NAVEGAR SEGURO POR INTERNET

Josep R. Pegueroles, Juan José Alins
Profesores Asociados del Departamento de Ingeniería Telemática
Universitat Politècnica de Catalunya
teljpv@mat.upc.es, juanjo@mat.upc.es

En nuestras rutinarias incursiones en Internet, cada vez es más frecuente encontrarnos con páginas que nos informan que son seguras. En la mayoría de los casos se trata de webs que ofrecen servicios bancarios o venta de productos y en las que los datos que se intercambian servidor y cliente deben ser confidenciales. Muchas veces, sin prestarle más atención, aceptamos esos mensajes que nos muestra nuestro navegador y que nos indican si la información que vamos a enviar podrá ser leída por alguien a quien no va dirigida. En este artículo se pretende explicar brevemente cuáles son los mecanismos que podemos usar para protegernos frente a esos peligros y cómo podemos conseguir ofrecer esos mismos servicios de seguridad a través de un servidor web seguro.

1. INTRODUCCIÓN.

Internet, en sus inicios, no fue pensada como una red intrínsecamente segura. Lo principal era lograr comunicaciones eficaces, sin preocuparnos por si los datos que se intercambiaban en estas comunicaciones podían ser interceptados por agentes intrusos. Rápidamente, la Red fue creciendo y evolucionando tecnológicamente y el uso que se hacía de ella forzó la necesidad de garantizar servicios de seguridad a esas comunicaciones. Inicialmente, esa seguridad se ofreció mediante soluciones a nivel de aplicación, pero con la aparición del comercio electrónico y la necesidad de realizar pagos no presenciales a través de la Red, se vió necesario incorporar mecanismos de seguridad a más bajo nivel. La solución fue impulsada por Netscape, y consiste en una capa de seguridad adicional responsable de proporcionar los servicios requeridos, esta capa se denominó SSL o Secure Sockets Layer.

Figura 1 *Icono de seguridad del programa Netscape*

En los siguientes apartados explicaremos qué es lo que hay detrás de los servicios de seguridad que nos ofrece nuestro navegador, en otras palabras, qué es lo que ha tenido que ocurrir para que el candado de nuestro navegador se cierre(**Figura 1**) Una vez ya sepamos cómo funciona por dentro la seguridad de nuestro navegador de Internet explicaremos brevemente cómo podemos instalar un servidor web seguro mediante software de libre distribución.

El resto del artículo se estructura de la siguiente manera. En la sección 2 nos encargamos de la descripción del protocolo Secure Sockets Layer, necesario para entender el funcionamiento de un servidor web seguro. En la sección 3 explicamos qué son y para qué sirven las Autoridades de Certificación y los Certificados y qué papel desempeñan en los servicios web seguros. A continuación, en la sección 4 explicaremos brevemente el comportamiento de los navegadores web. Finalmente, en el apartado 5 expondremos las herramientas de libre distribución disponibles para la realización de un servidor web seguro y se detallarán los pasos a seguir para conseguir un servidor de este tipo con el software presentado.

2. EL PROTOCOLO SSL/TLS.

Debido a la inexistencia de medidas de seguridad TCP/IP se desarrolló un protocolo con el que se puede conseguir conexiones seguras entre dos máquinas conectadas a Internet.

El SSL o Secure Sockets Layer es un protocolo inicialmente desarrollado por Netscape, ubicado entre la

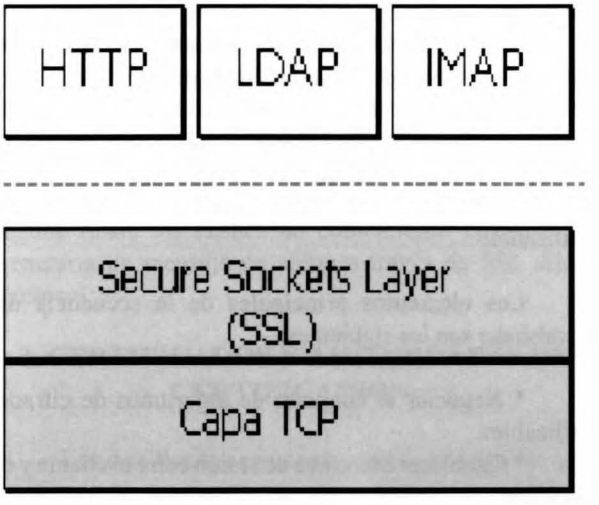

Figura 2 *Icono de seguridad del programa Netscape*

capa de transporte fiable (capa TCP) y la capa de aplicación (por ejemplo http), tal como se ve en la **Figura 2**. SSL proporciona un canal seguro entre cliente y servidor y ofrece la posibilidad de autenticación de las dos partes, la utilización de funciones de hash para la integridad y el cifrado para la privacidad de los datos.

Este protocolo fue diseñado para poder aceptar un gran número de opciones en cuanto a algoritmos utilizados para cifrar, para realizar hashes y firmas digitales. Estos algoritmos son negociados tanto por el cliente como por el servidor al principio de la conexión.

La **Tabla 1** muestra las distintas versiones de este protocolo, de las cuales la más ampliamente utilizada es la SSL v.3.0 ya que los navegadores todavía no se han actualizado a las nuevas versiones y las anteriores mostraron vulnerabilidades.

Versión	Origen	Descripción
SSL2.0	Netscape	Primera versión implementada.
SSL3.0	Netscape	Revisión con parche a ataques, soporte códigos no RSA y encadenamiento de certificados
TLS1.0	IETF	Actualización de la capa MAC a HMAC, mensajes estándar.

Tabla 1 *Versiones de SSL.*

2.1 Establecimiento de una sesión.

La sesión SSL se establece siguiendo la secuencia de handshake entre el cliente y el servidor, tal como se muestra en la **Figura 3**. Esta secuencia puede variar dependiendo de si el servidor o el cliente pueden o deben enviar un certificado.

Una vez iniciadas, las conexiones SSL, se pueden reutilizar ya que cada una de ellas tiene un identificador de sesión que el servidor recordará. Cuando el cliente se vuelva a conectar, enviará este identificador de sesión y si no ha caducado su validez, el servidor inmediatamente aceptará una conexión cifrada con los últimos valores utilizados.

Los elementos principales de la secuencia de handshake son los siguientes:

* Negociar el conjunto de algoritmos de cifrado utilizables.
* Establecer una clave de sesión entre el cliente y el servidor.
* Identificar el servidor al cliente. (opcional)
* Identificar el cliente al servidor. (opcional)

El primer paso, la negociación de la gama de algoritmos de cifrado utilizables, posibilita tanto al cliente como al servidor escoger unos algoritmos que puedan utilizar los dos.

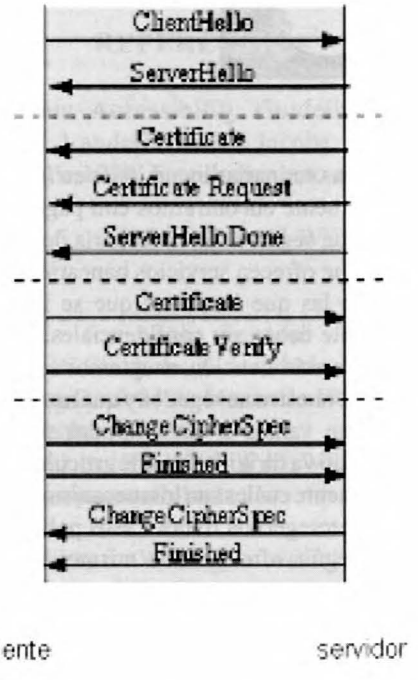

Figura 3 *Establecimiento de una sesión SSL. (simplificado)*

Los algoritmos a usar deben ser tanto para la autenticación e intercambio de claves como para el cifrado de los datos y el cálculo de funciones resumen o de hash.

Autenticación e intercambio de claves.

El fin general del intercambio de claves es crear un secreto común que sólo conozcan las dos partes, cliente y servidor, que después se utilizará para cifrar los mensajes y comprobar la integridad de los mismos. SSL/TLS ofrece tres tipos de autenticación:

* Autenticación del cliente y del servidor.
* Autenticación del servidor únicamente.
* Ningún tipo de autenticación.

En los dos primeros casos, el canal es seguro contra ataques de hombre en el medio (donde una tercera entidad suplanta completamente al cliente o al servidor), pero en el tercero SSL/TLS es vulnerable a este tipo de ataques. Servidores anónimos no pueden autentificar a los clientes. Si el servidor requiere un certificado al cliente, este debe de ser verificable por una Autoridad Certificadora válida. De hecho, ambas partes son responsables de comprobar que los certificados presentados por la otra parte son verificables por una Autoridad Certificadora válida.

En el tercer tipo, las sesiones completamente anónimas pueden ser establecidas utilizando RSA o Diffie-Hellman para el intercambio de claves. En este caso, el servidor, en vez de enviar un certificado, sólo envía una clave sin ningún tipo de firma, lo que le hace completamente anónimo pero vulnerable al ataque de hombre en el medio.

Para los dos primeros tipos se pueden utilizar tanto un intercambio de claves con certificados RSA o DH firmados por una autoridad RSA o DSS. En cualquiera de estos casos el servidor y/o el cliente, a parte de verificar el certificado, comprobarán que ése es el certificado del otro extremo cifrando el hash de un número aleatorio (reto) con su clave privada y enviándolo al otro extremo. El otro extremo, como dispone de clave pública (que antes se ha enviado) puede descifrar el hash que después de varias modificaciones será el que se utilizará como clave del cifrado simétrico.

Algoritmo simétrico para la transferencia de datos.

Existen las siguientes opciones para el cifrado por clave simétrica:

DES (con claves de 40 y 56 bits), triple DES (clave de 168 bits), RC4 (con claves de 40 y 128 bits), RC2 (clave de 40 bits) e IDEA (128 bits. También ofrecen la posibilidad de no cifrar en ningún sentido la transferencia de datos.

Función de hash.

SSL/TLS ofrece la opción de utilizar tanto MD5 como SHA-1 para generar el mensaje de autentificación de código (MAC).

2.2 Secuencia del protocolo de handshake.

El protocolo de handshake utiliza otros tres protocolos:

* SSL Handshake Protocol para establecer la sesión SSL
* SSL Change Cipher Spec Protocol para ponerse de acuerdo en la clave a utilizar en el cifrado simétrico.
* SSL Alert Protocol para el envío de mensajes de alerta y error entre el cliente y el servidor.

Estos protocolos, así como los datos de la aplicación están encapsulados dentro de SSL Record Protocol, tal como se muestra en la **Figura 4.**

Al iniciarse la conexión, el SSL Record Protocol no utiliza ningún algoritmo ni MAC, por lo que la comunicación se hace sin ningún tipo de cifrado. En cuanto el cliente y el servidor compartan un secreto y se hayan puesto de acuerdo en el algoritmo de cifrado simétrico a utilizar,

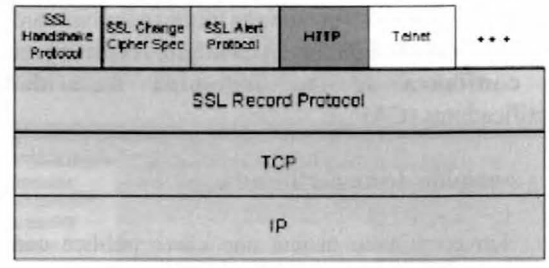

Figura 4 *Stack del protocolo SSL.*

usarán el SSL Change Cipher Spec para notificar al SSL Record Protocol que a partir de ese momento empiece a cifrar y a verificar los datos con la clave proporcionada.

2.3 Transferencia de datos.

El SSL Record Protocol introducido anteriormente es el utilizado para enviar los datos entre el cliente y el servidor. En la **Figura 5** se puede observar un organigrama de la estructura de este protocolo.

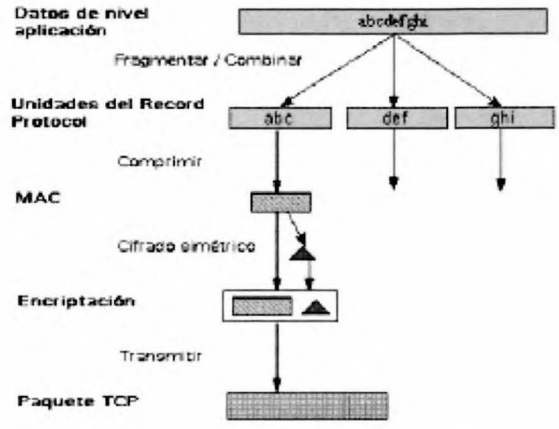

Figura 5 *Stack del protocolo SSL.*

Lo primero que hace el SSL Record Protocol es fragmentar los datos que le proporciona la aplicación en unidades más pequeñas, opcionalmente se comprimen cada una de estas. Una vez comprimida la unidad, se realizará un hash (con MD5 o SHA1) para ser utilizado más tarde como verificación de la integridad de la unidad. Después, la unidad y el hash se cifrarán y se transmitirán.

El proceso inverso se utilizará en recepción. Si por alguna razón los hashes no coincidiesen entonces se generaría un mensaje de alerta a través de SSL Alert Protocol.

3. CERTIFICADOS Y AUTORIDADES DE CERTIFICACIÓN.

Los certificados son documentos electrónicos que contienen la asociación de una entidad con su clave pública, estos documentos están notarizados, es decir, los

firman agencias de confianza. Sería el equivalente a un notario, que es aquella persona en la que se supone confían distintas partes integrantes en un litigio. A estas agencias de confianza se las denomina Autoridades Certificadoras.(CA)

Contenido de un certificado.

Un certificado asocia una clave pública con la identidad real de una determinada persona, servidor o entidad conocida como el Subject. La información sobre el Subject incluye el Distinguished Name (información de identificación) y la clave pública. También incluye la identificación y la firma de la Autoridad Certificadora (CA) que ha emitido el certificado (o Issuer) y el periodo de validez del certificado. Opcionalmente puede tener información adicional o extensiones e información administrativa para el uso de la CA: versión, número de serie. (**Figura 6**)

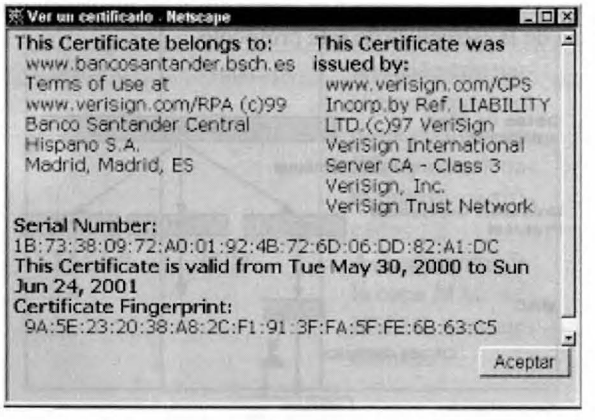

Figura 6 *Contenido de un certificado.*

El distinguished name se define por el estándar X.509. Este estándar define los campos, los nombres de los campos, y las abreviaciones que se usan. Los campos son el Common Name (CN) que indica el nombre de la identidad a certificar, el Organization (O) cuyo contenido es la organización a la que pertenece la identidad, la Organizational Unit (OU) o departamento a la que pertenece la identidad certificada, Locality (L) o ciudad de la entidad. State/Province (SP) y Country (C) o país. (**Figura 7**).

Una CA puede definir una norma (o policy) especificando cuales de los campos anteriores son obligatorios y cuales opcionales. Un ejemplo habitual es que el navegador Netscape requiera que el CN que presenta un servidor tenga el nombre coincidente con el dominio que dice servir.

El formato binario de un certificado se define usando la notación ASN.1. Esta notación especifica los contenidos y las reglas de codificación definen como se traduce esta información a su formato binario.

Figura 7 *Campos de un certificado.*

La codificación binaria de un certificado se define usando Distinguished Encoding Rules (DER) que se basan en el más general Basic Encoding Rules (BER). Para aquellas transmisiones que no pueden ser en forma binaria se traducen a forma ASCII usando codificación en base64. Esta versión de codificación se denomina codificación PEM y es la que aparece entre las líneas — BEGIN CERTIFICATE — y — END CERTIFICATE —. (**Figura 8**)

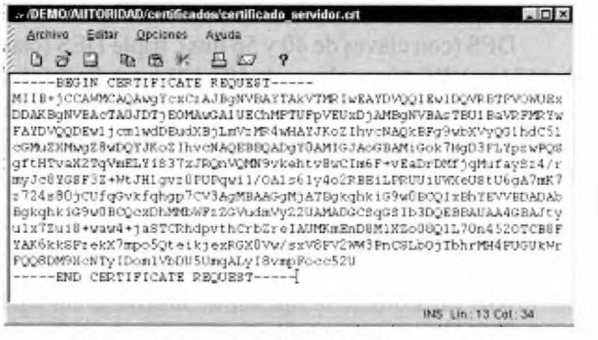

Figura 8 *Formato PEM de un certificado.*

Autoridades Certificadoras

En un sistema criptográfico de clave pública cada usuario debe hacer accesible su clave pública a cualquiera que quiera establecer una comunicación con él. Esto podría conseguirse, por ejemplo, introduciéndola en una base de datos de libre acceso. El sistema expuesto, en cambio, presenta el problema que alguien suplante nuestra identidad mediante la modificación de dicha clave en la base de datos de libre acceso.

Para dificultar ese ataque los certificados deben estar certificados, o lo que es lo mismo, los certificados de cada usuario deben estar firmados por alguna Autoridad Certificadora (CA) con un sistema de firma digital de clave pública.

Cuando a una CA le llega una petición de certificación, esa CA debe verificar que la clave pública que se quiere certificar se corresponda con los datos personales.

Cuando se quiere establecer un protocolo criptográfico con alguien, se obtiene el certificado firmado por la CA y se verifica la firma de la CA.

Para hacer esto es preciso disponer del certificado de la CA para obtener su clave pública y poder verificar.

Asimismo, un certificado de una CA puede estar firmado por una CA de nivel superior. El sistema funciona sobre una estructura jerárquica de CAs.

Los certificados de más alto nivel deben ser conocidos por todos los usuarios sin que sea posible su engaño. Todo el mundo debe poder confiar en ellos. Los certificados de más alto nivel están autofirmados.

Estas empresas certificadoras son los emisores (Issuers) de los certificados y son las encargadas de administrarlos y mantenerlos. Los servicios que deben prestar son los siguientes:

* Verificar las peticiones de certificados. Es decir, confirman que quien pide el certificado es quien dice ser que es.
* Procesar peticiones de certificado. O lo que es lo mismo, realizar el certificado y firmarlo.

También es posible crear una CA propia, no firmada por ninguna CA de jerarquía superior, en la que un grupo cerrado de usuarios de internet confiará. Esta solución se adopta mayoritariamente para intranets y servicios corporativos.

Mantenimiento de Certificados

Hacer una CA requiere una sólida estructura de mantenimiento ya que no sólo deben emitirse certificados sino que también se debe determinar hasta cuando son válidos, se deben renovar al finalizar el periodo de validez y se debe mantener una lista de certificados revocados (en cualquier momento una entidad debe poder invalidar su propio certificado),a esta lista se la denomina CRL o Certification Revocation List.

4. RECONOCIMIENTO DE CERTIFICADOS POR PARTE DE NAVEGADORES Y ESTABLECIMIENTO DE UNA SESIÓN SEGURA SOBRE SSL.

Los navegadores de internet están configurados para confiar, por defecto, en algunas CAs como por ejemplo Verisign (**Figura 9**). Esta configuración por defecto es extremadamente importante porque implica que todas las entidades incluidas en nuestra lista de ACs válidas o firmates tienen la potestad de garantizar que un sitio es quien dice ser que es. Dicho de otra forma, si algún site malicioso consigue un certificado firmado (o garantizado) por cualquiera de las entidades incluidas en esa lista nuestro navegador no tendrá forma de detectarlo y confiaremos en él, de forma que le enviremos toda la información que requiera pensando que está en buenas manos, cuando en realidad se hará un uso ilícito de ella.

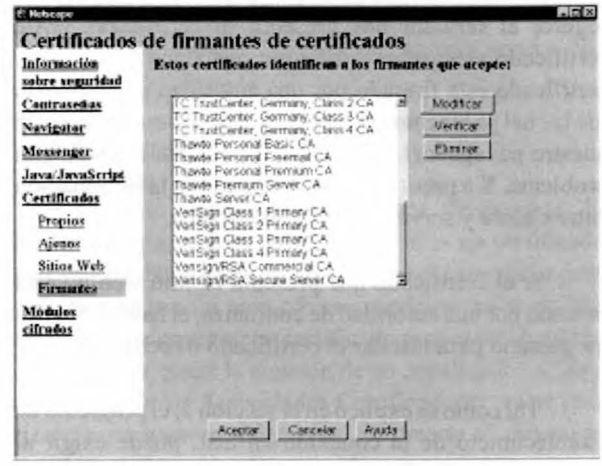

Figura 9 *Lista de Autoridades de Certificación aceptadas por el brower.*

Por este motivo es fundamental estar muy seguro de la legitimidad de una Autoridad Certificadora antes de incluirla en nuestro navegador.

Netscape usa un certificado del tipo x.509 que permite al servidor autentificarse frente a los clientes que realizan conexiones SSL.

Durante el protocolo de establecimiento de la conexión segura o Handshake SSL, el servidor presenta un certificado al cliente como garantía de quien dice ser que es. Este certificado está firmado por una Autoridad Certificadora CA. Si el navegador tiene en su base de datos el certificado del servidor o bien el certificado del servidor está firmado por una Autoridad Certificadora que conoce el navegador, el Handshake se finalizará correctamente. Es decir, se podrán poner de acuerdo en una clave de sesión y unos algoritmos criptográficos que usarán para establecer la conexión segura.

Netscape permite a los usuarios añadir nuevos certificados de confianza a su base de datos. Esta base de datos es accesible en todo momento al usuario de Netscape a través de del botón «Security» de la «barra de herramientas de navegación»(Figura 9).

En general, los navegadores pueden tratar dos tipos de certificados:

* Un certificado de una Autoridad Certificadora, firmado por la misma autoridad certificadora y que la identifica. Este tipo de certificados se denomina «autofirmados». Si añadimos un certificado de este tipo a nuestra base de datos, cada vez que nos conectemos a un servidor (vía https://) que nos presente un certificado

firmado por dicha Autoridad Certificadora, el navegador confiará en el servidor.

* Un certificado de Site (sitio web). Es un certificado de servidor. Cuando un cliente se conecta a un servidor seguro, el servidor nos presenta su certificado. (Este certificado siempre está firmado por alguna CA). Si el certificado está firmado por una autoridad de confianza (de la cual ya tenemos su certificado en la base de datos de nuestro navegador), la conexión se realizará sin ningún problema. Y a partir de ese momento toda la información entre cliente y servidor viajará encriptada.

Si el certificado que presenta el sitio web no está firmado por una autoridad de confianza, el navegador nos irá guiando para instalar el certificado o rechazarlo.

Tal como se explicó en la sección 2, el protocolo de establecimieto de la conexión en SSL puede exigir al cliente que se identifique mediante un certificado, por este motivo nuestro navegador también debe disponer de certificados que garanticen nuestra identidad. (Figura 10)

5. INSTALACIÓN DE UN SERVIDOR SEGURO CON SOFTWARE DE LIBRE DISTRIBUCIÓN SOBRE PLATAFORMA LINUX.

Los programas y aplicaciones con código fuente de libre distribución son desarrolladas por voluntarios de todo el mundo que se comunican a través de internet. Estos programas y/o sistemas operativos son por lo general de una alta calidad ya que colaboran cientos e incluso miles de personas en su desarrollo, participando simplemente por afición. Cualquiera puede añadir sus valoraciones o implementaciones al programa.

Normalmente la mayoría de los programas desarrollados bajo este método se distribuyen bajo licencias denominadas BSD o GNU. Hay pocas diferencias entre ellas y tienen en común que se puede utilizar el código fuente para cualquier aplicación, ya sea comercial o no, sin coste alguno. Un análisis a fondo de este tipo de licencias se puede encontrar en la Free Software Foundation.

A la contra, el mayor problema que presentan dicho tipo de aplicaciones es que nadie responde de ellas. Es decir, cualquier empresa o particular es libre de obtener el código fuente y utilizarlo, ahora bien, si en algún momento el programa deja de funcionar correctamente, no existe ningún tipo de garantía.

Sistemas operativos de libre distribución.

Dentro del apartado de los sistemas operativos existen varios disponibles con el código fuente. Todos ellos son sistemas UNIX de los cuales, se puede encontrar dos posibilidades Linux y BSD.

Linux. Este popular Sistema operativo fue desarrollado originalmente por Linus Torvalds en la Universidad de Helsinki, Finlandia, como un proyecto para una de las asignaturas de su carrera. Al poco tiempo de hacerlo público, entusiastas de todo el mundo se unieron al proyecto, y ahora este sistema operativo es uno de los más estables y eficientes del mercado. Funciona bajo CPUs Intel 80386 y superiores, Sun SPARC, Motorola 68000, PowerPC, MIPS, ARM y Digital Alpha. Se distribuye bajo la licencia de GNU.

BSD. Originalmente desarrollado en la universidad de California en Berkeley, EEUU es un sistema operativo altamente estable, del cual cabe destacar su subsistema de red TCP/IP que está considerado como uno de los mejores existentes. De BSD existen tres versiones gratuitas, FreeBSD , NetBSD y OpenBSD.

Programas de libre distribución para instalar un servidor web seguro.

Apache. Existen diversas implementaciones de servidores http (sin SSL) con código fuente de libre distribución, pero sin duda la más ampliamente aceptada es el Apache. Este servidor funciona bajo casi cualquier plataforma Unix y bajo Windows NT y copa aproximadamente un 53% de la cuota de mercado de los servidores WWW mundiales.

Sobre este servidor se desarrollaron originalmente unas extensiones para poder ser utilizado con SSL, y la implementación pasó a ser conocida como Apache-SSL. Basado en OpenSSL, este servidor proporciona perfectamente capacidades de SSL con http.

OpenSSL. OpenSSL es una librería de programación con código fuente de libre distribución que implementa y proporciona servicios de SSL. Esta librería poporciona tanto servicios de criptografía (cifrado por clave simétrica, clave pública, hashes, certificación X.509, etc) como servicios específicos sobre el protocolo SSL/TLS, como podría ser la manipulación automática de sockets on SSL.

OpenSSL también dispone de un programa llamado igual que la librería, para poder acceder a las funciones de las mismas sin tener que programar nada. Es decir, si por ejemplo se quiere generar un hash MD5 de un fichero, mediante este programa se puede. También se pueden generar claves RSA, crear certificados, cifrar por RC5, etc. Esta librería está basada en la librería SSLeay

ModSSL. A mediados de 1998, y debido a serias divergencias sobre el futuro desarrollo del proyecto Apache-SSL, Ralph Engelschall, uno de sus impulsores, decidió tomar el código hasta entonces creado y lanzar un nuevo proyecto basado en él, pero con amplias diferencias sobre nuevas metas y funcionamiento interno respecto a su predecesor Apache-SSL.

Este proyecto es conocido como mod_ssl y sorprendentemente ha dado como resultado un servidor

https de libre distribución altamente configurable, versátil y rápido. El código es muy limpio y está altamente documentado.

Pasos de la instalación.

A continuación se explican lo más detalladamente posible los pasos a seguir para conseguir prestar un servicio http seguro. Dicha instalación se probó sobre una distribución SuSE 7.0 y sus resultados se pueden encontrar en https://bruce.upc.es/~ssade. En todos los comandos el $ significa el prompt del sistema y los parámetros en negrita las deberás sustituir por tus opciones. El carácter \ significa cambio de línea pero no de comando. Aunque para no sería necesario, para evitar problemas durante la instalación es aconsejable obtener permisos de root.

Conseguir los paquetes necesarios.

Bájate los fuentes de los paquetes Apache, mod_ssl y OpenSSL y descomprímelos en tu ordenador.

```
$ftp://ftp.apache.org/dist/\
  apache_1.3.12.tar.gz
$ftp://ftp.modssl.org/source/\
  mod_ssl-2.6.4-1.3.12.tar.gz
$ftp://ftp.openssl.org/source/\
  openssl-0.9.5a.tar.gz
$ tar zxvf apache_1.3.12.tar.gz
$ tar zxvf mod_ssl-2.6.4-1.3.12.tar.gz
$ tar zxvf openssl-0.9.5a.tar.gz 2.
```

OpenSSl. Instala en tu sistema Linux el paquete OpenSSL para disponer en tu ordenador de las librerías criptográficas necesarias. Este paquete no sólo proporcionará a mod_ssl las librerías requeridas en el momento de la compilación del Apache parcheado sino que también permite realizar acciones propias de Autoridades Certificadoras, como por ejemplo emitir certificados, firmarlos,... El uso de estas funciones sería necesario si quisiéramos un servidor seguro personalizado, creándonos nosotros mismos nuestro certificado. Este objetivo queda fuera del alcance de este artículo, de todas formas se remite al lector interesado a la documentación de la librería OpenSSL para resolver como se realizarían dichas acciones.

```
$ cd openssl-0.9.5a
$./config - prefix=\
/directorio/donde/quieres/los/ejecutables
$ make
$ make install
$ cd ..
$ export PATH=$PATH:\
/directorio/de/los/ejecutables/bin
```

ModSSL. Parchea el código fuente de Apache con la extensión ModSSL.

```
$ cd mod_ssl-2.6.4-1.3.12
$ ./configure \
—with-apache=../apache_1.3.12 \
—with-ssl=../openssl-0.9.5a \
—prefix= \
/directorio/donde/quieres/los/ejecutables
```

Apache. Compila e instala el paquete apache con los parches de seguridad de mod_ssl.

```
$ cd ..
$ cd apache_1.3.12
$ make
```

Crea un certificado de servidor de pruebas. El certificado que se crea con esta acción es un certificado únicamente válido para poder arrancar el navegador pero difícilmente servirá para ofrecer servicios reales de http seguro. Para conseguir un certificado propio válido tienes dos opciones: pedir la emisión de un certificado válido a cualquiera de las Autoridades Certificadoras reales (por ejemplo en www.verisign.com) o crearte tú mismo tu propio certificado mediante la librería OpenSSL.

```
$ make certificate
$ make install
```

Borra los ficheros temporales de compilación (opcional)

```
$ rm -rf apache_1.3.12
$ rm -rf mod_ssl-2.6.4-1.3.12
$ rm -rf openssl-0.9.5a
```

Ejecuta tu SSL-aware Apache y pruébalo

```
$ /directorio/ejecutables/bin/httpd -DSSL
$ netscape https://nombre.de.tu.servidor.
```

Si todo ha funcionado correctamente deberías ver la página de pruebas del servidor Apache.

REFERENCIAS.

Camps, P. Análisis y desarrollo de un entorno de seguridad sobre plataformas de libre distribución. PFC UPC 2000

Pastor, J. Sarasa, M.A. Criptografía digital. Fundamentos y Aplicaciones. Colección textos docentes. Publicaciones Universitarias Zaragoza.

Schneier, B. Applied Cryptography Protocols, Algorithms and Source Code in C John Wiley & Sons, Inc 1994

REFERENCIAS A HIPERVÍNCULOS.

http://www.apache.org
http://www.freebsd.org
http://www.gnu.org
http://www.linux.org
http://www.modssl.org
http://www.netbsd.org
http://www.netcraft.org
http://www.openbsd.org
http://www.openssl.org
http://www.ssleay.org
https://bruce.upc.es/~ssade