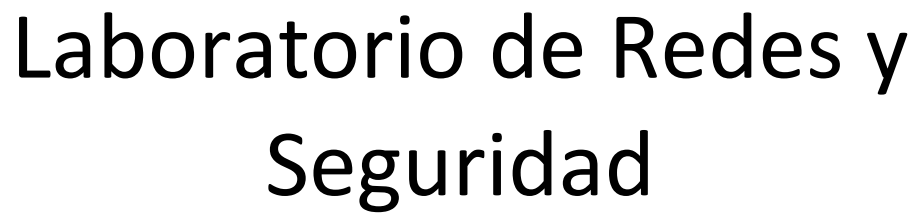



Laboratorios de docencia




CALIFICACIÓN:

	Manual de prácticas del Laboratorio de Administración de Redes	Código:	MADO-32
		Versión:	02
		Página	140/174
		Sección ISO	8.3
		Fecha de emisión	28 de julio de 2017
Facultad de Ingeniería		Área/Departamento: Laboratorio de Redes y Seguridad	
La impresión de este documento es una copia no controlada			

Práctica 10

Mecanismos de Seguridad, Firma Digital

Control

	Manual de prácticas del Laboratorio de Administración de Redes	Código:	MADO-32
		Versión:	02
		Página	141/174
		Sección ISO	8.3
		Fecha de emisión	28 de julio de 2017
Facultad de Ingeniería		Área/Departamento: Laboratorio de Redes y Seguridad	
La impresión de este documento es una copia no controlada			

1. Objetivos de Aprendizaje

- El alumno se familiarizará con herramientas básicas relacionadas con la seguridad en la red, las cuales podrán ser estudiadas a profundidad en subsecuentes materias.
- El alumno realizará una aplicación que permita el proceso de firmado de un documento a través de la infraestructura de claves asimétricas, en el lenguaje orientado a objetos Java.

2. Conceptos teóricos

En diversas organizaciones, tanto públicas como comerciales, el uso de Internet se ha convertido en un foro principal para hacer negocios, sirviendo como medio principal para la publicidad, el *marketing*, las ventas y la atención al cliente. Ha permitido que las empresas crezcan y a las grandes corporaciones, expandir su dominio.


Junto a las oportunidades que ofrece Internet se encuentran riesgos de seguridad importantes. Los servidores Web pueden ser sustituidos y a veces, las páginas Web han sido desconfiguradas. Los datos privados de los consumidores podrían ser revelados. Las transacciones financieras pueden ser falsificadas. Los cortafuegos de las empresas pueden ser infringidos y las redes de la empresa saboteadas. Todos estos escenarios conllevan a un uso seguro de Internet.

La comunicación segura dentro de una red debe cumplir con las siguientes características: confidencialidad, autenticación, integridad del mensaje, disponibilidad y control de acceso. Las 3 primeras características se han considerado componentes claves de una red segura, sin embargo, se han añadido en las últimas décadas la necesidad de mantener la red operando.

El concepto de firma digital fue introducido por Diffie y Hellman en 1976, siendo un bloque de caracteres que acompaña a un documento acreditando quién es su autor (autenticación) y que no ha existido ninguna manipulación posterior de los datos (integridad).

El proceso de firmado digital se inicia cuando el autor de un documento utiliza su clave secreta dentro del esquema de cifrado asimétrico, a la que sólo él tiene acceso, esto impide que pueda negar su autoría (revocación o no repudio). De esta forma el autor es vinculado al documento de la firma. El software del autor aplica un algoritmo hash sobre el texto a firmar, obteniendo un extracto de longitud fija y absolutamente específico del mensaje. Un cambio mínimo en el mensaje dará lugar a una cadena hash distinta. El extracto tiene una longitud de 128 a 160 bits, dependiendo del algoritmo utilizado, entre los que se encuentran: MD5 o SHA-1. El algoritmo más utilizado en el proceso de encriptación asimétrica es el RSA.

La validez de la firma es probada por cualquier persona que disponga de la clave pública del autor.

	Manual de prácticas del Laboratorio de Administración de Redes	Código:	MADO-32
		Versión:	02
		Página	142/174
		Sección ISO	8.3
		Fecha de emisión	28 de julio de 2017
Facultad de Ingeniería		Área/Departamento: Laboratorio de Redes y Seguridad	
La impresión de este documento es una copia no controlada			

La identificación indubitable de las personas y organizaciones se sustenta en la firma digital que opera bajo la denominada Infraestructura de Clave Pública.

La “despapelización”, como resultado de la utilización de la firma digital, da origen al Documento Digital, garantizando la autenticidad, integridad, no repudio y eventualmente la confidencialidad del mismo.

Es así que vistos los tiempos en que vivimos, es aplicable entre otras a:


- Identificación como usuario ante redes internas o externas (abiertas o cerradas).
- Correos electrónicos.
- Identificación de sitios en Internet.
- Transacciones EDI (Electronic Data Interchange).
- Comercio electrónico.
- Información que se obtenga de Internet.
- Transacciones financieras.
- Software y hardware.
- Comercio exterior.
- Comercio interno.
- Toda documentación que precise movilizarse rápidamente o por el contrario que posea un alto costo de movilización

Aplicaciones sobre la firma digital

- Firma y/o cifrado de correo electrónico, tanto interno como externo.
- Firma y/o cifrado de documentos (Pericias, dictámenes, planos, software, políticas, procedimientos, normativas, minutas, y otros).
- Identificación de personas ante sistemas internos en redes locales y abiertas (intranets), sitios Web (sin necesidad de registrar datos) determinación implícita del perfil de usuario.
- Identificación de sistemas ante el usuario (¿Cómo sé que es el sistema que dice ser?).
- Auditoría de transacciones.
- Seguridad al operar comercialmente (Compra-venta de acciones, transacciones bancarias, operaciones con tarjeta de crédito, y otras).
- Identificación de los componentes físicos de una red (Computadores, routers, teléfonos celulares, y otros).

Aplicaciones sobre el documento digital firmado digitalmente

- Fidelización de documentos digitalizados.
- Recibos de pago.
- Certificaciones.
- Cheque electrónico.
- Solicitudes de prestación de servicios
- Adjudicaciones.
- Factura electrónica.
- Invitaciones.

	Manual de prácticas del Laboratorio de Administración de Redes	Código:	MADO-32
		Versión:	02
		Página	143/174
		Sección ISO	8.3
		Fecha de emisión	28 de julio de 2017
Facultad de Ingeniería		Área/Departamento: Laboratorio de Redes y Seguridad	
La impresión de este documento es una copia no controlada			

- Promociones.
- Remitos de entrega.
- Solicitudes de adhesión.
- Actas.
- Planificaciones.
- Reservas o turnos para distintas prestaciones (Talleres, médicos, hoteles, pasajes, etc.).
- Autorizaciones de prestaciones médicas.
- Historia clínica única.
- Confirmaciones.
- Diseños
- Acreditaciones de puntaje.
- Órdenes de compra.
- Contratos.
- Planos.
- Circulares internas y/o externas.
- Cotizaciones de bienes y servicios (tanto el pedido como la cotización y condiciones del proveedor).
- Receta médica electrónica.
- Declaraciones juradas.
- Proyectos.

Es indudable que otras tantas aplicaciones particulares hacen o pueden hacer uso de la tecnología. Sin embargo, detrás de ésta, por las connotaciones que conlleva, la certificación por autoridad competente especializada del adecuado funcionamiento de las aplicaciones, acorde con las normativas en la materia, las transforma en un tema sumamente delicado. A modo de ejemplo, una aplicación que mayoritariamente contempla el uso de certificados digitales y concordantemente la firma digital es la de correo electrónico y si bien su funcionamiento es razonablemente adecuado a esta situación, adolecen de garantías contra defectos que le son imputables, más aún de cara a esta nueva realidad digital/legal.

3. Equipo y material necesario

Equipo del Laboratorio:


- JDK instalado en el sistema operativo Linux.

4. Desarrollo

Modo de trabajar

La práctica se desarrollará en parejas.

La aplicación Java a realizar se compone de dos programas, el primero hará el firmado del documento y el segundo verificará la firma de dicho documento. El programa que realiza el primer procedimiento debe cumplir con los siguientes requerimientos:

	Manual de prácticas del Laboratorio de Administración de Redes	Código:	MADO-32
		Versión:	02
		Página	144/174
		Sección ISO	8.3
		Fecha de emisión	28 de julio de 2017
Facultad de Ingeniería		Área/Departamento: Laboratorio de Redes y Seguridad	
La impresión de este documento es una copia no controlada			

- Solicitar mediante la entrada estándar, el nombre del archivo que contiene la información a firmar (Archivo A).
- La salida del programa que firma el documento, deben ser dos archivos, el primero contendrá la clave pública generada y el segundo la firma del documento (Archivo B, Archivo C).
- El programa que recibe el Archivo A debe generar internamente el par de claves.

El programa que realiza el segundo procedimiento debe cumplir con los siguientes requerimientos:

- Solicitar mediante la entrada estándar, el nombre del archivo que contiene la clave pública generada en el primer punto, además del nombre del archivo que contiene la información firmada (Archivo B, Archivo C).
- Indicar mediante una cadena si la firma es correcta.

4.1 Firma Digital

Una firma digital es un bloque de caracteres que se anexa a un documento con el fin de acreditar quién es su autor, se basa en la criptografía de clave pública, donde quién firma el mensaje lo cifra con su clave privada de forma que puede ser descifrado por todo aquel que posea la clave pública, correspondiente a la clave privada empleada para firmar el mensaje.

NOTA: La firma digital no hace cifrado de mensajes, únicamente garantiza el origen.

1. ¿Cuál es el objetivo de la firma digital?


mecanismo criptográfico que permite al receptor de un mensaje firmado digitalmente identificar a la entidad originadora de dicho mensaje (autenticación de origen y no repudio), y confirmar que el mensaje no ha sido alterado desde que fue firmado por el originador (integridad).

El cifrado de un mensaje con la clave privada consume un tiempo elevado de proceso, por lo que resulta imprescindible contar con un texto más corto que el mensaje original, éste es obtenido mediante una función hash y comúnmente denominado huella digital o fingerprint. Si el mensaje original se altera, también varía la huella digital y lo que se cifra con la clave privada no es el mensaje original, sino la huella digital, el resultado se añade al documento a transmitir.

4.2 Criptografía

4.2.1 Criptografía de clave privada

La criptografía tradicional también denominada de clave secreta se basa en que tanto emisor como receptor, cuentan con una misma clave, que es empleada por el primero para cifrar el mensaje, dando

	Manual de prácticas del Laboratorio de Administración de Redes	Código:	MADO-32
		Versión:	02
		Página	145/174
		Sección ISO	8.3
		Fecha de emisión	28 de julio de 2017
Facultad de Ingeniería		Área/Departamento: Laboratorio de Redes y Seguridad	
La impresión de este documento es una copia no controlada			

lugar a un mensaje ilegible, el segundo utiliza la misma clave de cifrado para descifrar y obtener el mensaje original.

II. Investigue la principal desventaja de la criptografía tradicional.

El mayor inconveniente es la necesidad de comunicar la clave compartida. Esto debe hacerse con mucho cuidado para asegurarse de que la clave no sea revelada a usuarios no autorizados. También puede haber un problema con el número de claves utilizadas. Si la llave es revelada la información esta comprometida

4.2.2 Criptografía de clave pública

En este tipo de cifrado, el emisor y receptor cuentan con claves distintas. La clave pública es aquella que todo el mundo conoce y que puede ser empleada por cualquiera para cifrar mensajes. La clave privada es aquella que únicamente conoce quién envía el mensaje y es capaz de descifrar los mensajes generados por la clave pública.

III. Investigue el algoritmo de cifrado más comúnmente utilizado por este tipo de cifrado.

RSA (Rivest, Shamir y Adleman) es un sistema criptográfico de clave pública desarrollado en 1979, que utiliza factorización de números enteros. Es el primer y más utilizado algoritmo de este tipo y es válido tanto para cifrar como para firmar digitalmente. La seguridad de este algoritmo radica en el problema de la factorización de números enteros. Los mensajes enviados se representan mediante números, y el funcionamiento se basa en el producto, conocido, de dos números primos grandes elegidos al azar y mantenidos en secreto.

Una de las principales ventajas de la criptografía asimétrica es la simplificación de la administración de claves al permitir que varias personas utilicen un par de claves, únicamente existe un problema: ¿cómo distribuir una clave pública de forma que el usuario pueda encontrarla y saber que es válida?

4.3 Firma de un documento


El programa que genera las claves para firmar un documento, hará uso del API de Seguridad del JSDK.

4.3.1 Inicie la máquina virtual e ingrese como usuario redes.

4.3.2 Cree una carpeta en /home/redes con el nombre de **Practica10aINICIALES**, de manera que sea donde almacene los archivos generados.

NOTA: La palabra INICIALES son las iniciales de su nombre y apellidos.

4.3.3 Emplee el comando **nano GeneraFirmaINICIALES.java** y teclee el siguiente código, el cual es la estructura inicial del programa, ver Figura No. 1.

	Manual de prácticas del Laboratorio de Administración de Redes	Código:	MADO-32
		Versión:	02
		Página	146/174
		Sección ISO	8.3
		Fecha de emisión	28 de julio de 2017
Facultad de Ingeniería		Área/Departamento: Laboratorio de Redes y Seguridad	
La impresión de este documento es una copia no controlada			

```

import java.io.*;
import java.security.*;

class GeneraFirmaMADR{

    public static void main (String [] args) {
        /*verificando el numero de argumentos de entrada*/
        if (args.length!=1) {
            System.out.println ("Sintaxis del programa es: java GeneraFirmaMADR archivoaFirmar");
        } else try {
            /*En este bloque se debe colocar el código faltante*/
        } catch (Exception e)
        {
            System.out.println("El error es " + e);
        }
    }
}

```

Figura No. 1. Código de la estructura inicial para generar la firma

```


import java.io.*;
import java.security.*;
class GeneraFirmaINICIALES
{
    public static void main (String [] args)
    {
        /*Verificando el numero de argumentos de entrada*/
        if (args.length!=1)
        {
            System.out.println ("Sintaxis del programa es: java
            GeneraFirmaINICIALES archivoaFirmar");
        }
        else try
        {
/*En este bloque se debe colocar el código faltante*/
        }
        catch (Exception e)
        {
            System.out.println("El error es " + e);
        }
    }
}

```

El programa importa el paquete java.security.*; ya que ahí se encuentran las clases necesarias para generar las claves y firmar un documento. De la misma manera se importa el paquete java.io.*; porque en ese paquete se encuentran las clases necesarias para manipular los archivos de entrada.

NOTA: Inserte las siguientes líneas del código que se encuentran en negritas y cursiva, en el programa debajo de la línea ***/*En este bloque se debe colocar el código faltante*/***

4.3.4 Para generar una firma digital, se requiere una clave privada que inicie el proceso. La clase KeyPairGenerator permite esa funcionalidad mediante el siguiente código:

	Manual de prácticas del Laboratorio de Administración de Redes	Código:	MADO-32
		Versión:	02
		Página	147/174
		Sección ISO	8.3
		Fecha de emisión	28 de julio de 2017
Facultad de Ingeniería		Área/Departamento: Laboratorio de Redes y Seguridad	
La impresión de este documento es una copia no controlada			

KeyPairGenerator genClave=KeyPairGenerator.getInstance("RSA");

4.3.5 Inicie el objeto genClave a través del método initialize, el cual tiene dos argumentos:

- Tamaño de clave, el cual es de 1024 bits.
- La fuente aleatoria, en el caso de la práctica haremos uso de la clase Secure Random.

***SecureRandom aleatorio=SecureRandom.getInstance("SHA1PRNG","SUN");
genClave.initialize(1024, aleatorio);***

El objeto SecureRandom, pretende aleatorizar el estado interno del propio generador utilizando el algoritmo de generación de números pseudoaleatorios llamado SHA1PRNG.

4.3.6 El siguiente paso es la generación del par de claves para almacenarlas en objetos del tipo PrivateKey y PublicKey, mediante el siguiente código:

***KeyPair pardeClaves= genClave.generateKeyPair();
PrivateKey privada=pardeClaves.getPrivate();
PublicKey publica=pardeClaves.getPublic();***

NOTA: Guarde los cambios efectuados en el código

4.3.7 El paso final de esta primera aplicación es el firmado de los datos, ya que se han creado la clave pública y la clave privada. Cree un archivo de nombre datosaFirmar.txt con una nueva ventana de la aplicación de bloc de notas y guarde el archivo en la carpeta creada con el siguiente contenido:


"Pensar en seguridad se vuelve necesario una vez que permitimos que nuestros recursos computacionales entren en contacto con el resto del mundo. Un puerto de comunicación abierto casi siempre está expuesto a ataques externos o a abusos; es necesario tomar medidas de seguridad para evitar el mal uso de los recursos. "

NOTA: Guarde los cambios efectuados en el código.

4.3.8 Una firma digital se crea o bien se verifica usando una instancia de la clase Signature, mediante el siguiente código:

Signature firma=Signature.getInstance ("MD5withRSA");

4.3.9 Antes de que el objeto Signature sea usado para firmar o verificar datos, requiere ser inicializado, el cual requiere de la clave privada.

	Manual de prácticas del Laboratorio de Administración de Redes	Código:	MADO-32
		Versión:	02
		Página	148/174
		Sección ISO	8.3
		Fecha de emisión	28 de julio de 2017
Facultad de Ingeniería		Área/Departamento: Laboratorio de Redes y Seguridad	
La impresión de este documento es una copia no controlada			

firma.initSign(privada);

4.3.10 Antes de firmar es necesario suministrar al objeto “firma” los datos a firmar, los cuales se almacenan dentro de un archivo. Las siguientes líneas abren el archivo, su contenido lo guarda en un buffer y posteriormente se lo hace llegar al objeto “firma”.

```
FileInputStream archivo=new FileInputStream(args[0]);
BufferedInputStream buferEntrada=new BufferedInputStream(archivo);
byte[] buffer=new byte[1024];
int longitud;

while (buferEntrada.available() !=0)
{
    longitud =buferEntrada.read(buffer);
    firma.update(buffer, 0, longitud);
}
buferEntrada.close();
```

4.3.11 Finalmente, una vez que se han suministrado los datos al objeto “firma”, se genera la firma digital de los datos.

byte[] firmaReal=firma.sign();


4.3.12 Generada la firma es necesario enviarla a un archivo de la misma manera que la clave pública para el proceso de verificado de firma digital.

```
/*Guardando los datos firmados en un archivo*/
FileOutputStream archivoFirma=new FileOutputStream("firmaINICIALES.txt");
archivoFirma.write(firmaReal);
archivoFirma.close();

/*Guardando la clave pública en un archivo*/
byte[] clave=publica.getEncoded();
FileOutputStream clavePublica=new FileOutputStream("clavePublicaINICIALES.txt");
clavePublica.write(clave);
clavePublica.close();
```

El método getEncoded obtiene los bytes codificados de la clave pública y luego se envían a un archivo.

NOTA: Guarde los cambios efectuados en el código.

	Manual de prácticas del Laboratorio de Administración de Redes	Código:	MADO-32
		Versión:	02
		Página	149/174
		Sección ISO	8.3
		Fecha de emisión	28 de julio de 2017
Facultad de Ingeniería		Área/Departamento: Laboratorio de Redes y Seguridad	
La impresión de este documento es una copia no controlada			

4.3.13 Debe ubicarse en el directorio `/home/redes/Practica10aINICIALES`, para compilar el programa mediante la instrucción **`javac GeneraFirmaINICALES.java`**, en la línea de comando y observe el resultado.

4.3.14 Ejecute el programa con la instrucción **`java GeneraFirmaINICALES datosaFirmar.txt`** y observe en su directorio de trabajo la creación de los archivos esperados.

4.4 Verificación de la firma de un documento

La verificación de la firma de un documento implica que quien recibe el mensaje, en primer lugar aplica la función hash al documento recibido, descifra la huella digital cifrada y compara ésta con la obtenida al procesar el documento, si son iguales el documento no ha sido alterado y efectivamente ha sido enviado por quien firma.

IV. Indique los argumentos que debe recibir el programa que verifica la firma digital de un documento.

El documento a verificar, la firma digital (o en su defecto la llave pública).

4.4.1 Emplee el comando **`nano VerificaFirmaINICALES.java`** y teclee el siguiente código, el cual es la estructura inicial del programa, ver Figura No. 2.


NOTA: No olvide guardar el archivo `VerificaFirmaINICALES.java` en la misma carpeta `/home/redes/Practica10aINICIALES`.

```
import java.io.*;
import java.security.*;
import java.security.spec.*;

class VerificaFirmaINICALES
{
    public static void main (String [] args)
    {
        /*verificando el numero de argumentos de entrada*/
        if (args.length!=3)
        {
            System.out.println
            ("Sintaxis del programa es: java VerificaFirmaINICALES archivo
            ");
        }
        else
        {
            try
            {
                /*En este bloque se debe colocar el código faltante*/
            }
            catch (Exception e)
            {
                System.err.println ("El error es de " + e.toString() );
            }
        }
    }
}
```

Figura No. 2. Código de la estructura inicial para verificar la firma

```
import java.io.*;
import java.security.*;
```

	Manual de prácticas del Laboratorio de Administración de Redes	Código:	MADO-32
		Versión:	02
		Página	150/174
		Sección ISO	8.3
		Fecha de emisión	28 de julio de 2017
Facultad de Ingeniería		Área/Departamento: Laboratorio de Redes y Seguridad	
La impresión de este documento es una copia no controlada			

```
import java.security.spec.*;

class VerificaFirmaINICIALES
{
    public static void main (String [] args)
    {
        /*Verificando el numero de argumentos de entrada*/
        if (args.length!=3)
        {
            System.out.println
            ("Sintaxis del programa es: java VerificaFirmaINICIALES
            archivoaFirmarINICIALES.txt clavepublicaINICIALES.txt firmaArchivo");
        }
        else try
        {
            /*En este bloque se debe colocar el código faltante*/
        }
        catch (Exception e)
        {
            System.err.println ("El error es de "+ e.toString() );
        }
    }
}
```


El programa importa el paquete java.security.spec, ya que éste contiene la clase X509EncodedKeySpec.

NOTA: Inserte las siguientes líneas del código que se encuentran en **negritas y cursiva** en el programa debajo de la línea **/*En este bloque se debe colocar el código faltante*/**

- 4.4.2** El primer paso consiste en importar los bytes codificados de la clave pública del archivo que lo contiene y convertirlos en un objeto del tipo PublicKey mediante el siguiente código:

```
FileInputStream clavepublica=new FileInputStream (args[0]);
byte[] clave=new byte[clavepublica.available()];
clavepublica.read(clave);
clavepublica.close();
```

El arreglo de bytes clave contiene los bytes codificados de la clave pública.

	Manual de prácticas del Laboratorio de Administración de Redes	Código:	MADO-32
		Versión:	02
		Página	151/174
		Sección ISO	8.3
		Fecha de emisión	28 de julio de 2017
Facultad de Ingeniería		Área/Departamento: Laboratorio de Redes y Seguridad	
La impresión de este documento es una copia no controlada			

- 4.4.3** El siguiente paso consiste en obtener el valor de la clave pública, para lo cual hacemos uso de una clase KeyFactory que proporciona conversión entre claves opacas (del tipo Key) y especificaciones de claves, que son representaciones transparentes del material de la clave. Primero es necesario una especificación de clave mediante el estándar X.509, con el siguiente código:

X509EncodedKeySpec pubKeySpec=new X509EncodedKeySpec(clave);

- 4.4.4** Se requiere de un objeto KeyFactory para realizar la conversión, éste debe trabajar con claves RSA.

KeyFactory keyFactory = KeyFactory.getInstance("RSA");

- 4.4.5** Empleando el objeto keyFactory se genera un objeto PublicKey de la siguiente manera:

PublicKey pubKey=keyFactory.generatePublic(pubKeySpec);

- 4.4.6** El siguiente paso consiste en introducir los bytes firmados desde el archivo especificado en el segundo argumento de la línea de comandos:

***FileInputStream archivoFirmado=new FileInputStream(args[1]);
byte[] firmaVerificada=new byte[archivoFirmado.available()];
archivoFirmado.read(firmaVerificada);
archivoFirmado.close();***

Hasta este punto el arreglo de bytes firmaVerificada contiene los bytes de la firma de documento.

- 4.4.7** Una firma se verifica usando una instancia de la clase Signature, definiendo los algoritmos utilizados en el proceso de la firma del documento:


Signature firma=Signature.getInstance ("MD5withRSA");

- 4.4.8** Inicialice el objeto Signature con el método verificar que recibe como argumentos la clave pública:

firma.initVerify(pubKey);

- 4.4.9** Suministre al objeto firma los datos para los cuales se generó la firma, éstos se encuentran en el archivo original.

***FileInputStream datos=new FileInputStream(args[2]);
BufferedInputStream
bufferEntrada=new BufferedInputStream(datos);***

	Manual de prácticas del Laboratorio de Administración de Redes	Código:	MADO-32
		Versión:	02
		Página	152/174
		Sección ISO	8.3
		Fecha de emisión	28 de julio de 2017
Facultad de Ingeniería		Área/Departamento: Laboratorio de Redes y Seguridad	
La impresión de este documento es una copia no controlada			

```

byte[] buffer=new byte[1024];
int longitud;

while (buferEntrada.available() !=0)
{
    longitud= buferEntrada.read(buffer);
    firma.update(buffer,0,longitud);
}
buferEntrada.close();

```

4.4.10 El proceso de verificación de la firma permite reportar el resultado.

```

boolean verifica=firma.verify(firmaVerificada);
System.out.println ("Verificación de la firma " +verifica);

```

NOTA: Guarde los cambios efectuados en el código.

4.4.11 Compile el programa mediante la instrucción ***javac VerificaFirmaINICIALES.java*** en la línea de comandos.

4.4.12 Ejecute el programa con la instrucción ***java VerificaFirmaINICIALES clavepublicaINICIALES.txt firmaINICIALES.txt datosaFirmar.txt*** y observe el resultado.


4.4.13 Modifique el archivo clavePublica.txt en el bloc de notas, ejecute nuevamente el programa con la instrucción ***java VerificaFirmaINICIALES clavepublicaINICIALES.txt firmaINICIALES.txt datosaFirmar.txt*** y observe el resultado.

V. Investigue el error obtenido en el punto anterior.

Lo que nos indica es que la llave de seguridad es inválida, ya que la longitud es mayor a la esperada; lo cual es el comportamiento que esperamos, pero este error debió ser manejado en el código

NOTA: Ejecute nuevamente la instrucción ***java GeneraFirmaINICIALES datosaFirmar.txt*** para restablecer la clave pública.

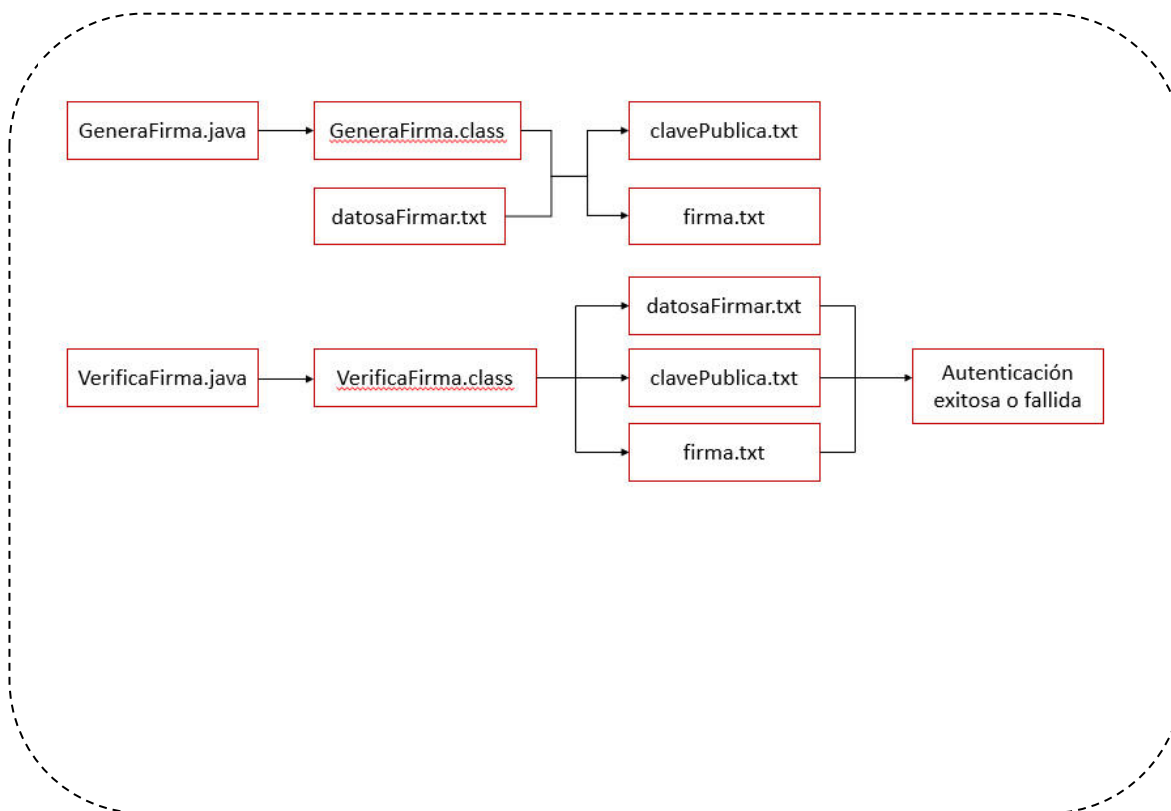
4.4.14 Modifique el archivo datosaFirmar.txt en el bloc de notas, ejecute nuevamente el programa con la instrucción ***java VerificaFirmaINICIALES clavePublicaINICIALES firmaINICIALES datosaFirmar.txt*** y observe el resultado.


	Manual de prácticas del Laboratorio de Administración de Redes	Código:	MADO-32
		Versión:	02
		Página	153/174
		Sección ISO	8.3
		Fecha de emisión	28 de julio de 2017
Facultad de Ingeniería		Área/Departamento: Laboratorio de Redes y Seguridad	
La impresión de este documento es una copia no controlada			

VI. Defina el escenario que representa el punto anterior.

Escenario donde un intruso ha obtenido acceso al archivo original, sin firmar. En el peor de los casos

Elabore un diagrama en el cuál indique el funcionamiento y los elementos de los programas realizados anteriormente.



	Manual de prácticas del Laboratorio de Administración de Redes	Código:	MADO-32
		Versión:	02
		Página	154/174
		Sección ISO	8.3
		Fecha de emisión	28 de julio de 2017
Facultad de Ingeniería		Área/Departamento: Laboratorio de Redes y Seguridad	
La impresión de este documento es una copia no controlada			

5. Conclusiones

Gutierrez Silvestre Griselda:

Con este ejercicio en java se comprobo que la autenticación por clave permite mantener la seguridad en el envio y recibo de archivos con información confidencial.


En este ejemplo se vio como a través de una clave se puede comprobar si eres la persona que es destinada para leer cierto archivo.

Sánchez Bautista Velia:

Los objetivos de la practica se cumplieron, desde el inicio se dio a conocer los conceptos básicos sobre firmas digitales y seguridad. Para esta practica no tuvimos complicaciones, editamos el código proporcionado para que nos diera el mensaje cifrado

Bibliografía:

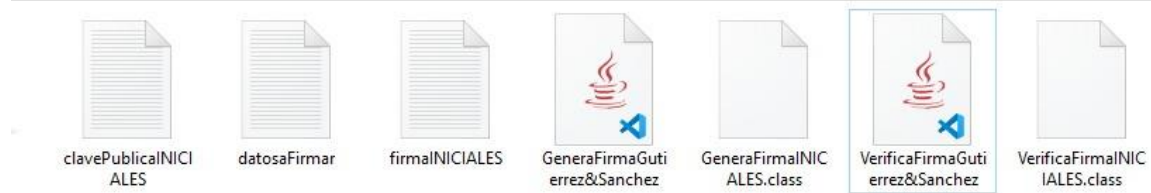
<https://criptografiafirmasdigitales.wordpress.com/2014/09/26/ventajas-y-desventajas-de-la-criptografia/>
<http://www.sedice.com/modules.php?name=Forums&file=viewtopic&t=27492>
<https://es.wikipedia.org/wiki/RSA>
<https://juncotic.com/rsa-como-funciona-este-algoritmo/>


	Manual de prácticas del Laboratorio de Administración de Redes	Código:	MADO-32
		Versión:	02
		Página	155/174
		Sección ISO	8.3
		Fecha de emisión	28 de julio de 2017
Facultad de Ingeniería		Área/Departamento: Laboratorio de Redes y Seguridad	
La impresión de este documento es una copia no controlada			

PRÁCTICA 10
Mecanismos de Seguridad, Firma Digital
Cuestionario Previo


1. ¿Qué son los mecanismos de seguridad?
2. ¿A qué se refieren los mecanismos de seguridad generalizados? y menciona 2 ejemplos.
3. ¿A qué se refieren los mecanismos de seguridad específicos? y menciona 3 ejemplos.
4. Investigue en qué consisten las funciones hash y mencione al menos 3 ejemplos.
5. Investigue en qué consisten los algoritmos de clave pública.
6. Mencione al menos 3 algoritmos de cifrado simétrico o de criptografía tradicional.
7. Investigue el contenido básico del estándar X.509.
8. Investigue el uso de las sentencias try y catch dentro del lenguaje java
9. Investigue el funcionamiento del método getPrivate y getPublic

arios > gray_ > Documents




 Selecciónar Símbolo del sistema

```
C:\Users\gray_\Documents>javac "GeneraFirmaGutierrez&Sanchez.java"
C:\Users\gray_\Documents>java "GeneraFirmaGutierrez&Sanchez.java" datosafirmar.txt
```

 Símbolo del sistema

```
C:\Users\gray_\Documents>java "VerificaFirmaGutierrez&Sanchez.java" clavePublicaINICIALES.txt firmaINICIALES.txt datosafirmar.txt
Verificación de la firma true
C:\Users\gray_\Documents>
```

 Símbolo del sistema

```
C:\Users\gray_\Documents>java "VerificaFirmaGutierrez&Sanchez.java" clavePublicaINICIALES.txt firmaINICIALES.txt datosafirmar.txt
Verificación de la firma true
C:\Users\gray_\Documents>java "VerificaFirmaGutierrez&Sanchez.java" clavePublicaINICIALES.txt firmaINICIALES.txt datosafirmar.txt
El error es de java.security.spec.InvalidKeySpecException: java.security.InvalidKeySpecException: IOException: DerInputStream.getLength(): lengthTag=126, too big.
C:\Users\gray_\Documents>
```

Modificando el archivo

```
C:\Users\gray_\Documents>java "GeneraFirmaGutierrez&Sanchez.java" datosafirmar.txt
C:\Users\gray_\Documents>java "VerificaFirmaGutierrez&Sanchez.java" clavePublicaINICIALES.txt firmaINICIALES.txt datosafirmar.txt
Verificación de la firma false
```