Assignment No.

Title: Message integrity check using SHA-1.

Aim: A message is to be transmitted using network resources from one machine to another calculate and demonstrate the use of a Hash value equivalent to SHA-1. Develop program in C++/Python/Scala/Java using Eclipse.

Objective:

- To learn client server communication using socket programming
- Use of hashing algorithm to check message Integrity.

Theory:

File verification is the process of using an algorithm for verifying the integrity or authenticity of a computer file. This can be done by comparing two files bit-by-bit, but requires two copies of the same file, and may miss systematic corruptions which might occur to both files. A more popular approach is to also store hashes of files, also known as message digests, for later comparison.

Integrity Verification:

File integrity can be compromised, usually referred to as the file becoming corrupted/modified by the hacker with bad intentions. A file can become corrupted/modified by a variety of ways: faulty storage media, errors in transmission, write errors during copying or moving, software bugs, by hackers and so on. Hash-based verification ensures that a file has not been corrupted by comparing the file's hash value to a previously calculated value. If these values match, the file is presumed to be unmodified. Due to the nature of hash functions, hash collisions may result in false positives, but the likelihood of collisions is often negligible with random corruption.

Authenticity Verification:

It is often desirable to verify that a file hasn't been modified in transmission or storage by untrusted parties, for example, to include malicious code such as viruses or backdoors. To verify the authenticity, a classical hash function is not enough as they are not designed to be collision resistant; it is computationally trivial for an attacker to cause deliberate hash collisions, meaning that a malicious change in the file is not detected with by a hash comparison. In cryptography, this attack is called a preimage attack.

For this purpose, cryptographic hash functions are employed often. As long as the hash sums cannot be tampered with- for example, if they are communicated over a secure channel - the files can be presumed to be intact. Alternatively, digital signatures can be employed to assure tamper resistance.

Hashing Function:

A hash function is any function that can be used to map data of arbitrary size to data of fixed size. The values returned by a hash function are called hash values, hash codes, digests, or simply hashes. One use is a data structure called a hash table, widely used in computer software for rapid data lookup. Hash functions accelerate table or database lookup by detecting duplicated records in a large file. They are also useful in cryptography. A cryptographic hash function allows one to easily verify that some input data maps to a given hash value, but if the input data is unknown, it is deliberately difficult to reconstruct it (or equivalent alternatives) by knowing the stored hash value. This is used for assuring integrity of transmitted data, and is the building block for HMACs, which provide message authentication.

The Secure Hash Algorithm is a family of cryptographic hash functions published by the National Institute of Standards and Technology (NIST) as a U.S.Federal Information Processing Standard (FIPS), including:

• **SHA-0:**

A retronym applied to the original version of the 160-bit hash function published in 1993 under the name "SHA". It was withdrawn shortly after publication due to an undisclosed "significant flaw" and replaced by the slightly revised version SHA-1.

• SHA-1:

A 160-bit hash function which resembles the earlier MD5 algorithm. This was designed by the National Security Agency (NSA) to be part of the Digital Signature Algorithm. Cryptographic weaknesses were discovered in SHA-1, and the standard was no longer approved for most cryptographic uses after 2010.

• SHA-2:

A family of 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-bit words where SHA-512 uses 64-bit words. There are also truncated versions of each standard, known as SHA-224, SHA-384, SHA-512/224 and SHA-512/256. These were also designed by the NSA.

• SHA-3:

A hash function formerly called Keccak, chosen in 2012 after a public competition among non-NSA designers. It supports the same hash lengths as SHA-2, and its internal structure differs significantly from the rest of the SHA family.

SHA 1 Algorithm:

Step 1: Append Padding Bits….

Message is "padded" with a 1 and as many 0's as necessary to bring the message length to 64 bits fewer than an even multiple of 512.

Step 2: Append Length....

64 bits are appended to the end of the padded message. These bits hold the binary format of 64 bits indicating the length of the original message.

Step 3: Prepare Processing Functions….

SHA1 requires 80 processing functions defined as:

f(t;B,C,D) = (B AND C) OR ((NOT B) AND D) (0 <= t <= 19)

f(t;B,C,D) = B XOR C XOR D (20 <= t <= 39)

f(t;B,C,D) = (B AND C) OR (B AND D) OR (C AND D) (40 <= t <=59)

f(t;B,C,D) = B XOR C XOR D (60 <= t <= 79)

Step 4: Prepare Processing Constants....

SHA1 requires 80 processing constant words

Step 5: Initialize Buffers….

SHA1 requires 160 bits or 5 buffers of words (32 bits)

Step 6: Processing Message in 512-bit blocks (L blocks in total message)….

This is the main task of SHA1 algorithm which loops through the padded and appended message in 512-bit blocks.

Input and predefined functions:
M[1, 2, ..., L]: Blocks of the padded and appended message

f(0;B,C,D), f(1,B,C,D), ..., f(79,B,C,D): 80 Processing Functions

K(0), K(1), ..., K(79): 80 rocessing Constant Words

H0, H1, H2, H3, H4, H5: 5 Word buffers with initial values

MD5 Algorithm:

The MD5 algorithm is a widely used hash function producing a 128-bit hash value. Although MD5 was initially designed to be used as a cryptographic hash function, it has been found to suffer from extensive vulnerabilities. It can still be used as a checksum to verify data integrity, but only against unintentional corruption.

Like most hash functions, MD5 is neither encryption nor encoding. It can be reversed by brute-force attack and suffers from extensive vulnerabilities as detailed in the security section below.

MD5 was designed by Ronald Rivest in 1991 to replace an earlier hash function MD4.The source code in RFC 1321 contains a "by attribution" RSA license. The MD5 hash function receives its acronym MD from its structure using Merkle–Damgård construction.

Conclusion: Thus we have successfully generated the hash values of message transmitted using SHA algorithms, and verified the message Integrity property.