"Protocolo TLS"

Redes de Computadores I ELO-322

Universidad Técnica Federico Santa María
Departamento de Electrónica

Grupo de Trabajo:	Javier Navarro	201021056-1
	Gilian Ubilla	201021013-8
	Mario Tejeda	201021049-9
Profesor:	Agustín González	

28 de julio de 2014

Resumen

El siguiente informe trata sobre el proyecto final correspondiente a la asignatura Redes de Computadores I. El tema abordado es un protocolo que permite la transferencia segura de datos: Transport Layer Security (TLS). Para comenzar se abordarán aspectos rudimentarios sobre seguridad en redes de computadoras. Luego se hará un breve paso por la evolución del protocolo, desde sus inicios hasta llegar a lo que actualmente se utiliza. Finalmente se explicará el funcionamiento del protocolo para luego mostrar el proceso de comunicación entre cliente y servidor, el cual será contrastado con un caso práctico a través de Wireshark.

Introducción

A medida que las redes de computadoras comenzaron a desarrollarse, se buscaron nuevas aplicaciones para dar uso a esta vasta herramienta y pronto fue necesario contar con servicios que permitieran dar seguridad a los datos que eran compartidos entre hosts. El correo electrónico sirve como un claro ejemplo. Las cuentas pueden ser utilizadas para transferir información de gran importancia y la escaces de seguridad puede provocar que algún tercero mal intencionado intercepte los mensajes compartidos y lleve a cabo acciones de suplantamiento o robo de información. Fue así como en 1995 la empresa de software Netscape Communications desarrolló SSL 1.0, para aplicaciones seguras en transacciones comerciales electrónicas. Las fallas encontradas en las primeras versiones darían paso a sucesivas mejoras y a una estandarización como protocolo cuyo resultado se convirtió en lo que hoy es conocido como Transpor Layer Securitiy.

¿Qué es TLS y por qué es necesario?

TLS (Transport Layer Security) es un protocolo de comunicación , en el cual, se establece una conexión segura entre cliente y servidor por medio de un canal cifrado. Este ha evolucionado de su antecesor SSL (Secure Sockets Layer) ambos creados por Netscape[1].

El intercambio de información con TLS/SSL se realiza en un entorno seguro y libre de ataques. La última propuesta de estándar está documentada en la referencia RFC 2246.

TLS/SSL son ampliamente necesarios en el tránsito de información debido:

- Prevenir escuchas (eavesdropping).
- Evitar la falsificación de la identidad del remitente Mantener la integridad del mensaje en una aplicación cliente-servidor.
- Establecer una conexión segura entre dos partes.
- Aplicaciones distintas deben poder intercambiar parámetros criptográficos sin necesidad de que ninguna de las dos conozca el código de la otra.
- Permitir la incorporación de nuevos algoritmos criptográficos.
- Eficiencia. Los algoritmos criptográficos son costosos computacionalmente, por lo que el protocolo incluye un esquema de cache de sesiones para reducir el número de sesiones que deben inicializarse desde cero.

Cifrado y encriptación.

Antes de adentrar a TLS, es necesario comprender ciertos aspectos básicos en la seguridad en el proceso de comunicación en las redes, para ello se definen ciertos conceptos básicos tales como llaves públicas y privadas y certificados digitales.

Se ha de definir la criptografía como la creación de técnicas para el cifrado de datos. Teniendo como objetivo conseguir la confidencialidad de los mensajes.

En criptografía, el cifrado es un procedimiento que utiliza un algoritmo de cifrado con cierta clave (clave de cifrado) transformando un mensaje, sin atender a su estructura lingüística o significado, de tal forma que sea incomprensible o, al menos, difícil de comprender a toda persona que no tenga la clave secreta (clave de descifrado) del algoritmo.

El juego de caracteres (alfabeto) usado en el mensaje sin cifrar puede no ser el mismo que el juego de caracteres que se usa en el mensaje cifrado.

Aunque el cifrado pueda volver secreto el contenido de un documento, es necesario complementarlo con otras técnicas criptográficas para poder comunicarse de manera segura. Puede ser necesario garantizar la integridad la autenticación de las partes.

Cifrado simétrico y asimétrico.

Un sistema de cifrado simétrico es un tipo de cifrado que usa una misma clave para cifrar y para descifrar. Las dos partes que se comunican mediante el cifrado simétrico deben estar de acuerdo en la clave a usar de antemano. Una vez de acuerdo, el remitente cifra un mensaje usando la clave, lo envía al destinatario, y éste lo descifra usando la misma clave.

[1]Netscape Communications Corporation es una empresa de software famosa por ser la creadora del navegador web Netscape Navigator.

La compañía fue fundada como Mosaic Communications Corporation el 4 de abril de 1994 por Marc Andreessen y Jim Clark

El principal problema con los sistemas de cifrado simétrico no está ligado a su seguridad, sino al intercambio de claves. Una vez que el remitente y el destinatario hayan intercambiado las claves pueden usarlas para comunicarse con seguridad, pero ¿qué canal de comunicación que sea seguro han usado para comunicar la clave entre ellos? Sería mucho más fácil para un atacante intentar interceptar una clave que probar las posibles combinaciones del espacio de claves. Otro problema es el número de claves que se necesitan. Si tenemos un número n de personas que necesitan comunicarse entre ellos, entonces se necesitan n(n-1)/2 claves para cada pareja de personas que tengan que comunicarse de modo privado. Esto puede funcionar con un grupo reducido de personas, pero sería imposible llevarlo a cabo con grupos más grandes.

Los sistemas de cifrado de clave pública se inventaron con el fin de evitar por completo el problema del intercambio de claves. Un sistema de cifrado de clave pública usa un par de claves para el envío de mensajes. Las dos claves pertenecen a la misma persona a la que se ha enviado el mensaje. Una clave es pública y se puede entregar a cualquier persona. La otra clave es privada y el propietario debe guardarla para que nadie tenga acceso a ella. El remitente usa la clave pública del destinatario para cifrar el mensaje, y una vez cifrado, sólo la clave privada del destinatario podrá descifrar este mensaje.

Este protocolo resuelve el problema del intercambio de claves, que es inherente a los sistemas de cifrado simétricos. No hay necesidad de que el remitente y el destinatario tengan que ponerse de acuerdo en una clave. Todo lo que se requiere es que, antes de iniciar la comunicación secreta, el remitente consiga una copia de la clave pública del destinatario. Es más, esa misma clave pública puede ser usada por cualquiera que desee comunicarse con su propietario. Por tanto, se necesitarán sólo n pares de claves por cada n personas que deseen comunicarse entre ellas.

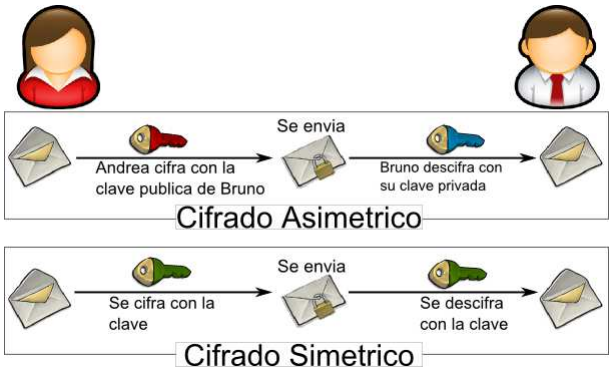


Figura 1: Cifrado Simétrico y Asimétrico

Certificado digital y firma digital.

El Certificado Digital es el único medio que permite garantizar técnica y legalmente la identidad de una persona en Internet. Se trata de un requisito indispensable para que las instituciones puedan ofrecer servicios seguros a través de Internet. Además:

El certificado digital permite la firma electrónica de documentos. El receptor de un documento firmado puede tener la seguridad de que éste es el original y no ha sido manipulado y el autor de la firma electrónica no podrá negar la autoría de esta firma.

El certificado digital permite cifrar las comunicaciones. Solamente el destinatario de la información podrá acceder al contenido de la misma.

Un Certificado Digital consta de una pareja de claves criptográficas, una pública y una privada, creadas con un algoritmo matemático, de forma que aquello que se cifra con una de las claves sólo se puede descifrar con su clave pareja.

El titular del certificado debe mantener bajo su poder la clave privada, ya que si ésta es sustraída, el sustractor podría suplantar la identidad del titular en la red. En este caso el titular debe revocar el certificado lo antes posible, igual que se anula una tarjeta de crédito sustraída.

Otra utilidad de los Certificados Digitales es que posibilitan el envío de mensajes cifrados: utilizando la clave pública de un Certificado, es posible cifrar un mensaje y enviarlo al titular del Certificado, quien será la única persona que podrá descifrar el mensaje con su clave privada.

Ya con estos conceptos es posible que el lector se sumerja en TLS y su funcionamiento.

Evolución de TLS

SSL fue desarrollado por Netscape con el fin de habilitar la posibilidad de transacciones comerciales seguras en la web. En 1994 se consigue la versión SSL 1.0, sin embargo, esta no es liberada y solo circula a nivel interno de la empresa debido a deficiencias importantes en sus servicios, como no entregar protección de integridad en los datos y no utilizar número de secuencia durante un intercambio lo que permitía realizar ataques de replicación.

En 1995 luego de corregir algunas fallas, se entrega SSL 2.0. A pesar de estas mejoras, aun hay fallas importantes que provocan que sea considerado un protocolo desaprobado para su uso en aplicaciones de seguridad. Una falla importa se presentan durante la etapa de Handshake, en la cual no se aplica cifrado por lo que un atacante puede engañar al usuario para que se elija un método de cifrado poco efectivo.

En 1996 se entrega SSL 3.0 como un rediseño completo del protocolo. Esta versión añade mejores opciones de cifrado y habilita la autenticación de certificados. Las mejoras en seguridad permiten que se considere como un protocolo aprobado en aplicaciones de seguridad.

En 1999, tras un esfuerzo por la IETF, se estandarizó el protocolo resultando el RFC 2264 el cual pasó a ser conocido como TLS 1.0, una actualización de SSL 3.0 (aveces mencionado como SSL 3.1). Si bien los cambios no son excesivos se especifica que hay problemas de compatibilidad entre TLS 1.0 y SSL 3.0, pero que el protocolo implementa un modo de operación que al restar servicios de seguridad lo permite.

En 2006 y 2008 se presentan TLS 1.1 y TLS 1.2 respectivamente para añadir mejoras de seguridad en consecuencia de fallas detectadas y mejoras en los distintos tipos de ataque. Actualmente TLS 1.2 es el protocolo por defecto en gran parte de los buscadores más utilizados. La figura 2 muestra el soporte de TLS en algunos navegadores.

Navegador	TLS 1.0	TLS 1.1	TLS 1.2
Chrome 0-22	Sí	No	No
Chrome 22-29	Sí	Si	No
Chrome 30-(35)	Sí	Si	Sí
Firefox 2-	Sí	No	No
Firefox 27-(30)	Sí	Sí	Sí
IE 9	Sí	No	No
IE 10	Sí	Deshabilitada	Deshabilitada
IE 11	Sí	Sí	Sí
Opera 10-(23)	Sí	Deshabilitada	Deshabilitada
Safari 5-6	Sí	No	No
Safari 7	Sí	Sí	Sí

Figura 2: Soporte de TLS en navegadores

Funcionamiento de TLS

El protocolo TLS tiene multitud de aplicaciones en uso actualmente. La mayoría de ellas son versiones seguras de programas que emplean protocolos que no lo son. Hay versiones seguras de servidores y clientes de protocolos como HTTP, SMTP, IMAP y POP3.

El protocolo TLS se ejecuta en una capa entre los protocolos de aplicación como: HTTP sobre SSL/TLS es HTTPS, ofreciendo seguridad a páginas web para aplicaciones de comercio electronico, utilizando certificados de clave pública para verificar la identidad de los extremos. Visa, MasterCard, American Express y muchas de las principales instituciones financieras han aprobado SSL para el comercio sobre Internet. SSH utiliza TLS por debajo. SMTP y NNTP pueden operar también de manera segura sobre TLS. POP3 e IMAP4 sobre TLS son POP3S e IMAPS. TLS también puede ser usado para tunelizar una red completa y crear una red privada virtual (VPN), como en el caso de OpenVPN [1].

Además, el protocolo se subdivide en 2 capas (o protocolos) principales, la capa Handshake TLS y la capa de registro TLS. La capa handshake se ubica entre la capa de registro y la aplicación y es encargada de generar mensajes correspondientes a cada situación. La capa de registro se ubica sobre la capa de transporte y recibe los mensajes desde la capa de Handshake y los acondiciona para ser enviados.

Protocolo handshake TLS

El protocolo de Handshake es el que permite a ambas partes autenticarse mutuamente y negociar el cifrado antes de intercambiar datos, para posteriormente establecer la conexión. A continuación, se detalla el protocolo de negociación para una nueva sesión de conexión a iniciar, en general, entre un browser (cliente) y un servidor.

Client Hello: El cliente envía un mensaje al servidor para dar inicio a una conexión cifrada, especificando una lista de conjuntos de cifrados, métodos de compresión soportados, la versión del protocolo SSL/TSL más baja permitida y la versión SSL/TLS más alta soportada. Además, se envía una cadena de 32 bytes aleatorios que sirven para generar la clave simétrica (más adelante en el protocolo). También, se incluye un identificador de sesión (ID), en caso de intentar "retomar" una sesión establecida con anterioridad y hacer un handshaking más corto

Server Hello: El servidor contesta al cliente escogiendo un cifrado, enviando ID de sesión correspondiente, la versión de TLS a utilizar en la conexión, el tipo de compresión de datos y otra cadena de 32 bytes aleatorios

Certificate: El servidor ofrece su certificado a ser verificado por el cliente

Server Key Exchange: Con este mensaje el servidor ofrece para el cifrado asimétrico entre cliente y servidor la clave pública firmada con la clave del certificado.

Server Hello Done: El servidor da por concluida su fase de negociación asimétrica.

Client Key Exchange: El cliente, tras haber comprobado y validado el certificado, genera el premaster secret (48 bytes) cifrado con la clave pública del servidor y se lo envía. Luego, cliente y servidor pueden generar el master secret aplicando funciones hash a la cadena random de 32 bytes y al premaster secret, usado para el cifrado simétrico de datos.

Change Cipher Spec: Con este mensaje el cliente informa que sus mensajes sucesivos estarán cifrados con el cifrado simétrico acordado.

Finished: El cliente da por finalizada su fase de negociación asimétrica. El mensaje que finaliza el handshake incluye un hash de todos los mensajes de handshake que garantiza la integridad de la comunicación.

Change Cipher Spec: El servidor descifra con su clave privada el premaster secret enviado por el cliente y genera por su cuenta el master secret. Desde este momento, cliente y servidor han logrado establecer una clave simétrica. Con este mensaje el servidor indica que sus mensajes desde este momento serán cifrados simétricamente.

Finished: El servidor finaliza su fase de negociación asimétrica. Se verifica todo el proceso de handshake haciendo un hash a todos los mensajes concatenados anteriores. Con este protocolo de negociación

finalizado, se logra crear un canal bajo el protocolo TLS que garantiza que todos los datos enviados por el protocolo de aplicación serán cifrados

A continuacion, se muestra la captura de paquetes de una sesión TLS usando Wireshark, para visualizar el envío de paquetes en una negociación. En particular, se usa el protocolo HTTPS para la conexión a www.google.cl; en donde el browser Google Chrome usa la versión TLS 1.2 por defecto[2].

Source	Destination	Protocol	Puerto src	Length	Info
192.168.0.16	173.194.42.232	TLSv1.2	51178	269	Client Hello
173.194.42.232	192.168.0.16	TLSv1.2	https	1484	Server Hello
173.194.42.232	192.168.0.16	TLSv1.2	https	1059	Certificate
192.168.0.16	173.194.42.232	TLSv1.2	51178	172	Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
173.194.42.232	192.168.0.16	TLSv1.2	https	292	New Session Ticket, Change Cipher Spec, Encrypted Handshake Message
173.194.42.232	192.168.0.16	TLSv1.2	https	103	Application Data
173.194.42.232	192.168.0.16	TLSv1.2	https	91	Application Data
192.168.0.16	173.194.42.232	TLSv1.2	51178	103	Application Data

Figura 3: Protocolo de negociación TLS

Como se observa de la figura 3, la conexión abre un puerto con conexión segura (51148) y algunos pasos de la negociación se encapsulan en un solo paquete, dada su poca longitud. Luego de la negociación, se empieza la conexión mediante el canal TLS con el paso de los datos por medio de TCP. El paquete Certificate de parte del servidor, aparte del envío de su certificado , también envía los mensajes de tipo Server Key Exchange y Server Hello Done

Protocolo de registro TLS

Mientras el protocolo de handshake se encarga de negociar los parámetros de seguridad, es en la capa de registro donde se realizan las operaciones de formación de cada registro con sus campos correespondientes, fragmentado, compresión y cifrado que aseguran los datos, mediante el MAC. Todo el trabajo que está a cargo del registro TLS es totalmente transparente para la mayoría de aplicaciones. Este protocolo también es responsable de identificar los diferentes tipos de mensajes (handshake, alert, cambio de cifrado o datos), asi como la obtención y verificación de la integridad de cada mensaje

Normalmente los datos de un registro corresponden a un mensaje de la subcapa superior, pero también es posible juntar en un mismo registro dos o más mensajes, siempre que todos pertenecen al tipo indicado por content_type. También puede pasar que un mensaje se fragmente en diversos registros, si su longitud es superior a 16384 bytes antes de comprimir.

El código de autenticación de mensaje (MAC) es calculado y adherido, mientras que los datos son encriptados usando el cifrado negociado. Una vez completados estos pasos, los datos cifrados se transmiten a la capa transporte mediante TCP. En el extremo receptor, se aplican los mismo pasos pero a la inversa; descifrar datos utilizando el cifrado negociado, verificar MAC, extraer y entregar los datos a la aplicación. La figura 4 muestra la estructura de un mensaje de datos en TLS, en este caso los datos como el MAC y Padding vienen encriptados, sin embargo se identifica el tipo de dato de por el campo content_type

```
⊟ Secure Sockets Layer
  ⊟ TLSv1.2 Record Layer: Application Data Protocol: http
      Content Type: Application Data (23)
      Version: TLS 1.2 (0x0303)
      Length: 32
      Encrypted Application Data: 5481973f9fe38c7f3030e63de9d7899f648ae291c4e50a6e...
```

Figura 4: Mensaje de aplicación TLS usando HTTP

[2]En el Anexo, se detalla en profundidad la captura de la negociación

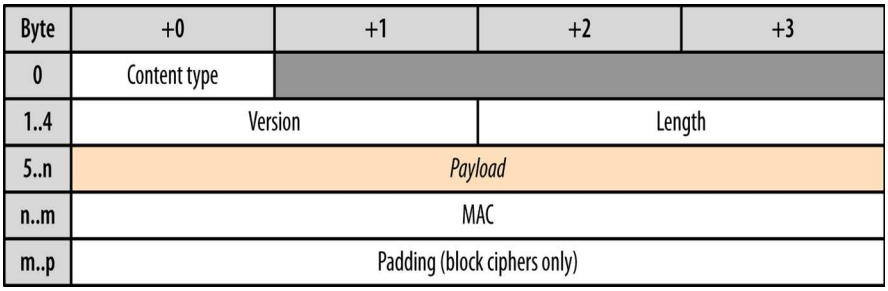


Figura 5: Registro TLS

El registro genérico de protocolo TLS (figura 5)define los 4 posibles valores en el campo content_type. El tipo Handshake (content_type=22) es el que se usa en el establecimiento de la conexión cliente-servidor; cada paso del handshaking tiene un nombre asociado y a su vez un valor en el campo "Message type". El tipo ChangeCipherSpec (content_type=20) es enviado por el cliente y servidor para notificar que los siguientes registros estarán protegidos bajo un nuevo tipo de cifrado. El tipo Alert (content_type=21) es para manejo de errores, los cuales pueden ser un "mensaje de aviso", para el fin normal de la conexión, o un "error fatal" . Un error fatal provoca el fin de la conexión de manera abrupta y la invalidación del identificador de sesión; son ejemplos de errores fatales un MAC incorrecto, tipo de mensaje inesperado, error de negociación, etc. El tipo Application (content_type =23) contiene los datos cifrados de la aplicación, llevándolos a la capa de transporte. El campo Version especifica la versión del protocolo que se utiliza. El campo Length, el largo del mensaje. El campo Payload contiene el mensaje enviado por el protocolo correspondiente. El campo MAC se utiliza para llevar a cabo un seguimiento de la conversación entre cliente y servidor con el fin de evitar la intervención de terceros. El campo Padding solo contiene data cifrada, la cual puede corresponder a datos cifrados de la capa de aplicación.

Conclusiones y Comentarios

Se concluye que TLS es un protocolo de gran importancia para muchas aplicaciones en la red. Es un estándar en comunicación segura y los navegadores y servidores lo utilizan por defecto.
Gracias a los servicios que proporciona los terminales en una red pueden autenticarse sin tener contacto directo y llevar a cabo operaciones de extrema importancia (como el pago de arancel universitario) sin temor a que terceros hagan mal uso de la información intercambiada (i.e. información de cuenta bancaria). Finalmente, si bien el protocolo busca proporcionar seguridad, siempre será posible romperla, sin embargo, las continuas mejoras que se han realizado permiten convertir esto en una tarea difícil.

Referencias

[1] http://deic.uab.es/material/26118-ssl.pdf. Protección de nivel de transporte: SSL/TLS/WTLS.

[2] http://lobobinario.blogspot.com/2011/02/analizando-ssl-ii-de-iii-aplicaciones-y.html. Analizando SSL (II de III): Aplicaciones y Estructura

[3] http://tools.ietf.org/html/rfc5246#section-4.RFC 5246. The Transport Layer Security (TLS) Protocol Version 1.2

[4] http://chimera.labs.oreilly.com/books/1230000000545/ch04.html#_testing_and_verification. Chapter 4: Transport Layer Security (TLS)

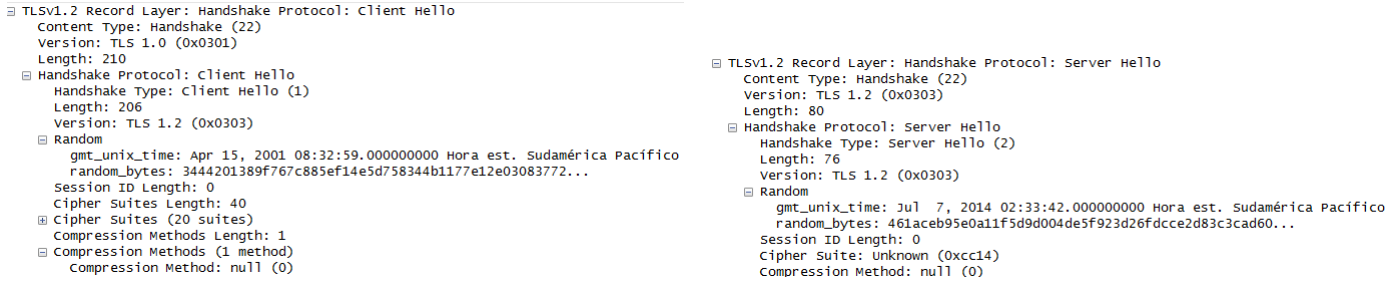
[5] http://portalae.sci.uma.es:8080/export/sites/default/uma/documentos/criptografia_certificado_digital_firma_d

Anexos

Anexo 1: Detalle del proceso de negociación TLS

Se muestra el detalle del handshake en una sesión TLS nueva, desde la petición del cliente para iniciar una conexión hasta el cifrado simétrico de los datos a enviar por el protocolo de capa aplicación. Los paquetes que se presentan son los mismos mostrados en la seccion "Protocolo Handshake TLS".

La figura 6 muetra la petición por parte del cliente para establecer una nueva sesión TLS, en la cual se detalla que el protocolo TLS 1.0 y TLS 1.2 son laa versiones mínima y máxima a aceptar para la conexión. Además, el tipo de mensaje Client Hello corresponde a 1 y de los 32 bytes aleatorios, hay 4 que corresponden a la hora exacta de la solicitud, esto cuenta como un sello de tiempo por motivos de seguridad. El identificador de sesion corresponde al cero y se ofrecen 20 suites de cifrados distintas a elección del servidor. Finalmente, ofrece solo un tipo de método de compresión, el nulo. El servidor contesta estos requerimientos con el tipo de versión 1.2 de TLS a utilizar, la cadena aleatoria de 32 bytes de respuesta, que incluye otra vez el sello de tiempo, y la elección de la suite de cifrado 0xcc14; el tipo de mensaje Server Hello corresponde al 2



(a) ClientHello (b) ServerHello

Figura 6: Negociación de parámetros

El servidor prosigue con el protocolo enviando su certificado de autenticidad , el envío de su clave pública y la finalización por parte suya del protocolo, con un Server Hello Done de 4 byte de longitud. Los valores de tipo de mensaje que identifican esta secuencia son 11, 12 y 14 respectivamente

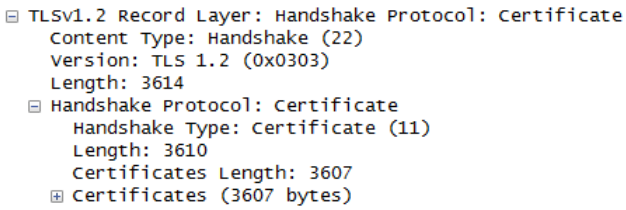


Figura 7: Envío de certificado por parte del servidor

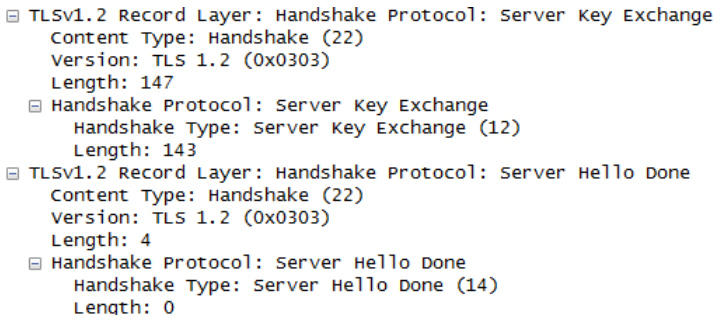
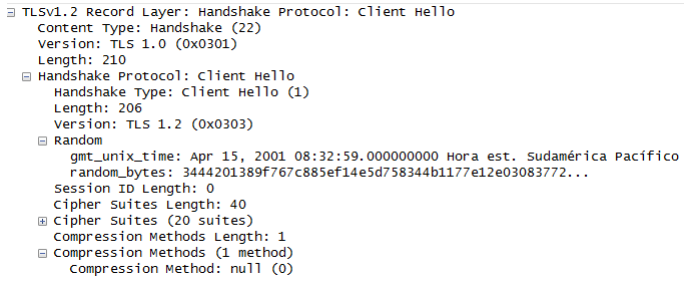
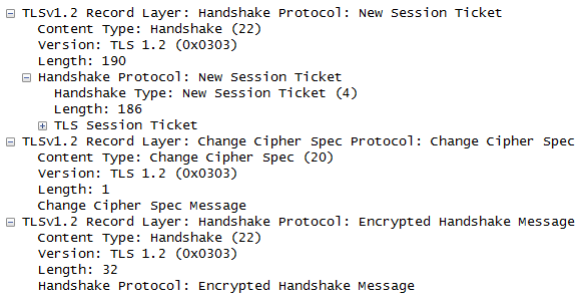


Figura 8: Server Key Exchange y Server Hello Done

En la figura 9 se observa por parte del cliente y servidor el intercambio de claves, con lo cual se pasa de un cifrado asimétrico a simétrico de datos. Notar que este cambio se identifica con el mensaje Change Cipher Spec de apenas 1 byte de longitud. El mensaje New Session Ticket es una extensión de TLS que le permite al cliente volver a iniciar una conexión TLS sin necesidad de rehacer el handshake completo nuevamente, con ello el servidor guarda una copia del estado de la sesión TLS en proxy



(a) Client Key Exchange y envío de premaster secret



(b) New Session Ticket y cambio de cifrado del servidor

Figura 9: Cambio a cifrado simétrico del cliente y servidor