Cryptographic hash functions

A hash function $h(\mathtt{m})$ is a message digest; in some sense, the message is condensed. Hash functions are routinely used to check integrity or for error detection of transmitted messages. Nick and Alex must agree on a hash function. If Nick is sending a message to Alex, he might create a hash of the message and transmit it along with the message. After receiving the message, Alex creates a hash of the message that he received using the hash function that he and Nick have agreed to use. The two hashes should be the same. If they are, Alex can assume that the message has not been altered intentionally or unintentionally during transmission.

Hash functions should accept messages of any length as input, produce a fixed-length output, and be fast.

A hash function that will be used for cryptographic purposes must have some other properties:

1. A cryptographic hash function should be ***one-way***. Knowing an output h of the hash function it should computationally infeasible to find a message $\mathtt{m}$ which hashes to that output; i.e., for which $h(\mathtt{m}) = h$. (This property is called ***pre-image resistant***.)

The number of inputs is much larger than the number of outputs; so, there will exist messages $\mathtt{m}_1$ and $\mathtt{m}_2$ such that $\mathtt{m}_1 \neq \mathtt{m}_2$ but $h(\mathtt{m}_1)=h(\mathtt{m}_2)$. Such an occurrence is called a ***collision***. Collisions will occur, but collisions should be unlikely.

2. A cryptographic hash function should also be ***second pre-image resistant*** – given a message $\mathtt{m}_1$, it should be computationally infeasible to find another message $\mathtt{m}_2$ with $\mathtt{m}_1 \neq \mathtt{m}_2$ having $h(\mathtt{m}_1)=h(\mathtt{m}_2)$. This is also called ***weak collision resistant***.

3. A cryptographic hash function should be ***strongly collision resistant***. It should be computationally infeasible to find two different inputs that have the same hash; i.e., it should be computationally infeasible to find messages $\mathtt{m}_1 \neq \mathtt{m}_2$ having $h(\mathtt{m}_1)=h(\mathtt{m}_2)$.

There are two widely used families of cryptographic hash functions – the MD family (MD = message digest) and the SHA family (SHA = secure hash algorithm). Ron Rivest and RSA laboratories developed the MD family. NIST and NSA developed the SHA family.

MD2 was published in 1989, MD4 in 1990, and MD5 in 1992. Collisions have been shown for all of them. MD was published in 2008 and was a SHA-3 candidate.

SHA-0 was published in 1993 and was used only briefly. SHA-1 was published in 1995; problems were discovered in 2005. SHA-2 was published in 2001 and consists of a family of six hash functions (having four possible hash lengths: 224-, 256-, 384-, and 512-bit hashes). SHA-3 was announced in 2015 and was previously named Keccak.

SHA-3 uses sponge functions. The others use the Merkle-Damagård construction. Hash functions can be constructed from encryption functions, but, of course, the property of having an inverse must be destroyed.

Hash functions allows authentication to occur without double encryption of the entire message.

Nick and Alex must agree on a hash function. Then Nick can (for security) send his message using Alex's public key. Also, he creates a hash of the plaintext and (for authentication) sends it using his private key. Using his private key, Alex decrypts the ciphertext encrypted with his public key and creates a hash of the plaintext using the hash function that he and Nick have agreed to use. Alex also decrypts the ciphertext of the hash using Nick's public key. The two hashes should be the same. If they are, Alex can assume that the message is secure and that it came from Nick.