Message Authentication Codes (MACs)

Debdeep Mukhopadhyay

Assistant Professor

Department of Computer Science and Engineering

Indian Institute of Technology Kharagpur

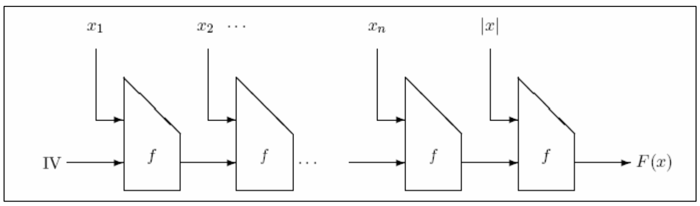
INDIA -721302

Objectives

- **Security Notions of MACs**
- **NMACs and HMACs**
- **CBC-MACs**

Unkeyed Hash Functions

- **We have studied un-keyed hash functions**
  - **Merkle Damgard Construction**
  - **iterative in nature**



What are MACs?

- **Message Authentication Codes**
- **They are keyed hash functions**
- **Needed for message integrity**
  - **One possible construction could be to make the IV (Initialization Vector) of hash functions secret.**

Constructing MAC by making IV secret

- **Consider for simplicity a hash function:**
  - **with no pre-processing steps**
  - **with no final output transformation.**
  - **Thus, every input message is a multiple of t, where compress: $\{0,1\}^{m+t} \rightarrow \{0,1\}^m$**
  - **Key K is of m bits**
- **Given x and $h_k(x)$ (MAC) we have to construct another valid pair.**
  - **Can we do that efficiently?**

Constructing MAC by making IV secret

- **$h_K(x)$=compress(K,x)**
- **Consider x||x', where x,x' are of t bits.**
- **Thus, $h_k(x||x')$=compress($h_k(x)$,x')**
  - **which can always be computed, even though key is secret!**
  - **this can be also attacked to those cases where padding is required and there is a pre-processing step.**

Hash with pre-processing step

- **Consider, $y=x||pad(x)$, such that $|y|=rt$**
- **Let w be any bit string:**
  - **st. $x'=x||pad(x)||w$**
  - **$y'=x||pad(x)||w||pad(x')$, $|y'|=r't$, $r'>r$**
- **Note that the attacker knows $z_r=h_K(x)$**

Computing $h_K(x')$ from $h_K(x)$

- **The attacker can obtain the value even without knowing K:**
  - **$z_{r+1}=compress(h_K(x)||y_{r+1})$**
  - **$z_{r+2}=compress(z_{r+1}||y_{r+2})$**

  …

  …
  - **$z_{r'}=compress(z_{r'-1}||y_{r'})$**
  - **$h_K(x')=z_{r'}$**

What is security of MAC?

- **Attacker is allowed to request for q valid MACs on $x_1,x_2,\ldots,x_q$**
- **Thus he obtains the list:**

  $((x_1,y_1),(x_2,y_2),\ldots,(x_q,y_q))$
- **Forgery: If he is able to output (x,y), where x is not among the q values queried for, then we say that the pair is a forgery.**
- **If the probability is ε, then adversary is an (ε,q) forger.**

Nested MAC (NMAC)

Suppose that (X,Y,K,G) and (Y,Z,L,H) be two hash families.

The composition of these hash families is the hash family $(X,Z,M,G\circ H)$ in which M=K×L and $G\circ H=\{g\circ h: g\in G, h\in H\}$ where $(g\circ h)_{(K,L)}(x)=h_L(g_K(x))$ for all $x\in X$.

A Result

- The nested MAC is secure provided that the following two conditions hold:
  - H is a secured MAC, given a fixed unknown key.
  - G is collision-resistant, given a fixed unknown key.

Adversaries

- Three kinds of adversaries:
  - forger for the nested MAC (big MAC attack)
  - forger for the little MAC (small MAC attack)
  - collision finder for the hash, when the key is secret (unknown key collision attack)

Theorem

Suppose $(X, Z, M, G \circ H)$ is a nested MAC. Suppose there does not exist an $(\varepsilon_1, q+1)-$ *collision attack* for a randomly chosen function $g_K \in G,$ when the key K is secret. Further, suppose that there does not exist an $(\varepsilon_2, q)-$ *forger* for a randomly chosen function $h_L \in H,$ where L is secret. Finally suppose there exists an $(\varepsilon, q)-$ *forger* for the nested MAC, for a randomly chosen function $(g \circ h)_{(K,L)} \in G \circ H.$ Then $\varepsilon \le \varepsilon_1 + \varepsilon_2.$

- **Result Proved in the class…**

Hash based MAC (HMAC)

- **HMAC is a nested MAC algorithm proposed by FIPS Standard.**
- **It constructs a MAC from an unkeyed hash function, namely SHA-1.**
  - **K: 512 bit key.**
  - **x is the message to be authenticated.**
  - **ipad and opad are 512 bit constants.**

HMAC

- **ipad=3636…36; opad=5C5C…5C**
- **Thus the 160 bit MAC is defined as follows:**

$$HMAC_K(x) = SHA-1((K \oplus opad) \,||\, SHA-1((K \oplus ipad \,||\, x))$$

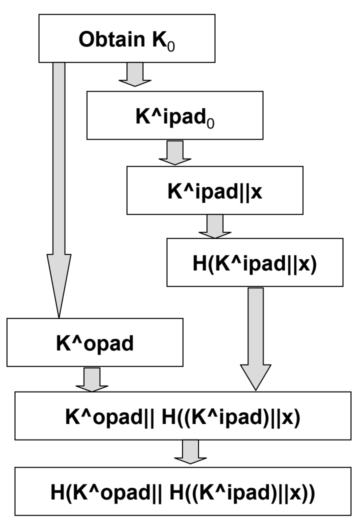


Illustration of the HMAC function

Security Arguments

- **First application of SHA-1 is assumed to be unknown key collision resistant.**
- **Second application of SHA-1 is assumed to be a secured MAC.**
- **Second SHA-1 needs only one compress function to be computed.**
- **Note that the "extension attack" is prevented in NMAC (or HMAC) because h_L avoids the exposure of $g_K(x)$.**

CBC-MAC

Each is of block length t

CBC-MAC(x, K)

denote $x = x_1 \| \cdots \| x_n$
IV $\leftarrow 00 \cdots 0$
$y_0 \leftarrow$ IV
for $i \leftarrow 1$ **to** n
  do $y_i \leftarrow e_K(y_{i-1} \oplus x_i)$
return (y_n)

Endomorphic Block Cipher

Attack on CBC-MAC

Set $q \approx 1.17 \times 2^{t/2}$ be an integer.
Choose q distinct bit strings of length t, which we denote $x_1^1, \ldots, x_1^q$.
Choose q random bit strings of length t, which we denote $x_2^1, \ldots, x_2^q$.
Let $x_3, \ldots, x_n$ be fixed bit strings of length t.
Construct: $x^i = x_1^i \| \ldots \| x_n^i$, for $1 \le i \le q$.
Here for $3 \le k \le n$, $x_k = x_k^i$, for each i.
Note that $x^i \ne x^j$ if $i \ne j$, as $x_1^i \ne x_1^j$.

Attack on CBC-MAC

- **The attacker now queries the hash value of the q, x^i values.**
- **Due to the Birthday Paradox, there is a collision with probability ½.**
- **Let $h_K(x^i)=h_K(x^j)$. This happens if and only if $y_2^i=y_2^j$, which happens if and only if :**

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

Attack on CBC-MAC

- Let x_δ be a non-zero bit string of length t.
- Define: $v = x_1^i \,||\, (x_2^i \oplus x_\delta) \,||\, \cdots \,||\, x_n^i$

  and $w = x_1^j \,||\, (x_2^j \oplus x_\delta) \,||\, \cdots \,||\, x_n^j.$
- The attacker now requests the MAC of v.
- The MAC of w also is the MAC of v.
- So, he publishes (w, MAC of v) as a valid pair.
- Thus, we have an (1/2, $O(2^{t/2})$)-forger.

Points to Ponder

- What would have happened if the hash function g, in the NMAC construction, would have been unkeyed?
- Why are different ipad and opads used?

References

- **D. Stinson, Cryptography: Theory and Practice, Chapman & Hall/CRC**
- **M. Bellare, R. Canetti, H. Krawczyk, "*Keying Hash Functions for Message Authentication*", *1996***

Next Days Topic

- **More Number Theoretic Results**