Programación de servicios y procesos

Act5.4. Firmado digital de documentos

Francisco José García Cutillas | 2FPGS_DAM

Índice

Ejercicio 1

Firmado digital de documentos

Diseña y desarrolla un sistema en Java que solicite un fichero y genere la firma digital del mismo.

Crea el programa capaz de validar que el documento firmado no ha sufrido alteraciones y que el emisor es quien dice ser.

Clase Main

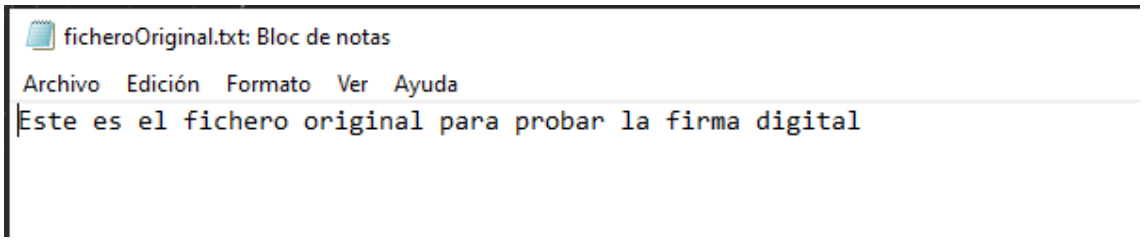
```java
package com.mycompany.psp_act5_4;

import java.security.Signature;

/**
 *
 * @author Fran
 */
public class PSP_Act5_4 {

    private static String FICHERO_ORIGINAL = "ficheroOriginal.txt";
    private static String FICHERO_MODIFICADO = "ficheroModificado.txt";

    public static void main(String[] args) {

        try{

            Signature signature = Signature.getInstance( algorithm: "DSA");
            signature.initSign( privateKey: ClavesManager.getClavePrivada());
            signature.update( data: FICHERO_ORIGINAL.getBytes());
            byte[] firma = signature.sign();

            signature.initVerify( publicKey: ClavesManager.getClavePublica());
            signature.update( data: FICHERO_ORIGINAL.getBytes());

            if (signature.verify( signature: firma)) {

                System.out.println( x: "Fichero verificado correctamente");

            } else {

                System.out.println( x: "¡Atención! El fichero no es fiable");

            }

        } catch (Exception ex){

            System.out.println("Error: " + ex.getMessage());

        }
    }
}
```

Clase ClavesManager

```java
package com.mycompany.psp_act5_4;

import java.io.File;
import java.io.FileOutputStream;
import java.nio.file.Files;
import java.security.KeyFactory;
import java.security.KeyPair;
import java.security.KeyPairGenerator;
import java.security.NoSuchAlgorithmException;
import java.security.PrivateKey;
import java.security.PublicKey;
import java.security.spec.EncodedKeySpec;
import java.security.spec.PKCS8EncodedKeySpec;
import java.security.spec.X509EncodedKeySpec;

/**
 *
 * @author Fran
 */
public class ClavesManager {

    private static final String FICHERO_CLAVE_PUBLICA = "clavePublica.key";
    private static final String FICHERO_CLAVE_PRIVADA = "clavePrivada.key";

    //Método que genera el par de claves pública y privada. Dewvuelve un KeyPair
    public static KeyPair generarClaves() throws NoSuchAlgorithmException {

        KeyPairGenerator generador = KeyPairGenerator.getInstance( algorithm: "DSA");
        generador.initialize( keysize: 512);
        KeyPair claves = generador.generateKeyPair();

        return claves;

    }

    //Método para guardar las claves en un fichero
    public static void guardarClaves(KeyPair claves) throws Exception {

        FileOutputStream fos = new FileOutputStream( name: FICHERO_CLAVE_PUBLICA);
        fos.write( b: claves.getPublic().getEncoded());
        fos.close();

        fos = new FileOutputStream( name: FICHERO_CLAVE_PRIVADA);
        fos.write( b: claves.getPrivate().getEncoded());
        fos.close();

    }

    //Método para obtener la clave privada de un fichero de clave
    public static PrivateKey getClavePrivada() throws Exception {

        File ficheroClavePrivada = new File( pathname: FICHERO_CLAVE_PRIVADA);
        byte[] bytesClavePrivada = Files.readAllBytes( path: ficheroClavePrivada.toPath());
        KeyFactory keyFactory = KeyFactory.getInstance( algorithm: "DSA");
        EncodedKeySpec publicKeySpec = new PKCS8EncodedKeySpec( encodedKey: bytesClavePrivada);
        PrivateKey clavePrivada = keyFactory.generatePrivate( keySpec: publicKeySpec);

        return clavePrivada;

    }

    //Método para obtener la clave pública de un fichero de clave
    public static PublicKey getClavePublica() throws Exception {

        File ficheroClavePublica = new File( pathname: FICHERO_CLAVE_PUBLICA);
        byte[] bytesClavePublica = Files.readAllBytes( path: ficheroClavePublica.toPath());
        KeyFactory keyFactory = KeyFactory.getInstance( algorithm: "DSA");
        EncodedKeySpec publicKeySpec = new X509EncodedKeySpec( encodedKey: bytesClavePublica);
        PublicKey clavePublica = keyFactory.generatePublic( keySpec: publicKeySpec);

        return clavePublica;

    }

    public static void main(String[] args) {

        try {

            KeyPair claves = generarClaves();
            guardarClaves(claves);

        } catch (Exception ex) {

            System.out.println("Error: " + ex.getMessage());

        }

    }

}
```

Para comenzar, vamos a ejecutar primero la clase ClavesManager con su método main, el cual nos generará un par de claves pública y privada que serán almacenadas cada una en un fichero con extensión ".rsa".

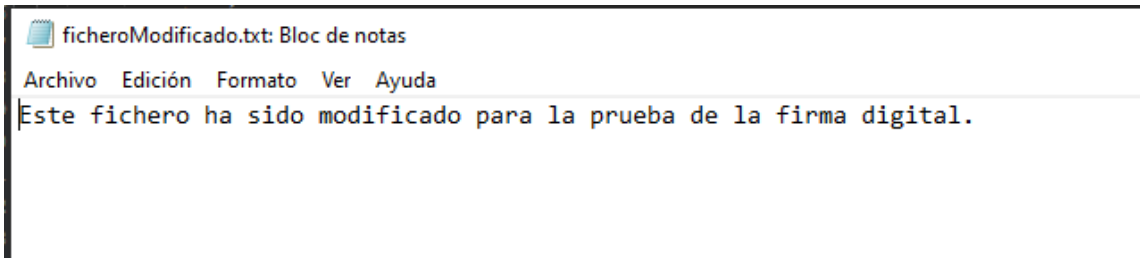
Una vez que tenemos las claves creadas y guardadas, vamos a probar con el fichero original. Lo firmamos y luego volvemos a comprobar la integridad del mismo fichero. Éste sería el contenido del fichero original.



Obtendríamos el siguiente resultado.

```
--- exec-maven-plugin:3.0.0:exec (default-cli) @ PSP_Act5_4 ---
Fichero verificado correctamente
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  0.524 s
Finished at: 2024-02-10T14:14:42+01:00
------------------------------------------------------------------------
```

Ahora vamos a probar a firmar el original, y comprobar otro fichero que ha sido modificado y no tiene el mismo contenido.




```
Signature signature = Signature.getInstance( algorithm: "DSA");
signature.initSign( privateKey: ClavesManager.getClavePrivada());
signature.update( data: FICHERO_ORIGINAL.getBytes());
byte[] firma = signature.sign();

signature.initVerify( publicKey: ClavesManager.getClavePublica());
signature.update( data: FICHERO_MODIFICADO.getBytes());
```

```
----------------------< com.mycompany:PSP_Act5_4 >----------------------
Building PSP_Act5_4 1.0-SNAPSHOT
--------------------------------[ jar ]---------------------------------

--- exec-maven-plugin:3.0.0:exec (default-cli) @ PSP_Act5_4 ---
¡Atención! El fichero no es fiable
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  0.368 s
Finished at: 2024-02-10T14:16:32+01:00
------------------------------------------------------------------------
```

Podemos observar que al comprobar que la firma no es la misma, nos avisa de que el fichero ha sido modificado.