PROGRAMACIÓN DE SERVICIOS Y PROCESOS

TÉCNICO EN DESARROLLO DE APLICACIONES MULTIPLATAFORMA

La criptografía
Clave pública y privadas
Firmas y Certificados digitales
Funciones HASH

Vamos a ver cómo se realiza el proceso de encriptado de la información, además de estudiar los dos modelos más comunes de la criptografía, el modelo de **clave privada** y el modelo de **clave pública**.

Por último, vamos a aprender cómo se puede aplicar la criptografía a los programas que desarrollemos mediante las **firmas digitales** y los **certificados digitales**, mediante funciones HASH.

Concepto de criptografía

La palabra criptografía proviene del griego "**cripto**", que significa **secreto**, y "**grafía**", que significa **escritura**, por lo que la palabra criptografía significa **escritura secreta**.

La criptografía fue creada y está siendo utilizada (entre otros usos) para poder enviar información confidencial o **mensajes privados** a ciertas personas u organizaciones.

Los pasos a seguir para poder aplicar la criptografía a un mensaje son los siguientes:

1. Se escribe el **mensaje 'normal'**, es decir, sin encriptar.

2. Se utilizan **técnicas de encriptado**, más o menos sofisticadas para codifica el mensaje deseado.

3. Se puede **enviar el mensaje** ya encriptado por una línea de **comunicaciones seguras, o no seguras.** Si estamos en el segundo caso, el mensaje enviado puede ser interceptado. Esto no entrañaría ningún peligro inicialmente, ya que estaría encriptado. No obstante, si el método de encriptación llevado a cabo es conocido o descifrado por el ente que lo intercepta, la información quedaría al descubierto.

4. Cuando el **receptor reciba** el mensaje, aplicará la **técnica de desencriptado** para poder ver el mensaje original.

Junto a la criptografía podemos destacar el criptoanálisis, que es una ciencia que se dedica a estudiar la **fortaleza o robustez que tienen los sistemas criptográficos**, pudiendo comprobar cómo de seguros son en realidad.

Mediante el criptoanálisis se están mejorando día a día los sistemas criptográficos.

Existen diferentes tipos de criptografía:

- Criptografía simétrica.
- Criptografía de clave pública o criptografía **asimétrica**.
- Criptografía con umbral.
- Criptografía basada en identidad.

- Criptografía basada en certificados.
- Criptografía sin certificados.
- Criptografía de clave aislada.

Los tipos de criptografía más utilizados en un **entorno profesional**, son los de **criptografía simétrica** y **asimétrica**, que son los que vamos a estudiar más adelante en esta unidad.

Tecnologías de la seguridad de la información

Algunas de las principales tecnologías referentes a la seguridad de la información en informática son:

- **Cortafuegos**,
- Administración de **cuentas de usuarios**,
- **Detección y prevención** de intrusos,
- **Antivirus**,
- Infraestructura de **llave publica**,
- **Capas de Socket Segura** (**SSL**),
- **Conexión única** "Single Sign On - **SSO**",
- **Biométrica**,
- **Cifrado**.

Aplicaciones de la criptografía

Una de las aplicaciones más emergentes de la criptografía, y sobre la que más se puede estar innovando en la actualidad, es el concepto de la **cadena de bloques** blockchain, ya que éste, utiliza **diferentes tipos de criptografía** para garantizar la **seguridad de las transacciones**.

En primer lugar, se utiliza el tipo de criptografía HASH, mediante la cual, se pueden **convertir grandes cantidades de información** en una combinación de **letras y números** única, y muy difícil de imitar. Con esto queremos decir que básicamente se van a resumir enormes cantidades de información, pudiéndose comprobar rápida y fácilmente, que **todos los procesos realizados por los nodos de la blockchain coincidan**. Por otra parte, el usar un código HASH nos va a permitir la creación de las **claves públicas y privadas**, con las que se **reciben y envían criptomonedas**.

Dentro de los datos de la **cadena blockchain**, son utilizadas **diferentes capas** de criptografía, que solamente pueden ser resueltos por **ordenadores** de una **potencia considerable**.

Ya en otro ámbito, concretamente en la cotidianeidad de nuestro día a día, uno de los usos más comunes que tiene la criptografía está en Internet.

Cuando accedemos a un sitio web denominado como HTTPS (fácilmente visible en el apartado de la URL de nuestro navegador favorito), éste utiliza el **protocolo** de seguridad denominado **SSL** (que estudiaremos en la siguiente unidad), por sus siglas en inglés, **Secure Sockets Layer**. Este protocolo se encarga de **cifrar todos los datos** que el usuario pueda enviar al servidor utilizando **diferentes algoritmos criptográficos**.

Un último ejemplo del uso de la criptografía está en el uso de cualquier **sistema financiero** virtual, como puede ser **PayPal**.

Así que recuerda, cada vez que accedas a determinadas páginas webs, realices una compra online..., estás haciendo uso de todo el potencial que nos ofrece la criptografía.

La mejor encriptación

En cuanto a la elección del mejor encriptado, habrá que prestar especial atención a los aspectos de:

- Longitud de la cadena: La cadena encriptada **más larga** será la **más complicada de revertir** en caso de ataque.
- Tipo de caracteres usados: Una encriptación será más segura si usa **diferentes tipos de caracteres**, como pueden ser letras minúsculas y mayúsculas, números y símbolos especiales.
- El tiempo de encriptado: El tiempo que se tarda en encriptar la información también es muy importante, ya que podemos tener una información muy segura, pero **tardar en encriptarse** mucho tiempo, lo cual **no es aconsejable**.

Encriptación de la información

Podemos definir formalmente la encriptación o el cifrado de la información como el proceso por el que la información o los **datos** que se desean proteger son **codificados**, dando lugar a un **texto** que parece ser **aleatorio**, o sin sentido para los humanos.

Igualmente, podemos definir formalmente la desencriptación como la operación inversa a la encriptación, mediante la cual, los datos encriptados se transforman mediante las **técnicas inversas** del algoritmo utilizado, para **encriptarlos en el texto original**.

Es importante conocer los siguientes conceptos básicos para poder hablar de criptografía y encriptación:

- Texto plano:

  Se refiere al texto **original**, sin aplicar ningún algoritmo de encriptación.

- Texto cifrado:

  Se refiere al texto resultado de **aplicar el algoritmo** de encriptación al texto original.

- Algoritmo de cifrado o algoritmo criptográfico:

  Es el algoritmo que utilizaremos para poder **encriptar o cifrar** el texto plano para dar lugar al texto cifrado. **Junto al algoritmo** de cifrado **existe una clave**.

- Clave:

  Cadena de caracteres que serán la **base para el algoritmo** de cifrado, y que permitirán pasar del texto plano al cifrado. Cada clave diferente proporcionará como salida un **texto cifrado diferente**. Esta clave puede ser **simétrica o asimétrica**.

Como resumen podemos decir que el proceso de encriptado de la información se puede representar mediante la siguiente fórmula:

Cifrado → F_K (M) = C

Donde F es el algoritmo que vamos a utilizar para cifrar la información, K será la clave de cifrado, M será el mensaje que queremos cifrar y C será el texto ya cifrado.

Dicho de otra manera:

Para conseguir un cifrado de mensaje: aplicamos un algoritmo de cifrado usando la clave de cifrado sobre el mensaje a cifrar, lo que dará lugar al texto ya cifrado.

Principios de la criptografía

Vamos a ver cuáles son los principios de la criptografía.

Las propiedades que son más deseables en un sistema criptográfico fueron anunciadas por August Kirchhoff en el año 1883. De entre ellas podemos destacar las siguientes:

1. **Si el sistema no es teóricamente irrompible, en la práctica al menos sí debe serlo**.

    Esto se debe a que ningún sistema es 100% seguro y tarde o temprano se terminará rompiendo aunque sea teóricamente irrompible.

2. La **efectividad** que tiene el sistema **no debe depender** de que el **diseño** del mismo sea **secreto**.

3. La clave que vamos a usar tanto para encriptar como para desencriptar los mensajes debe ser **fácilmente memorizable**.

    De esta manera evitamos tener que escribirlas y que puedan ser sustraídas.

4. Los sistemas criptográficos deben dar resultados **alfanuméricos**.

    De esta forma los sistemas serán mucho más seguros y las claves más difíciles de romper.

5. El sistema de criptografía debe ser **operado únicamente por una única** persona para que así sea todo más seguro.

6. El **sistema** debe ser **fácil de utilizar** para que de esta forma la persona que se encarga del sistema no tenga problemas.

Según un test:

El diseño del sistema puede ser público.

Criptografía de clave privada o simétrica

La criptografía de clave simétrica o privada es un método de encriptado que utiliza una **clave** que es secreta, la cual **solo pueden conocer el emisor y el receptor**. Este tipo de criptografía es m**uy apropiada** si queremos garantizar la **confidencialidad**.

A este tipo de criptografía se la denomina simétrica porque la **clave de encriptado y desencriptado** es exactamente la **misma**.

Podemos decir que las principales características de la criptografía simétrica son:

1. La clave es secreta, debiendo conocerla solo las partes involucradas en la comunicación, es decir, el **emisor y el receptor**.
2. Se utiliza la misma clave para cifrar y para descifrar los mensajes de la comunicación.
3. Estos algoritmos de cifrado suelen ser muy rápidos y **no** suelen **aumentar el tamaño** del mensaje, es debido a esto que son muy apropiados para **encriptar grandes cantidades** de texto.

La criptografía de clave privada también tiene una serie de inconvenientes, pudiendo destacar los siguientes:

• Como la clave de cifrado y descifrado es la misma, en el momento del envío de ésta por parte del emisor al receptor, este mensaje puede ser **interceptado** y una persona no deseada puede **hacerse con ella**.

• Como las claves se utilizan en una única comunicación, si se desean comunicar varias personas, deberá haber **una clave para cada combinación** de personas diferentes que vayan a comunicarse, generando así una enorme cantidad de claves.

Una alternativa para **solucionar los problemas de distribución de claves** y todo lo que se deriva de ello, pueden ser la **criptografía asimétrica** y la **criptografía híbrida** que estudiaremos a continuación.

Un ejemplo práctico donde se utilizaba este tipo de criptografía fue la **máquina Enigma**, utilizada por la **Alemania Nazi** en la segunda guerra mundial para **cifrar sus comunicaciones**.

Máquina enigma

Criptografía de clave pública o asimétrica

La criptografía de clave pública surgió para solucionar el problema de distribución de claves que sufría la criptografía de clave privada, permitiendo así tanto al **emisor** como al **receptor** poder **poner en común unas claves mediante un canal** (incluso no seguro) de comunicación.

A este tipo de criptografía se le denomina asimétrica porque las **claves** de encriptado y desencriptado **son diferentes**, a diferencia de la criptografía de clave privada.

Podemos listar las características más importantes de la criptografía asimétrica:

1. Tanto emisor como receptor tienen en su poder un par de claves, una que es **pública**, que es conocida por todo el mundo y que el hecho de conocerla no implica conocer ningún tipo de información sobre la clave privada inversa, y otra que es **privada**, que la **conoce únicamente su poseedor**.
2. Todas las parejas de claves sirven únicamente con ellas mismas, es decir, que son complementarias, y el proceso **no funcionará** si alguna de ellas es **cambiada**.
3. Las claves de encriptado y desencriptado únicamente se pueden generar una vez, de esta forma, es **prácticamente imposible** que dos personas obtengan las mismas claves.
4. Cuando un mensaje es cifrado con la clave pública, únicamente se va a poder descifrar con la clave privada que sea la inversa a esa clave pública.
5. Cuando ciframos un mensaje con la clave privada, estamos demostrando que nosotros hemos sido quienes hemos cifrado dicho mensaje.

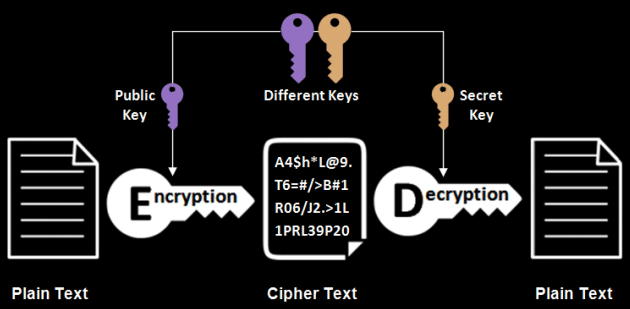
La gran ventaja que ofrece este tipo de criptografía es que **ya no** existe el problema de la **distribución de claves**.

La criptografía de clave pública también tiene algunos inconvenientes:

- Estos algoritmos son algo más lentos.
- Hay que poder garantizar que la clave pública es realmente **de quien dice ser**.

Lo más óptimo sería utilizar una **combinación** de criptografía de clave pública y de clave privada, lo que se conoce como **criptografía híbrida**.

Criptografía asimétrica usada en Bitcoin



Accesos de usuario seguros

Una de las cosas más comunes que deben hacer los programadores de aplicaciones son los accesos de los usuarios a las mismas.

Estos accesos nunca deben implementarse en **texto plano**, ya que con un simple **analizador de tráfico de red** podríamos obtener las **claves de acceso** de una forma extremadamente sencilla.

La idea es que las **claves de acceso** estén **encriptadas en la base de datos** de usuarios de la aplicación, realizando así **dos tareas de forma simultánea**:

1. La primera sería que, al estar esas claves **encriptadas** en la base de datos, si alguien consigue entrar a ella **no podrá obtenerlas**.
2. La segunda es que estamos dotando de una **alta seguridad a los accesos** de los usuarios.

Para realizar estos accesos podríamos seguir los siguientes **pasos**:

- **Obtener la clave** que introduce el usuario.
- **Encriptarla**.
- Una vez encriptada, enviarla por **petición segura HTTPS** para comprobar si el **login en correcto**.

Esquema de acceso seguro

```
// Obtenemos las claves
usuario = obtenerUsuario()
Clave = obtenerClave()

// Encriptamos la clave
Clavesegura = encriptar(clave)

// Comprobamos si el acceso es OK
Si comprobarAcceso(usuario, clavesegura)
        // entramos al sistema
sino
        // mostramos un error
```

Firma digital y certificados digitales

Las firmas digitales son el equivalente a las firmas personales, pero en un entorno tecnológico, es decir, su objetivo es identificar al firmante inequívocamente, pero en lugar de hacerlo en papel, se llevaría a cabo de forma digital.

Las firmas digitales están basadas en:

- criptografía de **clave pública**,
- resumen de **mensajes HASH**.

Un **resumen de mensajes** (**Message-Digest** Algorithm, en inglés) es un algoritmo que para encriptar, toma como entrada un **mensaje con una longitud variable** y lo convierte en un **resumen de una longitud fija**. Algunos algoritmos de este tipo son el **MD5** y el **SHA**.

Otro elemento más que interviene en el proceso de criptografía son los certificados digitales. Éstos se diseñaron para resolver el problema de la confianza que han de depositar las dos partes involucradas en la comunicación, es decir, un certificado digital es un documento electrónico **firmado por un tercero** (entidad certificadora) que da fe de los datos de la firma digital empleada. De forma genérica, podemos decir que vienen a ser como el **notario de la firma digital**.

Según un test:

El certificado digital… Esto garantiza que la persona que ha enviado el mensaje es quien dice ser.

Una entidad certificadora es una organización que **se responsabiliza** de la veracidad de los datos de los firmantes digitales, y, por tanto, de la **emisión y validez** de los certificados oportunos. Creándolos y aportando mecanismos que permitan poder **revocarlos, suspenderlos, y comprobar su validez**.

Extrapolando todos estos conceptos a un entorno de programación, podemos destacar que en **Java** se usa la clase **MessageDigest** y los **algoritmos** que podemos emplear son:

1. **MD2**,
2. **MD5**,
3. **SHA-1**,
4. **SHA-256**,
5. **SHA-384**
6. y **SHA-512**.

Cada vez que vayamos a encriptar un texto deberemos controlar la **excepción** NoSuchAlgorithmException mediante un bloque try-catch.

En el método **getInstance** es donde podremos **indicar cualquiera** de los **algoritmos** listados anteriormente para cifrar el mensaje con una **función HASH**.

Código para encriptar texto con HASH

```
try {
	String password = "esta_es_mi_contraseña_1234";
	MessageDigest md = MessageDigest.getInstance("SHA-256");
	md.update(password.getBytes());
	byte byteData[] = md.digest();
	StringBuilder sb = new StringBuilder();
	for (int i = 0; i < byteData.length; i++) {
		sb.append(Integer.toString((
				byteData[i] & 0xff) + 0x100, 16).substring(1));
	}
	System.out.println("Contraseña -> " + password);
	System.out.println("MD5 -> " + sb.toString());
} catch (NoSuchAlgorithmException error) {
	System.out.println("Error: " + error.toString());
}
```

Ejemplo de funciones HASH

Vamos a ver un ejemplo de codificación con función hash en Java.
En primer lugar, lo primero que vamos a hacer es introducir la contraseña que queremos cifrar y mostrarla.

Ahora vamos a cifrar en primer lugar con el algoritmo hash SHA-256. Para cifrar con cualquier algoritmo hash necesitamos la clase MessageDigest. Con el método getInstance vamos a indicar el tipo de algoritmo con el que queremos cifrar el texto.

Una vez lo hemos indicado, con el método update codificamos el texto y el resultado será con la función digest y vendrá dado en un **array de tipo byte**.
Ahora, mediante un StringBuilder vamos a ir traduciendo el texto cifrado, ya que viene dado en bytes, y de esta forma lo vamos a pasar a string y lo mostramos.
Vamos a repetir el mismo proceso con el algoritmo SHA-512.
Y, por último, haremos el mismo proceso con el algoritmo MD5, mostrando el resultado de los tres algoritmos para comprobar cuál es más seguro. Deberemos realizar todo esto dentro de un **bloque try-catch** para controlar esta excepción.

Ejecutamos el programa para ver el resultado:

- Introducimos una contraseña, por ejemplo, "perro", y aquí tenemos el resultado con el algoritmo SHA-256.
- Con el 512 es más seguro ya que es más largo y con el MD5.
- Si volvemos a ejecutar e introducimos otra contraseña, por ejemplo, "examen", tenemos los resultados de codificar "examen" con los tres algoritmos.

```
Contraseña -> examen
SHA-256 -> 854a557fb0868de7fe6e432766f141f446d3f34f5d9c7e50e0dac94c817d32f9
SHA-512 ->
be30e8557b748fd8d69a11685d36661c5c067ad1331747c4d24c1cd5d621211ace7e2d4f732c58
c9fe57e09768e3cfbf5ca5c9dcc7b3203101f61b433d94ddfb
MD5 -> 32bd3b82800c20c82f979e3cf1b26917
```

Código

```
try {
    Scanner teclado = new Scanner(System.in);
    System.out.println(
            "Introduce la contraseña a cifrar:");
    String password = teclado.nextLine();

    System.out.println("Contraseña → " + password);

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

    md.update(password.getBytes());
    byte byteData[] = md.digest();

    StringBuilder sb = new StringBuilder();
    for (int i = 0; i < byteData.length; i++) {
        sb.append(Integer.toString((
                byteData[i] & 0xff) +
                0x100, 16).substring(1));
    }

    System.out.println("SHA-256 → " + sb.toString());

    // Ciframos con SHA-256
    MessageDigest md2 = MessageDigest.getInstance(
            "SHA-512");

    md2.update(password.getBytes());
    byte byteData2[] = md2.digest();

    StringBuilder sb2 = new StringBuilder();
    for (int i = 0; i < byteData2.length; i++) {
        sb2.append(Integer.toString((
                byteData2[i] & 0xff) +
                0x100, 16).substring(1));
    }

    System.out.println("SHA-512 → " + sb2.toString());

    // Ciframos con MD5
    MessageDigest md3 = MessageDigest.getInstance(
            "MD5");
```

```
    md3.update(password.getBytes());
    byte byteData3[] = md3.digest();

    StringBuilder sb3 = new StringBuilder();
    for (int i = 0; i < byteData3.length; i++) {
        sb3.append(Integer.toString((
                byteData3[i] & 0xff) +
                0x100, 16).substring(1));
    }

    System.out.println("MD5 -> " + sb3.toString());
} catch (NoSuchAlgorithmException error) {
    System.out.println("Error: " + error.toString());
}
```