9.9.3 Hashing Facts

A *hash* is a function that takes a variable-length string (message) and compresses and transforms it into a fixed-length value. When received, this hashed value is decrypted or unhashed into the original text so the recipient can understand the message. This process guarantees that there is no interception during the communication and that the file or message has not been tampered with. Hashing is used to authenticate messages, store passwords, maintain data integrity, and more.

Important facts about hashes include the following:

- Hashes ensure the data integrity of files and messages in transit.
- Hashes do not ensure confidentiality (in other words, hashes are not used to encrypt data).
- A hash is a one-way function. You cannot reproduce the message by running it through the same hash or a different hash.
- The hash value (output) is also referred to as a message digest or digital fingerprint.
- The sender and the receiver use the same hashing algorithm on the original data. If the hashes match, you can assume the data is unmodified.
- The larger the message digest, the more secure the hash.
- Even a small change to the original message generates a completely new hash.

The predominate hashing algorithms in use today are:

- MD5, which was developed by RSA (Rivest-Shamir-Adleman). MD5 generates a message digest of 128 bits. It follows its precedents, MD4, MD3, and so on.
- SHA-1, which was developed by NIST and NSA. SHA-1 generates a message digest of 160 bits. Some other versions are SHA-0, SHA-2, and SHA-3.
- RIPEMD, which was developed by the COSIC research group. RIPEMD generates a message digest of 128, 160, 256, or 320 bits. It was developed as an alternative to the government-sponsored SHA hash function.

Hashing is often used for the following:

Use	Description
File Integrity	Hashes are often used to prove the integrity of downloaded files. After a file is downloaded, the recipient creates a hash of the file. If the recipient's hash matches the hash of the original file, you know that: ▪ The downloaded file is complete (no missing parts). ▪ The downloaded file was not corrupted during the download process. ▪ The downloaded file is the same as the original and has not been altered by inserting malicious code or replaced with a virus or other destructive file. For this reason, files available for download are typically not encrypted, as the has proves their data integrity.
Secure Logon Credential Exchange	Hashes can be used to secure logon credentials during the exchange. The password is used as the key to perform a hash on a challenge text value, and only the hashed value is passed (not the password). The receiving host uses the same method to compare the hashes to verify the identity of the user. Examples of protocols that use this method are: ▪ LANMAN ▪ NTLM ▪ CHAP ▪ MS-CHAP

Be aware of the following regarding hashes:

- Strong hash outputs should contain a large number of bits. This makes it more difficult for an attacker to duplicate the hash value.
- Hashes should be produced from the entire message, not just a portion of the message.
- Good hashing algorithms have high amplification, also known as the avalanche effect. This means that a small change in the message results in a big change in the hashed value.
- A *collision* is when two different messages produce the same hash value. This is an indication that a stronger hashing algorithm should be used.
- A *collision resistance* is a hash algorithm's ability to avoid the same output from two guessed inputs.
- A *birthday attack* is a brute force attack in which the attacker hashes messages until one with the same hash is found.

TestOut Corporation All rights reserved.