Proyecto FPMADdigital

Recursos digitales y multimedia para Formación Profesional

Comunidad de Madrid

Dirección General de Educación Secundaria, Formación Profesional y Régimen Especial

CONSEJERÍA DE EDUCACIÓN, UNIVERSIDADES, CIENCIA Y PORTAVOCÍA

Unión Europea

Fondo Social Europeo

"El FSE invierte en tu futuro"

Financiado como parte de la respuesta de la Unión a la pandemia de COVID-19

FPMADdigital
Recursos digitales y multimedia para Formación Profesional

CFGS Desarrollo de Aplicaciones Multiplataforma

módulo profesional

0490 - Programación de Servicios y Procesos

unidad didáctica

05 Técnicas de programación segura

resultados de aprendizaje

05 Protege las aplicaciones y los datos definiendo y aplicando criterios de seguridad en el acceso, almacenamiento y transmisión de la información

Comunidad de Madrid
Dirección General de Educación Secundaria, Formación Profesional y Régimen Especial
CONSEJERÍA DE EDUCACIÓN, UNIVERSIDADES, CIENCIA Y PORTAVOCÍA

Unión Europea
Fondo Social Europeo
"El FSE invierte en tu futuro"
Financiado como parte de la respuesta de la Unión a la pandemia de COVID-19

Resultados de aprendizaje y unidades didácticas

Resultados de aprendizaje y unidades didácticas

RESULTADOS DE APRENDIZAJE					UNIDAD DIDÁCTICA
1	2	3	4	5	
X					1.- Programación multiproceso
	X				2.- Programación concurrente y asíncrona
		X			3.- Programación de comunicaciones en red
			X		4.- Generación de servicios en red
				X	5.- Técnicas de programación segura

Unidades didácticas y materiales asociados

Unidades didácticas y materiales multimedia

RRAA					UU.DD	Material Multimedia
1	2	3	4	5		
X					1.- Programación multiproceso	1.1 Contenidos básicos 1.2 Ejemplos aplicados
	X				2.- Programación concurrente y asíncrona	2.1 Contenidos básicos 2.2 Ejemplos aplicados
		X			3.- Programación de comunicaciones en red	3.1 Contenidos básicos 3.2 Ejemplos aplicados
			X		4.- Generación de servicios en red	4.1 Contenidos básicos 4.2 Ejemplos aplicados
				X	5.- Técnicas de programación segura	5.1 Contenidos básicos 5.2 Ejemplos aplicados

Repositorios de materiales y prácticas

Repositorio de materiales y prácticas

Todos los proyectos mostrados, así como otros materiales utilizados en las unidades didácticas los podrás encontrar completos en:

https://github.com/joseluisgs/FP-NextGen-ProgramacionServiciosProcesos

Cualquier error o propuestas de mejora se publicarán en el repositorio indicado.
Gracias por tu colaboración.

Contenidos

1. Seguridad en las comunicaciones
2. Sistemas criptográficos
3. SSL/TSL
4. Secure Sockets
5. JWT

Seguridad en las comunicaciones

Seguridad en las comunicaciones

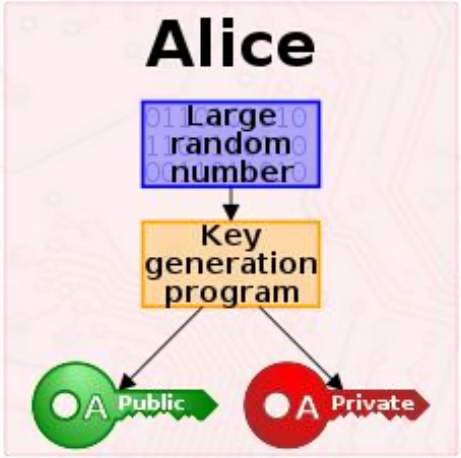
A la hora de asegurar nuestras comunicaciones, nuestro objetivo es alcanzar:

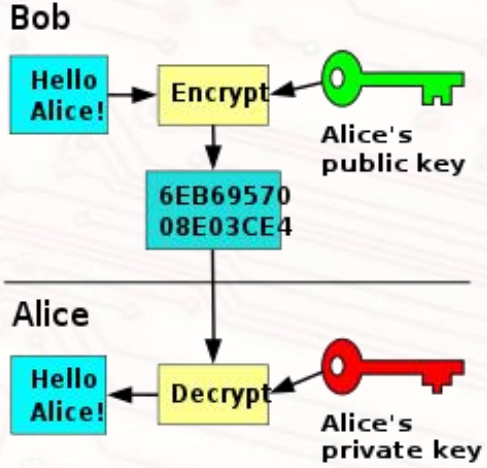
- **Confidencialidad**: garantiza que la información sea accesible únicamente a personal autorizado.
- **Integridad**: garantiza la corrección y completitud de la información.
- **Vinculación**: permite vincular un documento o transacción a una persona o un sistema de gestión criptográfico automatizado.
- **Autenticación**: proporciona mecanismos que permiten verificar la identidad del comunicador.

Sistemas criptográficos

Sistemas criptográficos

- **De una sola vía**: Una vez cifrados no se pueden descifrar. Se usa para comprobar la integridad del fichero o como parte de la firma digital. Ejemplo **BCrypt**, MD5 o **SHA**.
- **Simétricos**: Se usa la misma clave para cifrar y descifrar. Una sola clave, por ejemplo **AES**.
- **Asimétricos**: Se usa una clave para cifrar y otra clave para descifrar, por lo tanto hay que tener **un par de claves (pública y privada)**. Lo que cifra una, lo descifra la otra. Por ejemplo **RSA**





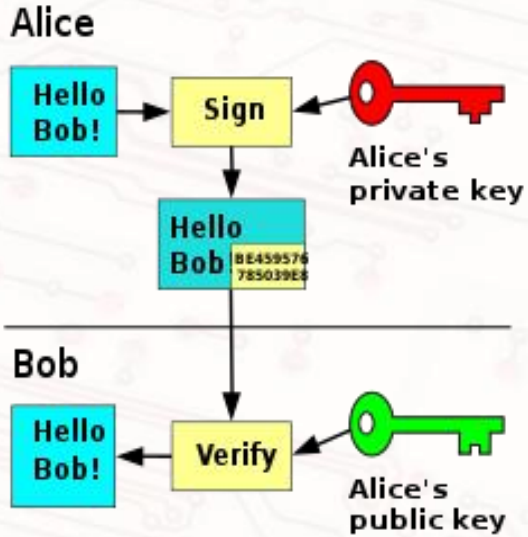
Wikipedia

Seguridad en las comunicaciones

Firma Digital: Se puede decir que es una mezcla de cifrado asimétrico con funciones hash con el objetivo de:

- Comprobar la **integridad** de la información (que no haya cambiado).
- Comprobar la **identidad** del emisor.

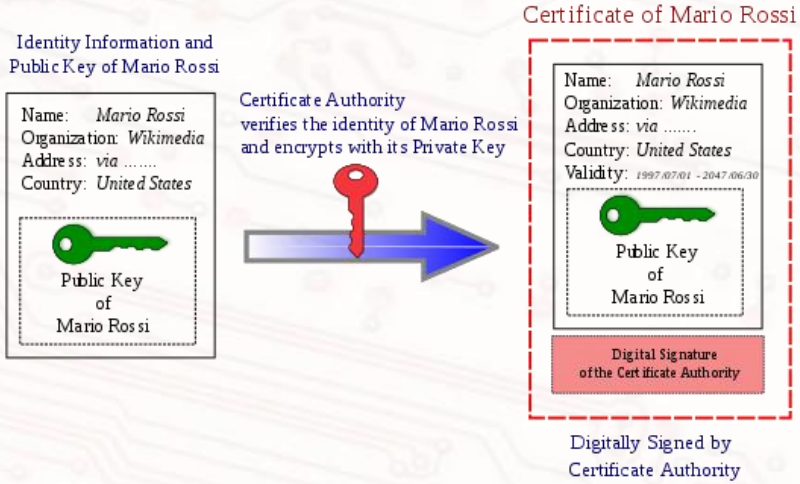
El **emisor**, sobre el **fichero a firmar** se crea un **resumen hash**. Este resumen **se cifra con nuestra clave privada** y se adjunta junto al fichero original. El **receptor** recibe el mensaje y **descifra con la clave pública del emisor (si se puede ya sabemos que es él quien lo envía**) posteriormente **calcula el resumen hash** del mensaje, **si ambos resúmenes coinciden** (el que ha descifrado y el que ha calculado) es que el **mensaje no se ha modificado** hasta la entrega.



Wikipedia

Sistemas criptográficos

Un **certificado digital** o certificado electrónico es un fichero informático firmado electrónicamente por un prestador de servicios de certificación, que vincula unos datos de verificación de firma a un firmante, de forma que únicamente puede firmar este firmante, y confirma su identidad.
Tiene una estructura de datos que contiene información sobre la entidad (por ejemplo una clave pública, una identidad o un conjunto de privilegios).
La firma de la estructura de datos del certificado agrupa la información que contiene de forma que no puede ser modificada sin que esta modificación sea detectada.

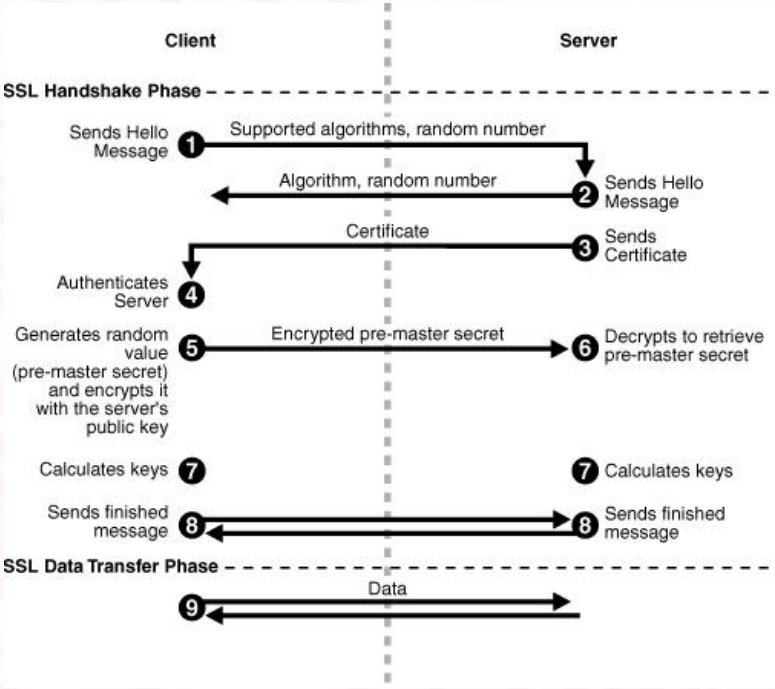


Wikipedia

SSL/TSL

SSL/TSL

- Las claves de sesión se usan para intercambiar datos a través de un medio, por ejemplo socket y servicios webs (es decir, por medios de comunicación compartidos). Combina cifrado asimétrico y simétrico: **Handshake**.
- El cliente envía un mensaje ClientHello especificando una lista de conjunto de cifrados.
- Recibe un registro ServerHello, en el que el servidor elige los parámetros de conexión a partir de las opciones ofertadas con anterioridad por el cliente.
- Cuando los parámetros de la conexión son conocidos, cliente y servidor intercambian certificados y negocian una clave secreta (simétrica) que intercambian cifrando dicha clave, con la clave pública del par contrario, para que este pueda descifrarla con su clave secreta.
- Desde ese momento la comunicación queda protegida.



Oracle Java Doc

Secure Sockets

Secure Sockets

```
/ServidorSeguro/cert/> keytool -genkey -alias claveSSL -keyalg RSA -keystore AlmacenSSL.jks -storepass
1234567
¿Cuáles son su nombre y su apellido?
  [Unknown]:  José Luis González
¿Cuál es el nombre de su unidad de organización?
  [Unknown]:  PSP
¿Cuál es el nombre de su organización?
  [Unknown]:  IES Luis Vives
¿Cuál es el nombre de su ciudad o localidad?
  [Unknown]:  Leganés
¿Cuál es el nombre de su estado o provincia?
  [Unknown]:  Madrid
¿Cuál es el código de país de dos letras de la unidad?
  [Unknown]:  ES
¿Es correcto CN=José Luis González, OU=PSP, O=IES Luis Vives, L=Leganés, ST=Madrid, C=ES?
  [no]:  si

Generando par de claves RSA de 2.048 bits para certificado autofirmado (SHA256withRSA) con una validez de 90
días
        para: CN=José Luis González, OU=PSP, O=IES Luis Vives, L=Leganés, ST=Madrid, C=ES
```

```
/ServidorSeguro/cert/> keytool -export -alias claveSSL -keystore AlmacenSSL.jks -storepass 1234567 -file
CertificadoSSL.cer
Certificado almacenado en el archivo <CertificadoSSL.cer>
```

Secure Sockets

```
/ClienteSeguro/cert/> keytool -import -alias claveSSL -file CertificadoSSL.cer -keystore UsuarioAlmacenSSL.jks -storepass
0987654
Propietario: CN=José Luis González, OU=PSP, O=IES Luis Vives, L=Leganés, ST=Madrid, C=ES
Emisor: CN=José Luis González, OU=PSP, O=IES Luis Vives, L=Leganés, ST=Madrid, C=ES
Número de serie: 61802abdab152e6c
Válido desde: Mon Jun 06 20:47:04 CEST 2022 hasta: Sun Sep 04 20:47:04 CEST 2022
Huellas digitales del certificado:
         SHA1: EE:9C:05:38:81:99:74:57:36:74:F5:8C:71:81:84:86:00:8E:EF:22
         SHA256: A8:FD:63:49:9D:91:08:25:15:0F:83:9D:F7:5E:22:1A:13:17:FC:0F:0A:3D:04:1C:BD:D3:4E:F3:64:B1:6B:BE
Nombre del algoritmo de firma: SHA256withRSA
Algoritmo de clave pública de asunto: Clave RSA de 2048 bits
Versión: 3

Extensiones:

#1: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
KeyIdentifier [
0000: F7 02 65 7E F3 20 BA 17   49 2C 0D 90 7E 2A CE F2  ..e.. ..I,...*..
0010: CC 0D BB ED                                        ....
]
]

¿Confiar en este certificado? [no]:  si
Se ha agregado el certificado al almacén de claves
```

Secure Sockets

Servidor

```
SSLServerSocketFactory serverFactory = (SSLServerSocketFactory)
SSLServerSocketFactory.getDefault();
ServerSocket serverSocket = serverFactory.createServerSocket(port);
```

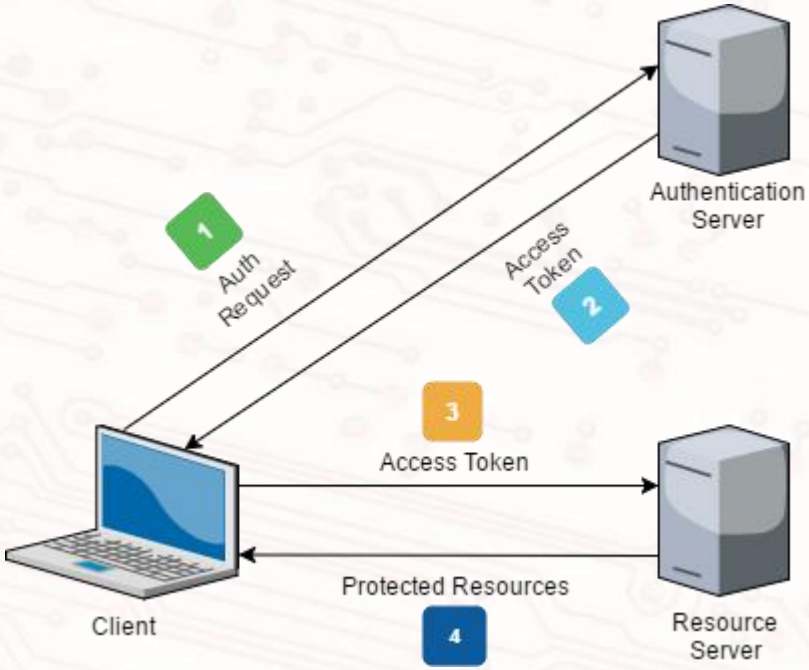
Cliente

```
SSLSocketFactory clientFactory = (SSLSocketFactory) SSLSocketFactory.getDefault();
Socket client = clientFactory.createSocket(server, port);
```

JWT

JWT

JSON Web Token (abreviado JWT) es un estándar abierto basado en JSON para la creación de tokens de acceso que permiten la propagación de **identidad y privilegios** o claims en inglés.
Por ejemplo, un servidor podría generar un token indicando que el usuario tiene privilegios de administrador y proporcionarlo a un cliente. El cliente entonces podría utilizar el token para probar que está actuando como un administrador en el cliente o en otro sistema.
El token está **firmado por la clave del servidor**, así que el cliente y el servidor son ambos capaces de verificar que el token es legítimo.
El token **puede caducar** pasado un periodo de tiempo, y por lo tanto queda invalidado.



fullstackmark.com

Conclusiones

¡Vamos con la práctica!

"El único sistema seguro es aquél que está apagado en el interior de un bloque de hormigón protegido en una habitación sellada rodeada por guardias armados"

-- Gene Spafford

Comunidad de Madrid
Dirección General de Educación Secundaria, Formación Profesional y Régimen Especial
CONSEJERÍA DE EDUCACIÓN, UNIVERSIDADES, CIENCIA Y PORTAVOCÍA

Unión Europea
Fondo Social Europeo
"El FSE invierte en tu futuro"
Financiado como parte de la respuesta de la Unión a la pandemia de COVID-19