1.Tujuan

- Memahami dasar security dan kriptografi
- Memahami proteksi domains dan permissions
- Bagaimana menambahkan permissions pada MIDlet Suite
- Bagaimana membuat MIDlet Suite menggunakan NetBeans Mobility Pack
- Bagaimana membuat message digest menggunakan SATSA
- Bagaimana melakukan enkripsi menggunakan symmetric keys

2. Latar Belakang

Kriptografi adalah cabang dari ilmu matematika yang memiliki banyak fungsi dalam pengamanan data. Kriptografi adalah proses mengambil message dan menggunakan beberapa fungsi untuk menggenerasi materi kriptografis (sebuah digest atau message terenkripsi).
Kriptografi adalah salah satu dari teknologi yang digunakan dalam layanan security seperti integrity, confidentiality, identity dan non repudiation.

3. Percobaan

Percobaan 1: Digest MIDlet

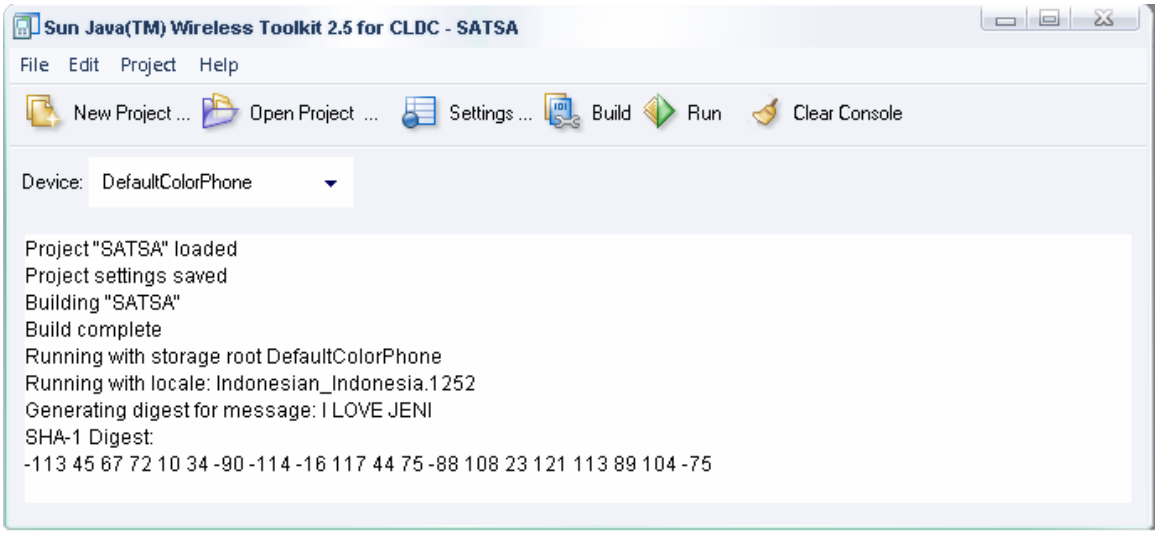
```
import javax.microedition.midlet.*;
import javax.microedition.lcdui.*;
import java.security.*;

public class DigestMidlet extends MIDlet {
    public void startApp() {
        String message = "I LOVE JENI";
        System.out.println("Generating digest for message: "+message);
        byte[] digest = generateDigest(message.getBytes());
        System.out.println("SHA-1 Digest:");
        for (int i=0;i<digest.length;i++){

          System.out.print(digest[i]+" ");
        }
        System.out.println();
    }
```

```
    public void pauseApp() {
    }
    public void destroyApp(boolean unconditional) {
    }
    public byte[] generateDigest(byte[] message){
        String algorithm = "SHA-1";
        int digestLength = 20;
        byte[] digest = new byte[digestLength];
        try{
            MessageDigest md;
            md = MessageDigest.getInstance(algorithm);
            md.update(message,0,message.length);
            md.digest(digest,0,digestLength);
        }catch(Exception e){
            System.out.println("Exception :"+e.getMessage());
        }
       return digest;
}
}
```

Hasil :



Percobaan 2 : SymmetricChiperMIDlet

```
import javax.microedition.midlet.*;
import javax.microedition.lcdui.*;
import java.security.*;
import java.security.spec.*;
import javax.crypto.*;
import javax.crypto.spec.*;
import javax.microedition.securityservice.*;

public class SymmetricChiperMidlet extends MIDlet {
    private static final byte[] key = {
        (byte) 0xab,(byte) 0xcd,(byte) 0xef,(byte) 0x88,
        (byte) 0x12,(byte) 0x34,(byte) 0x56,(byte) 0x78
    };
    String message = "I LOVE JENI!";

    public SymmetricChiperMidlet(){
        try{
            System.out.println("Original Message : "+message);
            printBytes(message.getBytes());
            byte[] encryptedMessage =
encrypt("DES/ECB/PKCS5Padding",message.getBytes(),key,"DES");
```

```
            System.out.println("Encrypted Message : ");
            printBytes(encryptedMessage);
            byte[] decryptedMessage =
decrypt("DES/ECB/PKCS5Padding",encryptedMessage,key,"DES");
            System.out.println("Decrypted Message : ");
            printBytes(decryptedMessage);
        }catch(Exception e){
            System.out.println("Exception : "+e.getMessage());
            e.printStackTrace();
        }
    }
    private void printBytes(byte[] bytes){
        for(int i=0;i<bytes.length;i++){
            System.out.print(toHex(bytes[i])+" ");
        }
        System.out.println();
    }
    private String toHex(byte b){
        char d1= toHexDigit((byte) ((b>>4)& 0x0f));
        char d2= toHexDigit((byte) (b & 0x0f));
        return ("0x"+d1+d2);
    }
    private char toHexDigit(byte x){
        if (x > 9){
            return ((char)('A'+(x-10)));
        }else {
            return((char)('0'+x));
        }
    }
    private byte[] encrypt(String algorithm,byte[] message,byte[] keybytes,String keyAlgo)
throws
NoSuchAlgorithmException,NoSuchPaddingException,InvalidKeyException,ShortBufferException,Ill
egalBlockSizeException,BadPaddingException {
        Cipher cipher = Cipher.getInstance(algorithm);
        Key key = new SecretKeySpec(keybytes,0,keybytes.length,keyAlgo);
        int blockSize = 16;
        int cipherLength = ((message.length/blockSize)+((message.length%blockSize)>0?1
:0))*blockSize;
        cipher.init(Cipher.ENCRYPT_MODE,key);
        byte[] cipherText = new byte[cipherLength];
        cipher.doFinal(message,0,message.length,cipherText,0);
        return (cipherText);
    }
    public byte[] decrypt(String algorithm,byte[] cipherText,byte[] keybytes,String keyAlgo)
throws
NoSuchAlgorithmException,NoSuchPaddingException,InvalidKeyException,ShortBufferException,Ill
egalBlockSizeException,BadPaddingException {
        Cipher cipher = Cipher.getInstance(algorithm);
        Key key = new SecretKeySpec(keybytes,0,keybytes.length,keyAlgo);
        cipher.init(Cipher.DECRYPT_MODE,key);
```

```
        byte[] decrypted = new byte[message.length()];
        cipher.doFinal(cipherText,0,cipherText.length,decrypted,0);
        return(decrypted);
    }
    public void startApp() {
    }
    public void pauseApp() {
    }
    public void destroyApp(boolean unconditional) {
    }
}
```

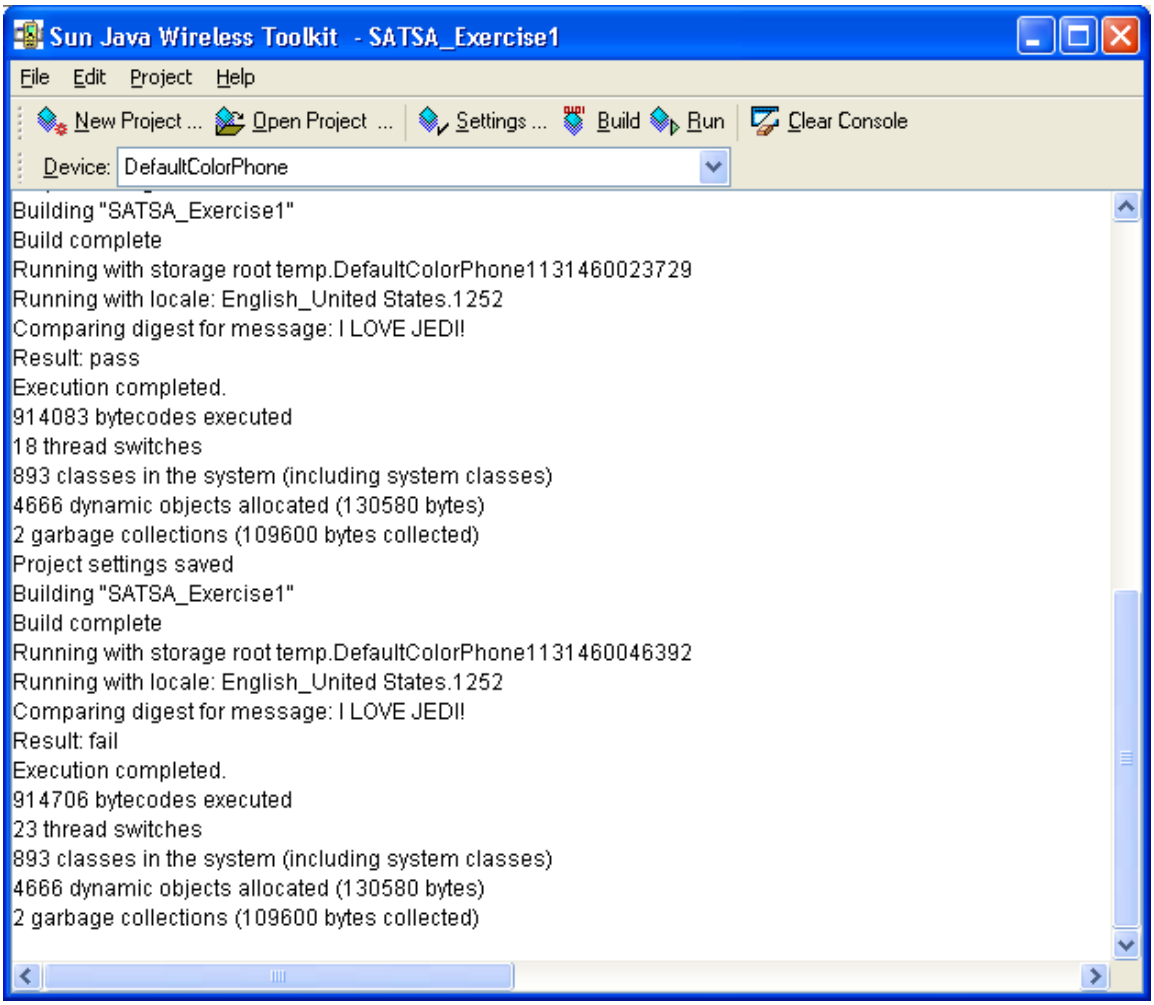
LATIHAN

7.7.1 Verifikasi Integritas Message Digest

Buatlah sebuah method yang akan mem-verifikasi integritas dari sebuah pesan dengan cara membandingkan antara digest dari message tersebut dengan message asli.

public boolean verifyDigest(byte[] message, byte[] originalDigest)

Hasil :


Sun Java Wireless Toolkit - SATSA_Exercise1
File Edit Project Help
New Project ... Open Project ... Settings ... Build Run Clear Console
Device: DefaultColorPhone
Building "SATSA_Exercise1"
Build complete
Running with storage root temp.DefaultColorPhone1131460023729
Running with locale: English_United States.1252
Comparing digest for message: I LOVE JEDI!
Result: pass
Execution completed.
914083 bytecodes executed
18 thread switches
893 classes in the system (including system classes)
4666 dynamic objects allocated (130580 bytes)
2 garbage collections (109600 bytes collected)
Project settings saved
Building "SATSA_Exercise1"
Build complete
Running with storage root temp.DefaultColorPhone1131460046392
Running with locale: English_United States.1252
Comparing digest for message: I LOVE JEDI!
Result: fail
Execution completed.
914706 bytecodes executed
23 thread switches
893 classes in the system (including system classes)
4666 dynamic objects allocated (130580 bytes)
2 garbage collections (109600 bytes collected)

Jawaban:

```
import javax.microedition.midlet.*;
import javax.microedition.lcdui.*;
import javax.microedition.io.*;

public class Exercise61 extends MIDlet implements CommandListener, Runnable {
    private Display display;
    private Command exitCommand = new Command("Exit", Command.EXIT, 1);
    private Command okCommand = new Command("Fetch", Command.OK, 1);
    private TextField urlField = new TextField(
            "URL",
            "http://localhost:8084/Chapter07/",
            64, TextField.URL);
    private StringItem message = new StringItem("URL Fetcher", "");
    HttpConnection connection = null;

    public void startApp() {
        display = Display.getDisplay( this );
        display.setCurrent(new MainForm());
```

```
}

public void pauseApp() {
}

public void destroyApp(boolean unconditional) {
}

class MainForm extends Form {
    MainForm(){
        // Setup the Form
        super("Exercise 6.1");
        addCommand(exitCommand);
        addCommand(okCommand);
        append(urlField);
        append(message);
        setCommandListener(Exercise61.this);
    }
}
```

```
public void commandAction(Command c, Displayable d){
    if(c == exitCommand){
        notifyDestroyed();
    }
    if (c == okCommand){
        try {
            Thread t = new Thread(this);
            t.start();
        } catch (Exception e){}
    }
}

public void run(){
    try {
        String url = urlField.getString();
        connection = (HttpConnection)
        Connector.open(url);
```

```
StringBuffer buff = new StringBuffer(1024);
buff.append("Response Code: ");
buff.append(connection.getResponseCode());
buff.append("\n");

buff.append("Response Mesg: ");
buff.append(connection.getResponseMessage());
buff.append("\n");

buff.append("Length: ");
buff.append(connection.getLength());
buff.append("\n");

buff.append("Content Type: ");
buff.append(connection.getType());
buff.append("\n");

buff.append("getEncoding: ");
buff.append(connection.getEncoding());
```

```
buff.append("\n");

buff.append("Date: ");
buff.append(connection.getDate());
buff.append("\n");

buff.append("Expiration: ");
buff.append(connection.getExpiration());
buff.append("\n");

buff.append("Last Modified: ");
buff.append(connection.getLastModified());
buff.append("\n");

buff.append("URL: ");
buff.append(connection.getURL());
buff.append("\n");

buff.append("Request Method: ");
```

```
            buff.append(connection.getRequestMethod());
            buff.append("\n");


            message.setText(buff.toString());
        } catch (Exception ex){


        }
    }
}
```