Disposition 2: Authenticity

Mathias Ravn Tversted

January 6, 2020

Table of contents

Authentication, Secret-Key systems

Definition of security: Definition 6.1

Consider any adversary who runs in time much less than what exhaustive key search would take. The authentication scheme is secure if no such adversary can play the game specified above and in the end produce a message m_o and a MAC c_o such that $V_K(m_o, c_o) = \top$ and $m_o \notin \{m_1, ...M_t\}$

In more human language: even if the adversary has seen some number of messages with valid MACs, he still cannot come up with a message that was not sent and a valid MAC for that message. This is the case even if the adversary can choose the valid messages to be sent.

Cipher Block Chaining MAC

CBC-MAC: If a Block Cipher, such as AES is secure, then it can be used to construct a MAC.
How is it built: We encrypt the same message, but using an all-zero *IV*, and use the last block as the MAC.
Intuition: The last block depends on the input in a complicated way, and is therefore extremely difficult to forge.

Authenticity

A secret-key system for Authentication consists of 3 algorithms: G, MAC, V. G generates a key, MAC authenticates messages, V verifies received messages. G usually produces a key by just outputting a random bit string of fixed length. MAC takes as input m, K and outputs a MAC. $c = MAC_K(m)$. Then m, c is sent who will run $V_K(m, c) = \{\top, \bot\}$. We require the following

$$V_k(m, MAC_K(m)) = \top$$

Difference between signature and MAC

- Secret Key schemes rely on all parties having the same keys. Only parties that have the key can be convinced that they are talking with each other, but they cannot convince others (who do not share the key)
- With public key schemes, everyone can verify the message because everyone is allowed to have the public key.

Why simplistic RSA signatures are not secure

- RSA relies on the identity
  $s^e \bmod n = (m^d \bmod n)^e \bmod n = m$
- But an adversary can choose an s such that
  $m = s^e \bmod n$
- Now m would be a “signed” message

Hash functions: Properties and examples

Hash functions can solve speed and security problems. Their desirable properties for hash functions are as follows

- Should take message of any length
- Should produce output of fixed length
- Speed similiar to secret-key systems
- Collisions should be hard to produce. That is, hard to find $x, y : x \neq y$ and $h(x) = h(y)$.

Examples of hash functions in use are

- **SHA-256**: Produces 256-bit outputs
- **MD5**: Totally borken
- **SHA-1**: Collisions have been found

Hash-and-sign

Since hash functions typically produce output that is smaller than the input, cryptograhy on hashed inputs is typically faster. Since it is not computationally viable to produce collisions, it is hard to reverse the hash of a message. Therefore it is at least as convincing to sign a hash as the message itself. This leads to new algorithms

- $S'_{sk}(m) = S_{sk}(h(m))$
- $V'_{pk}(m, s) = V_{pk}(h(m), s)$

Now the previous attack still works, but having it sign forged messages is much harder.

Table-based

For a finite number of messages, sender and receive agrees in advance on a table of t-bit MACs. Finding a correct MAC happens with probability 2^{-t}. Finding a MAC requires guessing a random t-bit string.

Authenticity, Public-Key Systems

Public-Key systems

Consists of 3 algorithms, G, S, V.

- S authenticates (signs)
- V verifies message (just like SK-case)
- G generates key pair
- k_{sk} signs
- k_{pk} is used to verify messages
- Requires distribution of public keys

Definition 6.3: Definition of security

The adversary is given the public key pk. He is allowed to specify any number of messages that he wants, say $m_1, ...m_t$. And he is given a valid authenticators $c_1 = S_{sk}(m1), ...c_t = S_{sk}(m_t)$ for these messages. The public-key authentication scheme is secure if the adversary cannot efficiently produce a message m_o and an authenticator c_o such that $V_{pk}(m_o, c_o) = \top$ and $m_o \notin \{m_o, ..., m_t\}$.

Replay Attacks

A replay attack is when a message is intercepted, and is then sent again by an adversary. This means that even authenticated messages could be attacked. Some remedies include

- Storing all the messages. Takes up a lot of space
- Store last seen message with a counter. This has problems if messages arrive late
- Have receiver send a nonce R to sender. Send message + MAC computed from the message and nonce.