Cryptography

Cunsheng Ding, HKUST

COMP3631

Lecture 20: Keyed Hash Functions and the HMAC

Main Topics of This Lecture

1. Keyed hash functions
   - The definition of keyed hash functions
   - An important security protocol using a keyed hash function h_k
   - Security requirements for the underlying keyed hash function h_k in the security protocol
2. The HMAC, a method for converting a hash function and a secret key into a keyed hash function

Recall that hash functions were discussed in Lectures 16 and 17.

Part I: Keyed Hash Functions

Keyed Hash Functions

Definition: A **keyed hash function** h_k is a function from a **subset** of A^* to B^ℓ, where k is a secret parameter from a key space $\mathcal{K}$, A and B are two finite alphabets, and ℓ is a fixed positive integer.

For any x, $h_k(x)$ is called the **hash value** or **message authentication code (MAC)**.

Remark: A keyed hash function h_k is a family of hash functions $\{h_k : k \in \mathcal{K}\}$. It is a **secret function** without knowing the secret key k.

Example: $h_k(m) = \text{SHA256}(m) \oplus k$, where $k \in \mathcal{K} := \{0,1\}^{256}$.

Applications: data authentication (data integrity) and data origin authentication.

Attention: In many applications the hash algorithm $h_k(x)$ (i.e., the internal structure) is known but the key k is unknown to attackers.

Design Requirements for Keyed Hash Functions

Question: What are the security requirements for a keyed hash function used in a security system?

Answer: It depends on the specific security system using it.

In other words, security requirements for a keyed hash function differ from security system to security system.

An Important Security Protocol (1)

Suppose that Alice and Bob share a secret key for a keyed hash function.

$$\text{Alice} \Longrightarrow m||h_k(m) \Longrightarrow \text{Bob}$$

Suppose that an adversary has total control of the communication channel.

Remark: The secret hash value $h_k(m)$ is called a **message authentication code (MAC)**.

Attention: This protocol is used in real-world security systems such as IP Security, Secure Socket Layer, Transport Layer Security, HTTPS, Secure Shell, and VPNs.

Question: Which security services could this protocol provide?

An Important Security Protocol (2)

Suppose that Alice and Bob share a secret key for a keyed hash function.

$$\text{Alice} \Longrightarrow m||h_k(m) \Longrightarrow \text{Bob}$$

Question: Which security services could this protocol provide?

Answer: If the keyed hash function is "well designed", the protocol can provide the data integrity and data origin authentication service. Why?

- Only Alice and Bob can create such a piece of data, as others do not know the secret key k.
- Any modification of the text $m||h_k(m)$ will destroy the relation between the first part m and the second part $h_k(m)$ of the original message $m||h_k(m)$ with high probability and can be detected with high probability.

An Important Security Protocol (3)

Suppose that Alice and Bob share a secret key for a keyed hash function.

$$\text{Alice} \implies m||h_k(m) \implies \text{Bob}$$

Suppose that an adversary has total control of the communication channel.

Question: How should Bob authenticate a received text c?

Answer: See the next page.

An Important Security Protocol (4)

Suppose that Alice and Bob share a secret key for a keyed hash function.

$$\text{Alice} \Longrightarrow m||h_k(m) \Longrightarrow \text{Bob}$$

Authentication checking by Bob: Bob partitions a received text $c = c_1||c_2$, where c_2 has the same length as the hash value. Then he computes $h_k(c_1)$. Finally, he compares $h_k(c_1)$ with c_2.

If $h_k(c_1) \neq c_2$, he is sure that the text c was not created by Alice.

If $h_k(c_1) = c_2$, he concludes that the text c was created by Alice.

- In the case that $h_k(c_1) = c_2$, Bob's conclusion could be wrong. But if h_k is "**well designed**" (to be explained later), the probability of making a wrong conclusion in this case should be very small.

Question: How should h_k be designed so that the probability is small?

An Important Security Protocol (5)

Suppose that Alice and Bob share a secret key for a keyed hash function.

$$\text{Alice} \implies m||h_k(m) \implies \text{Bob}$$

Question: How should the h_k in this protocol be designed?

Answer: It depends on feasible attacks on this protocol.

An Important Security Protocol (6)

Suppose that Alice and Bob share a secret key for a keyed hash function.

$$\text{Alice} \Longrightarrow m||h_k(m) \Longrightarrow \text{Bob}$$

The random attack: Carol randomly picks up a text c and sends it to Bob. The attack is successful if Bob accepts c as a text from Alice.

Success probability: Probability$(c_2 = h_k(c_1))$, where c_2 is a random string having the same length as the hash value, and c_1 is a random string.

Security requirements w.r.t. this attack:

- The length of the hash value is long enough (say, 128 bits). Why?
- The hash values are more or less uniformlly distributed for each fixed key k. Why?

An Important Security Protocol (7)

Suppose that Alice and Bob share a secret key for a keyed hash function.

$$\text{Alice} \implies m||h_k(m) \implies \text{Bob}$$

The second attack: Carol stops the message $m||h_k(m)$ in the middle of the transmission, and tries to find a different message m' such that $h_k(m) = h_k(m')$ by analysing the structure of the keyed hash function. Then Carol sends the modified message $m'||h_k(m)$ to Bob.

Security requirements w.r.t. this attack: For a given message m, it should be computationally infeasible to find a different message m' such that $h_k(m) = h_k(m')$ without knowing the key k.

Remark: This attack may work if h_k is not well designed. See the next slide for an example.

The First Example of Keyed Hash Functions

Example: Let E_k be the encryption transformation of one-key cipher and h be a hash function. Then $h_k := E_k \circ h$ is a keyed hash function, where $\circ$ denotes the function composition.

Remark: The hash function h and the internal structure of the cipher are publicly known.

Warning: Any pair of collision of h must be a pair of collision of h_k. In such case it is possible to find a pair of collision for a keyed hash function h_k without knowing the secret key k! See the second attack on the previous slide.

Security requirements for the two building blocks of the h_k used in this protocol: The one-key cipher should be computationally secure and h should have the weak collision resistance property.

An Important Security Protocol (8)

Suppose that Alice and Bob share a secret key for a keyed hash function.

$$\text{Alice} \Longrightarrow m||h_k(m) \Longrightarrow \text{Bob}$$

The third attack: After receiving one or several $m_i||h_k(m_i)$, Carol tries to find the secret key k shared by Alice and Bob.

If Carol finds the key k, she can impersonate Alice and Bob at any time.

Security requirements w.r.t. this attack: Given one or several $m_i||h_k(m_i)$, it should be computationally infeasible to recover the key k.

Remark: This attack may work if h_k is not well designed. See the next slide for an example.

The Second Example of Keyed Hash Functions

Example: Let h be a hash function with hash values of ℓ bits. Define $h_k(m) := h(m) \oplus k$, where k is a secret key of ℓ bits. Then h_k is a keyed hash function.

Question: Assume that the hash function h used in this keyed hash function is publicly known. Is the authentication protocol using this h_k secure with respect to the third attack?

Answer: No. One pair $(m, h_k(m))$ can be used to recover the secret key $k = h_k(m) \oplus h(m)$.

Remark: This is a simple method for converting a hash function h and a secret key k into a keyed hash function. **It is a very bad construction!**

An Important Security Protocol (9)

Suppose that Alice and Bob share a secret key for a keyed hash function.

$$\text{Alice} \implies m||h_k(m) \implies \text{Bob}$$

In view of the three attacks on this protocol, it is required that h_k have the following properties:

- Given one or several $m_i||h_k(m_i)$, it should be computationally infeasible to recover the key k.
- For any given message m, it should be computationally infeasible to find a different message m' such that $h_k(m) = h_k(m')$ without knowing the key k.
- The hash values should be long enough and should be more or less uniformly distributed for each key k.

Part II: the HMAC

A Summary of the HMAC

- HMAC is a method for converting a hash function and an authentication key into a keyed hash function.
- It is widely used in real-world security systems.

HMAC: A Specific Construction

- h a hash function with good confusion and diffusion, with n-bit hash value.
- b is a chosen positive integer and $8|b$.
  - The upper bound on the key size in bits.
- K is the secret key with size at most b bits.
- $\overline{K}$ is K padded with 0's on the left so that the result is b bits in length.
- ipad $=$ 00110110 repeated $b/8$ times.
- opad $=$ 01011100 repeated $b/8$ times.

$$\text{HMAC}_{(h,K)}(m) = h\{(\overline{K} \oplus \text{opad})||h[(\overline{K} \oplus \text{ipad})||m]\}.$$

Some Questions about the Design of HMAC

- What is the purpose of using the two constant binary strings?
  - This is to change half of the bits in $\overline{K}$ in two different ways with ipad and opad and then mix the bits in $\overline{K} \oplus$ ipad and $\overline{K} \oplus$ opad with the message bits in m using the given h with good confusion and diffusion.
- Why are the ipad and opad designed in that way?
  - To change half of the bits in $\overline{K}$ in two different ways.
  - To make the two changes in $\overline{K} \oplus$ ipad and $\overline{K} \oplus$ opad **"complementary"**.
- Why h is used twice?
  - For efficiency and security consideration.

Security of the HMAC

Conclusion: It depends in some way on the cryptographic strength of the underlying hash function. For details, see:

M. Bellare, R. Canetti and H. Krawzyk, *Keying hash functions for message authentication,* Advances in Cryptology – Crypto' 96, LNCS 1109, Springer-Verlag, 1996.

Online Demo of the HMAC

In the following URL, you find the demo of the HMAC with some hash functions:

`https://8gwifi.org/hmacgen.jsp`