Using Hash Functions to Ensure Software Integrity

John Mathieson
US Air Force (WR-ALC)
Systems & Software Technology Conference
Salt Lake City, Utah
19 May 2011

Software Integrity

- Software Integrity – level of confidence that software has not been modified intentionally or unintentionally from its original configuration(e.g. baseline).

- Related Topics Software Assurance, Trustworthiness

Software Integrity

- Problem: How to verify integrity of electronic data? Are copies of files, the same as the original? Have files been modified?
- Data (files) moved between different mediums (CDs, Tapes, Floppy Disks, Hard Drives, network, etc.) and different computers and Operating Systems (Microsoft Windows, UNIX, VMS, etc.) must remain unchanged.

Software Integrity

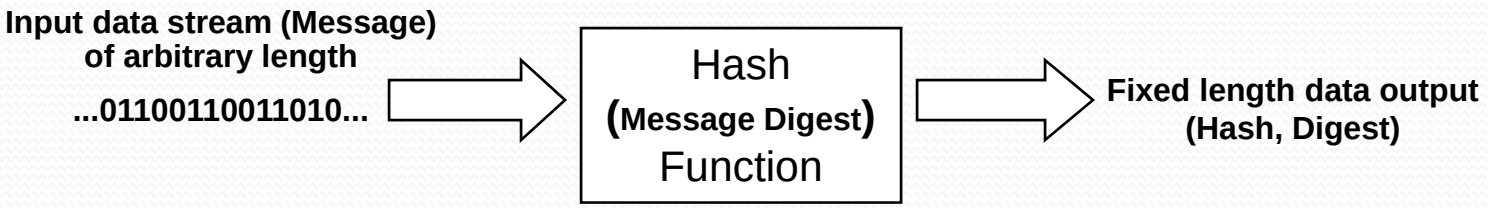
- Problem: How to track electronic data?
  - Must have a unique identifier (e.g. file size, date, name)
  - For different revisions, file name & size may not be unique. Date can be easily changed, intentionally or unintentionally.
- Derive file identifier from unique file data. (e.g. data changes, identifier changes)

Software Integrity

- **Solution: Create an Electronic "Fingerprint" of Data**
- Requirements
  - Every fingerprint must be unique, no matter how small the file differences.
  - No file modification to create fingerprint
  - Highly portable. Can be created, stored, and moved between different computers (hardware) and operating systems (Windows, UNIX, LINUX, VMS, etc.)
  - Small size. Does not require much additional storage space.

Solution- Hash Function

- Use a Hash Function (also sometimes called message digest, cryptographic hash function, one way encryption)
- A hash function takes as input, an arbitrary length stream of data (message) and generates a fixed length output (digest).



Hash Function

- Goals
  - Create a unique output, given unique (input) data. That is for every unique set of input data (e.g. file) there should be a unique(only 1) output (hash) of fixed length… (ideally).
  - Hash function is a one way function. This means if you have the function output, you can't (easily) reverse it and regenerate the original message (input).

Terminology

- A well designed hash function avoids the following (ideally)
  - Collision
    - Two unique inputs produce the same output
    - All hash functions will have collisions given an arbitrarily long input message and a fixed length output. Well designed hash functions will produce less collisions and they will be harder to find.
  - Pre-image
    - For a given a output (message), the input can be derived from it.
    - The hash is meant to be one way (i.e. The input can not be derived (easily) from the output.

Hash Algorithms

- Secure Hash 1(SHA-1), SHA-256, SHA-512, MD4, MD5, and more.
- Federal Information Processing Standard (FIPS)
  - FIPS-180-1 (Secure Hash Standard)
    - **SHA-1** 160 bit
      - For an input < 2^{64} bits (18,446,744,073,709,551,616) [1]
      - Generates 160 bit hash,
      - 40 char hexadecimal string **c62a99e8ee63adf8c9c34d303123f3a02376c290**

1. exabyte = 1 million terabytes

Creating File Hashes

- Numerous programs (cmdline and GUI) exist to create/verify hashes (e.g. md5summer, md5sum, sha1sum, etc.)
- You tell program which files to generate hashes for and store the results in a new file. The “hash file” would contain the path/filename and its hash.

Verifying Files Against Stored Hashes

-When verifying file(s) integrity, the hash program can be run in “check” mode to match existing files against stored hashes.
- If hashes don’t match 100%, files are not identical.

Sample file with stored hashes

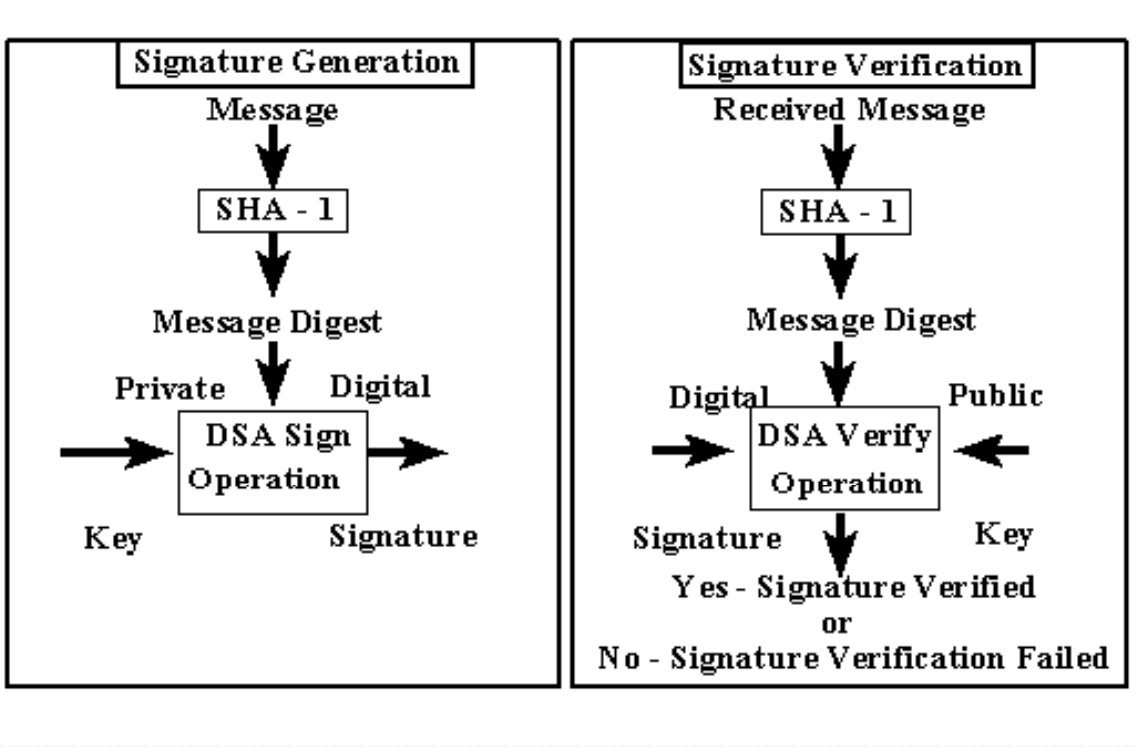
Secure Hash-1 (160 bit, 40 chars, hexadecimal string) [PATH/]Filename


```
bcbefc14e728e9f7ae0e369f5b3f58077e62f24b *autoexec.bat
906826abda839ee4269607c3e0b9241c0863d4fe *config.sys
1c8f520143514e6e035349be90631f4502a4ef09 *hosts/hosts.doc
bcfa5565419869e4c9e9850bbf27a210dd4daf7b *hosts/misc.txt
e5851ccc700a1d490f02d3279a91586e105dfc36 *start.bat
32d378a21392a8b770d6087b036b4d1455413efa *test.drv
```

Digital Signature & Hash Relationship

Digital Signatures work by “signing” a unique representation of the message(file). The Hash(message digest) of the message is that representation.

From FIPS 180-1



Using SHA-1 with Digital Signature Algorithm

Implementation

- Create hashes of original data at time of baselining.
- Store hashes with original code. One extra file can contain all the hashes. Create a “master” hash of the “hash file” and store it separately from the original files.
- For verification, verify “stored” master hash, against hash file, then use hash file to verify all the rest of the files.
- Process automated or performed “manually”.

Who Benefits?

- Everyone who “touches” the software from cradle to grave.
- Essentially creates a “tamper seal” on the software. Software changed, seal is “broken” (i.e. hash or “fingerprint” is changed).
- Developer → End User can easily verify the software is identical to the baselined version.

Hash uses

- Identification
  - National Institute of Standards & Technology (NIST)

    NIST has set up National Software Reference Library (NRSL) It is a repository of known software, file profiles, and their signatures (CRC32, MD4, MD5, SHA-1) to assist Department of Justice, federal, state, and local law enforcement with computer forensics.
- Verification
  - Computer security tools

    Intrusion detection (Tripwire). Store file signatures in secure database and use to check for unauthorized modification.

Data Configuration Management

- Configuration Management of data is a process not a product.
- This method provides a way to help track and verify the integrity of electronic data (i.e. software) from cradle to grave.

Contact Info

- Robins AFB, GA
- John.Mathieson@robins.af.mil
- 402 EMXG/402 EMXSS/MXDEAB
- DSN 468-1399
- Commercial (478) 926-1399