Computing MACs, Hashes and HMACs for messages

SUBJECT NAME: CRYPTOGRAPHY AND NETWORK SECURITY

SUBJECT CODE: CS6008

MODULE: 8

NAME	BHUVANESHWAR S
REG.NO	2019103513
DATE	11/06/2022

AIM:

To compute MACs, Hashes and HMACs for a given message.

TOOLS INVOLVED:

- JAVA
- CMD PROMPT
- VISUAL STUDIO CODE

PROBLEM DESCRIPTION:

1) **MESSAGE AUTHENTICATION CODE**

MAC algorithm is a symmetric key cryptographic technique to provide message authentication. For establishing MAC process, the sender and receiver share a symmetric key K. Essentially, a MAC is an encrypted checksum generated on the underlying message that is sent along with a message to ensure message authentication.

2) **HASHING**

Hashing is an algorithm performed on data such as a file or message to produce a number called a hash. The hash is used to verify that data is not modified, tampered with, or corrupted. In other words, you can verify the data has maintained integrity. A key point about a hash is that no matter how many times you execute the hashing algorithm against the data, the hash will always be the same if the data is the same.

3) **HMAC**

Hash-Based Message Authentication Code (HMAC) is a specific type of message authentication code (MAC) involving a cryptographic hash function and a secret cryptographic key. As with any MAC, it may be used to simultaneously verify both the data integrity and authenticity of a message.

HMAC can provide authentication using a shared secret instead of using digital signatures with asymmetric cryptography. It trades off the need for a complex public key infrastructure by delegating the key exchange to the communicating parties, who are responsible for establishing and using a trusted channel to agree on the key prior to communication.

INPUT:

Getting a input message from the user.

OUTPUT:

Computing MACs, Hashes and HMACs for a given messages.

SCREENSHOT:

1) **MAC**

FILENAME: macGen.java

```
import java.security.Key;
import java.security.SecureRandom;
import java.util.Scanner;

import javax.crypto.KeyGenerator;
import javax.crypto.Mac;

public class macGen {
```

```
    public static void main(String args[]) throws Exception {

        KeyGenerator keyGen = KeyGenerator.getInstance("DES");

        Scanner sc = new Scanner(System.in);
        SecureRandom secRandom = new SecureRandom();


        keyGen.init(secRandom);


        Key key = keyGen.generateKey();
        Mac mac = Mac.getInstance("HmacSHA256");
        mac.init(key);
        System.out.print("[+]\tEnter plain text :\t");
        String input = sc.nextLine();
        String msg = new String(input);
        byte[] bytes = msg.getBytes();
        byte[] macResult = mac.doFinal(bytes);

        System.out.println("Messafe Digest : ”);
        System.out.println(new String(macResult));
    }
}
```

OUTPUT

```
PS E:\clg 6th sem\Crypto&Net Security\assignment\hashes> javac macGen.java
PS E:\clg 6th sem\Crypto&Net Security\assignment\hashes> java macGen
[+]     Enter plain text :      College of Engineering, Guindy - Anna University
Messafe Digest :
?#[??ú%??ºTÜ ?ùk?|={ÉR7Q?↕>◄Y?x@
PS E:\clg 6th sem\Crypto&Net Security\assignment\hashes>
```

2) HASHING USING SHA256

Sha256Gen.java

```
import java.math.BigInteger;
import java.nio.charset.StandardCharsets;
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;
import java.util.Scanner;


class sha256Gen {
    public static byte[] getSHA(String input) throws NoSuchAlgorithmException
    {

        MessageDigest md = MessageDigest.getInstance("SHA-256");

        return md.digest(input.getBytes(StandardCharsets.UTF_8));
    }

    public static String toHexString(byte[] hash)
```

```
    {

        BigInteger number = new BigInteger(1, hash);


        StringBuilder hexString = new StringBuilder(number.toString(16));


        while (hexString.length() < 64)
        {
            hexString.insert(0, '0');
        }

        return hexString.toString();
    }


    public static void main(String args[])
    {
        Scanner sc = new Scanner(System.in);
        try
        {
            System.out.println("[+] Enter message : ");

            String s1 = sc.nextLine();
            System.out.println("\n" + s1 + " - SHA256 : " + toHexString(getSHA(s1)));
        }

        catch (NoSuchAlgorithmException e) {
            System.out.println("Exception thrown for incorrect algorithm: " + e);
        }
    }
}
```

OUTPUT:

```
PS E:\clg 6th sem\Crypto&Net Security\assignment\hashes> javac sha256Hash.java
PS E:\clg 6th sem\Crypto&Net Security\assignment\hashes> java sha256Gen
[+] Enter message :
College of Engineering, Guindy - Anna University

College of Engineering, Guindy - Anna University - SHA256 : 45c89bc6a4a0975e416839adf1206c253913bf285c95776cd88ff48c6f338fb1
```

3) HMAC

hmacGen.java

```
import java.math.BigInteger;
import java.util.Base64;
import java.util.Scanner;

import javax.crypto.Mac;
import javax.crypto.spec.SecretKeySpec;

public class hmacGen{
```

```
 private static class HMAC {

  public static byte[] hmac256(String secretKey,String message){
   try{
    return hmac256(secretKey.getBytes("UTF-8"), message.getBytes("UTF-8"));
   }catch(Exception e){
    throw new RuntimeException("Failed to generate HMACSHA256 encrypt",e);
   }
  }

  public static byte[] hmac256(byte[] secretKey,byte[] message){
   byte[] hmac256 = null;
   try{
    Mac mac = Mac.getInstance("HmacSHA256");
    SecretKeySpec sks = new SecretKeySpec(secretKey, "HmacSHA256");
    mac.init(sks);
    hmac256 = mac.doFinal(message);
    return hmac256;
   }catch(Exception e){
    throw new RuntimeException("Failed to generate HMACSHA256 encrypt ");
   }
  }
 }

 public static void main(String args[]){
  Scanner sc = new Scanner(System.in);
  System.out.println("[+] Enter message : ");

  String s1 = sc.nextLine();
  byte[] hmacSha256 = HMAC.hmac256("secreT1_", s1);
  System.out.println(String.format("Hex: %032x", new BigInteger(1, hmacSha256)));

  String base64HmacSha256 = Base64.getEncoder().encodeToString(hmacSha256);
  System.out.println("Base64: " + base64HmacSha256);
 }
}
```

OUTPUT

```
PS E:\clg 6th sem\Crypto&Net Security\assignment\hashes> javac hmacGen.java
PS E:\clg 6th sem\Crypto&Net Security\assignment\hashes> java hmacGen
[+] Enter message :
College of Engineering - Guindy, Anna University
Hex: 2936bd40b8b8053d759719fb082c92a17b043bc51de659a285278d3bb3913a68
Base64: KTa9QLi4BT11lxn7CCySoXsEO8Ud5lmihSeNO7OROmg=
PS E:\clg 6th sem\Crypto&Net Security\assignment\hashes> '
```