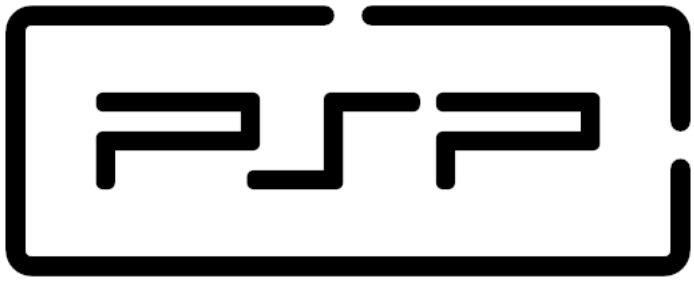

Programación de Servicios y Procesos

6.2 Funciones resumen

6.2 Funciones resumen

6.2.1. Funciones hash

Un *Message digest* o resumen de mensaje, más conocidos como **funciones hash**, es una marca digital de un bloque de datos. Existe un gran número de algoritmos diseñados para procesar esto resúmenes, los dos más conocidos son SHA-1 y MD5.

De un resumen cabe destacar las siguientes características:

- Para el mismo algoritmo, el resumen siempre tiene el mismo tamaño, independientemente del tamaño de los datos que se haya usado para generarlo.
- Es imposible recuperar la información original a partir de un resumen.
- El resumen no debe desvelar nada sobre los datos que se utilizaron para generarlo.
- Es computacionalmente inviable encontrar dos mensajes que tengan el mismo valor de resumen. Matemáticamente es altamente improbable, pero no imposible.
- Un pequeño cambio en los datos resumidos genera un resumen completamente diferente.

Los resúmenes se usan para generar identificadores únicos y confiables. A veces se les llama *checksum*, ya que sirven para comprobar si una descarga se ha realizado correctamente, generando su resumen y comparándolo con el que generó el archivo original.

> **Un hash no sirve para cifrar**
>
> Es importante destacar que, debido a que es imposible obtener los datos que generaron un resumen a partir del propio resumen, el resumen no se puede usar para cifrar información.
>
> Por el contrario, es un mecanismo que se usa para comparar. Su uso más extendido es con las contraseñas, ya que en las bases de datos se guarda un resumen en vez de la contraseña en claro. De esta forma, cuando se recibe una contraseña se genera su resumen y se compara con el valor almacenado.

6.2.2. MessageDigest

La clase *MessageDigest* permite a las aplicaciones implementar algoritmos de resumen criptográficamente seguros como SHA-256 o SHA-512

Para generar un hash con JCA se procede de la siguiente forma:

1. Se crea un objeto de la clase *MesageDigest* con el método estático *getInstance()* de la misma clase, especificando el nombre del algoritmo. Opcionalmente, se puede especificar el nombre del proveedor.
2. Se añaden datos con el método *update()*. Se puede añadir un byte o un array de bytes. Este método se puede invocar varias veces para ir añadiendo nuevos datos.
3. Se obtiene el valor de hash con el método *digest()*.
4. Si se quisiera calcular un nuevo hash, se invocaría el método *reset()* para volver a empezar el proceso.

A continuación podemos ver un ejemplo

```java
public class U6S2_MessageDigest {

    public static void main(String[] args) {
        String plaintext = "Esto es un texto plano.";
        try {
            // Obtenemos un ENGINE que implementa el algoritmo especificado
            // Se puede indicar cualquier algoritmo disponible en el sistema
            // SHA-224, SHA-512, SHA-256, SHA3-224, ...
            MessageDigest m = MessageDigest.getInstance("SHA-256");

            // Opcional - Reinicia el objeto para un nuevo uso
            // Por si queremos poner este código en un bucle y procesar más
            // de un mensaje
            m.reset();

            // Realiza el resumen de los datos pasados por parámetro
            // Si queremos procesar la información poco a poco,
            // debemos ir llamando al método update para cada bloque de datos
            m.update(plaintext.getBytes());

            // Completa el cálculo del valor del hash y devuelve el resumen
            byte[] digest = m.digest();

            // Mensaje de resumen
            System.out.println("Resumen (raw data): " + new String(digest));

            // Mensaje en formato hexadecimal
            System.out.println("Resumen (hex data): " + toHexadecimal(digest));


            // Información del proceso
            System.out.println("=> Algoritmo: " + m.getAlgorithm() + ", Provider: " + m.getProvider().getName() + " " +
        } catch (NoSuchAlgorithmException e) {
            System.err.println("No se ha encontrado la implementación del algoritmo MD5 en ningún Provider");
        }
    }

    static String toHexadecimal(byte[] hash) {
        String hex = "";
        for (int i = 0; i < hash.length; i++) {
            String h = Integer.toHexString(hash[i] & 0xFF);
            if (h.length() == 1) {
                hex += "0";
            }
            hex += h;
        }
        return hex.toUpperCase();
    }
}
```

y esta sería la salida proporcionada

```sh
Resumen (raw data): ◆Y◆"◆3□□□◆◆`b◆◆◆◆bs?;□◆◆◆□◆◆◆~E
Resumen (hex data): FB59D31122913314111B92CD60628ED7E7DE62733F3B10DEDAF303AAABE57E45
=> Algoritmo: SHA-256, Provider: SUN 11
```

6.2.3. MessageDigest con GnuPG

Con la suite GnuPG podemos generar resúmenes de archivos utilizando los algoritmos que nos proporciona la suite.

Algoritmos disponibles para GnuPG

Para ver la lista de algoritmos disponibles tenemos que mostrar la ayuda del comando

> gpg --help

y en la parte superior observamos la información de los algoritmos disponibles para cada tipo de servicio. En concreto, de resúmenes, en mi versión instalada:

Resumen: SHA1, RIPEMD160, SHA256, SHA384, SHA512, SHA224

Para generar un resumen de un archivo, ejecutamos el comando de la siguiente forma

```sh
gpg --print-md SHA256 filename.ext
```