PROJECT REPORT ON

**MD5**

**( Message digest version 5 )**

Submitted By:

**Almena (1905374)**

**Kshitij (1905514)**

**Pooja (1905330)**

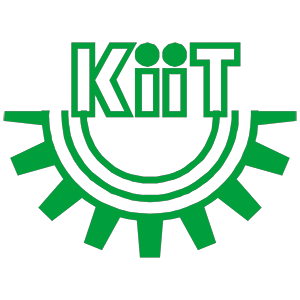
**Sarthak Rout(1905050)**

**Saurav(1905126)**

**Nischit(1905995)**

Under the Guidance of:

**Dr. Chittaranjan Pradhan**



School of Computer Science and Engineering

Kalinga Institute of Industrial Technology

(Deemed to be University)

BHUBANESWAR

APRIL 2021

ACKNOWLEDGEMENT

We would like to express our special thanks and gratitude to our Dr. Chittaranjan Pradhan who gave us the golden opportunity to do this wonderful project on the topic ‘**Message Digest (MD5)’**, which also helped us in doing a lot of research and we came to know about so many new things we are really thankful to him.

ABSTRACT

The MD5 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.

Keyword:—Hash Function, Collision Message digest Algorithm.

INDEX

Abstract 2

Index 3

1: INTRODUCTION 4

* 1. MD5 4
  2. Hash function 4

2: ALGORITHM 5-8

3: SECURITY ISSUES 9-12

3.1 Collision attack 9

3.2 Preimage attack 10

3.3 Brute force attack 11

3.4 Rainbow attack 11

4: REFERENCES 12

1 : INTRODUCTION

MD stands for message digest

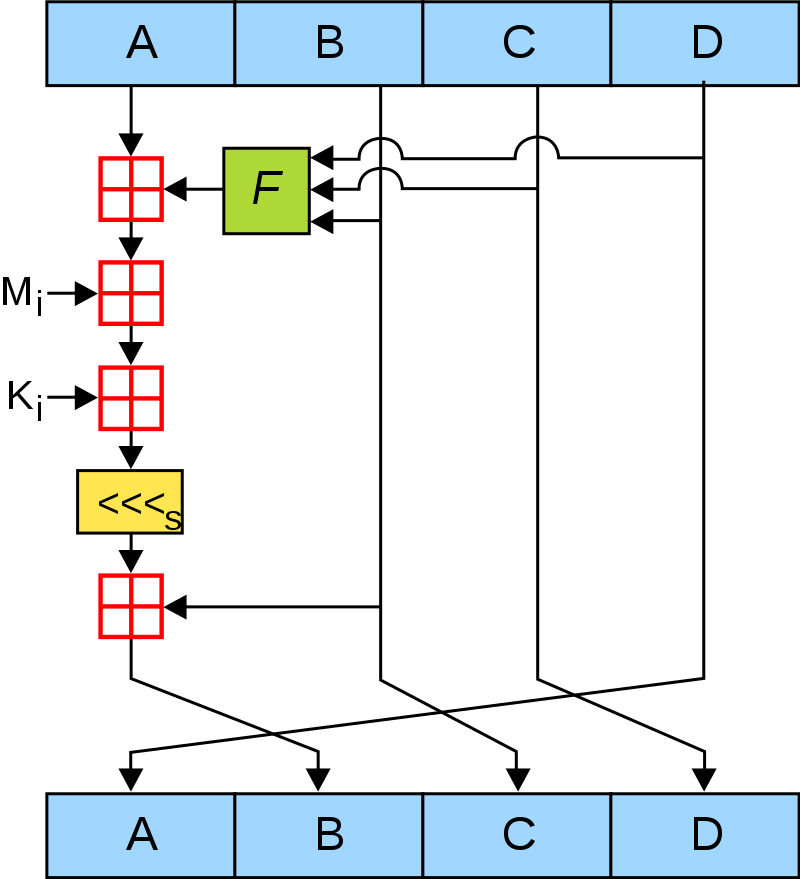
* 1. **MD5**

The MD5 message-digest algorithm was designed by Ronald Rivest in 1991 and it is an improved version of MD4 hashing algorithm .It is cryptographically broken but still widely used hash function producing a 128-bit hash value. Although MD5 initially designed to be used as cryptographic hash function , it has been found to suffer from extensive vulnerabilities. It can still be used as a [checksum](https://en.wikipedia.org/wiki/Checksum" \o "Checksum) to verify [data integrity](https://en.wikipedia.org/wiki/Data_integrity" \o "Data integrity), but only against unintentional corruption. It remains suitable for other non-cryptographic purposes, for example for determining the partition for a particular key in a [partitioned database](https://en.wikipedia.org/wiki/Partition_(database)" \o "Partition (database)). However, it is also used in other security protocols and applications such as SSH, SSL, and IPSec. Some applications strengthen the MD5 algorithm by adding a salt value to the plaintext or by applying the hash function multiple times.

**1.2 Hash function**

A hash function is a one-way encryption function that converts a numerical input value into another compressed numerical value and the output is always of fixed length. It is computationally hard to decipher the hash and any attempt to crack it is practically infeasible.

2: ALGORITHM



**Step1: Padding**

Padding means adding extra bits to the original message. So, in MD5 original message is padded such that its length in bits is corresponding to 448 modulo 512. Padding is done such that the total bits are 64 less, being a multiple of 512 bits length. Padding is done even if the length of the original message is already corresponding to 448 modulo 512. In padding bits, the only first bit is 1, and the rest of the bits are 0.

**Step2: Append the length**

After padding, 64 bits are inserted at the end, which is used to record the original input length. Modulo 2^64. Exceeds of the length more than 64 bits lower order is considered. At this point, the resulting message has a length multiple of 512 bits.

**Step 3: Divide the Input**

Now that the length of the message after adding have become the exact multiple of 512.The process include Dividing into 512 bits block.

**Step 4: Initialize chaining variable**

A four-word buffer (A, B, C, D) also known as chaining variable is used to compute the values for the message digest. Here A, B, C, D are 32- bit registers and are initialized in the following way

A=6DFEEY23

B=44ABCD22

C=AFDC2161

D=7A9CDE34.

These chaining variables are the hash value which we will get as 128-bit hash value of MD5 algorithm.

**STEP 5: Process Block**

**5.1 Consecutive element initialization**

The consecutive element of A, B, C, D which is a, b, c, d will get initialized with the chaining variable it contains. Thus, these a, b, c, d are the one which will go through nonlinear function and all the process block operation.

**5.2 Divide 512 bits block**

Now the algorithm loop starts and this loop continue until all 512 bits block are all done processing. These 512 bits blocks are divided into 16 further sub-block which are of 32 bits each.

**5.3 Four round operation**

Each of these 16 sub-blocks of 512-bit block has to go through 4 rounds with 4 different nonlinear functional operation.ie (F, G, H, I).

F(x, y, z) = (x ^ y) ∨ ((¬x) ∨ z)

G(x, y, z) = (x ^ y) ∨ (y ∨ (¬z))

H(x, y, z) = x ⊕ y ⊕ z

I(x, y, z) = y ⊕ (x ∨ (¬z))

For the first-round operation on these 16 block F nonlinear function will be performed in one of three variable a, b, c & d. Then the result is added with the fourth variable which is the result of adding constant(t[i]=232 \*ABS(sin(i))) and the block of text. Then it rotates that result to the right of the variable number of bits and adds the result to one of the a, b, c or d. Finally, the result replaces one of the a, b, c, d.

This same operation is performed for all the 16 sub-blocks before moving further for next nonlinear functional operation.

The same above operation is done 3 times more with other nonlinear function G, H, I. Then last hash value we will get after performing last of I operation is the hash value used as chaining variable for another round of operation on 512 bits block.

Finally, after the completion of the loop and performing this operation in all the 512 bits block the hash value we get in A, B, C, D chaining variable which is of 128 bits is the hash value generated with MD-5 algorithm.

Mathematical representation of these operation is

FF(a, b, c, d, Mi, s, ti) denotes a = b + ((a + F(b, c, d) + Mi + ti) <<< s)

GG(a, b, c, d, Mi, s, ti) denotes a = b + ((a + G(b, c, d) + Mi + ti) <<< s)

HH(a, b, c, d, Mi, s, ti) denotes a = b + ((a + H(b, c, d) + Mi + ti) <<< s)

I(b, c, d) + Mi + ti) <<< s)

{Mi denotes ith sub-block of the message (i varies from 0 to 15)}

1. SECURITY ISSUES

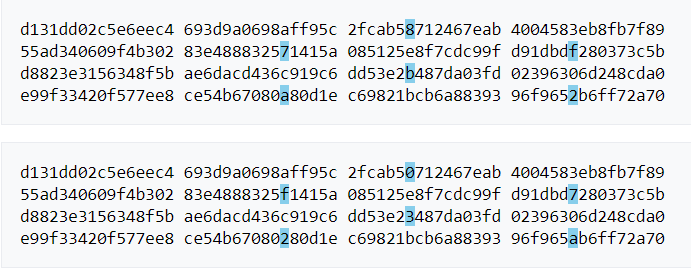
Like every cryptographic function, hashes are susceptible to brute-force attacks. The longer L is, the more work an attacker has to do to mount an attack; however, hashes with longer L also are usually slower to computer.There are three important attacks on hashes:

* A "collision attack" allows an attacker to find two messages M1 and M2 that have the same hash value in fewer than 2(L/2) attempts.
* A "first-pre image attack" allows an attacker who knows a desired hash value to find a message that results in that value in fewer than 2^L attempts.
* A"second-pre image attack" allows an attacker who has a desired message M1 to find another message M2 that has the same hash value in fewer than 2^L attempts.

**3.1 Collision attack**

In 1996, collisions were found in the compression function of MD5.A [collision attack](https://en.wikipedia.org/wiki/Collision_attack" \o "Collision attack) exists that can find collisions within seconds on a computer. MD5 provides no security over these collision attacks. A [collision attack](https://en.wikipedia.org/wiki/Collision_attack" \o "Collision attack) exists that can find collisions within seconds on a computer. MD5 hash prone to hash collision weakness, i.e. it is possible to create the same hash function for two different inputs.

An example MD5 collision, with the two messages differing in 6 bits, is:



Both produce the MD5 hash 79054025255fb1a26e4bc422aef54eb4. The difference between the two samples is that the leading bit in each [nibble](https://en.wikipedia.org/wiki/Nibble" \o "Nibble) has been flipped.

**3.2 Preimage attack**

A preimage attack on [cryptographic hash functions](https://en.wikipedia.org/wiki/Cryptographic_hash_function" \o "Cryptographic hash function) tries to find a [message](https://en.wikipedia.org/wiki/Message" \l "In_computer_science" \o "Message) that has a specific hash value. A cryptographic hash function should resist attacks on its [preimage](https://en.wikipedia.org/wiki/Preimage" \l "Inverse_image" \o "Preimage) (set of possible inputs). In April 2009, an attack against MD5 was published that breaks MD5's [preimage resistance](https://en.wikipedia.org/wiki/Preimage_resistance" \o "Preimage resistance). This attack is only theoretical, with a computational complexity of 2123.4 for full preimage.

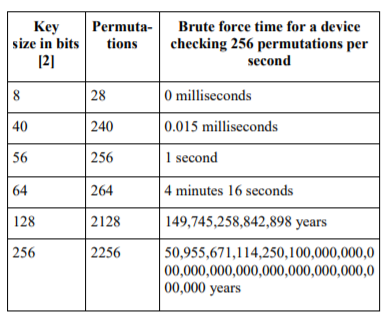
In the context of attack, there are two types of preimage resistance:

**preimage resistance**: for essentially all pre-specified outputs, it is computationally infeasible to find any input that hashes to that output.

**second-preimage resistance:** for a specified input, it is computationally infeasible to find another input which produces the same output.

**3.3 Brute force attack:**

In cryptograph, a brute force attack or exhaustive key search is a strategy that can in theory be used against any encrypted data by an attacker who is unable to take advantage of any weakness in an encryption system that would otherwise make his/her task easier. It involves systematically checking all possible key until the correct key is found. In the worst case, this would involve traversing the entire search space.



**Symmetric key length vs brute force combinations**

**3.4 Rainbow table:**

A rainbow table is a precomputed table for reversing cryptographic hash function, usually for cracking password hashes. Tables are usually used in recovering the plain text password, up to a certain length consisting of a limited set of characters. It is a form of time-memory trade of, using less CPU at the cost of more storage. Proper key derivation function employ sal to make this attack infeasible. Rainbow tables are a refinement of an earlier, simpler algorithm by Martin Hellman[21 that used the inversion of hashes by looking up precomputed hash chains

REFERENCES

Wikipedia MD5 <https://en.wikipedia.org/wiki/MD5>

Applied Cryptography\_ Protocols, Algorithms and Source Code in C-Bruce Schneier.

Cryptography Network Security by Behrouz Forouzan.