Introduction to Cryptography

Alvin Lin

January 2018 - May 2018

Message Authentication and Hash Functions

Message authentication is a procedure to verify that received messages from the alleged source have not been altered. Message authentication may also verify sequencing and timeliness. A **digital signature** is an authentication technique that also includes measures to counter repudiation by the source.

Message Authentication Functions

- Message encryption: the ciphertext of the entire message serves as its authenticator.
- Hash function: a function that maps a message of any length into a fixed-length hash value which serves as the authenticator.
- Message authentication code (MAC): a function of the message and a secret key that produces a fixed-length value that serves as the authenticator.

Hash Functions

A hash function H accepts a variable-length block of data M as input and produces a fixed-size hash value. The principal objective of a hash function is data integrity. Cryptographic hash functions are algorithms for which it is computationally infeasible to find a data object that maps to a pre-specified hash result (one-way property), or two data objects that map to the same hash result (the collision-free property). Hash functions are also commonly used for:

- Creating one-way password files

- Intrusion detection and virus detection
- Constructing a pseudorandom function or pseudorandom number generator

Message Authentication Code (MAC) Functions

A message authentication code is also known as a keyed has function. It is typically used between two parties that share a secret key to authenticate information exchanges. A MAC function takes a secret key and a data block as input and produces a hash value referred to as the MAC, which is then associated with the protected message. An attacker who alters the message will be unable to alter the associated MAC value without knowledge of the secret key.

Digital Signatures

Digital signatures operate similar to that of a MAC. The hash value of a message is encrypted with a user's private key. Anyone who knows the user's public key can verify the integrity of the message. An attack who wishes to alter the message would need to know the user's private key.

Digital Signatures vs MAC

The combination of hashing and encryption results in an overall function that is conceptually a MAC. Its goal is to be a function $E(K, H(M))$ for a variable length message M and a secret key K to produce a fixed length output that is secure against an opponent who does not know the secret key. In practice, specific MAC algorithms are designed to be more efficient that an encryption algorithm.

Attacks on Hash Functions

Brute Force Attacks do not depend on the specific algorithm, they only depend on the bit length. In the case of a hash function, the attack depends only on the bit length of the hash value. This method involves picking values and random and hashing each one until a collision occurs.

Cryptanalysis attacks are based on weaknesses in a particular cryptographic algorithm. They seek to exploit some property of the algorithm to perform some attack other than an exhaustive search.

If we have a hash with an output digest length n, we can find a preimage using a brute force search in 2^n operations. We can find a second preimage also with 2^n

operations. Thus we can find a collision with $\sqrt{2^n} = 2^{\frac{n}{2}}$ operations using a brute force search (birthday attack). Collision resistance causes most of the security problems with hash functions. It is related to the birthday problem.

Birthday Paradox

How many people are needed such that two of them have the same birthday with a probability of 0.5? Conventionally, one would think $\frac{365}{2} = 183$, but in fact only 23 people are required because the search takes $\approx \sqrt{2^n}$ steps. To deal with this paradox, hash functions need an output size of at least 160 bits.

Using Collision Attacks

For attacks based on hash collisions, an adversary wishes to find two messages or data blocks that yield the same hash function. The effort required is explained above by the birthday paradox. The strategy to use is as follows:

1. Suppose Alice is prepared to sign a correct message X by appending the appropriate n-bit hash code and encrypting that hash code with the private key.

2. An opponent generates $2^{\frac{n}{2}}$ varations X' of X, all with essentially the same meaning, and stores the messages and their hash values.

3. The opponent prepares a deceptive message Y for which Alice's signature is desired.

4. The opponent generates minor variations Y' of Y, all of which convey essentially the same meaning. For each Y', the opponent computes $H(Y')$ and checks for matches with any of the $H(X')$ until a match is found. The process continues until a Y' is generated with a hash value equal to the hash value of one of the X' values.

5. The opponent offers the valid message variation to Alice for signing. The signature can then be attached to to the fraudulent variation for transmission to the intended recipient.

Because the two varations have the same hash code, they will produce the same signature and the opponent is assured of success even though the encryption key is not know. The generation of many variations that convey the same meaning is not difficult. An opponent can insert extra spaces in between words or simply reword the message while retaining the meaning.

Examples of Cryptographic Hash Functions

Hash Algorithm	Date	Digest Length (bits)	Security
Message Digest 4 (MD4)	1990	128	Broken
Message Digest 5 (MD5)	1991	129	Broken
Secure Hash Algorithm 0 (SHA-0)	1993	160	Broken
Secure Hash Algorithm 1 (SHA-1)	1995	160	Broken
Secure Hash Algorithm 2 (SHA-2)	2002	224,256,384,512	Questionable
Secure Hash Algorithm 3 (SHA-3)	2015	224,256,384,512	Secure

Hash Functions Based On Cipher Block Chaining

A number of proposals have been made for hash functions based on using a cipher block chaining technique, but without using a secret key. One of the first proposals was that of Rabin, where we divide a message M into fixed sized blocks $M_1, M_2, \ldots, M_n$ and use a symmetric encryption system such as DES to compute the hash code h as:

$$\begin{aligned} H_0 &= \text{initial value} \\ H_i &= E(M_i, H_{i-1}) \\ h &= H_n \end{aligned}$$

This is similar to the cipher block chaining technique but uses no secret key. As with any hash code, this scheme is subject to the birthday attack. If the encryption algorithm is DES and only a 64-bit hash code is produced, the system is susceptible.

The Merkle-Damgård Construction

The Merkle-Damgård Construction was invented by Ralph Merkle and Ivan Damgård working independently. The message to hash is extended with padding bits (1000,000+) plus the message length. The padded message is fed through an iterated compression function. In 1989, Merkle and Damgå proved that if the compression function is collision-resistance, then the whole hash funcion will be collision resistance. Because of this proof, most hash functions proposed since then use the Merkle-Damgård construction or a variation. Possible attacks:

- Distinguishing attack
- Length extension attack

- Partial-message collision attack
- Fixed-point attack
- Multicollision attack
- Expandable message attack
- Herding attack

You can find all my notes at `http://omgimanerd.tech/notes`. If you have any questions, comments, or concerns, please contact me at alvin@omgimanerd.tech