Aplicación Facebook Messenger

- **Integrantes:**
  - Cristobal Carmona
  - Eduardo Reyes
  - José Cayo
  - Sebastian Ubiergo

- **Ramo:** [ELO322] Redes de computadores I
- **Profesor:** Agustín González

Resumen:

Facebook Messenger es una de las aplicaciones de mensajerías mas utilizadas hoy en día. Su creciente popularidad y nuestras ganas de aplicar lo aprendido en el curso fue motivación suficiente para darnos la tarea de investigar como es el funcionamiento de esta aplicación desde el punto de vista de Redes de Computadores. Las funcionalidades de la aplicación van desde mensajes sencillos de texto, hasta la realización de videollamadas. Además la aplicación funciona en múltiples sistemas. En el presente informe, principalmente nos concentramos en ver que protocolos utilizaba y en que capas se encontraban, utilizando la versión en escritorio de la aplicación, ya que fue mas sencillo trabajar con esta.. Así, descubrimos la existencia de el protocolo ***Transport Layer Security***, el cual es un protocolo intermediario entre la capa de aplicación y la capa de transporte, un protocolo criptográfico, que proporcionan comunicaciones seguras por una red, comúnmente Internet, y el cual es el mas importante que utiliza Facebook Messenger. Por eso entramos en detalle e investigamos el funcionamiento de sus subsprotocolos, así como el proceso de Handshaking entre estos.

Introducción:

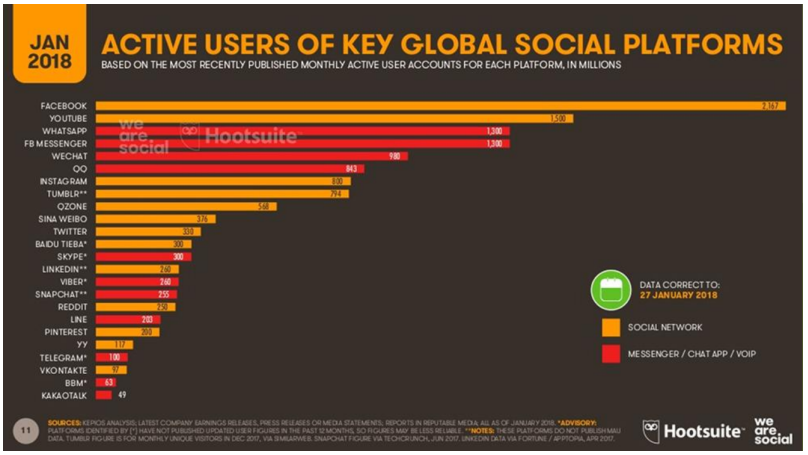

Figura 1: Estadísticas de uso de las redes sociales (Ver referencia 8).

Facebook Messenger es una de las plataformas sociales mas utilizadas en el mundo. Como se puede observar en el gráfico de arriba, solo es superado en usuarios por la plataforma de Facebook oficial y por Youtube, y está empatada en número de usuarios con Whatsapp,

otra aplicación de mensajería muy utilizada hoy en día. Al ser una aplicación muy utilizada, como grupo nos propusimos la tarea de investigar el como funciona por dentro, viendo que protocolos utiliza, aplicando los conocimientos aprendidos en clase.La aplicación es multi-plataforma,está disponible tanto en celulares (Android, iOS, etc) como en computadores de escritorios. Para efectos de investigación decidimos utilizar la versión de escritorio, ya que esta era mas fácil de analizar con el software *Wireshark*, así pudimos analizar los paquetes que se enviaban y recibian.

Facebook Messenger:

Como indica su nombre, es una aplicación de mensajería desarrollada por Facebook y lanzada el año 2011, con el fin de que los usuarios de Facebook pudieran comunicarse entre si a través de un chat completo, con muchas funcionalidades. Entre sus caracteristicass, además de poder enviar mensajes de texto, también tiene disponible el envió de imágenes, archivos, audios, emojis e incluso esta la opción de jugar minijuegos entre los participantes del chat, realizar videollamadas y planear eventos. Los chats que uno puede crear pueden ser 1 a 1, o muchos a muchos si se invitan mas personas a la conversación. También tiene disponible la opción de bloquear la comunación con algun usuario en especifico. Por ultimo, una característica importante de Facebook Messenger es que es multiplataforma (como se menciona en la introducción). Para poder enviar mensajes utiliza varios protocolos, de los cuales destacamos el mas importante, TLS (*Transport Layer Security*) que se describirá en detalle en las siguientes secciones.

TLS - Transport Layer Security:

El protocolo *Transport Layer Security* se utiliza entre la capa de aplicación y la de transporte para por encriptar los paquetes que viajan por el Internet. Para ello utiliza dos tipos de encriptación: Asimétrica y Simétrica.

Durante el inicio de una conexión se utiliza una encriptación asimétrica donde existen dos llaves, una que utiliza el cliente (llave pública) y otra el servidor (llave privada). A pesar de lo seguro de este sistema, resulta muy costo, por lo que solo se utilizará mientras el cliente y

el servidor se ponen de acuerdo para establecer una llave única y así pasar a una encriptación simetrica.

TLS - Subprotocolos:

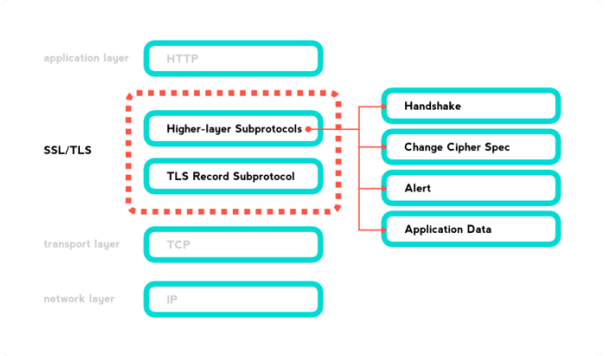

Figura 2: Capas del protocolo TLS (Ver referencia 7).

Como se observa en la figura 2, el protocolo TLS consta de dos subcapas: *TLS Record Subprotocol* y *Higher-layer Subportocols*.

TLS Record Subprotocol:

Esta capa se encuentra sobre TCP y está recibe los datos de la capa de aplicación, los fragmenta, comprime (opcional), agrega la autenticación del mensaje, encripta los datos y agrega el encabezado de TLS: Versión de TLS, tipo de registro de TLS y el largo de los datos.

Higher-layer Subportocols:

Esta capa trabaja bajo la capa de aplicación y está compuesta por 4 subprotocolos:

- **Handshake:** Aquí se autentifican los participantes en la conexión y se establece la llave que se utilizará para la encriptación simétrica.
- **Change Cipher Spec:** Se hace efectivo lo negociado en el *Hanshake* y se empiezan a cifrar los datos con la llave negociada.

- **Alert Protocol:** Si existe algún problema que afecte que afecte a la seguridad del protocolo se notifica con un mensaje de alerta.
- **Application Data Protocol:** Toma los datos de la capa de aplicación y los envía por el canal seguro.

TLS - Handshake:

En la siguiente imagen se puede apreciarla estructura del *Handshake* (Los mensajes con * son mensajes no siempre se envían):

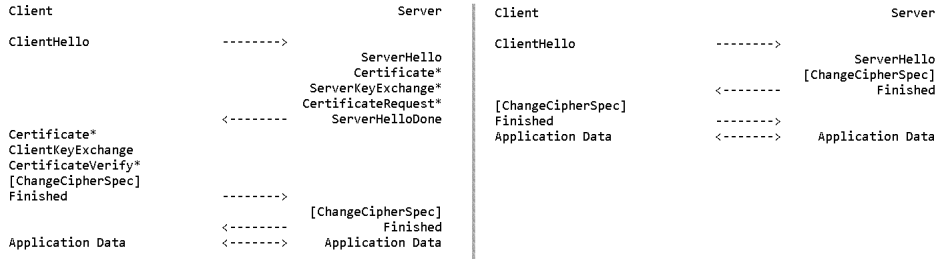

Figura 3: Capas del protocolo TLS: Izquierda es la versión completa y el de la derecha es la versión con solo los campos obligatorios (Ver referencia 6, páginas 36 y 37).

- **Client Hello:** Este mensaje comienza la negociación de handshake, se envía la lista de suites de cifrado que soporta el cliente para que el servidor elija la que mejor se adapte. También da la posibilidad de reiniciar una sesión previa a través de un campo SessionId.
- **Server Hello:** Es muy similar al Client Hello con la excepcion de que solo incluye un suite de cifrado y un método de compresión. También da la posibilidad al cliente de reiniciar una sesión previa a través del campo SessionId.
- **Certificate:** El cuerpo de este mensaje contiene una cadena de certificados de claves públicas. Las cadenas de certificados permiten que TLS admita jerarquías y PKIs (infraestructura de clave pública)
- **Server Key Exchange:** Este mensaje contiene los parámetros del algoritmo de intercambio de claves que el cliente necesita del servidor para poder trabajar posteriormente

con la encriptación simétrica.

- **Certificate Request:** Se usa cuando el servidor requiere autenticación de identidad del cliente. También indica qué autoridades de certificación se consideran confiables.
- **Server Hello Done:** Este mensaje finaliza la parte del servidor de la negociación del protocolo de enlace. No lleva información adicional.
- **Client Key Exchange:** Proporciona al servidor los datos necesarios para generar las claves para el cifrado simétrico. El formato de mensaje es muy similar a *Server Key Exchange*, ya que depende principalmente del algoritmo de intercambio de claves elegido por el servidor.
- **Certicate Verify:** El cliente envía este mensaje para demostrarle al servidor que posee la clave privada correspondiente a su certificado de clave pública.
- **Finish:** Este mensaje indica que la negociación de TLS está completa y suite de cifrado está activado. Ya se encuentra encriptado por lo que se envía después del mensaje de *Change Cipher Spec*.

Demostración Practica:

En la siguiente figura se observa que al momento de conectarse a Facebook Messenger, nuestro computador envío un *Client Hello*, a lo que el servidor respondió con un *Server Hello*, *Change Cipher Spec* y un *Encrypted Handshake Message* (que sería el mensaje *Finish*). Luego nuestro computador respondió con un *Change Cipher Spec* y un *Encrypted Handshake Message* para luego comenzar el intercambio de información de la aplicación mediante *Application Data*. Note que está es la versión que contiene solo los campos obligatorios que muestran en la figura 3.

```
37 3.659574225   10.112.15.177    179.60.193.38    TLSv1.2     583 Client Hello
38 3.664675356   179.60.193.38    10.112.15.177    TCP          66 443 → 60650 [ACK] Seq=1 Ack=518 Win=29184 Len=0 TSval=773189909 TSecr=4065328261
39 3.664911689   179.60.193.38    10.112.15.177    TLSv1.2     204 Server Hello, Change Cipher Spec, Encrypted Handshake Message
40 3.664944354   10.112.15.177    179.60.193.38    TCP          66 60650 → 443 [ACK] Seq=518 Ack=139 Win=30336 Len=0 TSval=4065328267 TSecr=773189909
41 3.665352626   10.112.15.177    179.60.193.38    TLSv1.2     109 Change Cipher Spec, Encrypted Handshake Message
42 3.668382452   10.112.15.177    179.60.193.38    TLSv1.2     151 Application Data
43 3.668701971   10.112.15.177    179.60.193.38    TLSv1.2     508 Application Data
```

Figura 4: Capturando intercambio de mensaje con Wireshark

Conclusión:

Al realizar este proyecto podemos concluir que Facebook Messenger usa un protocolo TLS entre la capa de aplicación y la capa de transporte para cifrar los datos que se envían a través de la red cuando se realiza una conversación en esta plataforma.Esto genera una gran confiabilidad hacia la plataforma ya que nuestros paquetes de datos enviados van a ser cifrados y no serán visibles en la red si es que alguien trata de capturar estos datos,y así de esta forma las conversaciones que se realizan a través de la aplicación van a ser privadas y seguras cuando estas se usen.

Referencias:

A continuación se presentan todos los enlaces para que se pueda acceder a las páginas con la información recaudada y las imágenes utilizadas:

1. https://www.ssl.com/article/ssl-tls-handshake-overview/
2. https://adictec.com/que-es-y-como-funciona-ssl-tls/
3. https://www.digicert.com/es/noticias/2011-06-03-ssl-renego.htm
4. https://www.swhosting.com/blog/transport-layer-security-tls-que-es-y-como-funciona/
5. https://tools.ietf.org/html/rfc2818
6. https://tools.ietf.org/html/rfc5246
7. http://blog.fourthbit.com/2014/12/23/traffic-analysis-of-an-ssl-slash-tls-session
8. https://wearesocial.com/blog/2018/01/global-digital-report-2018