Message Authentication

Confidentiality is not always enough!

Goals: prevent
- Masquerade
- Content modification
- Sequence modification
- Timing modification
- Source repudiation
- Destination repudiation

There are 2 general techniques to achieve these goals:
- Message Authentication Codes (MACs)
- Digital signatures

MACs can help to prevent 1-4
Signatures can help to prevent 1-6

What about encryption? This can also help to prevent masquerade, content modification, and sequence modification, but only if it is clear to the receiver what is a valid message.
- Therefore, encryption alone cannot guarantee to provide message authentication
MAC: uses a secret key to generate a small, fixed-size block of data that is appended to the message

Message plus the MAC are transmitted to the intended recipient. The recipient computes a new MAC on the message, using the same secret key, and compares this new MAC with the received MAC.

- If the MACs are the same, and if only the sender and receiver know the key, then the receiver is assured of at least three things:

Benefits of using a MAC:

1.

2.

3.
Let the size of the message be $N$, the size of the MAC be $n$, and the size of the key be $k$.

In general, $N > n$ and $N > k$.

What about the relative sizes of $k$ and $n$?

If $k > n$: attacker may need to do several rounds (with several (message, MAC) pairs in order to find the correct key.

If $k < n$: a single known (message, MAC) pair should suffice.

Along with these sizes, the MAC designer also needs to be careful about the overall construction of the algorithm.

Requirements of a MAC function:

1.

2.

3.

4.
MAC constructions

1. MAC based on a block cipher (e.g., DES)
   - DES in CBC mode with an IV of zero: FIPS PUB 113; ANSI X9.17
   
   - Widely implemented and used, but end up encrypting the entire message twice, with two different keys, if you need confidentiality

2. MAC based on a hash function (e.g., MD5 or SHA-1)
   - HMAC: RFC 2104; FIPS PUB 198; mandatory-to-implement for IPsec; used in other Internet protocols, such as SSL
   
   - Becoming widely used.
   
   - More efficient than construction (1) since hashing is typically faster than encryption

So, what is a hash function…?
Hash function ("cryptographic hash function"; "one-way function")

- Accepts a variable-size message as input (note: no key)
- Produces a fixed-size output ("hash code", "hash value", "hash", "message digest", "checksum", "cryptographic checksum")
- The checksum is a function of all the bits in the message (a change to any bit or bits in the message results in a change to the checksum with high probability)

Requirements of a cryptographic hash function

1.

2.

3.

4.

Note that collisions are inevitable (they cannot be avoided), but it must be infeasible to find any.
The Birthday Paradox

What is the minimum value of $k$ such that the probability is greater than 0.5 that at least two people in a group of $k$ people have the same birthday? (Assume each birthday is equally likely and ignore February 29th.)

(see Stallings, Figure 11.10)

Applying this to hash functions, we have the following:

Suppose we have a hash function, $H$, with an $m$-bit checksum. If $H$ is applied to $k$ random inputs, what must be the value of $k$ so that there is at least half a chance of finding a collision (i.e., $\text{Prob}[ H(x) = H(y) ] \geq 0.5$, for some inputs $x, y$)?

Note: these calculations give an upper bound (not a lower bound!) for the security of the hash function.
General structure of hash functions

There have been many different structures proposed over the years, but the most common (proposed by Merkle in 1979) is referred to as an iterated hash function:

- Input message is partitioned into $L$ blocks of $b$ bits each
- The final block (possibly padded) includes the message length
- The algorithm makes repeated use of a compression function, $f$, that takes two inputs ($b$-bit block and $n$-bit “chaining variable”) and produces an $n$-bit output (the updated chaining variable)
- The final chaining variable is the checksum

If $f$ is collision resistant, then so is the hash function. Therefore, can concentrate on constructing a good compression function…
Secure Hash Algorithm (SHA)

- SHA
- SHA-1
- SHA-2 (SHA-256; SHA-384; SHA-512)
- SHA-3

SHA-512

- Messages up to $2^{128}$ bits in length
- Input processed in 1024-bit blocks (iterated hash function)
- Produces a 512-bit checksum

Processing: Steps 1-5

(see Stallings, Figure 12.1, for a picture of SHA-512)
(see Stallings, Figure 12.4, for creation of 80-word seq.)
(see Stallings, Figure 12.2, for a picture of $F$)
(see Stallings, Figure 12.3, for a picture of one round)

Note: researchers have also explored the idea of using a block cipher as the compression function. A good example of this approach is Whirlpool
HMAC

Design objectives:
- To use, without modifications, available hash functions
- To allow easy replaceability of the hash function
- To preserve the performance of the hash function
- To use and handle keys in a simple way
- To have a well-understood cryptographic analysis of the strength of the MAC based on assumptions about the hash function

Algorithm:

\[
\text{HMAC}(k, m) = H[(k \oplus \text{opad}) || H[(k \oplus \text{ipad}) || m]]
\]

HMAC security:

- It has been proven that the probability of a successful attack on HMAC is equivalent to finding collisions in the hash function