**PRACTICAL: 8**

**AIM:**

An organization has transferred sensitive financial documents between departments over a network and wants to ensure that the files remain unchanged during transit. The IT team is tasked with using HashCalc and MD5 Calculator to confirm the integrity of these files by comparing hash values generated before and after the transfer. Verify the integrity of critical data files by generating and comparing hash values using HashCalc and MD5 Calculator, ensuring no tampering or corruption has occurred during transmission or storage.

**THEORY:**

The MD5 (message-digest algorithm) hashing algorithm is a one-way cryptographic function that accepts a message of any length as input and returns as output a fixed-length digest value to be used for authenticating the original message.

The MD5 hash function was originally designed for use as a secure cryptographic hash algorithm for authenticating digital signatures. But MD5 has been deprecated for uses other than as a noncryptographic checksum to verify data integrity and detect unintentional data corruption.

Although originally designed as a cryptographic message authentication code algorithm for use on the internet, MD5 hashing is no longer considered reliable for use as a cryptographic checksum because security experts have demonstrated techniques capable of easily producing MD5 collisions on commercial off-the-shelf computers. An encryption collision means two files have the same hash. Hash functions are used for message security, password security, computer forensics and cryptocurrency.

Ronald Rivest, founder of RSA Data Security LLC and professor at Massachusetts Institute of Technology, designed MD5 in 1991 as an improvement to a prior message-digest algorithm, MD4. Describing it in Internet Engineering Task Force (IETF) Request for Comments (RFC) 1321, "The MD5 Message-Digest Algorithm," he wrote:

The algorithm takes as input a message of arbitrary length and produces as output a 128-bit 'fingerprint' or 'message digest' of the input. It is conjectured that it is computationally infeasible to produce two messages having the same message digest, or to produce any message having a given prespecified target message digest. The MD5 algorithm is intended for digital signature applications, where a large file must be 'compressed' in a secure manner before being encrypted with a private (secret) key under a public-key cryptosystem such as RSA.

IETF suggests MD5 hashing can still be used for integrity protection, noting: "Where the MD5 checksum is used in line with the protocol solely to protect against errors, an MD5 checksum is still an acceptable use." However, it added that "any application and protocol that employs MD5 for any purpose needs to clearly state the expected security services from their use of MD5."

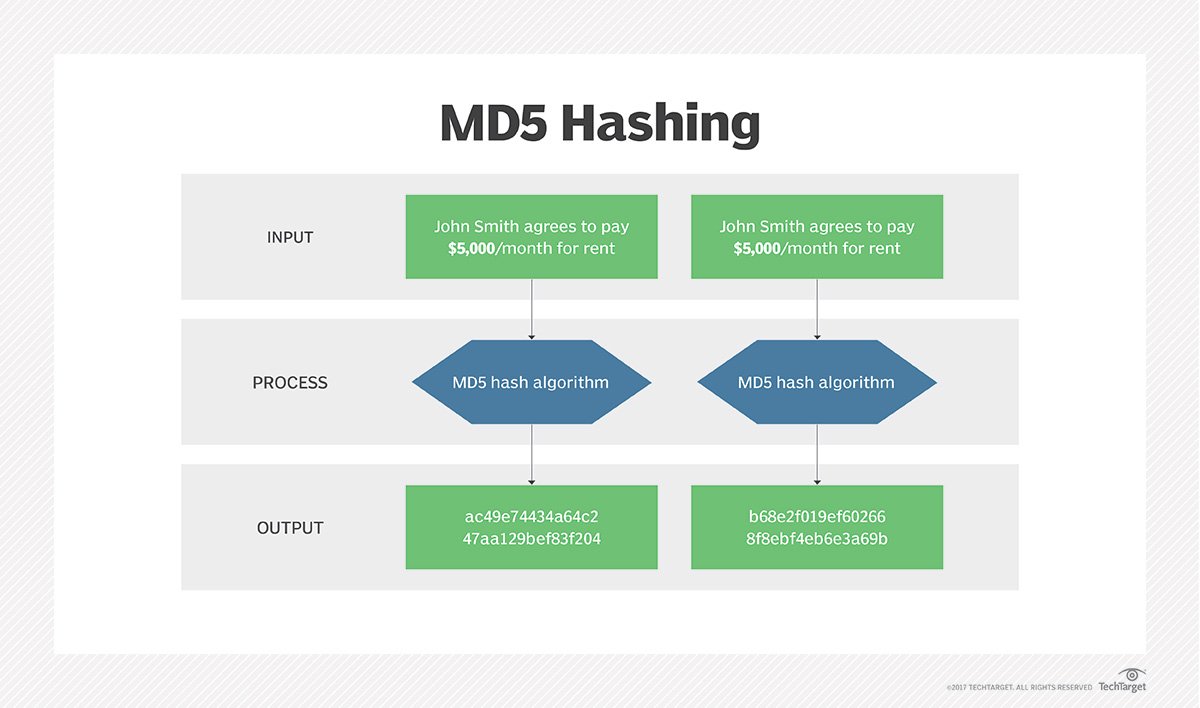


Figure 1: MD5

Message digests, also known as *hash functions*, are one-way functions; they accept a message of any size as input and produce as output a fixed-length message digest.

MD5 is the third message-digest algorithm Rivest created. MD2, MD4 and MD5 have similar structures, but MD2 was optimized for 8-bit machines, in comparison with the two later algorithms, which are designed for 32-bit machines. The MD5 algorithm is an extension of MD4, which the critical review found to be fast but potentially insecure. In comparison, MD5 is not quite as fast as the MD4 algorithm, but offered much more assurance of data security.

The MD5 message-digest hashing algorithm processes data in 512-bit strings, broken down into 16 words composed of 32 bits each. The output from MD5 is a 128-bit message-digest value.

Computation of the MD5 digest value is performed in separate stages that process each 512-bit block of data along with the value computed in the preceding stage. The first stage begins with the message-digest values initialized using consecutive hexadecimal numerical values. Each stage includes four message-digest passes, which manipulate values in the current data block and values processed from the previous block. The final value computed from the last block becomes the MD5 digest for that block.

The goal of any message-digest function is to produce digests that appear to be random. To be considered cryptographically secure, the hash function should meet two requirements:

1. It is impossible for an attacker to generate a message matching a specific hash value.
2. It is impossible for an attacker to create two messages that produce the same hash value.

MD5 hashes are no longer considered cryptographically secure methods and should not be used for cryptographic authentication, according to IETF.

**CODE:**

|  |
| --- |
| **N/A** |

**OUTPUT:**

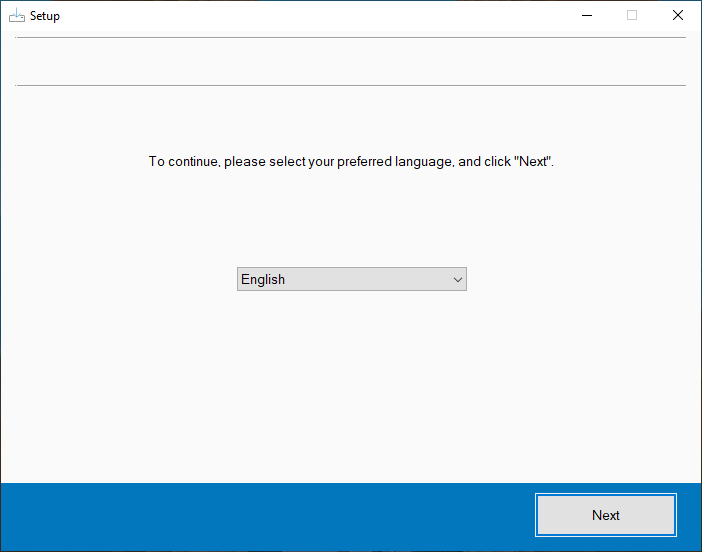


Figure 2:Start setting up HashCalc

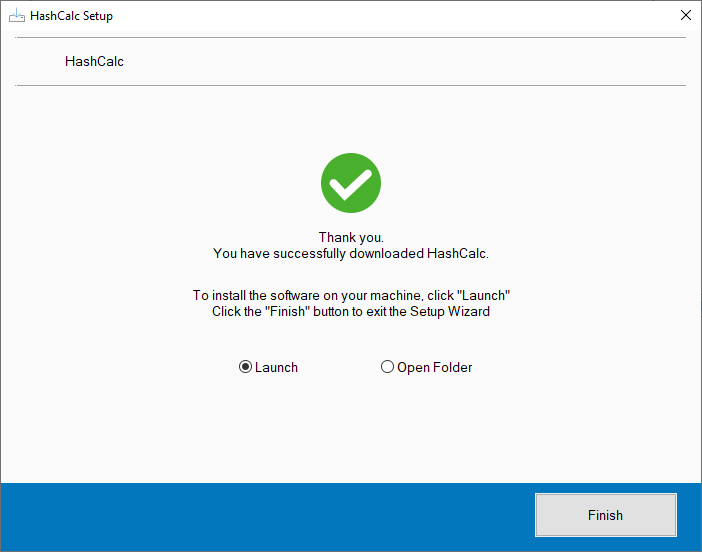


Figure 3:Finish setup of HashCalc and launch it

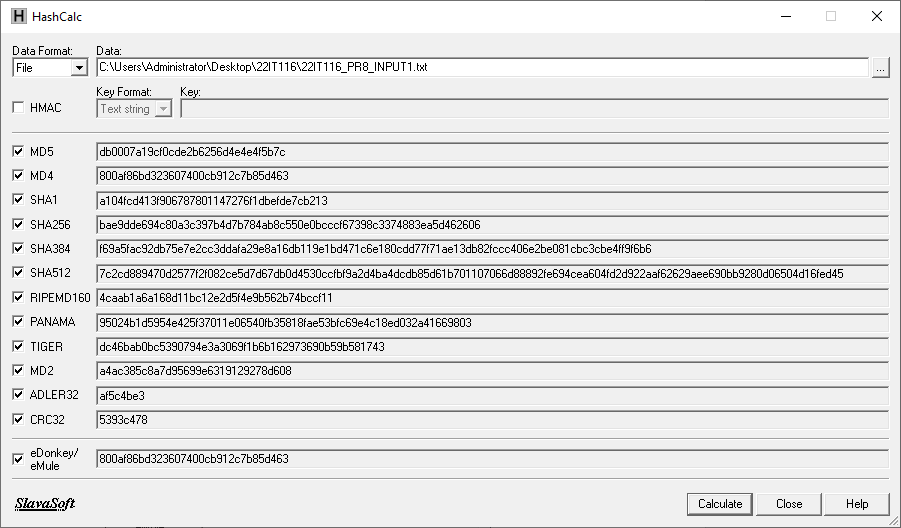


Figure 4:Hash values for Input1.txt file

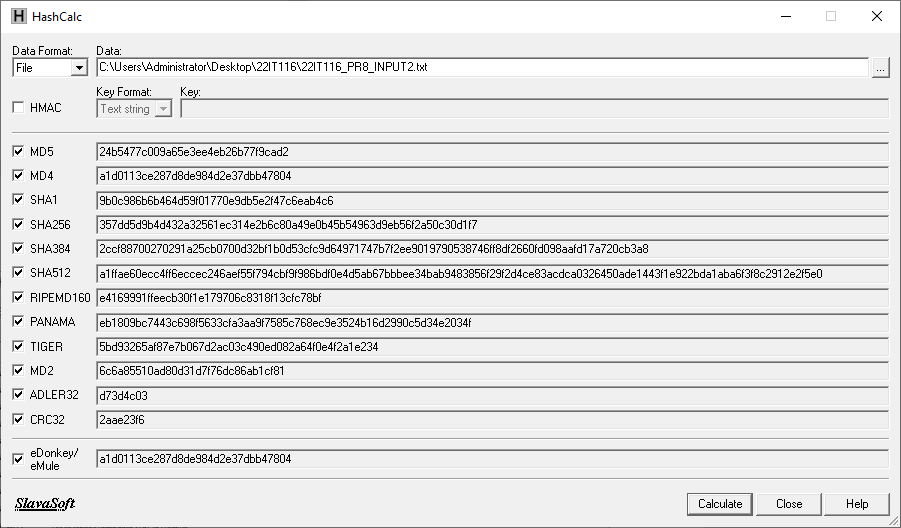


Figure 5:Hash values for Input2.txt file

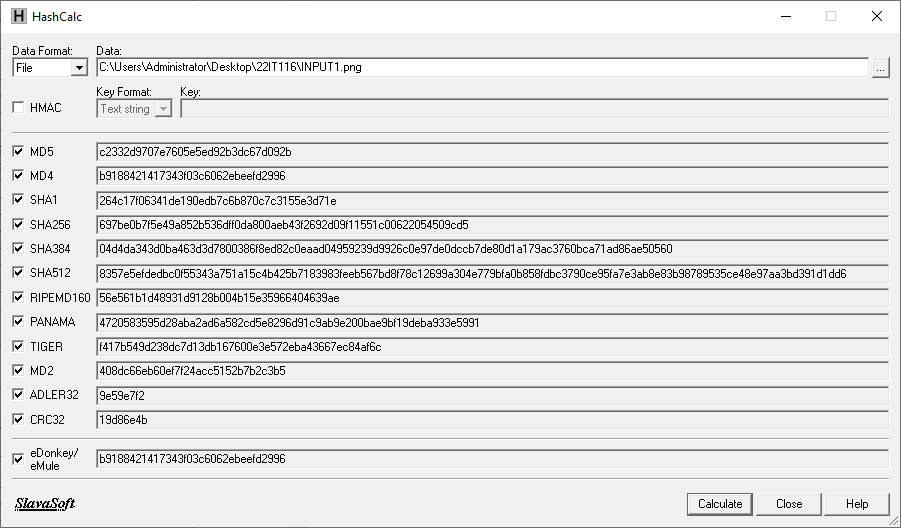


Figure 6:Hash values for INPUT1.png photo

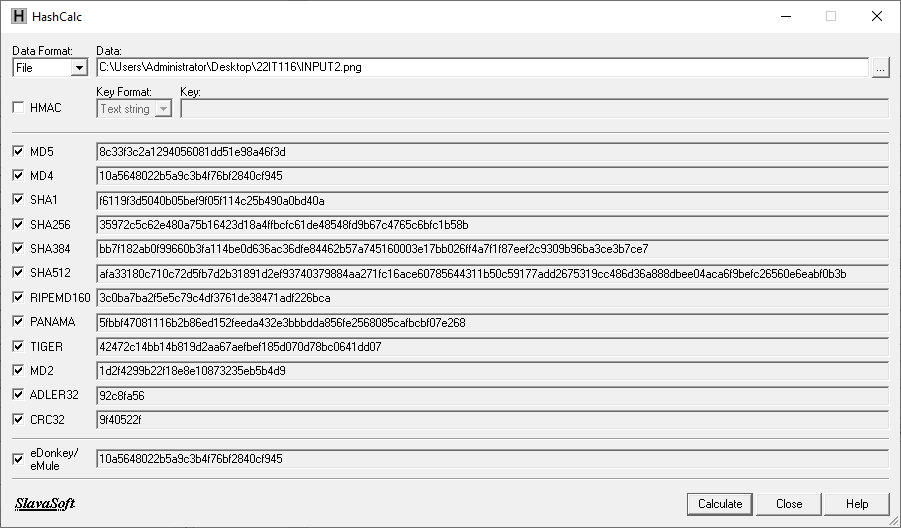


Figure 7:Hash values for Input2.png photo

**LATEST APPLICATIONS:**

* Hashi Corp Vault
* Acronis True Image
* VeraCrypt
* GnuPG (GPG)
* OpenSSL
* AWS S3

**LEARNING OUTCOME:**

In this practical, I will learn how to verify the integrity of sensitive financial documents transferred between departments using HashCalc and MD5 Calculator. I will generate hash values before and after the transfer, comparing them to ensure the files remain unchanged. Through this process, I will understand how hashing algorithms help detect data tampering or corruption, practice secure file transfer methods, and troubleshoot integrity issues, ensuring critical files are securely transmitted and stored.

**REFERENCES:**

1. HashCalc: <https://download.cnet.com/hashcalc/3000-2250_4-10130770.html>
2. MD5: <https://www.techtarget.com/searchsecurity/definition/MD5>
3. ChatGPT: <https://chatgpt.com/>