UNIVERSITY INSTITUTE OF ENGINEERING

Department of Computer Science & Engineering

Subject Name: Web And Mobile Security Lab

Subject Code: 20CSP-338

Submitted to: Renuka Ratten

Faculty name: Renuka Ratten

Submitted by: Pranjal Kumar

Name: Pranjal Kumar

UID: 20BCS3504

Section: 607

Group: B

Ex. No	List of Experiments	Conduct (MM: 12)	Viva (MM: 10)	Record (MM: 8)	Total (MM: 30)	Date	Remarks/Signature
1.1	Open any website on computer system and identify http packet on monitoring tool like Wireshark.					19/08/22	
1.2	Design a method to simulate the HTML injections and cross-site scripting (XSS) to exploit the attackers.					28/08/22	
1.3	Implementation of Cross site request forgery (XSRF) attack.					16/09/22	
1.4	Implementation of Design methods to break authentication schemes (SQL Injection attack).					04/10/22	
2.1	Write a program to generate message digest for the given message using the SHA/MD5 algorithm and verify the integrity of message.					19/10/22	
2.2	Perform Penetration testing on a web application to gather information about the system (Foot Printing).					20/10/22	

Experiment 2.1

Student Name: Pranjal Kumar **UID: 20BS3504**

Branch: CSE **Section/Group: 607-B**

Semester: 5th **Date of Performance: 19/10/22**

Subject Name: WMS Lab **Subject Code: CSP-338**

Aim: Write a program to generate message digest for the given message using the SHA/MD5 algorithm and verify the integrity of message.

Software/Hardware Requirements:

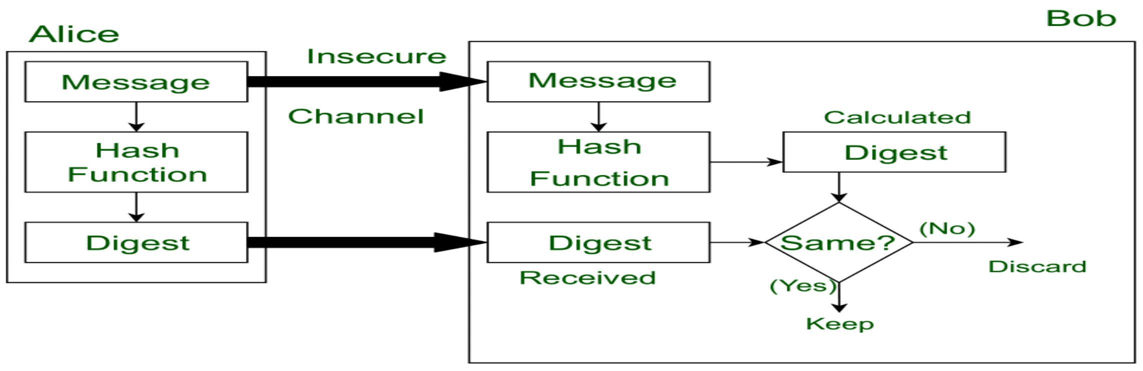
Windows 7 and above version.

Tools to be used:

1. Eclipse IDE
2. JDK (Java Development kit)
3. IntelliJ IDEA

INTRODUCTION

Message Digest is used to ensure the integrity of a message transmitted over an insecure channel (where the content of the message can be changed). The message is passed through a Cryptographic hash function. This function creates a compressed image of the message called **Digest**.



Steps/Method/Coding:
To calculate cryptographic hashing value in Java, **MessageDigest** Class is used, under the package java.security.
MessageDigest Class provides following cryptographic hash function to find hash value of a text as follows:

- MD2
- MD5
- SHA-1
- SHA-224
- SHA-256
- SHA-384
- SHA-512

1.This Algorithms are initialize in static method called **getInstance()**.
2. After selecting the algorithm it calculate the **digest** value and return the results in byte array.
3. BigInteger class is used, which converts the resultant byte array into its **sign- magnitude representation**.
4.This representation is then converted into a hexadecimal format to get the expected MessageDigest.

A message digest is a fixed size numeric representation of the contents of a message, computed by a hash function. A message digest can be encrypted, forming a digital signature.

Messages are inherently variable in size. A message digest is a fixed size numeric representation of the contents of a message. A message digest is computed by a hash function, which is a transformation that meets two criteria:

- The hash function must be one way. It must not be possible to reverse the function to find the message corresponding to a particular message digest, other than by testing all possible messages.
- It must be computationally infeasible to find two messages that hash to the same digest.

The message digest is sent with the message itself. The receiver can generate a digest for the message and compare it with the digest of the sender. The integrity of the message is verified when the two message digests are the same. Any tampering with the message during transmission almost certainly results in a different message digest.

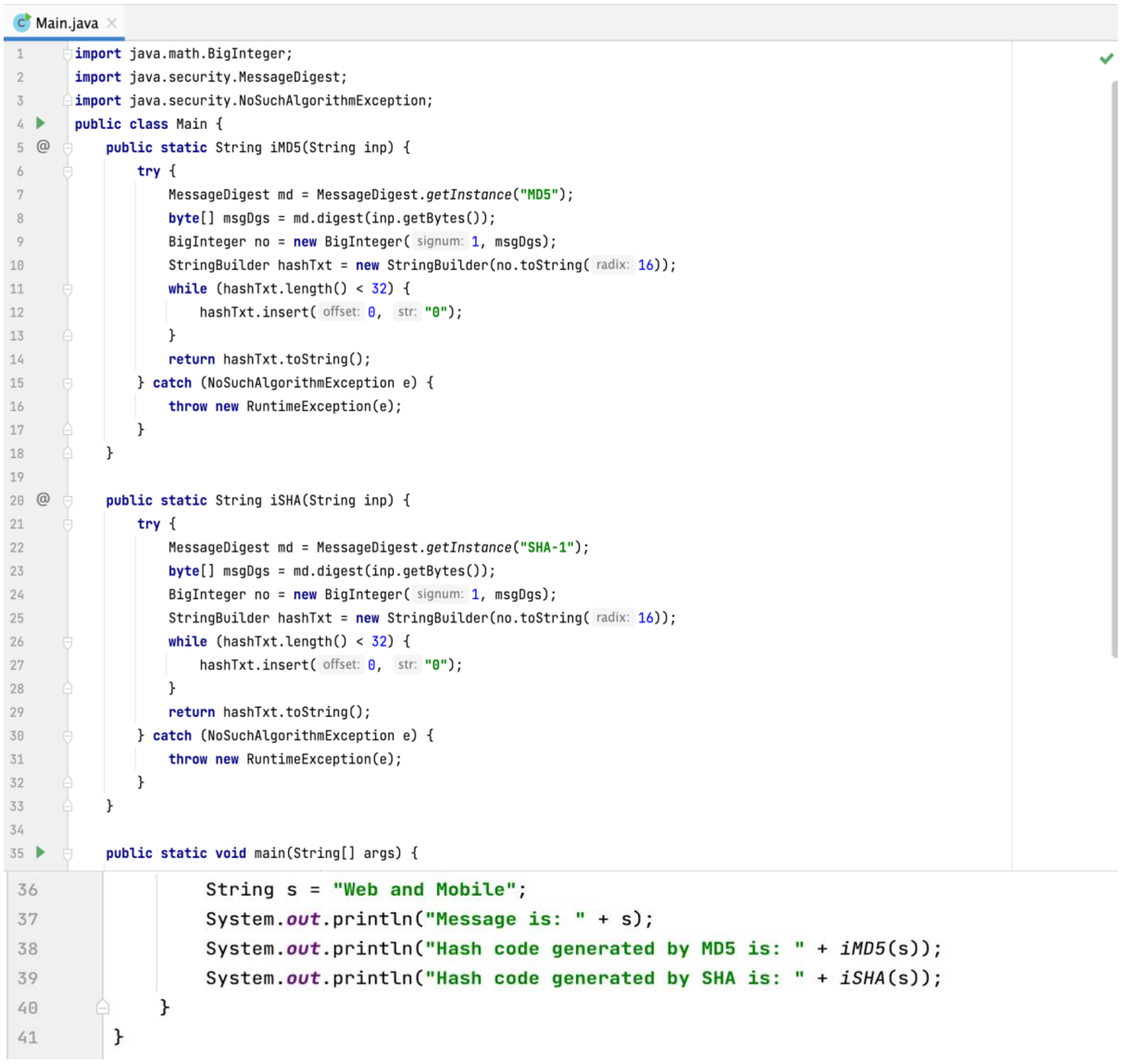
A message digest created using a secret symmetric key is known as a Message Authentication Code (MAC), because it can provide assurance that the message has not been modified.

The sender can also generate a message digest and then encrypt the digest using the private key of an asymmetric key pair, forming a digital signature. The signature must then be decrypted by the receiver, before comparing it with a locally generated digest.

Coding (MD5 algorithm & SHA algorithm):

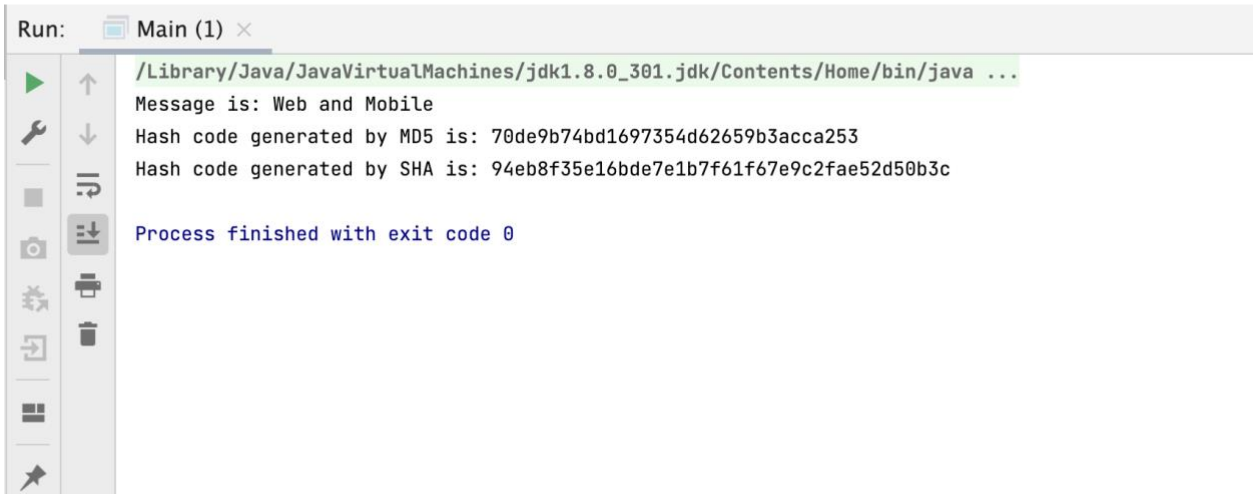
```
import java.math.BigInteger;

import java.security.MessageDigest;

import java.security.NoSuchAlgorithmException;

public class Main {

    public static String iMD5(String inp) {

        try {

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

            byte[] msgDgs = md.digest(inp.getBytes());

            BigInteger no = new BigInteger(1, msgDgs);

            StringBuilder hashTxt = new StringBuilder(no.toString(16));

            while (hashTxt.length() < 32) {

                hashTxt.insert(0, "0");

            }

            return hashTxt.toString();

        } catch (NoSuchAlgorithmException e) {
            throw new RuntimeException(e);
        }
    }
    public static String iSHA(String inp) {
        try {
            MessageDigest md = MessageDigest.getInstance("SHA-1");
            byte[] msgDgs = md.digest(inp.getBytes());
            BigInteger no = new BigInteger(1, msgDgs);
            StringBuilder hashTxt = new StringBuilder(no.toString(16));
            while (hashTxt.length() < 32) {
                hashTxt.insert(0, "0");
            }
            return hashTxt.toString();
        } catch (NoSuchAlgorithmException e) {
            throw new RuntimeException(e);
        }
    }
    public static void main(String[] args) {
        String s = "Web and Mobile";
        System.out.println("Message is: " + s);
        System.out.println("Hash code generated by MD5 is: " + iMD5(s));
        System.out.println("Hash code generated by SHA is: " + iSHA(s));
```

```
   }
}
```



```
import java.math.BigInteger;
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;
public class Main {
    public static String iMD5(String inp) {
        try {
            MessageDigest md = MessageDigest.getInstance("MD5");
            byte[] msgDgs = md.digest(inp.getBytes());
            BigInteger no = new BigInteger( signum: 1, msgDgs);
            StringBuilder hashTxt = new StringBuilder(no.toString( radix: 16));
            while (hashTxt.length() < 32) {
                hashTxt.insert( offset: 0,  str: "0");
            }
            return hashTxt.toString();
        } catch (NoSuchAlgorithmException e) {
            throw new RuntimeException(e);
        }
    }

    public static String iSHA(String inp) {
        try {
            MessageDigest md = MessageDigest.getInstance("SHA-1");
            byte[] msgDgs = md.digest(inp.getBytes());
            BigInteger no = new BigInteger( signum: 1, msgDgs);
            StringBuilder hashTxt = new StringBuilder(no.toString( radix: 16));
            while (hashTxt.length() < 32) {
                hashTxt.insert( offset: 0,  str: "0");
            }
            return hashTxt.toString();
        } catch (NoSuchAlgorithmException e) {
            throw new RuntimeException(e);
        }
    }

    public static void main(String[] args) {
        String s = "Web and Mobile";
        System.out.println("Message is: " + s);
        System.out.println("Hash code generated by MD5 is: " + iMD5(s));
        System.out.println("Hash code generated by SHA is: " + iSHA(s));
    }
}
```

OUTPUT:



```
/Library/Java/JavaVirtualMachines/jdk1.8.0_301.jdk/Contents/Home/bin/java ...
Message is: Web and Mobile
Hash code generated by MD5 is: 70de9b74bd1697354d62659b3acca253
Hash code generated by SHA is: 94eb8f35e16bde7e1b7f61f67e9c2fae52d50b3c

Process finished with exit code 0
```

Learning Outcomes:

Output is often known as hash values, hash codes, message digest. The length of output hashes is generally less than its corresponding input message length.

Evaluation Grid :

Sr. No.	Parameters	Marks Obtained	Maximum Marks
1.	Student Performance (Conduct of experiment) objectives/Outcomes.		12
2.	Viva Voce		10
3.	Submission of Work Sheet (Record)		8
	Total		30