SSL, Secure Sockets Layer y Otros Protocolos Seguros para el Comercio Electrónico

(Versión 1.0)
Puedes descargar la última versión de este documento de:
http://jo.morales0002.eresmas.net/fencasa.html

José María Morales Vázquez
e-mail: josemaria.morales@hispalinux.es

Master de Seguridad Informática, I Edición. Curso 2001/2002
Universidad Politécnica de Madrid. Organizado por ALI

Resumen: El comercio electrónico en Internet constituye una compleja operación en la que uno de los principales elementos es dar seguridad a vendedor y comprador de que la transacción comercial que están realizando se realiza sin intromisiones de ningún tipo y sin posibilidad de fraudes o engaños entre ninguna de ambas partes. Durante los últimos 10 años han ido surgiendo un número considerable de tecnologías y sistemas de pago electrónico que ofrecen las garantías de seguridad e integridad necesarias para realizar estas transacciones de una forma fiable. No obstante, este sigue siendo el mayor obstáculo (tanto técnico como psicológico) a vencer para que se produzca el definitivo despegue del comercio electrónico. Mientras no exista confianza, mientras los usuarios teman al fraude, mientras se desconozcan los sistemas de pago empleados y su fiabilidad, es difícil que esta oportunidad de negocio prospere. SSL, Secure Sockets Layer, fue diseñado y propuesto por Netscape Communications Corporation en 1994 junto con su primera versión de Netscape Navigator. Tras una accidentada historia inicial de revisiones alcanzó su madurez en 1995 con la versión 3.0, convirtiéndose hoy en día en la solución más extendida en los servidores que ofrecen servicios de comercio electrónico y dejando virtualmente en la cuneta a todos sus competidores a pesar de que no es ni el método más seguro ni el más idóneo para implantar este tipo de soluciones, puesto que fue diseñado como un protocolo seguro de propósito general. En este documento nos centraremos en realizar un detallado análisis del, hasta ahora, ganador en esta carrera pero sin olvidar por completo los méritos de sus competidores.

0 Índice

1 Introducción

La seguridad (o la aparente carencia de seguridad, según el lado desde el que miremos) es la barrera real y psicológica que es necesario franquear para el definitivo despegue del comercio electrónico. Los elementos que forman esta barrera son siete y van más allá de lo meramente físico (hardware) o lógico (protocolos, aplicaciones) e involucran factores tales como la legislación, la educación de los usuarios, etc. Estos siete elementos son:

- Confidencialidad.
- Integridad.
- Disponibilidad.
- No Repudio.
- Verificación de la Identidad.
- Validez Legal.
- Confianza de los Usuarios.

SSL, el protocolo dominante en la actualidad en el panorama del comercio electrónico, proporciona confidencialidad, integridad y verificación de la identidad de ambas partes (esta últimas característica sólo si se utiliza en conjunción de certificados digitales en ambosextremos, cosa que no suele ser frecuente). La disponibilidad es algo que se escapa del ámbito del protocolo y que se ve gravemente perjudicada por lo pesado del mismo (aunque existen soluciones hardware integradas con el y diseñadas a tal efecto que veremos en el capítulo 6 de este documento). La validez legal de una transacción comercial a través de Internet es algo fuera del objeto de este documento y la confianza de los usuarios habrá que ganarla poco a poco, interviniendo en ella mucho más la actitud de los medios de comunicación y la formación de los usuarios que la verdadera validez del protocolo. El estándar SSL no contempla la implementación del No Repudio de mensajes, aunque como veremos en el capítulo seis sería muy simple y fácil dar soporte a esta característica.

Sin embargo, decíamos en el resumen previo que SSL no está diseñado específicamente para el comercio electrónico ¿cómo es posible entonces que se adapte tan bien a los requisitos del mismo? En realidad no lo hace. En una transacción comercial electrónica existe una tercera parte involucrada que no se contempla en SSL: el banco. Cuando realizamos una transacción comercial usando SSL el cliente escoge la mercancía, realiza la orden de pedido y envía la información de pago (típicamente un número de tarjeta de crédito o débito) al vendedor. Este realiza las comprobaciones oportunas en el banco y una vez que obtiene la validez del medio de pago procesa el pedido. ¿Cual es el problema en este esquema? Es evidente: el vendedor no compromete ante el cliente ningún tipo de información sensible y, además, está protegido contra posibles fraudes (tarjetas falsas o caducadas) . El comprador, sin embargo, tiene que enviar información sensible al vendedor comprometiendo la misma ante la honestidad de este y, además, exponiéndose ante

posibles fraudes por parte del mismo. SSL, pues, aplicado a un esquema de comercio electrónico proporciona la máxima protección al vendedor en detrimento de la seguridad del comprador.

Mientras que algunos de los competidores de SSL (S-HTTP, PCT o TLS) son meras variantes más o menos mejoradas de hacer lo mismo que se han visto perjudicadas por la supremacía que durante largos años ha mantenido Netscape en este sector, existen dos protocolos (SET y Cybercash) que si ofrecen una solución a estos problemas y que, sin embargo, no han tenido la aceptación y difusión que debieran, al menos hasta el momento. CyberCash fue comprada a mediados de 2001 por Verisign, quien se limita a mantener el soporte a la escasa cartera de clientes existentes de esta tecnología con objeto de darle una muerte más o menos digna y tratar de potenciar su propia solución al problema del pago electrónico y SET, a pesar de estar arropada por VISA y Mastercard (dos de los grandes gigantes del pago magnético) no acaba de arrancar en el terreno del pago electrónico. Trataremos de estudiar el porque más adelante.

2 Análisis de lo Existente

No es fácil realizar una estadística fiable en este aspecto, pero todos coinciden en que la gran mayoría de transacciones comerciales que se realizan hoy en día en la red se protegen mediante SSL. La gran difusión del protocolo, la facilidad de implantación por parte de administradores de sistemas y la transparencia para el usuario que, en la mayoría de los casos ni siquiera se entera de que lo está usando lo ha convertido en el protocolo dominante en un terreno en el que existen opciones mucho mejores.

TLS, la llamada versión 3.1 de SSL, es el sucesor natural de este. Lo tiene todo para que esta sucesión se realice sin traumas: como el alías que lo acompaña indica (nos referimos a la 'etiqueta' de SSL 3.1) es meramente un SSL mejorado y no propietario que posee mecanismos de compatibilidad con SSL y viene ya de serie con la gran mayoría de los navegadores. Y sin embargo, la inercia en el mundo de los servidores es tal que la transición hacía el no acaba de realizarse. Tanto IIS como Apache, los dos servidores más extendidos (juntos suman más del 90% del total del parque de Internet) llevan soporte de TLS y, sin embargo, los administradores de sistemas siguen decantándose por 'las viejas soluciones'.

PCT es el clónico de SSL inventado por Microsoft. Poco merece la pena hablar de el: aporta pocas novedades técnicas y, en esta ocasión, las cosas no les han salido bien a los chicos de Redmon. Ni siquiera las últimas versiones de Explorer soportan ya este protocolo.

Tampoco se puede esperar ya nada de S-HTTP, un protocolo muy orientado a proporcionar seguridad exclusivamente al protocolo HTTP que, a pesar de ser posiblemente el protocolo seguro más antiguo de Internet, no ha sabido aprovecharse de esa ventaja, quizás or su orientación tan específica.

Ya en otro ámbito, IPSec ha sido postulado en ocasiones como sucesor o competidor de SSL. Sin embargo su mayor complejidad en cuanto a implantación y mantenimiento reducen su ámbito de actuación, al menos por el momento, a ámbitos de comunicaciones empresariales y no al mundo abierto y sin fronteras del comercio electrónico.

Y ya en el terreno de los protocolos realmente orientados a las transacciones comerciales electrónicas tenemos que hablar al menos de dos de ellos: Cybercash, el más antiguo y SET el más poderoso.

Cybercash introdujo en el panorama de los protocolos seguros conceptos como el de la pasarela de pago y mecanismos para proteger no sólo al vendedor, sino también al comprador. La especial complejidad de un esquema que no se vio acompañado por el estado de la técnica (cuando Cybercash atravesaba su momento de esplendor las

conexiones habituales en los ordenadores de los usuarios eran de 9K6 o 14K4) y la inmadurez de los sistemas PKI imprescindibles para su desarrollo lo condenaron a una larga peregrinación de cambios y adaptaciones al mercado hasta su definitiva muerte a manos de Verisign.

SET ha retomado el testigo de Cybercash. Utiliza un esquema muy similar y está arropado por los gigantes del pago electrónico y las grandes compañías de informática y telecomunicaciones. Las PKI se encuentran en un momento de extensión y reconocimiento (aunque aún adolecen de algunos serios problemas) y las comunicaciones ya no son lo que eran. No obstante sigue sin arrancar. El principal problema, según todos los expertos, es la complejidad de la infraestructura a desarrollar. El comercio electrónico es hoy en día, en su mayor parte, un mecanismo impulsivo y espontáneo y sería muy difícil convencer al usuario de la necesidad de obtener una certificación (que habría de adquirir de modo presencial) para realizar sus compras en Internet. Hay, de todas formas, muchos factores próximos que pueden cambiar este punto: la inminente implantación del DNI digital hará que todos tengamos en el bolsillo un certificado digital con las máximas garantías. Quizás sea ese el 'pistoletazo' de salida que necesita SET (al menos en nuestro país) y represente la definitiva salida de SSL de un mundo que no le corresponde por derecho.

3 Objetivos

Hay mucho escrito en Internet acerca de SSL, un protocolo con casi 10 años de existencia y ampliamente utilizado por los internautas. Basta con mirar las referencias que acompañan al capítulo de Bibliografía. El objetivo de este documento no es, pues, aportar nada nuevo a la comprensión del mismo, sino ordenar ideas, presentar de forma clara su mecanismo de funcionamiento y facilitar un marco de comparación del mismo con sus competidores pasados, presentes y futuros.

Los capítulos centrales de este documento son los comprendidos entre el 4 y el 8.

En el capítulo 4, el siguiente, se explica en detalle el funcionamiento de SSL y las particularidades de su funcionamiento. No se ha buscado en el un enfoque excesivamente técnico, sino todo lo contrario: que sea comprensible para el mayor número de personas. Quién no vea satisfechas sus necesidades puede consultar el *draw* disponible en Internet (ver Bibliografía) que es extraordinariamente ilustrativo.

En el capítulo 5 se presenta la competencia directa de SSL intentando, igualmente, alejarnos de características excesivamente técnicas y centrándonos en los mecanismos de funcionamiento y en las ventajas e inconvenientes que aportan sobre SSL

El capítulo 6 detalla los principales problemas, algunos de diseño y otros de implementación, que presenta el protocolo SSL. Se trata, en los casos en los que es posible, de aportar soluciones para aliviarlos o solventarlos.

En el capítulo 7 se enumeran sus principales debilidades. Igual que decíamos en el punto anterior, algunas son en realidad debilidades del protocolo y en otras ocasiones se trata de errores de implementación

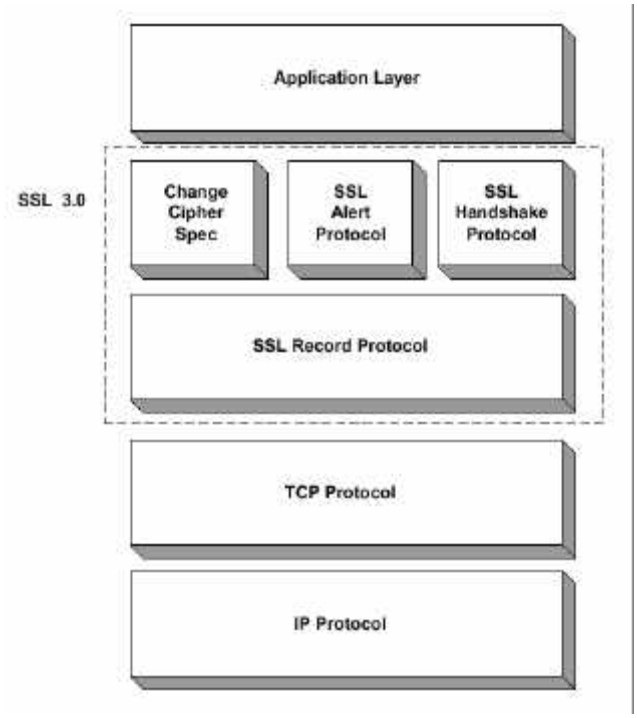
En el capítulo 8 se relaciona un completo historial de vulnerabilidades encontradas en SSL o en aplicaciones que lo utilizan a lo largo de toda su vida: desde 1994 hasta diciembre de 2002.

4 SSL, Secure Sockets Layer

Netscape desarrolló la primera versión de SSL en 1994. Está primera versión jamás fue implementada de forma pública. Tan sólo unos pocos meses después liberó una importante actualización que vino a llamarse SSL 2.0 y que si tuvo una implementación real a pesar de ir aquejada de importantes errores de diseño. En noviembre de 1995 Netscape publica la especificación para SSL 3.0 la cual, desde entonces, se ha convertido en el estándar 'de hecho' para comunicaciones seguras entre clientes y servidores en Internet.

El objetivo de Netscape era crear un canal de comunicaciones seguro entre un cliente y un servidor que fuese independiente del sistema operativo usado por ambos y que se beneficiara de forma dinámica y flexible de los nuevos adelantos en materia de cifrado a medida de que estos estuvieran disponibles. SSL fue diseñado como un protocolo seguro de propósito general y no teniendo en mente las necesidades específicas del comercio electrónico.

SSL trabaja sobre el protocolo TCP y por debajo de protocolos como HTTP, IMAP, LDAP, etc., y puede ser usado por todos ellos de forma transparente para el usuario. Opera entre la capa de transporte y la de sesión del modelo OSI (o entre la capa de transporte y la de aplicación del modelo TCP) y está formado, a su vez, por dos capas y cuatro componentes bien diferenciados.



El protocolo de registro (*Record Protocol*) se encarga de encapsular el trabajo de los elementos de la capa superior, construyendo un canal de comunicaciones entre los dos extremos objeto de la comunicación.

El verdadero corazón de SSL está en el protocolo de *Handshake* que es el encargado de intercambiar la clave que se utilizará para crear un canal seguro mediante un algoritmo eficiente de cifrado simétrico. También es responsabilidad de este protocolo coordinar los estados de ambos extremos de la transmisión.

El protocolo de Alerta es el encargado de señalizar problemas y errores concernientes a la sesión SSL establecida.

Por último, el *Change Cipher Spec Protocol* está formado por un único mensaje consistente en un único byte de valor 1 y se utiliza para notificar un cambio en la estrategia de cifrado.

A grandes rasgos, podríamos decir que SSL trabaja de la siguiente forma: en primer lugar intercambiamos una clave de longitud suficiente mediante un algoritmo de cifrado asimétrico. Mediante esa clave establecemos un canal seguro utilizando para ello un algoritmo simétrico previamente negociado. A continuación, toma los mensajes a ser transmitidos, los fragmenta en bloques, los comprime, aplica un algoritmo hash para obtener un resumen (MAC) que es concatenado a cada uno de los bloques comprimidos para asegurar la integridad de los mismos, realiza el cifrado y envía los resultados. El estado de todas estas operaciones son controladas mediante una máquina de control de estados. Una sesión SSL puede comprender múltiples conexiones. Adicionalmente, se pueden establecer múltiples sesiones SSL simultaneas.

A continuación veremos todos estos procesos con un poco más de detalle.

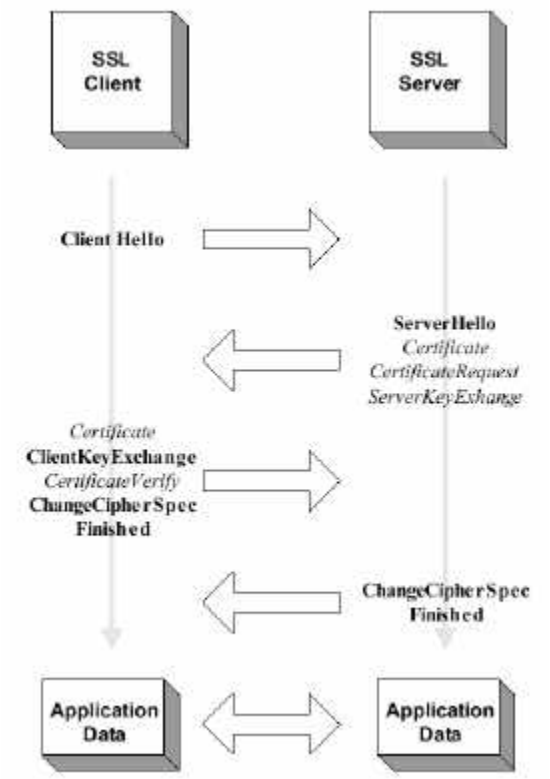
El protocolo de Handshake es el encargado de negociar los atributos de la sesión SSL que permitirán construir un canal seguro de comunicaciones.

En primer lugar el cliente envía un mensaje *Client Hello* al servidor el cual debe de responder con un mensaje similar de *Server Hello*. Estos mensajes son utilizados para dar a conocer ciertas características de ambos: versión del protocolo usada, algoritmos de cifrado conocidos y preferidos, longitudes máximas de clave que admite para cada uno de ellos, funciones hash y métodos de compresión a utilizar. Adicionalmente cliente y servidor pueden intercambiar dos números generados aleatoriamente para ser usados como sal. En este momento, además, el servidor asigna un identificador a la sesión y se hace constar la fecha y hora de la misma. El identificador de sesión es enviado al cliente en el mensaje de *Server Hello.* Si el servidor no respondiera con un mensaje de *Server Hello* o este no fuese valido o reconocible la sesión abortaría inmediatamente. Generalmente el servidor, el segundo en contestar, elige los algoritmos más fuertes de entre los soportados por el cliente. Si no hay acuerdo en este punto se envía un mensaje de error y se aborta la sesión.

A continuación del mensaje de *Server Hello*, el servidor puede envíar su Certificado (típicamente un X.509) de forma que sea autenticado por el cliente y que, además, este reciba su clave pública. Si no es así, le envía al cliente su clave pública mediante un mensaje de *Server Key Exchange* (o también si ha enviado su Certificado y este es únicamente para firma y autenticación). Está claro es que al menos uno de estos dos mensajes es necesario para establecer el canal seguro. Un último mensaje que puede enviar el servidor en esta fase de negociación es una solicitud de certificado al cliente. Por último, la fase concluye con el envío, por parte del servidor, de un mensaje de *Server Hello Done*.

Si el Servidor ha solicitado su certificado al cliente, este debe de responder con el o con un mensaje de alerta indicando que no lo posee. A continuación se envía un mensaje de *Client Key Exchange* donde el cliente envía al servidor, cifrada mediante la clave pública de este, la clave maestra, un número aleatorio generado por el y que actuará como clave del algoritmo simétrico acordado para el intercambio de datos. Por último, si el cliente ha enviado un certificado y este tiene capacidades de firma, enviará adicionalmente un mensaje de *Certificate Verify* firmado digitalmente con objeto de que el servidor pueda verificar que la firma es válida. En este punto el cliente da por concluida la fase mediante un mensaje de *Change Cipher Spec* seguido, inmediatamente, de un mensaje de *Finished* que ya va cifrado mediante los algoritmos y claves recién negociados.

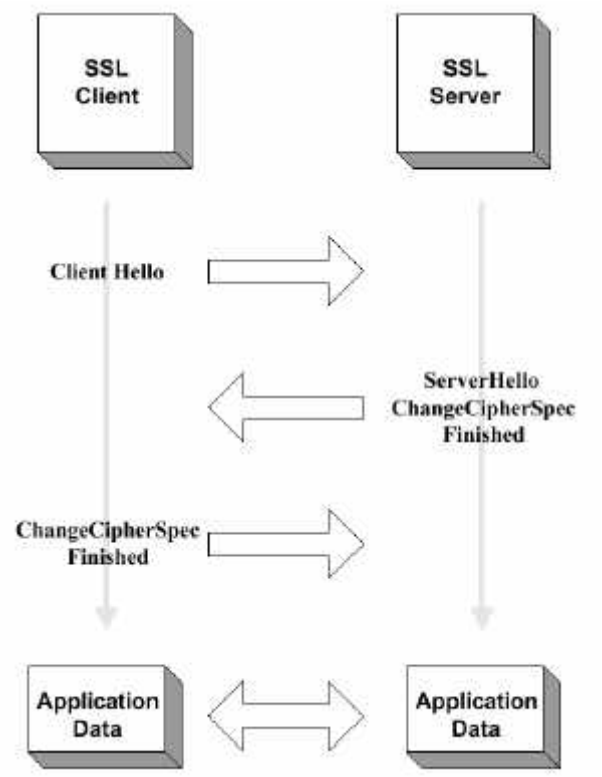
En respuesta, el servidor envía su propio mensaje de *Change Cipher Spec* y, a continuación, su mensaje de *Finished* cifrado con los parámetros negociados. En este momento finaliza la fase de *Handshake* y cliente y servidor pueden intercambiar datos libremente. Podemos ver un esquema de este intercambio de mensajes en la siguiente figura:



Durante la transmisión de datos los mensajes son fragmentados y comprimidos por el protocolo de registro antes de su envío y descomprimidos y reconstruidos por el mismo protocolo al otro extremo de la comunicación. El algoritmo de compresión utilizado es característico de la sesión y se negocia, como hemos visto, en la fase de *Handshake*.

SSL puede establecer múltiples conexiones dentro de una misma sesión o reanudar una sesión previamente interrumpida. En ambos casos el intercambio de mensajes de la fase *Handshake* es mucho más reducido como veremos a continuación.

El cliente envía un mensaje de *Client Hello* usando el identificador de la sesión previamente negociada. El servidor verifica si ese identificador es válido y en caso afirmativo devuelve un mensaje de *Server Hello* usando el mismo identificador de sesión. Acto seguido, envía al cliente un mensaje de *Change Cipher Spec* y a continuación un mensaje de *Finished* cifrado ya con los parámetros de la sesión reanudada. El cliente responde con sus propios mensajes de *Change Cipher Spec* y *Finished* y seguidamente comienzan a intercambiar datos. Podemos ver este proceso en la siguiente imagen:



Para facilitar la labor de los desarrolladores, NETSCAPE mantiene una excelente información sobre SSL. En la siguiente URL tenemos una traza completa y detallada del intercambio de mensajes y el contenido de los mismos entre cliente y servidor en dos casos distintos: usando RC4-40 y MD5 y usando RC4-128 y MD5. Para ambos casos tenemos los mensajes correspondientes a la primera conexión y a una supuesta reanudación de la misma:

http://wp.netscape.com/eng/ssl3/traces/index.html

Como hemos dicho anteriormente, SSL es capaz de trabajar de forma transparente con todos los protocolos que trabajan sobre TCP. La IANA tiene asignado un número de puerto por defecto a cada uno de ellos que podemos ver en la tabla siguiente:

identificador	**puerto TCP**	**descripción**
https	443	HTTP sobre SSL
smtps	465	SMTP sobre SSL
nttps	563	NTTP sobre SSL

ldaps	646	LDAP sobre SSL
telnets	992	TELNET sobre SSL
imaps	993	IMAP sobre SSL
ircs	994	IRC sobre SSL
pop3s	995	POP3 sobre SSL
ftps-data	989	FTP-Datos sobre SSL
ftps-control	990	FTP-Control sobre SSL

Estos puertos son válidos también para las implementaciones de estos mismos protocolos sobre TLS.

5 La Competencia de SSL

En este capítulo analizaremos la principal competencia que SSL tiene, o ha tenido, puesto que algunos de los protocolos que se mencionan ya han desaparecido o están a punto de hacerlo. Veremos aquí dos tipos de protocolos: los que conceptualmente se concibieron, al igual que SSL, como protocolos de carácter general para asegurar la información transmitida a través de un canal de comunicaciones y aquellos que se concibieron teniendo en mente el comercio electrónico como propósito específico.

En el primer grupo tenemos tres protocolos: S-HTTP, PCT, TLS e IPSec. Todos siguen el mismo esquema: en primer lugar se invoca un mecanismo para, mediante criptografía asimétrica, intercambiar una clave secreta de longitud suficiente y, en segundo lugar, se utiliza esta clave para cifrar la información transmitida mediante un algoritmo simétrico mucho más eficiente.

En el segundo grupo veremos CyberCash y SET. Ambos implementan una arquitectura mucho más adecuada y orientada para el comercio electrónico en base a uan tercera parte de confianza que actúa como intermediaria entre comprador, vendedor y los bancos de ambos.

5.1 S-HTTP, Secure HTTP

Secure HTTP (o S-HTTP) fue posiblemente el primer protocolo de seguridad implementado para Internet. Desarrollado por Enterprise Integration Technologies Inc. (EIT) en conjunción con RSA Data (los cuales formaron la empresa Terisa Systems a este efecto), vio la luz por primera vez en forma de borrador en Junio de 1994.

S-HTTP es una extensión del protocolo HTTP cuya finalidad es la transmisión de datos de forma segura sobre la web entre un cliente y un servidor. para ello trabaja sobre la capa de aplicación cifrando el contenido de los mensajes entre las dos máquinas usando para ello un sistema de cifrado basado en una pareja de claves pública y privada. S-HTTP permite la negociación inicial de la fortaleza del cifrado a usar (algoritmo y longitud de claves), permite la autenticación de ambos extremos mediante el uso de firmas digitales y verifica la integridad de los datos usando MAC (Message Authentication Code). S-HTTP es un protocolo muy flexible que permite que estas opciones (firma y cifrado) se usen o no de forma opcional.

La principal diferencia, en cuanto a diseño, con SSL es que S-HTTP fue concebido para la transmisión de mensajes individuales de forma segura mientras que SSL se diseñó para establecer una conexión segura permanente entre dos ordenadores.

Puesto que S-HTTP es un protocolo situado en la capa de aplicación es capaz de proporcionar características de no repudio de mensajes de forma individualizada a través de las firmas digitales, mientras que SSL, un protocolo de bajo nivel que opera en la capa de transporte, no lo hace (aunque podría). Por el mismo motivo (su situación en la pila de protocolos) S-HTTP es capaz de trabajar de forma conjunta con un cortafuegos, mientras que un cortafuegos que deba de soportar una comunicación SSL no tiene forma de saber que datos pasan a través de esa conexión.

La situación en la pila de comunicaciones como contrapartida da otras ventajas a SSL: sus servicios de seguridad son transparentes al usuario y a la aplicación y puede ser usado por otros protocolos aparte del HTTP. S-HTTP, sin embargo, se definió como una extensión de HTTP y sus servicios sólo están disponibles para este protocolo.

S-HTTP es, en definitiva, más flexible que SSL. Cada aplicación puede configurar el nivel de seguridad que necesita permitiendo mantener más conexiones o responder más rápidamente. SSL, por el contrario, puede beneficiarse más fácilmente de soluciones de optimización como cifradoras, balanceadores de carga, etc. Su menor número de opciones también lo hacen más fácil de mantener.

En cuanto al modo de operación de S-HTTP es substancialmente más seguro que el de SSL: resiste criptoanálisis de texto en claro, ataques de '*Man In the Middle*' y ataques de retransmisión de tramas. Es más robusto que SSL porque permite opciones de renegociación y reintento, mientras que su principal debilidad, documentada ampliamente, se encuentra en el mecanismo inicial de intercambio de claves.

5.2 PCT, Private Communication Technology

PCT, Private Communication Technology, fue la respuesta de Microsoft a la supremacía de Netscape en el mundo de las comunicaciones seguras. El primer borrador del mismo apareció en septiembre de 1995. PCT fue diseñado para soportar transacciones comerciales de forma segura y espontánea y, al igual que SSL, opera en la capa de transporte siendo, pues, independiente de la aplicación que lo maneja. PCT incorpora mecanismos de clave pública/privada de RSA para cifrado y autenticación de ambos extremos de la comunicación.

Las principales diferencias entre PCT y SSL están en la fase de negociación: Uno de los mas novedosos mecanismos que aporta PCT es que separa las negociaciones y mecanismos de autenticación y cifrado, permitiendo que las claves usadas en la autenticación excedan la longitud impuesta por las leyes de exportación de los EE.UU., no así las de cifrado. La segunda gran aportación de este protocolo es corregir un agujero de seguridad en el diseño de la fase de handshake de SSL (y que estudiaremos en el capítulo 7 del presente documento).

A pesar de estas dos evidentes ventajas sobre su directo competidor, la gran implantación de SSL ha dejado a este nuevo protocolo sin ninguna posibilidad de

competencia. En Mayo de 2000 y como último intento, Microsoft anunció las especificaciones para una versión convergente de SSL y PCT que vendría a llamarse STLP (Secure Transport Layer Protocol) de la cual nunca más se ha vuelto a hablar y que no ha llegado a tener ninguna implementación real.

5.3 Transport Layer Security, TLS

TLS nace de la mano del IETF en forma de RFC en enero de 1999. Se construye a partir de las especificaciones de SSL 3.0 y son tan semejantes que a veces se lo conoce como SSL 3.1. Por decirlo con las propias palabras de los autores de la especificación "[...] el protocolo TLS está basado en las especificaciones de SSL 3.0 publicadas por Netscape. Las diferencias entre este protocolo y SSL 3.0 no son grandes, pero si suficientes para que TLS 1.0 y SSL 3.0 no puedan interoperar (aunque TLS incorpora un mecanismo mediante el cual una implementación de TLS puede trabajar con SSL 3.0)".

Las principales diferencias entre SSL 3.0 y TLS 1.0 son las siguientes:

- En los mensajes *Certificate Request* y *Certificate Verify* del protocolo de *Handshake*. En SSL 3.0 si el servidor solicita un certificado al cliente para que se autentique, este debe de responder con el o con un mensaje de alerta advirtiendo de que no lo tiene. En TLS 1.0 si el cliente no posee certificado no responde al servidor de ninguna forma a este requerimiento.
- Cálculo de las claves de sesión. El mecanismo utilizado para construir las claves de sesión es ligeramente diferente en TLS 1.0.
- TLS 1.0 no soporta el algoritmo de cifrado simétrico FORTEZZA que si es soportado por SSL 3.0. Esto es debido a que TLS busca ser integramente público mientras que FORTEZZA es un algoritmo propietario.
- TLS utiliza un mecanismo diferente y más seguro en el cálculo del MAC.
- TLS 1.0 introduce nuevos códigos de alerta no contemplados por SSL 3.0
- TLS 1.0 introduce un nuevo mecanismo en el relleno de los bloques para frustrar ataques basados en el análisis de la longitud de los mensajes.
-

5.4 IPSec

Muchas de las funcionalidades que SSL presta están contempladas en IPv6, concretamente en el protocolo IPSec que puede usarse sobre el actual IPv4. IPSec proporciona autenticación, confidencialidad e integridad de datos, trabaja en la capa de red y posee protecciones efectivas contra la repetición de tramas y, gracias a su posición en la pila de protocolos, es capaz de trabajar con UDP y otros protocolos de la capa de transporte, uno de los inconvenientes de SSL y que veremos con un poco más de detalle en el punto 6 de este documento.

Por estos motivos algunos autores presentan a IPSec como el verdadero sustituto de SSL mientras que otros insisten en que ambos coexistirán juntos puesto que ofrecen soluciones óptimas para diferentes problemas. Nortel Networks presentó en un estudio de las soluciones aportadas por ambos protocolos una tabla en laque define ciertos criterios por los que decantarse por uno u otro protocolo:

	SSL/TLS	**IPSec**
Control de accesos		
Conexiones permanentes		✓
Conexiones efímeras o puestos móviles	✓	
Ambos tipos de acceso	✓	✓
Usuarios		
Todos los usuarios son empleados de la compañía		✓
No todos los usuarios son empleados de la compañía	✓	
No todos los usuarios son empleados de la compañía y, además, algunos trabajan con sus propios sistemas	✓	✓
Software cliente		
Todos los usuarios han de tener acceso a todos los recursos de la red		✓
Deseamos controlar el acceso a determinadas aplicaciones	✓	
Necesitamos niveles variables de control de acceso en las diferentes aplicaciones	✓	✓
Confidencialidad y Autenticidad		
Precisamos de un alto nivel de seguridad en el cifrado y autenticación		✓
La confidencialidad y autenticidad no son especialmente críticas en nuestros sistemas	✓	
Precisamos de niveles moderados de confidencialidad e integridad	✓	
Criticidad de los recursos accedidos		
Alta		✓
Moderada	✓	
Variable	✓	✓

Criticidad de las funciones realizadas		
Alta		✓
Moderada	✓	
Variable	✓	✓
Nivel técnico de los usuarios		
Entre moderado y alto		✓
Entre moderado y bajo	✓	
Implantación, flexibilidad y escalabilidad		
Deseamos una implantación rápida y facilidad de mantenimiento	✓	
Deseamos flexibilidad en las modificaciones futuras		✓
Ambas consideraciones son importantes	✓	✓

5.5 CyberCash

CyberCash fue una compañía fundada en 1994 por un grupo de expertos en ordenadores y medios de pago convencionales con el objetivo único de desarrollar un medio seguro de pago electrónico cuando este era un concepto que aún apenas se intuía. El proyecto fue incapaz e encontrar un mercado y la compañía sobrevivió a duras penas adaptándose a otros terrenos menos exigentes, siempre dentro del terreno de la seguridad informática, hasta su definitiva compra y absorción (además de su muerte programada, podríamos decir) por Verisign en mayo de 2001. CyberCash, que llegó a cotizar en bolsa, es hoy en día uno de los ejemplos 'de libro' analizados a la hora de estudiar la difícil supervivencia de una empresa en la web, a pesar de ofrecer una solución pionera y eficiente a un problema que hasta el momento no había sido resuelto.

CyberCash introduce a mediados de 1995 el primer modelo de pago por Internet que contempla la figura de una tercera parte, el propio servidor de CyberCash, que actúa como intermediario entre comprador y vendedor asegurando la transacción económica sin que el vendedor conozca jamás los datos de pago del cliente, pero garantizando que no existan fraudes. En 1996 introduce, siguiendo la misma filosofía, un sistema de transacciones electrónicas entre particulares.

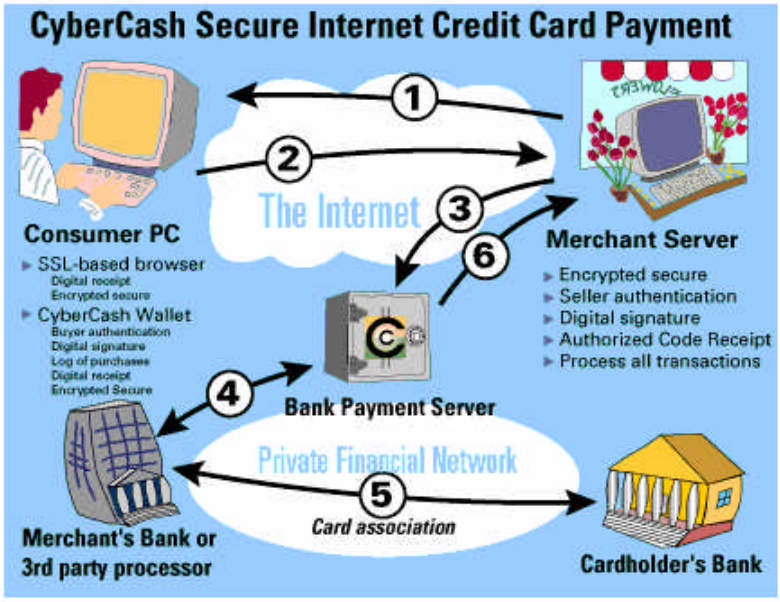
Para trabajar con CyberCash era preciso (usaremos a partir de ahora el pasado porque, como hemos dicho, aunque aún existe alguna implementación de este sistema no será por demasiado tiempo) instalar en el ordenador del cliente un software de 'cartera' que almacena en forma segura los datos relativos a los medios de pago. El sistema más popular para ello se denominaba InstaBuy y también ha desaparecido ya del mercado.

En cuanto a los algoritmos utilizados por el protocolo, Cybercash usaba una combinación de cifrado asimétrico (RSA) y simétrico (Triple DES). Para generar los hash utilizaba MD5.

El proceso de pago con Cybercash implica a cinco actores (cliente, vendedor, sus respectivos bancos y un servidor de CyberCash) y seis pasos principales:

- El cliente elige el producto o servicio que desea comprar y el vendedor le presenta en la web una página en la que detalla los precios y condiciones y le pide confirmación de la transacción
- El cliente acepta las condiciones y elige CyberCash como forma de pago. En este momento su ordenador lanza la aplicación de 'cartera' electrónica y le pide que elija la tarjeta que usara como forma de pago. Una vez que el cliente realiza su elección la información es enviada al vendedor cifrada y firmada de manera que no es ni accesible ni manipulable por este.
- El vendedor se queda con los datos relativos a los productos comprados dirección de entrega y envía al servidor de CyberCash la información de pago del cliente (cifrada y firmada por este) e incluye la información relativa a su banco para el cobro cifrada y firmada digitalmente.
- El servidor de CyberCash recibe la petición de transacción, descifra la información de cliente y vendedor, verifica la integridad de los datos y la identidad de los actores involucrados y, si todo está en orden, reexpide la información relativa a la transacción económica al banco del comerciante pero lo hace a través de una línea dedicada y no de Internet.
- El banco del vendedor envía una petición de autorización instantánea al banco del cliente a través de sus propios medios de comunicación y retransmite a CyberCash la respuesta, afirmativa o negativa, de este que ha debido de realizar sus propias comprobaciones: si la tarjeta está caducada o ha sido robada, si el saldo es suficiente, etc.
- Cybercash retransmite la respuesta del banco al vendedor y este anula la compra o procesa el pedido en función de esta.

Podemos ver un esquema del proceso completo en el siguiente gráfico:



Como vemos, Cybercash es un sistema de pago que se usaba junto con SSL y que incluso, en su etapa final, se adapto para ser compatible con SET. Era un sistema maduro y estable cuando SET se encontraba aún en forma de borrador e indiscutiblemente se adaptaba mucho mejor al problema del comercio electrónico que SSL o ninguno de sus otros competidores. ¿Cuales fueron sus errores?

Uno de ellos fue la necesidad de contar con software específico en el ordenador del cliente. Aunque el software de 'cartera' electrónica era gratuito, era preciso instalarlo. Cybercash trató de integrar su 'cartera' en el navegador y que esta fuera distribuida simultáneamente con ellos, pero ni Netscape ni Microsoft, cada uno con sus propios intereses en el mercado, accedió a ello.

La necesidad de registrarse previamente fue otro grave inconveniente para el éxito comercial de CyberCash. Para hacer una transacción comercial mediante SSL sólo es preciso disponer de un ordenador, un navegador instalado y conexión a Internet. Para usar Cybercash era preciso, además de la instalación de la 'cartera' electrónica antes mencionada, obtener un certificado digital e inscribirse en el servicio.

Por último, aunque Cybercash aseguraba que una transacción con su sistema duraba entre 15 y 20 segundos, la verdad es que no era así: La necesidad de involucrar a tantos actores y medios en tiempos en los que la calidad de las comunicaciones dejaba tanto que desear y el ancho de banda era un bien tan escaso aportaba muy mala imagen a este sistema de pago.

5.6 SET, Secure Electronic Transaction

En Febrero de 1996 VISA International, Mastercard International, Microsoft, IBM, Netscape y Verisign entre otras grandes compañías anuncian la creación de SETCo, una empresa cuyo objetivo sería la creación de un sistema seguro y universal de pago electrónico a través de Internet. La primera implementación y las especificaciones finales del mismo verían la luz en la segunda mitad de 1997. Las principales ventajas que aporta frente a SSL son las siguientes:

- Permite al cliente autenticar que el comerciante se encuentra autorizado para aceptar tarjetas de pago de forma segura y realizar transacciones económicas a través de Internet.
- Permite al vendedor autenticar la tarjeta de pago usada por el cliente en la transacción.
- Cada una de las informaciones involucradas en la transacción (datos de pago, de cobro, mercancía o servicios comprados y dirección de entrega de los mismos) es leída únicamente por el destinatario previsto.

SET usa criptografía asimétrica (RSA) para las firmas y para el cifrado de las claves de cifrado simétrico y de los datos bancarios y criptografía simétrica (DES) para la transmisión del resto de los datos involucrados en la transacción. Las claves usadas son de 128 bytes y SHA-1 la función hash usada para la verificación de integridad de los mensajes.

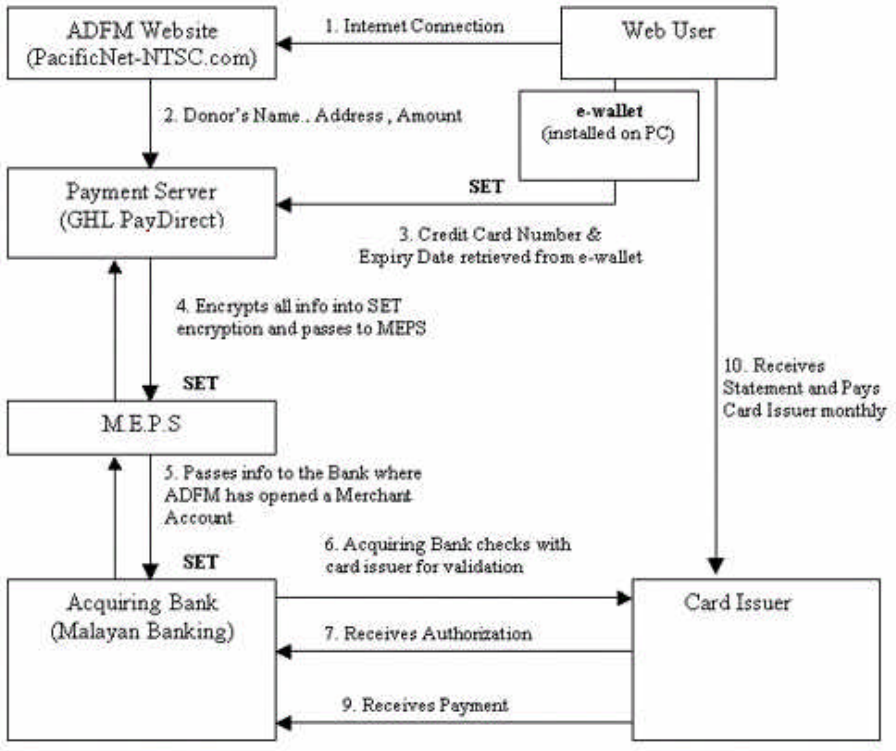
SET es, en cuanto a concepción, muy similar a CyberCash como veremos a continuación. El proceso de pago consta de 10 pasos:

- El cliente elige el producto o servicio que desea comprar y el vendedor le presenta en la web una página en la que detalla los precios y condiciones y le pide confirmación de la transacción.
- El cliente acepta las condiciones y elige SET como forma de pago. En este momento se inicia el protocolo mediante el envío por parte del vendedor de una descripción del pedido al cliente que arranca en el ordenador de este la aplicación de 'cartera' electrónica.
- El cliente comprueba la orden de pedido recibida, elige la forma de pago (si dispone de varias tarjetas registradas en la aplicación) y transmite la orden de pago al vendedor. La aplicación de cartera electrónica crea, a tal efecto, dos mensajes que envía al vendedor: uno de ellos contiene los datos del pedido, dirección de entrega, etc. y el otro las instrucciones de pago (número de tarjeta, banco emisor, etc.). El software de cartera genera un mensaje cifrado y firmado dualmente de forma que el vendedor pueda acceder tan sólo al primer mensaje y el banco a la segunda.
- El vendedor (en realidad el software de SET en su servidor) retransmite este mensaje, una vez recibido, a su banco
- El banco del vendedor descifra y valida la petición obteniendo las instrucciones de pago del cliente y verificando la integridad de los datos,

la identidad del vendedor y la validez de la transacción. Si todo es correcto se envía una petición de autorización al banco del cliente por los canales dedicados a tal efecto.

- El banco del cliente verifica la identidad del mismo, la validez de la tarjeta, la existencia de crédito y, si todo está en orden, autoriza la transacción.
- Cuando el banco del vendedor recibe la autorización genera y firma digitalmente un mensaje con el testigo de la autorización y transferencia de fondos que, una vez cifrado, se envía al vendedor.
- Cuando el vendedor recibe este mensaje y lo verifica, almacena el testigo de la transferencia, autoriza el suministro o proceso de los servicios demandados por el cliente y le envía a este un recibo de la compra que recoge la aplicación de 'cartera' electrónica y que actúa como justificante de la misma.
- Una vez procesado con éxito el pedido, el vendedor genera una petición de transferencia a su banco.
- Se hace efectivo el cargo en la cuenta del cliente.

A continuación podemos ver un esquema gráfico que ilustra este flujo de operaciones:



El protocolo definido por SET especifica el formato de los mensajes, las codificaciones y las operaciones criptográficas que deben usarse. No requiere un método particular de transporte, de manera que los mensajes SET pueden transportarse sobre HTTP en aplicaciones Web, sobre correo electrónico o cualquier otro método. Como los mensajes no necesitan transmitirse en tiempo real, es posible realizar implantaciones de SET eficientes basadas en correo electrónico u otros sistemas asíncronos.

Inicialmente SET solamente soportaba transacciones con tarjeta de crédito/débito, y no con tarjetas monedero. Posteriormente se extendió el estándar de manera que aceptara nuevas formas de pago. Actualmente está en curso un proyecto para incluir los certificados SET en las tarjetas inteligentes, de tal forma que el futuro cambio de tarjetas de crédito a tarjetas inteligentes pueda incorporar el estándar SET.

Bien apadrinado por grandes compañías con muchos intereses en juego y con un mecanismo eficiente y totalmente orientado al comercio electrónico ¿Cuales son los inconvenientes que está encontrando SET? En primer lugar, SET no resulta fácil de implantar, por lo que su despliegue está siendo muy lento. SET exige software especial para clientes, vendedores y bancos. En segundo lugar, los distintos involucrados en el desarrollo de estos productos cumplen con el estándar SET pero no siempre son compatibles entre si. Este es un problema que exige mayores esfuerzos de coordinación y más pruebas a escala mundial para asegurar la interoperabilidad. Estas barreras constituyen un obstáculo importante que seguirán retrayendo el despliegue SET en tanto no se alcance la convergencia de aplicaciones.

SET puede llegar a originar un conflicto en muchos vendedores a la hora de integrar los productos SET con sus sistemas internos de entrada de órdenes. Como la información sobre el número de la tarjeta de pago del comprador está cifrada e inaccesible al vendedor, pueden surgir problemas con sistemas internos que precisan para su propia contabilidad el número de la tarjeta del cliente. En tales casos, una posible solución sería que los propios comerciantes solicitaran los números de tarjeta de los compradores a las organizaciones de tarjetas de pago. Un inconveniente adicional de SET reside en su incapacidad para trabajar con pagos aplazados, modalidad muy extendida en países como España.

Uno de sus puntos fuertes constituye también su talón de Aquiles: la autentificación de todas las partes exige rígidas jerarquías de certificación, ya que tanto clientes como comerciantes deben adquirir certificados distintos para cada tipo de tarjeta de crédito, trámites que resultan engorrosos, para la mayoría de los usuarios. Se añade el problema de la revocación de certificados, la portabilidad de los mismos cuando el usuario trabaja en distintas máquinas y las cadenas de certificación. En definitiva, SET descansa sobre una infraestructura de clave pública (PKI) que en la actualidad dista mucho de ser perfecta.

La mayoría de los analistas coinciden en que, por el momento, SET no desbancará a SSL. Muchos de ellos dudan de que lo haga nunca y en lo que todos coinciden es que aún le queda un largo camino por recorrer.

6 Problemas de SSL (y algunas formas de solventarlos)

SSL es el protocolo seguro más usado y con más implementaciones útiles que existe en la actualidad. Eso es un hecho. Pero como hemos ido viendo a lo largo de este documento, ni es el mejor en todo ni el más idóneo para todas las circunstancias. El secreto de su éxito no ha estado sólo en su bondad, sino también en su versatilidad, su facilidad de implementación y su oportunidad (o lo que se llama estar en el momento justo y el lugar apropiado). A lo largo de los próximos dos capítulos veremos algunas cosas que SSL no hace bien o no hace en absoluto. Problemas, errores frecuentes, debilidades y, en algunos casos, formas más o menos costosas de solventarlas o aliviarlas.

6.1 Problemas con Otros Protocolos de la Capa de Transporte

Como hemos dicho anteriormente, SSL se coloca entre la capa de transporte y la de sesión y soporta, de forma transparente, la práctica totalidad de protocolos que se sitúan sobre el mismo: HTTP, FTP, IMAP, LDAP, etc. Pero, ¿que ocurre por debajo de el? Desgraciadamente no está concebido para trabajar con otros protocolos de la capa de transporte diferentes de TCP y, sobre todo, no orientados a conexión, algunos de ellos muy usados hoy en día (como UDP) y otros que han tenido su momento aunque ahora caigan en el desuso (como es el caso de IPX).

La extensa utilización de UDP ha propiciado, no obstante, soluciones más o menos eficientes que permiten, alejándose algo del estándar, utilizar SSL sobre UDP. El truco consiste en crear una sesión convencional de SSL sobre TCP en la que se realiza la negociación de parámetros y posteriormente cifrar los paquetes UDP con el fruto de esta negociación. Una de las características de UDP es que cada paquete ha de ser independiente de los demás y, por tanto, el hecho de que uno de ellos no llegue a destino no debe de influir en la transmisión. Cada paquete UDP, por tanto, debe de ser descifrable de forma independiente y debe de ir cifrado con su propia clave.

Existen otras implementaciones aún menos ortodoxas de solventar este problema pero que se escapan del objetivo y las posibilidades de este trabajo.

6.2 No Repudio de Transacciones

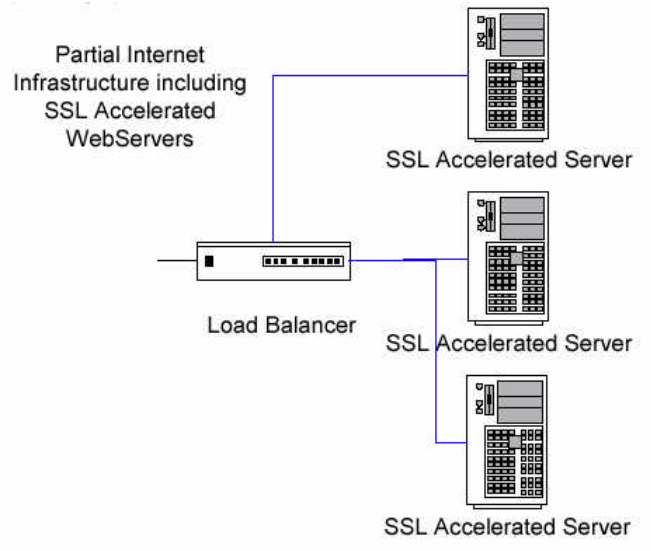
¿Qué es exactamente el no repudio de transacciones? Imaginemos que un comprador envía un mensaje pidiendo un producto a un vendedor. Posteriormente el comprador se retracta y dice que el jamás realizó esa petición. Un protocolo implementa No Repudio de mensajes cuando permite ante casos como este que el vendedor demuestre inequívocamente ante una tercera parte que el comprador y sólo el emitió el mensaje causa de litigo.

El estándar SSL no implementa esta característica, sin embargo su soporte sería muy sencillo en el caso de una comunicación SSL en la que ambas partes utilizan certificados con validez para firma electrónica. Basta, en este caso, con que ambas partes firmen todos sus mensajes antes de que sean cifrados con SSL. Las firmas serían comprobadas inmediatamente después de recibir cada mensaje por la parte opuesta y la sesión SSL sería abortada caso de recibir una firma inválida en cualquiera de los mensajes de forma similar a la utilizada por el protocolo S/MIME. Una alternativa válida a esto es la utilización de S/MIME sobre una conexión SSL.

6.3 Falta de Eficiencia

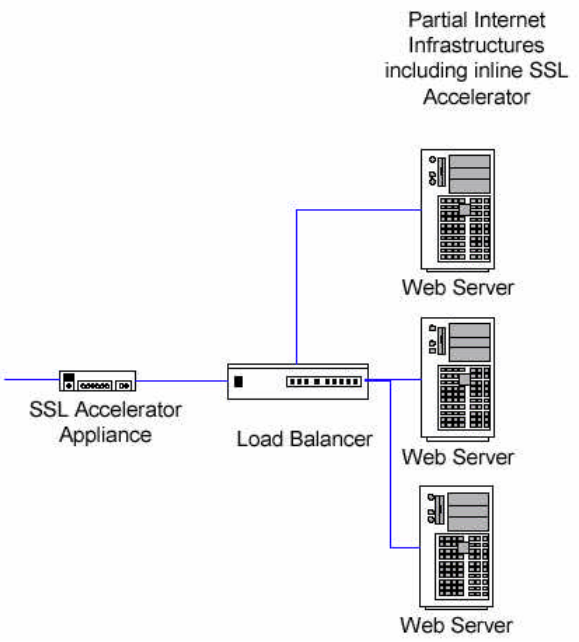
La seguridad casi siempre es un concepto opuesto a la eficiencia. Este es uno de los más dolorosos problemas que SSL debe de soportar. Entablar una sesión SSL es muy costosa en recursos, en particular durante el establecimiento de la conexión. Existen, no obstante, varias soluciones más o menos costosas para que esto no sea un problema. Cachear las sesiones para obviar la parte más costosa del proceso de establecimiento de una conexión a los clientes que han realizado una conexión previa (como se ha explicado en el punto 4 de este documento) es, quizás, la menos costosa, pero es una solución poco general que funciona correctamente con HTTP pero no con otros protocolos. Usar hardware dedicado para funciones criptográficas y balanceadores de carga más o menos sofisticados (algunos de ellos con hardware acelerador integrado en los mismos) es, quizás, la solución más eficiente para esto.

Tres son las estrategias más comúnmente utilizadas. La más sencilla consiste en utilizar tarjetas aceleradoras en los servidores objeto del tráfico SSL. Está solución puede ser válida en entornos reducidos (un sólo servidor) y muy estáticos (sin previsibles cambios de tecnología futura), pero se trata, posiblemente, de la solución menos eficiente y escalable a largo plazo. En la siguiente figura tenemos un esquema:



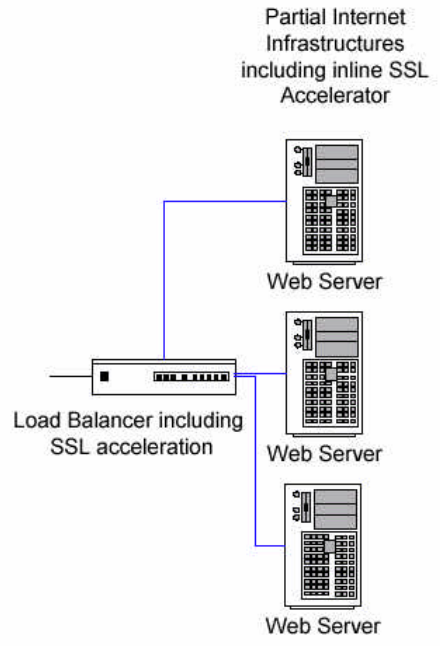
Los inconvenientes de este esquema son muchos: necesitamos un acelerador (y un certificado si es el caso) para cada servidor implicado, el coste de mantenimiento se multiplica por el número de servidores usados y perdemos la habilidad de SSL de renegociar rápidamente las conexiones interrumpidas. Un problema adicional es la dependencia del hardware: ante un posible cambio de plataforma en los servidores habrá que volver a realizar también una nueva inversión en las tarjetas cifradoras.

Para solucionar estos inconvenientes comenzaron a surgir dispositivos externos y autónomos dedicados exclusivamente al cifrado y descifrado SSL. Estos dispositivos actúan como terminador de la conexión SSL con el cliente, de forma que la comunicación entre el servidor y dicho dispositivo se realiza sin cifrar. El esquema más simple lo tenemos en la siguiente imagen:



Se trata de una solución mucho más económica y eficiente para grandes entornos corporativos. Es, además, independiente del hardware que utilicemos y centralizamos el mantenimiento de la configuración y control de la conexión SSL y la mayoría de los dispositivos actuales son ampliables en cuanto a potencia de cálculo se refiere, con lo cual se convierten en soluciones altamente escalables. Sin embargo, este esquema no nos resuelve aún el problema de la reanudación de sesiones interrumpidas puesto que el balanceador de carga recibe los mensajes sin información de la sesión SSL, con lo que no puede rutarlo hacia el servidor correcto.

Para solucionar este último punto han surgido nuevos dispositivos que integran el balanceo de carga con el cifrado SSL. En la siguiente figura podemos ver un esquema de este tipo de arquitecturas:



Al integrar el balanceador de carga y el cifrado/descifrado SSL, el propio dispositivo balanceador es capaz de descifrar el mensaje y tomar la decisión correcta en el encaminamiento del mismo hacia el servidor adecuado.

La eficiencia, al menos en los grandes entornos, ha dejado de ser un problema para SSL. Con las soluciones más modernas de esta tercera generación de dispositivos se logran tiempos de respuesta reales iguales a los obtenidos contra los mismos servidores usando HTTP plano y sin cifrado.

6.4 Errores en las implementaciones de SSL

Como veremos más adelante, en el capitulo 8 de este documento, la mayoría de los problemas que se han registrado en las diversas implementaciones de SSL a lo largo de su historia no se deben al diseño del protocolo en si, sino a errores en las implementaciones del mismo o en las aplicaciones que lo usan. En cualquier caso, una máxima que todos conocen en el terreno de la seguridad es que la falsa sensación de seguridad es muy peligrosa y no debemos de caer en el error de creernos a salvo por utilizar un protocolo que todos conocemos seguro despreciando que la explotación de un *buffer overflow* en nuestra implementación de SSL puede provocar que nuestro sistema sea vulnerable.

7 Debilidades de SSL

La versión 2.0 de SSL poseía muchas debilidades que la versión 3.0 se apresuró a corregir: una débil construcción de los MAC, errores en el mecanismo de *HandShake*, etc. Aunque, como decimos, la gran mayoría de estos errores están corregidos, la gran mayoría de nuestros navegadores y muchos servidores de Internet siguen aceptando, por motivos de compatibilidad, la versión 2.0 de este protocolo.

La versión 3.0 de SSL es suficientemente robusta para aguantar la gran mayoría de los ataques y la mayoría de las debilidades que analizaremos brevemente en este apartado no son universales, sino que afectan a determinadas implementaciones más o menos extendidas, del popular protocolo. Para más detalles al respecto, el capítulo siguiente del presente documento refleja un historial de las vulnerabilidades conocidas en las distintas implementaciones de este protocolo.

7.1 Almacenamiento de las Claves Privadas del Servidor SSL

Es claro y evidente que sin seguridad física no es posible la seguridad. Ningún servicio o servidor resiste un ataque físico si se tiene acceso al mismo porque, en último extremo, siempre es posible inutilizarlos o destruirlos. un servidor SSL no podía ser menos ya que la clave privada del mismo debe de residir físicamente en algún lugar del mismo.

Es evidente que las claves privadas del servidor deben de residir físicamente en algún sitio del mismo, pero ¿donde y de que forma protegidas? Una solución hardware (en forma de tarjeta aceleradora o dispositivo externo) es posiblemente la solución más segura puesto que todos podemos manejarnos con soltura en un sistema UNIX pero una clave que resida de alguna forma en uno de estos dispositivos es mucho más difícil de recuperar por alguien que no conozca la tecnología concreta del mismo.

En los casos en que no es posible una solución hardware no existen muchas protecciones posibles puesto que la clave privada del servidor SSL debe de usarse sin cifrar en la memoria del servidor durante el proceso de negociación de conexiones y un usuario que pueda acceder a la máquina con privilegios de root puede obtener un volcado y hacerse fácilmente con la misma. No podemos prevenirnos en modo alguno contra este tipo de ataques. Lo que si podemos evitar es que la clave resida sin cifrar en el disco del servidor, lo cual facilitaría mucho las cosas a un hipotético atacante: la clave debe de permanecer cifrada en el disco del servidor y protegida mediante una sólida frase-password que deberá de ser facilitada en el arranque del servicio por el administrador del sistema.

7.2 Recuperación del Contenido de Sesiones Previas

Hemos visto que, bajo algunas circunstancias, existen formas de robar la clave privada de un servidor SSL. Lo normal es que si sospechamos que dicha clave ha sido comprometida la sustituyamos inmediatamente para evitar que la confidencialidad del tráfico generado desde y hacia el servidor SSL atacado quede comprometida. Pero ¿que ocurre si previamente hemos grabado el tráfico de las conexiones con dicho servidor? Pues que, una vez con la clave secreta en nuestro poder, es posible descifrar la totalidad de los mensajes enviados en ellas. Para evitar esto, SSL posee un mecanismo que le permite usar claves efímeras diferentes por cada nueva sesión entablada.

7.3 Deficiencias en la Generación Aleatoria de Números

SSL como muchos otros sistemas de cifrado necesita apoyarse en sistemas de generación de números pseudo-aleatorios que, en gran medida, son la base de la robustez del sistema. Los PRNG's (*Pseudo Random Numbers Generators*) son algoritmos determinísticos capaces de producir números que, en apariencia, son totalmente aleatorios. Pieza clave en este tipo de sistemas es lo que se denomina semilla. Una semilla es una entrada que alimenta a un PRNG de forma que si esta es conocida la salida del PRNG se vuelve predecible. La semilla, por tanto, ha de ser también un número aleatorio para que el PRNG sea eficaz. Si la generación de números pseudo-aleatorios de una implementación de SSL no es lo suficientemente buena, el sistema funcionará pero las claves obtenidas de esta forma serán fácilmente obtenibles por una tercera parte malintencionada.

La mayoría de los dispositivos y acelerador SSL por hardware de la actualidad van acompañados de excelentes PRNG's que obtienen la semilla de entrada de forma electrónica. Las implementaciones por software usan diferentes trucos para recoger información de eventos externos que afectan a la máquina para convertirlos en la semilla de entrada del algoritmo. Existen, incluso, algunos modelos de máquinas UNIX que incluyen un dispositivo dedicado especialmente a generar números aleatorios para todas las funciones del sistema.

Para quien desee profundizar en estos sistemas, existe una extraordinaria librería para proporcionar números aleatorios seguros denominada EGADS (*Entropy Gathering and Distribution System*) diseñada por John Kelsey y John Viega y distribuida libremente para sistemas Windows y Unix desde la siguiente URL:

http://www.securesoftware.com/download_egads.htm

7.4 Longitud de las Claves

Hace apenas un par de años que los EE.UU. relajaron las leyes de exportación de algoritmos criptográficos. En Marzo de 2002 (aproximadamente un año después) Netcraft, la empresa líder en sondeos y estadísticas de servidores web, realizó un estudio de los servidores SSL existentes en Internet que usaban claves potencialmente vulnerables y la localización geográfica de los mismos. El resultado de este estudio desveló que alrededor del 18% de los servidores analizados era vulnerable a este tipo de ataques. En nuestro país la cifra ascendía a más del 30%. A continuación tenemos un resumen de la tabla obtenida en el estudio.

País	**Porcentaje de Servidores SSL vulnerables**
Canada	13,5%
EE.UU.	15,1%
Reino Unido	26,5%
España	31,9%
Francia	41,1%

Las causas de esta renuencia a usar claves mayores son varias. Por un lado está la ya mencionada ley de exportación de los EE.UU. Pero por otro, y según todos los expertos mucho más importante, está que usar claves más largas empobrece el tiempo de respuesta obtenido por los usuarios y encarece la inversión en hardware de la empresa que da el servicio mientras que el uso de claves cortas (e inseguras) es difícilmente apreciado por la inmensa mayoría de los usuarios.

7.5 Certificados y Listas de Revocación

En el año 2001 la empresa Verisign, posiblemente la más popular de las entidades emisoras de certificados digitales, emitió y firmó varios certificados que supuestamente iban destinados a la empresa Microsoft pero que en realidad no fueron a parar a sus manos. Para atajar las consecuencias de este tipo de errores, entre otras cosas, existen las listas de revocación de certificados o CRL's. Las CRL's son listas donde las autoridades de certificación consignan los números de serie de los certificados que, por uno u otro motivo, han dejado de ser válidos y confiables de forma que cualquier cliente de dichos servicios pueda verificarlos. No obstante, la tecnología de las PKI's no está en la actualidad lo suficientemente madura y a distribución de CRL's es, precisamente, uno de sus puntos más débiles de forma que puede pasar mucho tiempo desde que un certificado es comprometido hasta que la totalidad de los clientes tienen cuenta de ello.

7.6 Interceptando los mensajes de Client y Server Hello

Es relativamente sencillo interceptar los primeros mensajes intercambiados en el mecanismo de *Handshake* y manipularlos para lograr que la sesión SSL se establezca en condiciones óptimas para su posterior escucha, haciendo creer al cliente que, por ejemplo, el servidor sólo soporta la versión 2.0 del protocolo, longitudes de clave de 40 bits o que el único algoritmo común es el DES.

La única forma de impedir esto es configurar nuestros respectivos clientes y servidores para impedir que se establezca una sesión SSL bajo estas condiciones. SSL 3.0 o TLS 1.0 deben de ser los únicos protocolos permitidos por nuestros servidores y navegadores para establecer un canal seguro. Las longitudes de las claves han de ser, ahora que la restricción de los EE.UU. ha sido levantada, de 128 bits y RC4 y 3DES los algoritmos más 'débiles' cuyo uso debemos de permitir.

7.7 ¿Un Sniffer para SSL?

En diciembre de 1999 aparece dsniff, una colección de herramientas desarrolladas por Du Song con el objeto de realizar labores de auditoría sobre una red, tests de penetración, etc. Este poderoso conjunto de herramientas es capaz de robar la información de sesiones SSL incluso en un entorno de switchs. DSniff se distribuye originalmente para una gran diversidad de sistemas operativos Unix/Linux aunque existen también versiones disponibles para sistemas windows e incluso Mac.

- Página del autor y versiones para Unix/Linux:
  http://monkey.org/~dugsong/dsniff/
- Versión para Windows:
  http://www.datanerds.net/~mike/dsniff.html
- Versión para MAC OsX:
  http://blafasel.org/~floh/ports/osx/dsniff-2.3.osx.tgz

En Mayo de 2000, Eu-Jin Goh y Dan Boneh, dos estudiantes de la Universidad de Standford, desarrollaron un nuevo sniffer para SSL (versiones 2.0 y 3.0) y TLS. La finalidad de esta herramienta es educativa

- Página del proyecto:
  http://crypto.stanford.edu/~eujin/sslsniffer/

En agosto de 2002 aparece sslsniff, otro sniffer para SSL desarrollado por Mike Benham y que a partir de su versión 0.4 se distribuye con un certificado válido firmado por una autoridad de certificación. Se trata de una herramienta similar a dsniff capaz de robar una sesión SSL incluso en un entorno de switches .

- Página de sslsniff:
  http://www.thoughtcrime.org/ie.html

8 Historial de Vulnerabilidades en SSL

En este capítulo se pretende realizar una relación exhaustiva de las vulnerabilidades encontradas en el protocolo SSL o en las implementaciones que lo utilizan a l.o largo de toda su historia. Las principales fuentes de documentación han sido las inestimables y completas listas del CERT y de Security Focus. No obstante, aparecen en ella enlaces a otras fuentes de información cuando se considera necesario o especialmente aclaratorio.

La lista aparece ordenada cronológicamente.

8.1 1995

- En septiembre de 1995 se publica por primera vez una advertencia sobre la posibilidad de romper en un tiempo computacional moderado las claves de 40 bits usadas por algunos de los algoritmos de la versión 2.0 de SSL. El problema reside en el generador de números aleatorios usado por el navegador (PRNG) y los artífices del descubrimiento fueron David Wagner e Ian Golberg quienes lograron obtener la clave de sesión en unas pocas horas usando para ello un simple ordenador de los de aquella epoca. En el enlace de ftp que se acompaña aparece el código C que utilizaron para ello.

  http://online.securityfocus.com/advisories/454
  ftp://ftp.csua.berkeley.edu:21/pub/cypherpunks/cryptanalysis/unssl.c

8.2 1998

- Enero de 1998. Varios errores que posibilitan provocar un buffer overflow en distintos módulos de Apache (cfg_getline, mod_proxy, logresolve) permitirían leer las claves SSL que permanecen sin cifrar en la memoria del servidor web.

  http://online.securityfocus.com/advisories/1355

- En junio de 1998, el CERT (Coordination Center) publica un extenso report denunciando una vulnerabilidad en algunas implementaciones de productos que usan PKCS#1 (Public-Key Cryptography Standard #1) de RSA que permitiría obtener información confidencial de una sesión cifrada mediante SSL. PKCS#1 se usa durante la fase de negociación de la sesión SSL.

  http://www.cert.org/advisories/CA-1998-07.html
  http://online.securityfocus.com/advisories/260

http://online.securityfocus.com/advisories/265
http://online.securityfocus.com/advisories/274
http://online.securityfocus.com/advisories/1115
http://online.securityfocus.com/advisories/1311

8.3 1999

- En agosto de 1999, L0pht denuncia a través de Security Focus que pueden explotarse ciertas debilidades en el diseño del protocolo IRDP (ICMP Router Discovery Protocol) para realizar ataques de diferentes tipos sobre un host, y en particular en el caso que nos atañe posiblitar un ataque de Man-in-the-Middle sobre un servidor SSL.

  http://online.securityfocus.com/advisories/1675
  http://online.securityfocus.com/advisories/1685

- En noviembre de 1999 se publica en Security Focus un exploit que permite tirar un servidor Netscape Enterprise Server a través de un bug en el código del mecanismo de Handshake del servicio SSL.

  http://online.securityfocus.com/advisories/1828

- En diciembre de 1999 se publica un error en la implementación SSL del Internet Information Server (IIS) de Microsoft en sus versiones 3.0 y 4.0 que permitiría obtener texto en claro dentro de una conexión SSL

  http://online.securityfocus.com/advisories/1897

- En diciembre de 1999 se publica una nueva vulnerabilidad causada por un error en la implementación de ciertas librería de RSA. El problema sólo afectaba a la versión de EE.UU. de las mismas.

  http://online.securityfocus.com/advisories/1896
  http://online.securityfocus.com/advisories/1981
  http://online.securityfocus.com/advisories/1968
  http://online.securityfocus.com/advisories/1914
  http://online.securityfocus.com/advisories/1988

8.4 2000

- En febrero de 2000, el CERT publica una advertencia sobre la posibilidad de la inclusión de determinados scripts maliciosos en una página html generada dinámicamente que provocan el envío de información hacia un determinado servidor sin que el usuario lo advierta. En una conexión SSL, si se logra

introducir el script antes de que la conexión sea establecida entre cliente y servidor, podemos usarlo para robar información de dicha conexión o para enviar información maliciosa al servidor SSL

http://www.cert.org/advisories/CA-2000-02.html
http://online.securityfocus.com/advisories/2077
http://online.securityfocus.com/advisories/2082

- En marzo de 2000 se publican numerosos errores capaces de provocar un buffer overflow en Lynx, un popular navegador web que sólo admite texto disponible en diferentes versiones para varios sistemas operativos y con posiblidad de entablar sesiones SSL que se ven afectadas por esta vulnerabilidad. Las versiones afectadas son las anteriores a la 2.8.3pre.5.

  http://online.securityfocus.com/advisories/2127

- En mayo de 2000 el CERT publica un error que afecta a la forma de validación de una sesión SSL por parte del navegador de Netscape. El problema afecta a las versiones del navegador iguales o inferiores a la 4.72. El error permite que el navegador acepte un certificado inválido al entablar una sesión SSL sin mostrar un mensaje de advertencia al usuario.

  http://www.cert.org/advisories/CA-2000-05.html
  http://online.securityfocus.com/advisories/2211
  http://online.securityfocus.com/advisories/2218
  http://online.securityfocus.com/advisories/2217
  http://online.securityfocus.com/advisories/2238

- También en mayo de 2000 se publica un nuevo error en los navegadores de Netscape que permitiría a un posible atacante explotar nuevos errores en el tratamiento y validación de los certificados por parte del mismo. En esta ocasión la vulnerabilidad afecta a la versión 4.73 e inferiores del navegador.

  http://www.cert.org/advisories/CA-2000-08.html
  http://online.securityfocus.com/advisories/2255
  http://online.securityfocus.com/advisories/2303

- En junio de 2000 se descubre un problema similar al anterior pero que, en esta ocasión, afecta al navegador de Microsoft (Internet Explores). En esta ocasión el error desvela que cuando realizamos una conexión sobre un servidor SSL a través de un link situado en un frame o en una imagen, el Internet Explorer verifica que el certificado recibido ha sido firmado por una autoridad válida pero, incomprensiblemente, no comprueba si el nombre del servidor que lo envía corresponde con el que aparee en el certificado ni si la fecha de validez del certificado ha expirado. El problema afecta a las versiones 4.x y 5.x del

navegador y fue corregido a partir de la versión 5.5 o mediante Services Packs para algunas de las versiones anteriores.

http://www.cert.org/advisories/CA-2000-10.html
http://www.microsoft.com/technet/security/bulletin/ms00-039.asp
http://online.securityfocus.com/advisories/2305
http://online.securityfocus.com/advisories/2304

- En junio de 2000 se publica una nueva vulnerabilidad en la versión 4.72 de Netscape Navigator que puede comprometer la información de una sesión SSL. El problema es corregido por la versión 4.73 del navegador.

  http://online.securityfocus.com/advisories/2322

- Junio de 2000. Se detecta la omisión de las librerias de generación de números pseudo aleatorios en algunas versiones del sistema operativo FreeBSD para máquinas basadas en procesadores Alpha de Digital. El problema puede afectar de forma impredecible a cualquier aplicación que realice cifrados y utilice esas librerías, entre ellas todas las implementaciones de SSL. Las versiones de FreeBSD para plataformas Intel no estaban afectadas.

  http://online.securityfocus.com/advisories/2323

- Agosto de 2000. Se detectan dos errores en STRONG, el componente que realiza las veces de servidor web en el servidor PKI de NAI (Network Associates Inc.). Por un lado, la posibilidad de explotar una vulnerabilidad de buffer overflow sobre dicho servidor web y por otro la posibilidad de que un cualquier usuario se conecte al puerto 444 del servicio PKI sin que se requiera su autenticación.

  http://online.securityfocus.com/advisories/2473

- En octubre de 2000 se denuncia la posibilidad de explotar un buffer overflow sobre las versiones de curl y curl-ssl distribuidas con la versión 2.2 de Debian GNU/Linux. Curl es un popular cliente que facilita la obtención de ficheros y documentos de un servidor, diseñado para trabajar sin necesidad de interacción con el usuario.

  http://www.linuxsecurity.com/advisories/debian_advisory-804.html
  http://online.securityfocus.com/advisories/2750
  http://online.securityfocus.com/advisories/2752

- Octubre de 2000. Una nueva vulnerabilidad en las versiones 4.0 y 5.0 de IIS (Internet Information Server) permite que, bajo un muy restringido conjunto de circunstancias, un usuario malintencionado pueda robar a otro el control de una sesión SSL contra el servidor afectado.

http://online.securityfocus.com/advisories/2766

- En noviembre de 2000 se descubre un error en el servicio LDAP sobre SSL de las implementaciones que acompañan a las versiones 5.2, 6.x y 7.0 de Red Hat Linux.

  http://online.securityfocus.com/advisories/2752
  http://online.securityfocus.com/advisories/2835

- En diciembre de 2000 se encuentra un error en todas las implementaciones VPN de la empresa VPNet. El cliente de dichas implementaciones. el cual usa SSL para negociar las conexiones, responde en claro en alguno de los mensajes que deberían de ir cifrados desvelando información sensible.

  http://online.securityfocus.com/advisories/2946

- Para cerrar el año 2000 se descubren varios errores (uno de ellos con la disponibilidad de un exploit remoto) en las versiones inferiores a la 3.9 de stunnel, un popular paquete que ofrece un servicio de conexión SSL genérico para 'envolver' con el otros protocolos TCP, tales como pop3, ldap, etc.

  http://online.securityfocus.com/advisories/2975
  http://online.securityfocus.com/advisories/2985
  http://online.securityfocus.com/advisories/3024

8.5 2001

- En abril de 2001 se desvelan un grupo de potenciales problemas de seguridad en OpenSSL, la más popular de las implementaciones libres de SSL usadas, entre otros, por los servidores web Apache. Las versiones de OpenSSL anteriores a la 0.9.6.a sufren de importantes errores en el mecanismo de intercambio de claves de la versión 3.0 del protocolo, errores en la generación de números pseudo aleatorios, errores en los permisos de ciertos archivos sensibles, etc.

  http://online.securityfocus.com/advisories/3447
  http://online.securityfocus.com/advisories/3475

- En mayo de 2001 se publica en Security Focus una interesante discusión sobre las debilidades existentes en el mecanismo de números de secuencia iniciales usados por TCP y sobre como un protocolo para securizar un canal de comunicaciones que actúa sobre el (como SSL o TLS) se ve afectado por este error y no así un protocolo como IPSec que actúa bajo el en la capa 3 del modelo OSI.

http://online.securityfocus.com/advisories/3277

- En junio de 2001 se denuncia un bug sobre el paquete SSL de fetchmail susceptible de ser explotado de forma remota.

  http://online.securityfocus.com/advisories/3382
  http://online.securityfocus.com/advisories/3502

- En junio de 2001 asimismo se detecta un error en la implementación de LDAP sobre SSL de los productos de Microsoft.
  http://www.microsoft.com/technet/security/bulletin/MS01-036.asp

- En agosto de 2001 se detecta la posibilidad de explotar un buffer overflow wn el paquete de Telnet sobre SSL versión 0.16.3-1 distribuido junto con Debian GNU/Linux de nombre clave 'Potato'

  http://online.securityfocus.com/advisories/3497
  http://online.securityfocus.com/advisories/3500

- En septiembre de 2001 CISCO System hace pública una vulnerabilidad en la implementación de SSL de la versión 2.0 de su librería iCND mediante la cual es posible establecer una sesión SSL con el servidor afectado sin necesidad de contar con un certificado válido en el cliente a pesar de que este sea requerido por el servidor. La vulnerabilidad es corregida en la versión 2.0.1 de dicha implementación.

  http://www.cisco.com/warp/public/707/SSL-J-pub.html
  http://online.securityfocus.com/advisories/3562

- Octubre de 2001. Se detecta un error en w3m y w3m-ssl, las funciones que manejan las cabeceras de MIME para la implementación de este protocolo y distribuidas con la versión 2.2 de Debian Linux.

  http://online.securityfocus.com/advisories/3597

- En octubre de 2001 se denuncia una vulnerabilidad en la implementación de SSL en Stronghold, un servidor web basado en Apache y distribuido junto con algunas versiones de Red Hat Linux.

  http://securescannx.vigilante.com/tc/17227

- En diciembre de 2001 se deteca una nueva vulnerabilidad en STunnel.

  http://www.linuxsecurity.com/advisories/other_advisory-1769.html

http://online.securityfocus.com/advisories/3768

8.6 2002

El año 2002 merece una especial atención por ser el año negro de SSL. Un error en la implentación de ciertas librerías de OpenSSL, usadas por Apache Web Server, provocó la aparición de Slapper, un gusano que se propagó inmediatamente por la red de igual forma a como había ocurrido el año anterior con CodeRed, el gusano que usaba para difundirse un error en los servidores web de Microsoft.

- En febrero de 2002 se descubre un error en la librería mod_ssl de OpenSSl, utilizada por Apache.

  http://www.apache-ssl.org/advisory-20020301.txt
  http://online.securityfocus.com/advisories/3900
  http://online.securityfocus.com/advisories/3908
  http://online.securityfocus.com/advisories/3957
  http://online.securityfocus.com/advisories/3916
  http://online.securityfocus.com/advisories/3936
  http://online.securityfocus.com/advisories/3938
  http://archives.neohapsis.com/archives/bugtraq/2002-02/0313.html

- En julio de 2002 se encuentra un nuevo error sobre la misma librería mencionada anteriormente (mod_ssl).

  http://online.securityfocus.com/advisories/4253
  http://online.securityfocus.com/advisories/4310
  http://online.securityfocus.com/advisories/4309
  http://online.securityfocus.com/advisories/4321
  http://online.securityfocus.com/advisories/4314
  http://online.securityfocus.com/advisories/4316
  http://online.securityfocus.com/advisories/4324
  http://online.securityfocus.com/advisories/4340
  http://online.securityfocus.com/advisories/4364

- En agosto de 2002 se denuncia un error que afecta a la librería de SSL que maneja Konqueror, el navegador web del popular escritorio de Linux KDE. El error provoca que el mencionado navegador no verifique al establecer una conexión SSL si el certificado del Servidor es válido para la actividad que realiza contentándose con recibir un certificado firmado por una autoridad válida, independientemente de la actividad que se autorize en el.

  http://online.securityfocus.com/advisories/4411
  http://online.securityfocus.com/advisories/4435
  http://online.securityfocus.com/advisories/4461

- Agosto de 2002. Se denuncia una vulnerabilidad en la implementación de SSL de Internet Explorer que permite realizar facilmente un ataque de Man-in-the-Middle. Las versiones afectadas son las 5.x y 6.x de este navegador.

  http://www.thoughtcrime.org/ie-ssl-chain.txt

- Agosto de 2002. Hewlett Packart advierte de que sus sistemas HP Tru64 UNIX & HP OpenVMS se ven afectados por los errores encontrados en la implementación de las librerías de OpenSSL.

  http://online.securityfocus.com/advisories/4442

- Septiembre de 2002. Importante error en la validación de certificados realizada por Microsoft y que afecta a todas las versiones de sus sistemas operativos (exceptuando a windows95) y las implementaciones de sus navegadores web y sistemas de correo para ordenadores Mac.

  http://www.microsoft.com/technet/security/bulletin/MS02-050.asp

- Septiembre de 2002. El día 16 aparece la anunciada primera versión de Slapper, un gusano que se difunde automáticamente por la red a través de la vulnerabilidad de OpenSSL denunciada el 30 de julio y que afecta a la gran mayoría de implentaciones del servidor web Apache, el más popular de la red.

  Slapper utiliza para introducirse en la máquina un exploit que provoca un desbordamiento de buffer en las implementaciones de Apache Web Server sobre máquinas Linux. Slapper es capaz de realizar ataques distribuidos de denegación de servicio a través del puerto UDP 2002 de las máquinas infectadas. Su peligrosidad es muy baja puesto que su código no posee ningún payload dañino.

  Durante los siguientes días aparecieron tres nuevas variantes de este virus que se detectaron por primera vez los días 23 y 24 de septiembre y el día 1 de octubre respectivamente. Actúalmente no se encuentra en circulación.

  http://online.securityfocus.com/advisories/4496
  http://www.nipc.gov/warnings/advisories/2002/02-006.htm
  http://www.cert.org/advisories/CA-2002-23.html
  http://www.cert.org/advisories/CA-2002-27.html
  http://isc.incidents.org/analysis.html?id=167
  http://www.openssl.org/news/secadv_20020730.txt
  http://www.linuxandmain.com/modules.php?name=News&file=article&sid=161
  http://www.unam-cert.unam.mx/Boletines/Boletines2002/boletin-UNAM-CERT-2002-023.html

http://bvlive01.iss.net/issEn/delivery/xforce/alertdetail.jsp?oid=21184
http://www.f-secure.com/slapper/
http://www.pandasoftware.es/virus_info/enciclopedia/verficha.aspx?idvirus=37317
http://online.securityfocus.com/archive/1/291748/2002-09-11/2002-09-17/1
http://online.securityfocus.com/archive/1/291782/2002-09-11/2002-09-17/1
http://www.quands.info/alertes/html/openssl020914.html
http://dammit.lt/apache-worm/

- En septiembre de 2002 aparece una nueva vulnerabilidad en las librerias de OpenSSL usadas por Apache.

  http://online.securityfocus.com/advisories/4502

- Octubre de 2002. Varios errores que son susceptibles de ser explotados mediante buffers overflows que afecta an fetchmail-ssl.

  http://online.securityfocus.com/advisories/4520
  http://online.securityfocus.com/advisories/4540

- En octubre de 2002 se detecta un nuevo error en el módulo mod_ssl de la librería OpenSSL usada por Apache susceptible de posibilitar un ataque por Cross-Site Scripting. El Croos-Site Scripting es una técnica mediante la cual es posible conseguir que un servidor web legítimo envíe una página web falsa que contenga código malintencionado como respuesta a una petición del cliente.

  http://online.securityfocus.com/advisories/4591
  http://online.securityfocus.com/advisories/4632
  http://online.securityfocus.com/advisories/4774

- Noviembre de 2002. Nuevos errores en la validación de certificados por parte de la implementación SSL del escrtitorio KDE.

  http://online.securityfocus.com/advisories/4688
  http://online.securityfocus.com/advisories/4740

9 Conclusiones

Hay un hecho evidente y es que en el futuro cada vez habré más necesidad de contar con redes de comunicaciones seguras. Lo que no es tan fácil de prever es por cuanto tiempo mantendrá SSL su supremacía (en cuanto a universalidad, que no en cuanto a calidad) en estos terrenos. Lo lógico sería pensar que en el terreno de las comunicaciones seguras de propósito general se vea desplazado en algún momento por IPSec y en el terreno específico de las aplicaciones de comercio electrónico por SET u otro protocolo diseñado a tal efecto. El momento en el que esto ocurrirá es mucho más difícil de prever.

Por el momento y lo que hemos tratado de reflejar en este documento, SSL se mantiene en 'la cresta de la ola' porque no existe ninguna otra opción mejor. En el terreno de los protocolos de propósito general tenemos tres grandes grupos: los que son más restrictivos que SSL en cuanto a posibles aplicaciones (como S-HTTP), los que no aportan nada nuevo a lo que ya hace bien SSL y lo poco que aportan no decide a su favor la balanza debido al peso que la gran implantación de SSL le proporciona (como PCT o el mismo TLS) y los que aportando mejores mecanismos y más seguridad se ven desfavorecidos por su mayor coste de implantación y mantenimiento (lo cual es el caso de IPSec).

En el campo de los protocolos específicos para el comercio electrónico es la mayor complejidad (necesaria en su mayor parte) de los mismos lo que los mantiene aún en un segundo plano pese a su evidente idoneidad. Una mayor madurez de los sistemas de PKI's y una forma de distribución universal (que quizás se produzca con la generalización de los documentos de identidad digitales) es lo que parecen necesitar estos sistemas para su definitivo despegue. En cualquier caso, y si no cambia extraordinariamente el panorama en unos pocos meses, todo hace pensar que SSL cumplirá 10 años de vida con excelente salud y siendo aún el protocolo seguro más extendido en las redes de comunicaciones.

10 Referencias y Bibliografía Recomendada

La bibliografía que aquí se cita es muy interesante para ampliar los contenidos del presente documento. Parte de ella ha sido de inestimable ayuda para la confección del mismo.

Se ha dado una especial relevancia a la información disponible de forma libre y gratuita en Internet. Es mucha y buena la información que puede obtenerse en este medio sobre SSL y está escrita, en muchos casos, por personal muy vinculado directamente con el desarrollo del mismo. La práctica totalidad de la documentación usada para la confección de este documento se ha recogido a través de estas fuentes.

Como especial recomendación para quien haya echado en falta en este documento una descripción más técnica de SSL le recomiendo encarecidamente la lectura de las especificaciones del protocolo [2] proporcionadas por Netscape. Quien desee profundizar en las debilidades del mismo tiene a su disposición dos excelentes documentos: uno de Wagner y Schneier [3] y otro de Krawczyk [10].

Debido al formato de la página algunas direcciones aparecen cortadas y, por lo tanto, imposibilitan el acceso a la web correspondiente. Si esto sucede, unificar la URL en una sola línea antes de usarla en el navegador.

10.1 Bibliografía

1. E. Rescorla. *SSL and TLS, Designing and Building Secure Systems*. Addison-Wesley. 2000.
2. S. Feit. *TCP/IP, Arquitectura, protocolos, implementación y seguridad*. Osborne MCGraw-Hill. 1997.
3. W.R. Stevens. *TCP/IP Illustrated Volume 1: The Protocols.* Addison-Wesley. 1994.
4. J.Viega, M.Messier and P.Chandra. *Network Security with OpenSSL, Cryptography for Secure Communications.* O'Really. 2002.

10.2 RFC's

1. D. Eastlake, B. Boesch, S. Crocker and M. Yesil. *Cybercash Credit Card Protocol Version 0.8*.
   http://www.ietf.org/rfc/rfc1898.txt
2. T. Dierks and C. Allen. *The TLS Protocol Version 1*.0
   http://www.ietf.org/rfc/rfc2246.txt
3. C. Newman. *Using TLS with IMAP, POP3 and ACAP*
   http://www.ietf.org/rfc/rfc2595.txt

4. A. Schiffman and E. Rescorla. *The Secure Hypertext Transfer Protocol*. http://www.ietf.org/rfc/rfc2660.txt
5. A. Medvinsky, M. Hur. *Addition of Kerberos Cipher Suites to Transport Layer Security (TLS)*. http://www.ietf.org/rfc/rfc2712.txt
6. B. Aboba, D. Simon. *PPP EAP TLS Authentication Protocol*. http://www.ietf.org/rfc/rfc2716.txt
7. R. Khare, S. Lawrence. *Upgrading to TLS Within HTTP/1.1*. http://www.ietf.org/rfc/rfc2817.txt
8. E. Rescorla. *HTTP Over TLS*. http://www.ietf.org/rfc/rfc2818.txt
9. P. Hoffman. *SMTP Service Extension for Secure SMTP over Transport Layer Security*. http://www.ietf.org/rfc/rfc3207.txt
10. P. Chown. *Advanced Encryption Standard (AES) Ciphersuites for Transport Layer Security (TLS)*. http://www.ietf.org/rfc/rfc3268.txt

10.3 URL's

1. Introduction to SSL. http://developer.netscape.com/docs/manuals/security/sslin/contents.htm
2. *SSL 3.0 Especifications*. http://wp.netscape.com/eng/ssl3/
3. David Wagner and Bruce Schneier. *Analysis of the SSL 3.0 protocol*. http://www.cs.berkeley.edu/~daw/papers/ssl3.0.ps
4. Adam Shoctack. *An Overview of SSL (version 2)*. http://www.homeport.org/~adam/ssl.html
5. Allen Jones. *SSL Demystified*. http://www.windowswebsolutions.com/Articles/Index.cfm?ArticleID=16047
6. *Understanding Encryption and SSL*. http://developer.netscape.com/docs/manuals/enterprise/mngserv/security.htm
7. *How SSL Works*. http://developer.netscape.com/tech/security/ssl/howitworks.html
8. John Viega, Matt Messier and Pravir Chandra. *Seven Common SSL Pitfalls* http://www.onlamp.com/lpt/a/2493
9. John Viega, Matt Messier and Pravir Chandra. *Network Security with OpenSSL. Chapter 1: Introduccion*. http://www.oreilly.com/catalog/openssl/chapter/ch01.pdf

10. Hugo Krawczyk. *The order of encryption and authentication for protecting comunications (Or: how secure is SSL?).*
http://eprint.iacr.org/2001/045/
11. Ran Canetti and Hugo Krawczyk, *Analysis of Key-Exchange Protocols and Their Use for Building Secure Channels.*
http://eprint.iacr.org/2001/040/
12. Laurent Demailly. *Netscape (in) Security (problems).*
http://www.demailly.com/~dl/netscapesec/
13. Luciano Moreno. *Transacciones Seguras. SSL y SET*.
http://www.htmlweb.net/seguridad/ssl/ssl_1.html
14. Security Elements of Internet Transactions
http://commerceworld.ed.hkedcity.net/e_comm/security.php
15. *Apache SSL*.
http://www.apache-ssl.org/
16. *mod_SSL, The Apache Interface to OpenSSL*.
http://www.modssl.org/
17. *Secure Electronic Environment (S.E.E.) PKI Paper. The SSL/TLS handshake*.
http://www.e-government.govt.nz/docs/see-pki-paper-11/chapter14.html
18. *RSA Security. Secure Socket Layer & Transport Layer Security*.
http://www.rsasecurity.com/standards/protocols/ssl_tls.html
19. Nicko van Someren. *The Risks of Short RSA Keys for secure communications using SSL*.
http://www.securitytechnet.com/resource/rsc-center/vendor-wp/nCipher/wp_short_keys.pdf
20. Shannon Appel. *SSL FAQ*.
http://www.faqs.org/faqs/computer-security/ssl-talk-faq/
21. S. Henson. *OpenSSL PKCS#12 FAQ v1.81*
http://www.drh-consultancy.demon.co.uk/pkcs12faq.html
22. Justin Davies. *TLS CA and server key HOWTO*
http://palmcoder.net/files/howtos/SSL%20CA%20HOWTO/SSL_CA-HOWTO.html
23. *Stunnel, Universal SSL Wrapper*.
http://www.stunnel.org/
24. *SSLeay Programmer Reference*.
http://psych.psy.uq.oz.au/~ftp/Crypto/ssl.html
25. *SSLeay FAQ*.
http://psych.psy.uq.oz.au/~ftp/Crypto/
26. Gonzalo Álvarez Marañón. *Medios de pago en Internet: La necesidad de un canal seguro*.
http://www.iec.csic.es/criptonomicon/comercio/mediospago.html
27. Adam Shostack. *An Overview of SHTTP*.
http://www.homeport.org/~adam/shttp.html

28. J. Benaloh, B. Lampson, D. Simon, T. Spies and B. Yee. *Microsoft Corporation's PCT Protocol*.
http://activex.adsp.or.jp/english/specs/pct.htm
29. *IPSec Versus "Clientless" VPNs for Remote Access*.
http://www.checkpoint.com/products/downloads/ipsecwp.pdf
30. *SET Specification Books*.
http://www.setco.org/set_specifications.html
31. *SET Comparative Performance Analysis*.
http://www.setco.org/download/setco6.pdf
32. Peter Wayner. *CyberCash's Lesson in Web Survival*.
http://www.nytimes.com/library/tech/98/08/biztech/articles/10cybercash.html
33. Gonzalo Álvarez Marañón. *SET a fondo*.
http://www.idg.es/iworld/articulo.asp?id=103068&sec=iworld
34. *SSL versus SET*.
http://www-3.ibm.com/software/webservers/commerce/payment/cryptset.html
35. *IPsec and SSL; Complementary Solutions*.
http://www.securitytechnet.com/resource/rsc-center/vendor-wp/nortel/nn102260-110802.pdf
36. Muhammad Saad Ahsan and Travis Creason. *Set vs. SSL*.
http://islab.oregonstate.edu/koc/ece478/proj/2002RP/CA.pdf
37. Don Larson. *The Race to Secure Cyberspace*.
http://www.webdeveloper.com/security/security_race_cyberspace.html
38. *OpenSSL*.
http://www.openssl.org/
39. Lutz Jänicke. *Postfix/TLS - A TLS extension for POSTFIX*.
http://www.aet.tu-cottbus.de/personen/jaenicke/postfix_tls/doc/index.html
40. Patrick Koetter. *Adding TLS support to Postfix*.
http://postfix.state-of-mind.de/patrick.koetter/smtpauth/configuration/tls.html
41. Justin Davies. *Postfix SSL HOWTO*.
http://palmcoder.net/files/howtos/Postfix%20SSL/Postfix_SSL-HOWTO.html
42. Rick Kaseguma. *SSLWRAP*
http://www.quiltaholic.com/rickk/sslwrap/
43. *Choosing the Right SSL Acceleration Device*.
http://www.f5.com/solutions/archives/whitepapers/choosingssl.html
44. *Application Traffic Management*.
http://www.f5.com/solutions/tech/ApplicationTrafficManagement/

45. *SSL Accelerator Performance: Determining metrics and limiting factors*. http://www.simpleaccess.com/site_files/PDFs/SSLPerf.pdf
46. *F5's FIPS-Certified SSL Acceleration Products: Providing Top Level Security For Web Applications and Services*. http://www.securitytechnet.com/resource/rsc-center/vendor-wp/nCipher/F5_FIPS_White_Paper.pdf
47. *SSL in the E-commerce Saddle*. http://www.securitytechnet.com/resource/rsc-center/vendor-wp/nokia/ssl_in_the_e-commerce_saddle.pdf
48. *IP Clustering: Enhancing E-commerce SSL Acceleration*. http://www.securitytechnet.com/resource/rsc-center/vendor-wp/nokia/ssl_ipclustering.pdf
49. *Measuring SSL Accelerator Performance*. http://www.securitytechnet.com/resource/rsc-center/vendor-wp/nokia/ssl_accelerator_performance.pdf
50. *BIG-IP FIPS SSL Accelerator*. http://www.f5.com/f5products/bigip/FIPSSSL/
51. *BIG-IP SSL Accelerator*. http://www.f5.com/f5products/bigip/SSL400800/
52. *BIG-IP 2000 IP Application Switch*. http://www.f5.com/f5products/bigip/2000/
53. *BIG-IP Blade Controller*. http://www.f5.com/f5products/bigip/BladeController/BIGIPBLADE.html
54. *AEP SureWare Runner 1000 / 1000 IDE / 2000 Overview*. http://www.aep.ie/products_sure_runner.htm
55. *AEP SureWare Runner S1000 Overview*. http://www.aep.ie/products_sure_runner_s1000.htm
56. *NITROX, Security Macro Proccessors for SSL Acceleration*. http://www.cavium.com/nitrox-overview.htm

10.4 Foros de Discusión, Listas de Correo y Grupos de Noticias

1. *SSL-Talk, Secure Sockets Layer Discussion List.* Foro inactivo desde noviembre de 1998 pero que contiene mucha y valiosa información. http://groups.yahoo.com/group/ssl-talk/
2. *News sobre Criptografía de Netscape-Mozilla.* news://netscape.public.mozilla.crypto/
3. *BugTraq, Security Focus Mailing Lists.* http://online.securityfocus.com/cgi-bin/sfonline/subscribe.pl
4. *CERT Advisory Mailing List.* http://www.cert.org/contact_cert/certmaillist.html