## GOVERNMENT POLYTECHNIC, AMRAVATI

**(An Autonomous Institute of Govt. of Maharashtra)**



## MICRO-PROJECT REPORT ON

**“Password Encryption And Decryption”**

## In Partial fulfillment of Diploma in Computer Engineering

**In the subject of**

**Computer Security (CM5467)**

## By

## Ms. Shreya Borode (20CM003)

**Under the guidance of**

**Prof. V. R. Rathod**

## Lecturer in Computer Engineering

Department of Computer Engg.

Government Polytechnic Amravati,

## (2022-2023)

**Government Polytechnic, Amravati.**

(An Autonomous Institute of Govt. of Maharashtra)

**Department of Computer Science and Engg.**

*Certificate*

This is to certify that **Shreya Vilas Borode (20CM003)** of Fifth Semester Diploma in **Computer Engineering** has satisfactorily completed the Micro Project entitled “**Password Encryption And Decryption**” in Computer Security **(CM5467)** for the academic year 2022-23 as prescribed in curriculum.

Head of Department Lecturer in Computer Engineering

Dr. P. R. Satav Prof. V. R. Rathod

Place: Amravati Date: 19/12/22

**VISION:**

Provide skilled professionals in Computer Engineering to contribute towards the advancement of technology useful for society and the industrial environment.

**MISSION:**

* Train students with Computer Engineering knowledge to apply it in the general disciplines of design, deployment of software and integration of existing technologies for E-governance and for the benefit of society.
* Provide a learning ambience to enhance innovations, problem solving skills, leadership qualities, team spirit and ethical responsibilities.
* Provide an academic environment and consultancy services to the industry and society in the area of Computer Engineering
* Impart need based and value based education by providing exposure of latest tools and technologies in the area of Computer Engineering to satisfy the stakeholders.
* Upgrade and maintain facilities for quality technical education with continuous effort for excellence in Computer Engineering.

**Index**

|  |  |  |
| --- | --- | --- |
| **Sr. No.** | **Title** | **Page No.** |
| 1. | Part A | 5 - 7 |
| 2. | Part B | 8 - 10 |
| 3. | Information about Hashing | 11 |
| 4. | Explain the MD5 algorithm | 11 -13 |
| 5. | Java code for Encryption | 13 |
| 6. | Output | 14 |
| 7. | Information about Decryption | 15-16 |
| 8. | Conclusion | 17 |

# PART A - Plan

## 1.0 Title of Micro Project:

The title of the micro project is to **Encryption and Decryption of Password**

**2.0 Brief Introduction**:

The MD5 (message-digest algorithm) hashing algorithm is a one-way i.e., the cypher text password cannot be decrypted back to a plain text. It has a 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 and this is known as MD5 Hashing.

Ronald Rivest designed this algorithm in 1991 to provide the means for digital signature verification. 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

## Aim of the Micro-Project:

* + 1. To encrypt the password.
    2. To authenticate the password while keeping the password unreadable to third party or even the developer.
    3. To provide extra security to the information entered by the user.

## 4.0 Course Outcomes Integrated:

1. Identify risks faced by Computer Systems and Information hazard in various situations.
2. Apply cryptographic algorithms and protocols to maintain Computer Security.

## 5.0 Action Plan:

|  |  |  |  |
| --- | --- | --- | --- |
| **S.N.** | **Details of activity** | **Planned start date** | **Planned Finish date** |
| 1 | Started planning to design the basic structure of the project & discussed it  other group members | 02/10/22 | 15/10/22 |
| 2 | Collected basic information of MD5 Algorithm | 16/10/22 | 21/10/22 |
| 3 | Implemented the java code for encryption | 22/10/22 | 30/10/22 |
| 4 | Check the java code is properly run or not and take a screenshot. | 03/11/22 | 25/11/22 |
| 5 | Prepared brief introduction for Part A & Part B | 03/11/22 | 08/11/22 |

**6.0 Resources Required:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S. N** | **Name of Resource/material** | **Specifications** | **Qty** | **Remarks** |
| **1** | Books | * Applied Cryptography * Modern Cryptography * The secrets of MD5 Decryption – R book |  |  |
| **2** | Laptop | Windows 10, core i5, 8gb ram, 1tb hdd, 256gb SSD, intel integrated gpu | 3 |  |
| **3** | Program (Software) | Microsoft Word, Google Docs, |  |  |

## Team members name:

* + 1. **Ms. Shreya Borode (20CM003)**

## Ms. Purva Deshmukh (20CM007)

* + 1. **Ms. Srushti Deshmukh (20CM008)**

## Ms. Kalyani Hirulkar (20CM020)

* + 1. **Ms. Vedika Joshi (20CM024)**
    2. **Ms. Shreya Pande (20CM040)**

# PART B- Plan

## 1.0 Title of Micro Project:

The title of the micro project is to **Encryption and Decryption of Password.**

**2.0 Brief Introduction**:

The MD5 (message-digest algorithm) hashing algorithm is a one-way i.e., the cypher text password cannot be decrypted back to a plain text. It has a 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 and this is known as MD5 Hashing.

Ronald Rivest designed this algorithm in 1991 to provide the means for digital signature verification. 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

## Aim of the Micro-Project:

* + 1. To encrypt the password.
    2. To authenticate the password while keeping the password unreadable to third party or even the developer.
    3. To provide extra security to the information entered by the user.

## 4.0 Actual Procedure:

|  |  |  |  |
| --- | --- | --- | --- |
| **S.N.** | **Details of activity** | **Planned start date** | **Planned Finish date** |
| 1 | Started planning to design the basic structure of the  project & discussed it other group members | 02/10/22 | 15/10/22 |
| 2 | Prepared brief introduction for Part A & Part B | 16/10/22 | 21/10/22 |
| 3 | Collected basic information of MD5 Algorithm. | 22/10/22 | 30/10/22 |
| 4 | Implemented the java code for encryption | 03/11/22 | 25/11/22 |
| 5 | Check the java code is properly run or not and take a screenshot. | 03/11/22 | 08/11/22 |

**5.0 Resources Required:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S. N** | **Name of Resource/material** | **Specifications** | **Qty** | **Remarks** |
| **1** | Books | * Applied Cryptography * Modern Cryptography * The secrets of MD5 Decryption – R book |  |  |
| **2** | Laptop | Windows 10, core i5, 8gb ram, 1tb hdd, 256gb SSD, intel integrated gpu | 3 |  |
| **3** | Program (Software) | Microsoft Word, Google Docs,  Facebook, YouTube, Google, LinkedIn |  |  |

## 6.0 Outputs of the Micro-Project:

1. We learn the encryption Algorithm MD5
2. We implemented the java code for encryption of password and converted into cypher text
3. We came to know about the authentication method used to verify the password.
4. We came with the proper solution of the user password security.

## Skill Developed / Learning outcome of this Micro-Project:

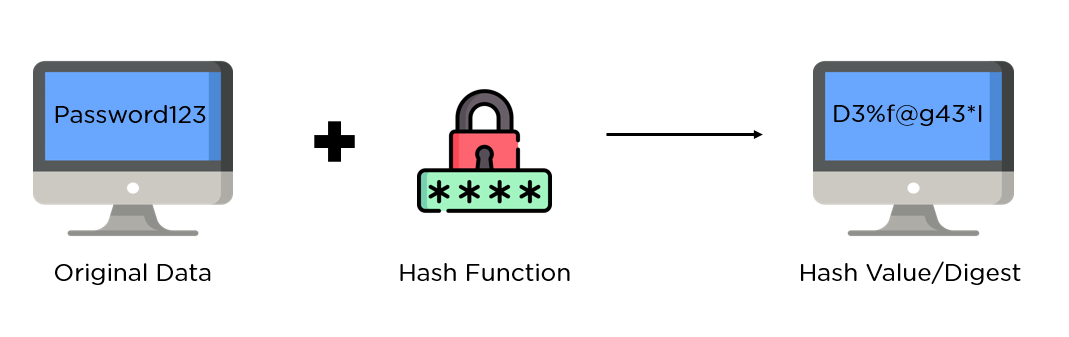
1. We have learnt encryption algorithm
2. We developed the java code for developing the cypher text
3. We learn about how to prepare reports.

## 8.0 Assessment by Faculty as per Rubrics

|  |  |  |  |
| --- | --- | --- | --- |
| Process Assessment (06) | Product Assessment (04) | Total Marks (10) | Signature of Faculty |
|  |  |  |  |

**What is Hashing?**

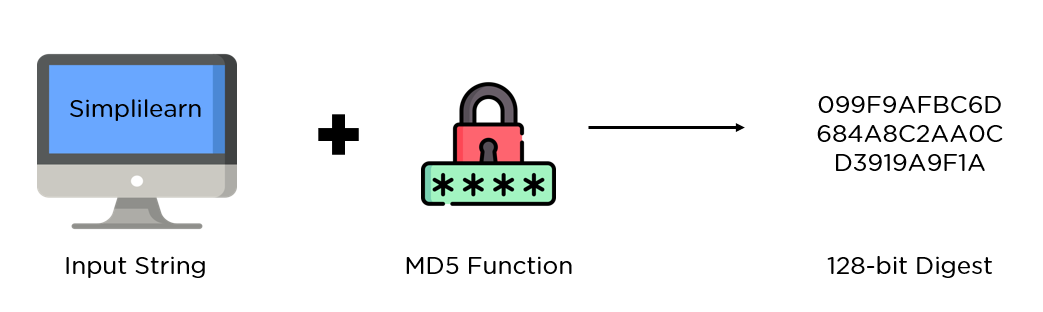
Hashing consists of converting a general string of information into an intricate piece of data. This is done to scramble the data so that it completely transforms the original value, making the hashed value utterly different from the original.



Hashing uses a hash function to convert standard data into an unrecognizable format. These hash functions are a set of mathematical calculations that transform the original information into their hashed values, known as the hash digest or digest in general. The digest size is always the same for a particular hash function like MD5, irrespective of input size.

**What is the MD5 Algorithm?**

MD5 (Message Digest Method 5) is a cryptographic hash algorithm used to generate a 128-bit digest from a string of any length. It represents the digests as 32-digit hexadecimal numbers.

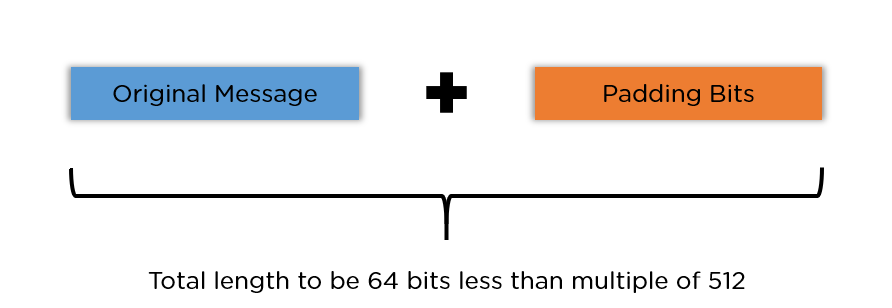


The digest size is always 128 bits, and thanks to hashing function guidelines, a minor change in the input string generate a drastically different digest. This is essential to prevent similar hash generation as much as possible, also known as a hash collision.

**In MD5 Algorithm there are four major sections of the algorithm:**

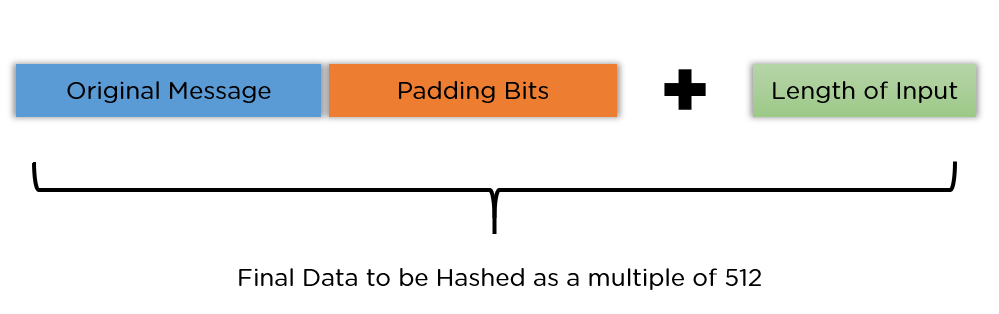
1. **Padding Bits**

When you receive the input string, you have to make sure the size is 64 bits short of a multiple of 512. When it comes to padding the bits, you must add one first, followed by zeroes to round out the extra characters.



1. **Padding Length**

You need to add a few more characters to make your final string a multiple of 512. To do so, take the length of the initial input and express it in the form of 64 bits. On combining the two, the final string is ready to be hashed.



1. **Initialize MD Buffer**

The entire string is converted into multiple blocks of 512 bits each. You also need to initialize four different buffers, namely A, B, C, and D. These buffers are 32 bits each and are initialized as follows:

1. A = 01 23 45 67
2. B = 89 ab cd ef
3. C = fe dc ba 98
4. D = 76 54 32 10

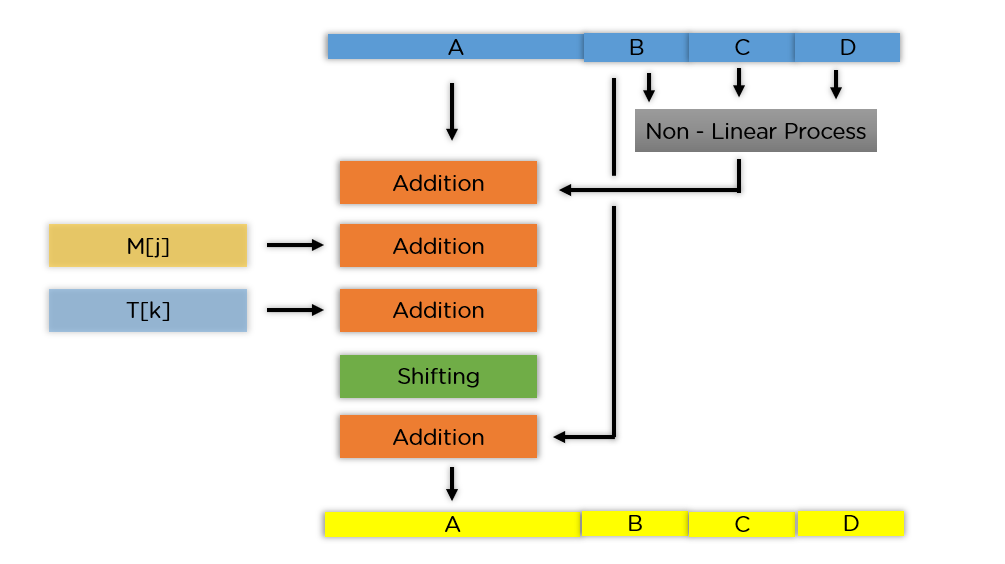
Micro-project Report on Entrepreneurship Development Page 12

1. **Process Each Block**

Each 512-bit block gets broken down further into 16 sub-blocks of 32 bits each. There are four rounds of operations, with each round utilizing all the sub-blocks, the buffers, and a constant array value.

This constant array can be denoted as T [1] -> T [64].

Each of the sub-blocks are denoted as M [0] -> M [15].



According to the image above, you see the values being run for a single buffer A. The correct order is as follows:

* + It passes B, C, and D onto a non-linear process.
  + The result is added with the value present at A.
  + It adds the sub-block value to the result above.
  + Then, it adds the constant value for that particular iteration.
  + There is a circular shift applied to the string.
  + As a final step, it adds the value of B to the string and is stored in buffer A.
  + The steps mentioned above are run for every buffer and every sub-block. When the last block’s final buffer is complete, you will receive the MD5 digest.

The non-linear process above is different for each round of the sub-block.

Round 1: (b AND c) OR ((NOT b) AND (d))

Round 2: (b AND d) OR (c AND (NOT d))

Round 3: b XOR c XOR d

Round 4: c XOR (b OR (NOT d))

**Java Code for Encryption:**

import java. security.MessageDigest;

import java.security.NoSuchAlgorithmException;

public class SimpleMD5Example

{

public static void main(String[] args)

{

String passwordToHash = "Group 1 Computer Security CM5467";

String generatedPassword = null;

try

{

// Create MessageDigest instance for MD5

MessageDigest md = MessageDigest.getInstance("MD5");

// Add password bytes to digest

md.update(passwordToHash.getBytes());

// Get the hash's bytes

byte[] bytes = md.digest();

// This bytes[] has bytes in decimal format. Convert it to hexadecimal format

StringBuilder sb = new StringBuilder();

for (int i = 0; i < bytes.length; i++) {

sb.append(Integer.toString((bytes[i] & 0xff) + 0x100, 16).substring(1));

}

// Get complete hashed password in hex format

generatedPassword = sb.toString();

} catch (NoSuchAlgorithmException e) {

e.printStackTrace();

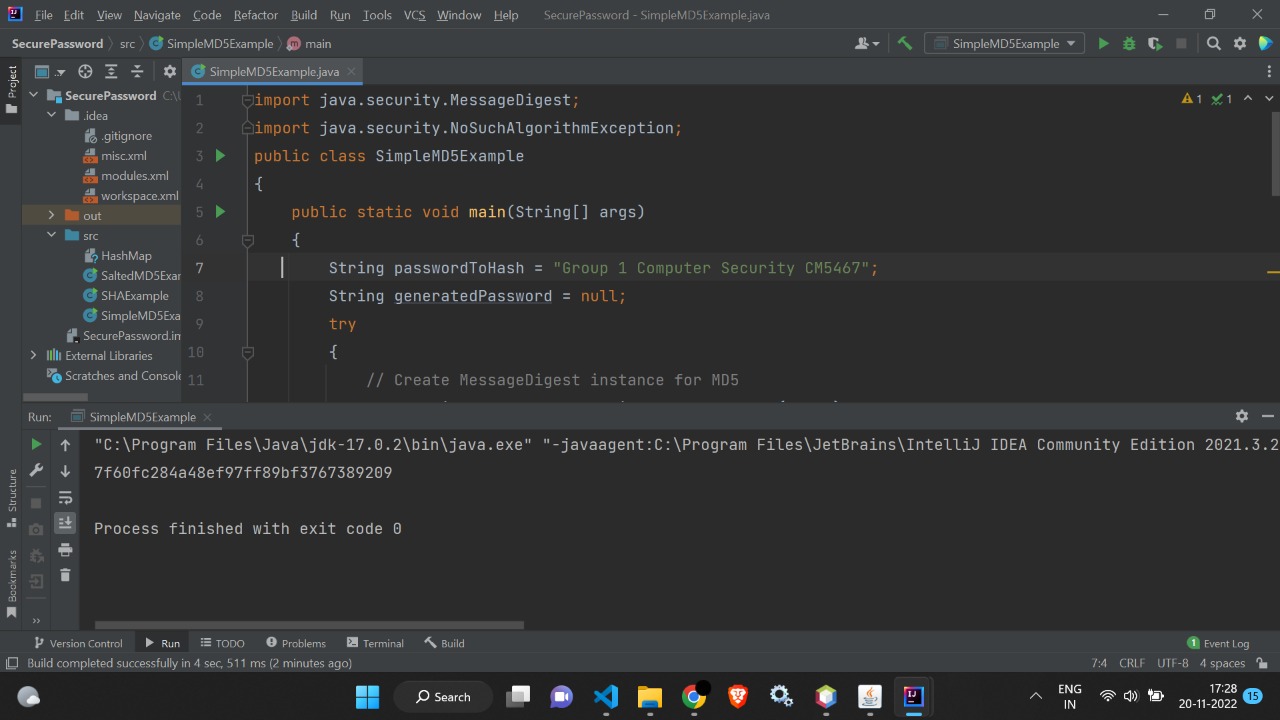
}

System.out.println(generatedPassword);

}

}

**Output**:



**Decryption Of MD5 Algorithms?**

The MD5 cryptographic algorithm only works one way. It’s possible to crypt a word into MD5 with Java, but there is no reverse function. In case of a password verification, the best practice is to also crypt to entered password and compare the result with the original one.

* The output is always 32 characters long, but you can hash anything in 32 characters.
* So, the MD5 algorithm output is not unique. Two word or files can have the same MD5 hash.
* Given this information, it’s not possible to reverse a hash to the original word.

The MD5 algorithm has a weakness we can exploit, each time you create a MD5 hash of a word, you get the same result.

There is no decryption algorithm for MD5, but there is a solution.  
For example, you now know that the MD5 hash from “Group 1 Computer Security CM5467” is 7fc0fc284848ef97ff89bf3767389209.  
If someone is looking for the word corresponding to this hash, there is a good chance that “Group 1 Computer Security CM5467” was the original password.

To verify the login credentials, they just encrypt the typed password in MD5 and compare this hash to the one stored in the database.

If there is a match, we consider that the login is valid

## Verification Of User :

## 

## 

**MD5 password verification in Java**

Once the password is encrypted and stored in the database, you can use a simple condition to check that the login attempt you try to validate is correct.

The idea is to compare the input password to the stored password for this user:

**Java Code :**

String password = " Group 1 Computer Security CM5467";

String stored\_password = "7fc0fc284848ef97ff89bf3767389209";

if(MD5(password).equals(stored\_password)) {

#Authentication success, do whatever was asked

System.out.println("Password Verified");

}

else {

#Doesn't match, display error and ask the password again

System.out.println("Password Not Matched");

**Advantages of MD5 :**

1. Easy to Compare small hashes
2. Storing passwords is convenient
3. Low resource consumption

**Conclusion:**

In this project we learnt the encryption algorithm of MD5 Hash and Implemented the Java code for encryption and user password verification. We learnt about privacy and security concerns to be followed to provide user security.