5.5 Hashing

A hash function, “**H”** for example, is a function that takes a variable-size input “**m”** and returns a fixed-size string. The value that is returned is called the hash value “**h”** or the digest. This can be expressed mathematically as “**h = H(m)”**. There are three properties a hash function should have:

- ●Variable length input with fixed length output. In other words, no matter what you put into the hashing algorithm, the same sized output is produced.
- ●H(x) is one-way; you cannot “un-hash” something.
- ●H(x) is collision-free. Two different input values do not produce the same output. A collision refers to a situation where two different inputs yield the same output. A hash function should not have collisions.

Hashing is how Windows stores passwords. For example, if your password is “**password”**, then Windows will first hash it, producing something like:

“**0BD181063899C9239016320B50D3E896693A96DF”.**

It then stores that hash in the SAM (Security Accounts Manager) file in the Windows System directory. When you log on, Windows cannot “un-hash” your password. In essence, Windows takes whatever password you type in, hashes it, and then compares the result with the equivalent in the SAM file. If they match (exactly) then you can log in.

Storing Windows passwords is one of many applications of hashing. For example, in computer forensics, hashing a drive before starting a forensic examination is common practice. You can always hash it again later to see whether anything has been changed (accidently or intentionally). If the second hash matches the first, then nothing has been changed.

In relationship to hashing, the term “**salt”** refers to random bits that are used as one of the inputs to the hash. Essentially, the salt is intermixed with the message that will be hashed. Salt data complicates dictionary attacks that use pre-encryption of dictionary entries. It is also effective against rainbow table attacks. To achieve maximum security, the salt value is kept secret and separate from the password database/file.

5.5.1 MD5

MD5 is a 128-bit hash that is specified by RFC 1321. It was designed by Ron Rivest in 1991 to replace an earlier hash function, MD4. In 1996, a flaw was found with the design of MD5. Although it was not a clearly fatal weakness, cryptographers began recommending the use of other algorithms, such as SHA-1. MD5’s biggest problem is that it has never been collision resistant.

5.5.2 SHA

The Secure Hash Algorithm is perhaps the most widely used hash algorithm today. Several versions of SHA exist in today’s world. SHA (all versions) is considered to be secure and collision free. The versions include:

- SHA-1: This 160-bit hash function resembles the MD5 algorithm. This was designed by the National Security Agency (NSA) to be part of the Digital Signature Algorithm.
- SHA-2: These are actually two similar hash functions, with different block sizes, known as SHA-256 and SHA-512. They differ in the word size; SHA-256 uses 32-byte (256 bits) words whereas SHA-512 uses 64-byte (512 bits) words. There are also truncated versions of each standard, known as SHA-224 and SHA-384. These were also designed by the NSA.
- SHA-3: This is the latest version of SHA. It was adopted in October of 2012.