute CRC.
- CRC designed to detect **random**, not malicious errors.

5

Secure MACs

Attacker's power: **chosen message attack**

- for $m_1, m_2, \ldots, m_q$ attacker is given $t_i \leftarrow S(k, m_i)$

Attacker's goal:

- produce some new valid message/tag pair (m,t).

$$(m,t) \notin \{ (m_1,t_1), \ldots, (m_q,t_q) \}$$

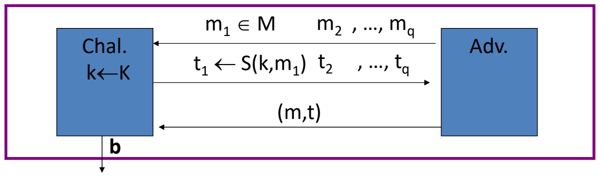
⇒ attacker cannot produce a valid tag for a new message

⇒ given (m,t) attacker cannot even produce (m,t') for $t' \neq t$

6

Secure MACs

- For a MAC I=(S,V) and adv. A define a MAC game as:



b=1 if V(k,m,t) = `yes' and $(m,t) \notin \{ (m_1,t_1), \ldots, (m_q,t_q) \}$

b=0 otherwise

Def: I=(S,V) is a **secure MAC** if for all "efficient" A:

$\text{Adv}_{\text{MAC}}[A,I]$ = Pr[Chal. outputs 1] is "negligible."

7

Example

Let I = (S,V) be a MAC.

Suppose S(k,m) is always 5 bits long

Can this MAC be secure?

Tag length:

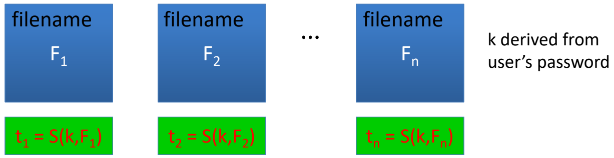
64bits, 96bits(TLS), 128bits

- ○ No, an attacker can simply guess the tag for messages
- ○ It depends on the details of the MAC
- ○ Yes, the attacker cannot generate a valid tag for any message
- ○

8

Example: protecting system files

Suppose at install time the system computes:



Later a virus infects system and modifies system files

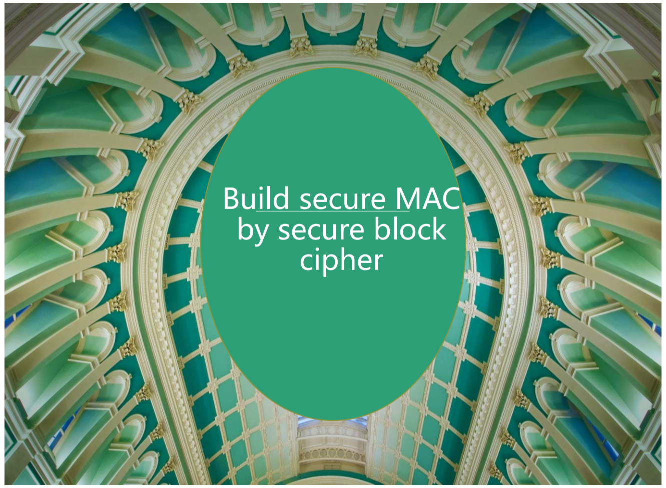
User reboots into clean OS and supplies his password
- Then: secure MAC ⇒ all modified files will be detected

9

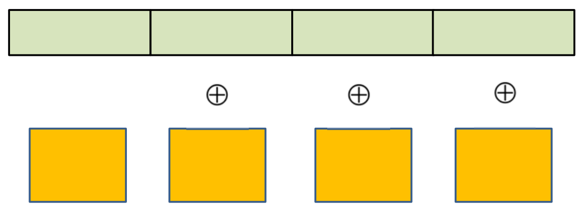
Examples

- AES: a MAC for 16-byte messages.
- Main question: how to convert Small-MAC into a Big-MAC ?
- Two main constructions used in practice:
  - **CBC-MAC** (banking – ANSI X9.9, X9.19, FIPS 186-3)
  - **HMAC** (Internet protocols: SSL, IPsec, SSH, …)

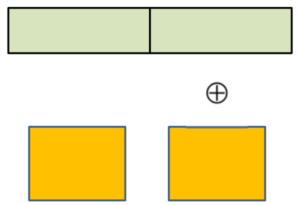
10



11



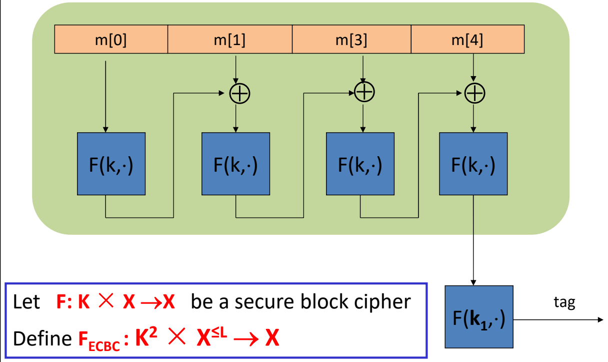
12



13

Construction 1: encrypted CBC-MAC

raw CBC



Let $F: K \times X \to X$ be a secure block cipher

Define $F_{ECBC}: K^2 \times X^{\leq L} \to X$

14

Why the last encryption step in ECBC-MAC?

Suppose we define a MAC $I_{RAW} = (S,V)$ where

$$S(k,m) = rawCBC(k,m)$$

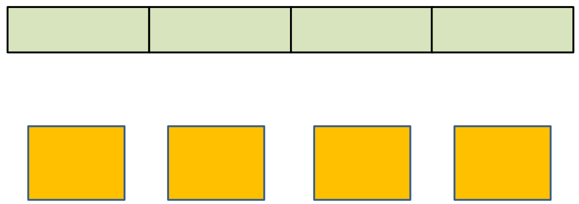
Then I_{RAW} is easily broken using a 1-chosen msg attack.

Adversary works as follows:

- Choose an arbitrary one-block message $m \in X$
- Request tag for m. Get $t = F(k,m)$
- Output t as MAC forgery for the 2-block message $(m, t \oplus m)$

Indeed: $rawCBC(k, (m, t \oplus m)) = F(k, F(k,m) \oplus (t \oplus m)) = F(k, t \oplus (t \oplus m)) = t$

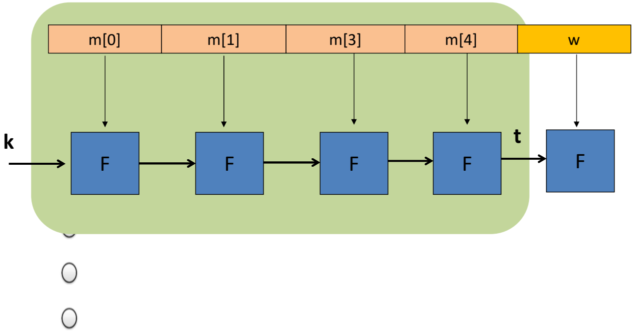
15



16

NMAC: why do we need the last step?

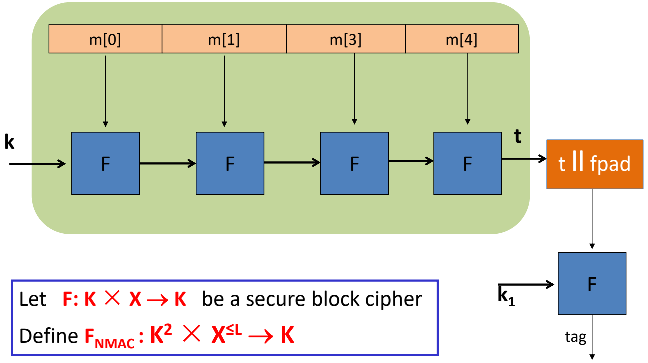
cascade



17

Construction 2: NMAC (nested MAC)

cascade



Let $F: K \times X \to K$ be a secure block cipher

Define $F_{NMAC}: K^2 \times X^{\leq L} \to K$

18

Why the last encryption step in NMAC?

NMAC: suppose we define a MAC I = (S,V) where

S(k,m) = cascade(k, m)

cascade(k, m) → cascade(k, m||w) for any w

- ○ This MAC is secure
- ○ This MAC can be forged without any chosen msg queries
- ○ This MAC can be forged with one chosen msg query
- ○ This MAC can be forged, but only with two msg queries

19

Comparison

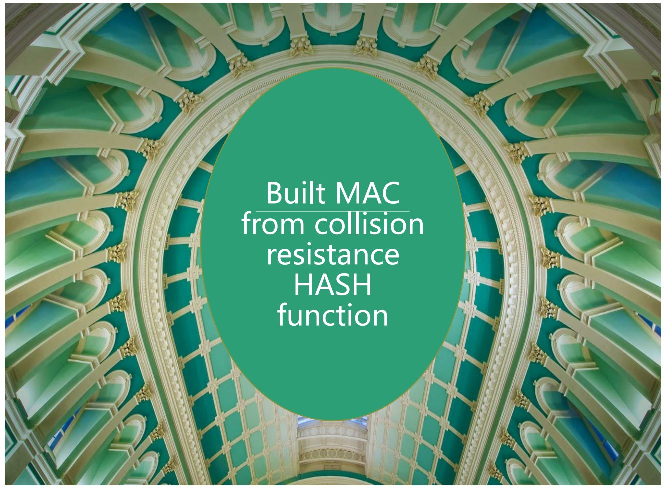
ECBC-MAC is commonly used as an AES-based MAC

- CCM encryption mode (used in 802.11i)
- NIST standard called CMAC

NMAC not usually used with AES or 3DES

- Main reason: need to change AES key on every block requires re-computing AES key expansion
- But NMAC is the basis for a popular MAC called HMAC

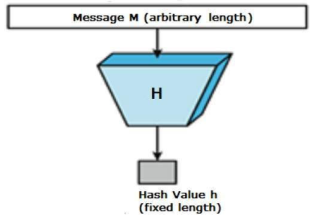
20



21

Hash function

- ✓ 输入长度可以是任意长度
- ✓ 输出是固定长度
- ✓ 给出任意的报文可以很轻松的算出哈希函数H（x）
- ✓ 哈希函数是个不可逆的函数，就是给出一个Y，其中Y=H（x）,你完全不能通过Y去推算出x
- ✓ 哈希函数不存在碰撞，就是不存在任意一个x',使H（x'）=H（x）



Example: SHA-256 (outputs 256 bits)

$md5("hello\ world") = 5eb63bbbe01eeed093cb22bb8f5acdc3$

22

实验：Linux中如何生成和校验hash值

- md5sum
- sha256sum

23

Collision Resistance

Let H: M →T be a hash function (|M| >> |T|)

A **collision** for H is a pair m_0 , $m_1 \in M$ such that:

$$H(m_0) = H(m_1) \quad \text{and} \quad m_0 \neq m_1$$

A function H is **collision resistant** if for all "eff" algs. A:

$Adv_{CR}[A,H]$ = Pr[A outputs collision for H]

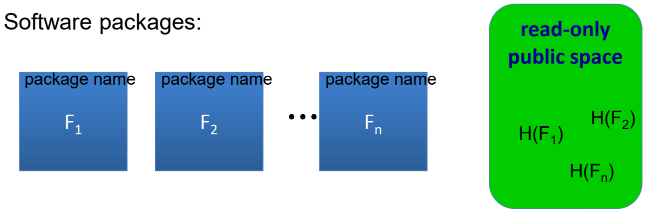
is "neg".

Example: SHA-256 (outputs 256 bits)

24

Protecting file integrity using C.R. hash

Software packages:



When user downloads package, can verify that contents are valid

H collision resistant ⇒
attacker cannot modify package without detection

no key needed (public verifiability), but requires read-only space

25

Generic attack on hash functions

Let H: M → $\{0,1\}^n$ be a hash function (|M| >> 2^n)

Generic alg. to find a collision **in time** **O($2^{n/2}$)** hashes

Algorithm:

1. Choose $\mathbf{2^{n/2}}$ random messages in M: $m_1, \ldots, m_{2^{n/2}}$
2. For i = 1, …, $2^{n/2}$ compute $t_i = H(m_i) \in \{0,1\}^n$
3. Look for a collision ($t_i = t_j$). If not found, got back to step 1.

How well will this work?

26

生日攻击

生日悖论：
在***k***个人中至少有两个人的生日相同的概率大于**0.5**时，**k**至少多大？

27

The birthday paradox

生日悖论：至少随机选多少位学生，会有两位的生日相同的概率大于50%。用数学语言描述如下：

Let $r_1, \ldots, r_n \in \{1,\ldots,B\}$ be indep. identically distributed integers.

Thm: when **n**= 1.2 × $\mathbf{B^{1/2}}$ then $\Pr[\,\exists i\neq j:\ r_i = r_j\,] \geq ½$

Proof: (for uniform indep. $r_1, \ldots, r_n$)

$$\Pr[\exists i\neq j: r_i = r_j] = 1 - \Pr[\forall i\neq j: r_i \neq r_j] = 1 - \left(\frac{B-1}{B}\right)\left(\frac{B-2}{B}\right)\cdots\left(\frac{B-n+1}{B}\right) =$$

$$= 1 - \prod_{i=1}^{n-1}\left(1-\frac{i}{B}\right) \geq 1 - \prod_{i=1}^{n-1} e^{-i/B} = 1 - e^{-\frac{1}{B}\sum_{i=1}^{n-1} i} \geq 1 - e^{-n^2/2B}$$

$1-x \leq e^{-x}$

$$\geq 1 - e^{-0.72} = 0.53 > \frac{1}{2}$$

$\frac{n^2}{2B} = 0.72$

28

生日攻击

- 生日攻击是基于下述结论：设散列函数 h 有2^m个可能的输出（即输出长 m 比特），如果 h 的 k 个随机输入中至少有两个产生相同输出的概率大于**0.5**，则
-
- $$k \approx \sqrt{2^m} = 2^{m/2}$$

对SHA-1，$k = 2^{160/2} = 2^{80}$

对SHA-256，$k = 2^{256/2} = 2^{128}$

对SHA-512，$k = 2^{512/2} = 2^{256}$

29

Generic attack

H: M $\rightarrow \{0,1\}^n$. Collision finding algorithm:

1.Choose $\mathbf{2^{n/2}}$ random elements in M: $m_1, \ldots, m_{2^{n/2}}$

2.For i = 1, …, $2^{n/2}$ compute $t_i = H(m_i) \in \{0,1\}^n$

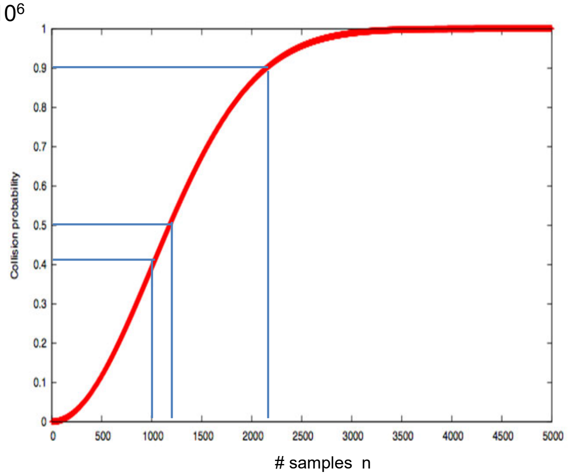
3.Look for a collision $(t_i = t_j)$. If not found, got back to step 1.

Expected number of iteration ≈ 2

Running time: $\mathbf{O(2^{n/2})}$ (space $O(2^{n/2})$)

30

B=10^6



31

王小云和她的团队

- 2005年2月，王小云和她的研究小组证明，散列函数SHA-1的强抗碰撞性没有想象中那么强。
- 能在2^{69}量级的运算内找到两个字符串x和y，使SHA-1(x) = SHA-1(y)
- 2005年8月，又降至2^{63}
- 过去认为是安全的方法会因为新技术和新方法的突破而变得不再安全。

32

Sample hash functions

AMD Opteron, 2.2 GHz (Linux)

	function	digest size (bits)	Speed (MB/sec)	generic attack time
	SHA-1	160	153	2^{80}
NIST standards	SHA-256	256	111	2^{128}
NIST standards	SHA-512	512	99	2^{256}
NIST standards	Whirlpool	512	57	2^{256}

33

34

HMAC

- **S(k, m) = H(k ll m)**
- Hash函数扩展长度攻击 （NMAC）

35

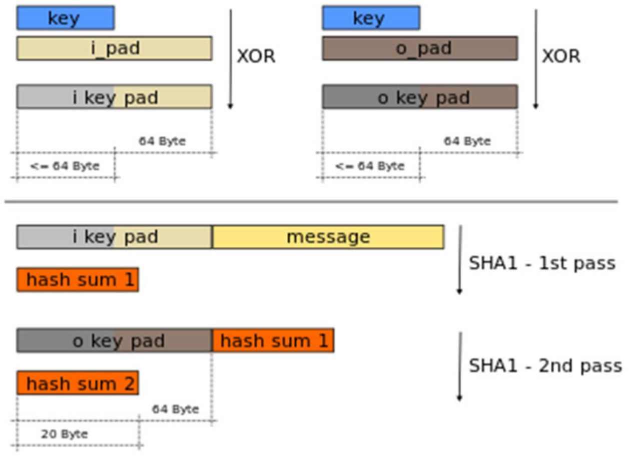
Standardized method: HMAC (Hash-MAC)

Most widely used MAC on the Internet.

Building a MAC out of a hash function:

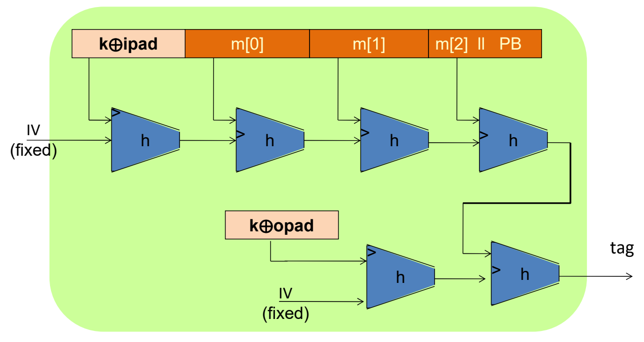
HMAC: $\mathbf{S(k, m) = H(k \oplus opad \parallel H(k \oplus ipad \parallel m))}$

36



37

HMAC in pictures



38