Práctica 3
Programación SSL

DIIC

Humberto Martínez Barberá

Índice

- Programación Java con SSL
  - Introducción a JSSE
  - Ejemplo 1. Migración TCP a SSL
  - Aspectos avanzados de JSSE
  - Ejemplo 2. Servidor HHTPS

Programación con JSSE (1)

- *Java Secure Sockets Extension* (JSSE)
  - Implementación Java 100% pura de SSL y TLS
  - JSSE incluido en el JDK 1.4 y posteriores
    - Soporta SSL v3.0 y TLS v1.0
    - Existe JSSE 1.0 para versiones anteriores del JDK
- Estructura y objetivo de JSSE
  - Abstrae al programador de los detalles del protocolo SSL
  - Las clases JSSE se encuentran en dos paquetes
    - `javax.net`
    - `javax.net.ssl`
  - JSSE hace uso extenso del patrón *Factoría*

Programación con JSSE (2)

- Funcionalidades similares a los sockets estándar
  - La clase `SSLSocket` hereda de `Socket`
  - La clase `SSLServerSocket` hereda de `ServerSocket`
- Se instancian haciendo uso de factorías
  - `SSLSocketFactory` instancia `SSLSocket`
  - `SSLServerSocketFactory` instancia `SSLServerSocket`
- Las factorías se pueden usar de dos formas
  - Haciendo uso del método `getDefault()`
  - Construyendo una factoría específica, ajustándola a las necesidades de la aplicación que se va a desarrollar

Ejemplo 1. Migración Simple (1)

- Servidor TCP simple
  - Por claridad no se usan hilos múltiples

```
import java.io.*;
import java.net.*;

ServerSocket   s;
Socket         c;
OutputStream   out;
InputStream    in;

try {
    s   = new ServerSocket (port);
    c   = s.accept ();
    out = c.getOutputStream ();
    in  = c.getInputStream ();
} catch (IOException e) {}
```

Ejemplo 1. Migración Simple (2)

- Servidor SSL simple

```
import java.io.*;
import javax.net.ssl.*;

SSLServerSocketFactory    fac;
SSLServerSocket  s;
SSLSocket                 c;
OutputStream              out;
InputStream               in;

try {
    fac  = (SSLServerSocketFactory) SSLServerSocketFactory.getDefault ();
    s    = (SSLServerSocket) fac.createServerSocket (port);
    c    = (SSLSocket) s.accept ();
    out  = c.getOutputStream ();
    in   = c.getInputStream ();
} catch (IOException e) {}
```

Ejemplo 1. Migración Simple (3)

- Cliente TCP simple

```
import java.io.*;
import java.net.*;

Socket          s;
OutputStream    out;
InputStream     in;

try {
    s   = new Socket (host, port);
    out = s.getOutputStream ();
    in  = s.getInputStream ();
} catch (IOException e) {}
```

Ejemplo 1. Migración Simple (4)

- Cliente SSL simple

```
import java.io.*;
import javax.net.ssl.*;

SSLSocketFactory        fac;
SSLSocket               s;
OutputStream            out;
InputStream             in;

try {
    fac = (SSLSocketFactory) SSLSocketFactory.getDefault ();
    s   = (SSLSocket) fac.createSocket (host, port);
    out = s.getOutputStream ();
    in  = s.getInputStream ();
} catch (IOException e) {}
```

Proveedor SunJSSE (1)

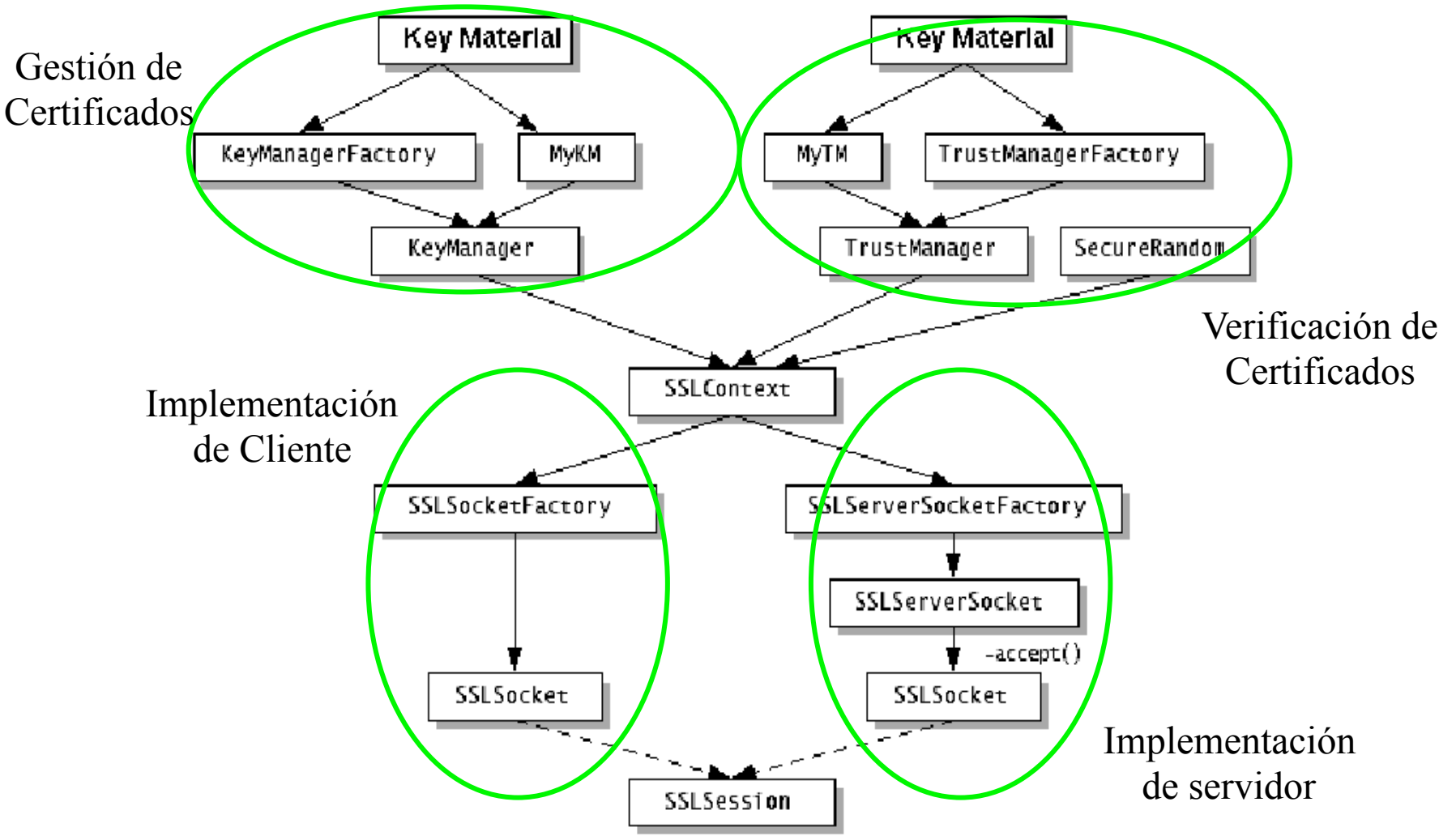
- Implementación de JSSE registrada en la JCA de Sun
  - *Java Cryptography Architecture*
- El proveedor incluye distintas suites de cifrado
  - Accesibles a través de la clase `SSLSocket`
    - `getSupportedCipherSuites()`
    - `getEnabledCipherSuites()`
    - `setEnabledCipherSuites()`
- El proveedor incluye distintos servicios de autenticación
  - Acceso a repositorio de certificados ("*JKS*", "*PKCS12*")
  - Certificados tipo X.509 ("*SunX509*")
  - Especificación del tipo de protocolo ("*SSLv3*", "*TLSv1*")

Proveedor SunJSSE (2)

Algoritmo Criptográfico	Proceso Criptográfico	Longitud Clave (bits)
RSA clave pública	Autenticación e intercambio de claves	2048 (autenticación) 2048 (intercambio) 512 (intercambio)
RC4	Cifrado	128 128 (40 efectivos)
DES	Cifrado	64 (56 efectivos) 64 (40 efectivos)
Triple DES	Cifrado	192 (112 efectivos)
Diffie-Hellman clave pública	Generación de claves	1024 512
DSA clave pública	Autenticación	1024

TLS_DHE_DSS_WITH_3DES_EDE_CBC_SHA

Arquitectura de JSSE



Generación de Certificados

- JSSE incluye una utilidad para generar certificados

```
canijo> keytool -genkey -keystore claves -keyalg rsa -alias humberto
```

  - Se necesitan dos passwords (keystore y certificado)

```
Enter keystore password: hola-1
What is your first and last name? [Unknown]: humberto.dif.um.es
What is the name of your organizational unit? [Unknown]: DIIC
What is the name of your organization? [Unknown]: Universidad de Murcia
What is the name of your City or Locality? [Unknown]: Murcia
What is the name of your State or Province? [Unknown]: Murcia
What is the two-letter country code for this unit? [Unknown]: ES
Is CN=humberto, OU=DIIC, O=Universidad de Murcia, L=Murcia,
ST=Murcia,
C=ES correct? [no]: yes

Enter key password for <humberto>
     (RETURN if same as keystore password): hola-2
```

Uso de Certificados

- El servidor envía un certificado si se le especifica
- Por defecto, la autenticación de cliente no se activa
  - El servidor la activa sobre su `SSLServerSocket`
    - `setNeedClientAuth(true)` <= se exige certificado
    - `setWantClientAuth(true)` <= sólo si lo soporta la suite

Ejemplo 2. Servidor HTTPS (1)

```
import java.io.*;
import java.net.*;
import javax.net.*;
import javax.net.ssl.*;
import java.security.*;
import java.util.*;

public class HttpsServer extends HttpServer {
   public void run() {
      ServerSocket listen;
      try {
         listen = getServer();
         while(true) {
            Socket client = listen.accept();
            ProcessConnection cc = new ProcessConnection(client);
         }
      } catch(Exception e) { System.out.println("Exception: "+e.getMessage()); }
   }

   public static void main(String argv[]) throws Exception {
      HttpsServer https = new HttpsServer();
      https.run();
   }
```

Ejemplo 2. Servidor HTTPS (2)

```
public static final int   HTTPS_PORT       = 443;
private String            keystore         = "claves";
private char[]            keystorepwd      = "hola-1".toCharArray();
private char[]            keypwd           = "hola-2".toCharArray();

   public ServerSocket getServer() throws Exception {
      KeyStore            ks;
      KeyManagerFactory   kmf;
      SSLContext          sslctx;
      ServerSocketFactory ssf;
      ServerSocket        ss;

      ks         = KeyStore.getInstance ("JKS");
      ks.load (new FileInputStream (keystore), keystorepwd);
      kmf        = KeyManagerFactory.getInstance ("SunX509");
      kmf.init (ks, keypwd);
      sslctx     = SSLContext.getInstance("SSLv3");
      sslctx.init (kmf.getKeyManagers (), null, null);
      ssf        = sslctx.getServerSocketFactory ();
      ss         = (SSLServerSocket) ssf.createServerSocket (HTTPS_PORT);
      return ss;
    }
}
```

Ejemplo 2. Servidor HTTPS (3)

- Al acceder desde un navegador da error
  - Certificados `keytool` no validados por ninguna CA

Bibliografía

- W. Stallings (2004) *Fundamentos de Seguridad en Redes*, Prentice Hall
- A.O. Alan, P. Freier, P.C. Kocher (1996) *The SSL Protocol Protocol Version 3.0*, Internet Draft