"Comparación de protocolos en seguridad de mensajería instantánea - WhatsApp v/s Telegram"

Carina Flores

Paula Nieto

Paul Rojas

Nicolas Villanueva

Universidad Tecnica Federico Santa Maria

Agosto 2018

Resumen

Los protocolos de seguridad son muy importantes hoy en día, un servicio que no asegure a sus usuarios que es seguro puede significar la perdida de clientes. Se explicarán los protocolos de seguridad usados por WhatsApp y Telegram con el objetivo de compararlos y analizar que factores podrían influir para evaluar la seguridad en la mensajeria de cada servicio. Finalmente, se mostrará capturas de un mismo mensaje enviado por los dos servicios, analizando que efectivamente estén usando el protocolo debidamente.

Introducción

La seguridad tiene una gran relevancia a la hora de elegir que servicio usar, especialmente en la mensajería instantánea. Los usuarios confían en los servicios que usan, y muchas veces entregamos información personal a través de estos servicios que esperamos nadie vea. Si no es confiable, los usuarios tienden a abandonar el servicio o reemplazarlo por otro, esto ocurrió con los usuarios de WhatsApp cuando se dio a conocer que este servicio no implementaba correctamente el protocolo que decían usar. Por este motivo, muchos usuarios se cambiaron a Telegram, aumentando su popularidad y forzando a WhatsApp a mejorar su seguridad.

Actualmente, WhatsApp utiliza Signal Protocol y Telegram utiliza MTProto (Mobile Transport Protocol). Estos son protocolos de la capa de aplicación, debido a que antes de enviar los mensajes estos se cifran y luego son enviados. De esta forma, se evita que los mensajes puedan ser legibles, gracias al grado de dificultad que tiene el cifrado, en caso de ser capturados cuando son transportados.

Un aspecto importante del trabajo, es que al referirnos con mensajes cifrados solo se tomará en cuenta el envío/recepción de estos, entre dos personas, es decir, no llamadas, videollamadas, imágenes o archivos; tampoco se considerará el cifrado en grupo ya que se trata de otra manera, el cual no es muy seguro y es de mayor complejidad. Esto por otro lado no asegura que los puntos finales sean seguros, ya que el mismo dispositivo puede ser vulnerado.

Finalmente lo que queremos lograr al finalizar este proyecto es garantizar si efectivamente WhatsApp cumple con su dicho “Simple. Seguro. Mensajería confiable.” y si a Telegram le corresponde la fama de app más segura en mensajería instantánea.

Descripción

Protocolo de seguridad de WhatsApp

El protocolo a estudiar usado por WhatsApp se desarrolló en conjunto con Open Whisper Systems, donde los detalles técnicos que disponen son limitados al público, Closed Source, y datan a finales del año 2017.

La seguridad que ofrece Whatsapp a sus usuarios es el cifrado E2EE, llamado extremo a extremo, este sistema de comunicacion debe asegurar que solo el emisor y receptor puedan leer lo que es enviado, y que ningún tercero; ni siquiera el servicio o aplicación que lo implementa lo puedan leer, un ejemplo de terceros es el ataque "Man-in-the-middle" donde el protocolo asegura la integridad del mensaje con la autenticación. Esto es porque los mensajes están seguros con un "candado" y solo el emisor y receptor tienen el "código/llave" para abrir y leer los mensajes.

El cifrado de extremo a extremo de WhatsApp para mayor protección, por cada mensaje que se envía tiene su propio candado y código único, es decir, no se puede descrifrar un mensaje del pasado con la "llave" del mensaje actual. Además, una vez llegado el mensaje al receptor este es borrado del servidor de WhatsApp.

Protocolo “The Signal”

Open Whisper Systems desarrolló un protocolo que es Opensource en cifrado de mensajes, orientado a una "sesión", el cual consiste en generar llaves, **PreKeys**, al momento de la instalación del servicio, para luego trabajar con un protocolo de establecimiento de claves que es un método de intercambio seguro de claves criptográficas, **Diffie–Hellman** (DH), combinado con el algoritmo **Double Ratchet** al momento de realizar la comunicación segura entre dos o más clientes en una "sesión" . WhatsApp implementó este protocolo trabajado de la siguiente manera, un total de seis llaves, tres de estas públicas las cuales se almacenan en un servidor, `Identity Key Pair`, `Signed Pre Key` y `One-Time Pre Key`, usando la primitiva Curve25519 para el cifrado de estas y generar una "Master Secret" que genera tres llaves de sesión, `Root Key`, `Chain Key` y `Message Key`.

Una vez establecida la "sesión", se puede comenzar a intercambiar mensajes los cuales son protegidos por la llave `Message Key` la cual usa un estándar de cifrado avanzado, AES256,

en modo CBC (Encadenamiento de bloque de cifrado) para cifrar/descifrar (vease figura 1 y 2) y HMAC-SHA256 para la autenticación.

Y como dijimos antes esta llave es efímera ya que la "sesión" no puede reconstruir la llave para recuperar un mensaje ya enviado/recibido, esto se logra con `Chain Key`, ya que esta avanza (en un solo sentido) para generar `Message Key` y asi un DH Handshake se debe hacer por cada mensaje enviado/recibido, permitiendo solo el avance y no retroceso para recuperar una llave previa.

Protocolo Telegram

El protocolo a estudiar usado por Telegram se desarrolló hacia 2013 bajo un estandar abierto, a base de API JAVA, siendo a su vez Closed Source.
Este sistema trabaja bajo la premicia "servidor-cliente", en donde se envía un mensaje que va cifrado al servidor, y el servidor lo envía al cliente destino. Haciendo multiples comprobaciones internas entre procesos, de tal manera que nadie pueda leer el mensaje en tránsito y verificar que no se haya modificado.

MTProto

Telegram actualmente trabaja con el protocolo MTProto 2.0. El protocolo está diseñado para acceder a una API de servidor desde aplicaciones que se ejecutan en dispositivos móviles. Consta de 3 componentes:

- **Componente de alto nivel** (lenguaje de consulta de API): define el método mediante el cual las consultas y respuestas de API se convierten en mensajes binarios.
- **Capa criptográfica** (autorización): define el método por el cual los mensajes se cifran antes de transmitirse a través del protocolo de transporte.
- **Componente de transporte**: define el método para que el cliente y el servidor transmitan mensajes a través de algún otro protocolo de red existente (como HTTP, HTTPS, TCP, UDP).

Al igual que Whatsapp, Telegram le da una clave única al cliente al momento de iniciarse en la aplicación `auth_key`, que es compartida entre el servidor-cliente entregada por el algoritmo Diffie-Hellman. El servidor tambien le entrega el `salt` (numero aleatorio que va

cambiando periodicamente) y `session_id` (numero aleatorio para distinguir sesiones individuales). Luego se generan 3 partes que contienen el texto cifrado (vease figura 3), para poder ser enviado a través de la red, las cuales son:

- **auth_key_id**: identifica a la clave de autenticación, compartida por el servidor y el usuario.
- **msg_key**: Mensaje de 128 bits, obtenido a través de la función hash SHA-256 tomando como entrada un fragmento de 32 bits de la clave de autenticación y el mensaje a encriptar (incluyendo el encabezado y padding), y seleccionando los 128 bits del medio del resultado. La msg_key se combina con otro fragmento de la auth_key en la KDF para generar una clave aes_key de 256 bits y un vector de inicialización aes_iv también de 256 bits. Estos dos valores se utilizarán para encriptar el mensaje (el cual incluye otros datos además del texto mismo, como salt, id de sesión, padding, entre otros).
- **encrypted_data**: Datos encriptados combinando la AES_KEY y AES IGE IV y el mensaje a ser encriptado, por medio del protocolo AES-IGE.

El receptor para poder desencriptar el mensaje deriva las claves AES_KEY y AES IGE IV y utilizando la msg_key (que recibió del emisor) y el fragmento de auth_key (que conoce). Luego utiliza estas claves para desencriptar el mensaje.

Resultado

Los protocolos que vimos son de la capa de aplicación y no podremos verlo directamente con wireshark, pero capturamos los paquetes con el objetivo de analizar el tamaño de los paquetes, el servidor destino y los datos.

En el caso de Whatsapp, mandamos un mensaje (vease captura 4) corto que tiene una longitud de 258 bytes, y notamos que el receptor (vease captura 5) recibe el mensaje desde un mismo servidor aunque el tamaño mensaje es levemente menor, esto es explicado porque Whatsapp desencripta parte del mensaje para guardar metadatos como la hora en que fue enviado el mensaje, número del emisor y receptor, por lo tanto, el mensaje disminuye su tamaño. Whatsapp necesita partes del mensaje pero esto no implica que lo desencripta todo.

Comparando el mensaje enviado desde Telegram (vease captura 6) vemos que la IP destino corresponde a la IP fuente en el mensaje de destino. Y el largo de este mensaje cambia en 120 bytes (vease captura 7). Esto es debido a que una parte del mensaje enviado **msg_key**, tiene texto relleno cifrado (como se dijo anteriormente) oscila entre 12 y 1024 bytes. En conlusión, podemos asumir que Telegram desencripta el mensaje y lo encripta, ya que el mensaje tiene un comportamiento aleatoriamente esperable de acuerdo a lo explicado.

Conclusión

Una vez visto los tipos de protocolos que usan WhatsApp y Telegram observamos que WhatsApp se preocupa de que el mensaje esté seguro entre el cliente-cliente, mientras que Telegram enfoca su seguridad entre el servidor-cliente, cliente-servidor. Así sus enfoques son distintos, por lo cual cada uno tiene sus ventajas y desventajas.

La ventaja de usar la seguridad implantada por WhatsApp sobre Telegram, es que no deja ningún rastro en sus servidores, a comparación de Telegram que todo el paquete es guardado en el servidor, incluyendo sus llaves de encriptación. Lo que hace que Telegram pueda acceder a nuestros mensajes, aunque ellos aseguran por sus Políticas de Privacidad que nunca las compartirán a otros entes.

Por otro lado, Telegram al tener los mensajes encriptados en sus servidores, los ocupan tambien para almacenar su copia de seguridad (historial de mensajes). Mientras Whatsapp al no guardar algo en sus servidores, para el servicio de historial se ocupa Google Drive, en donde los mensajes son almacenados en un texto plano (sin encriptar), lo que claramente es muy poco seguro.

Al intentar revisar el contenido de los mensajes enviados tanto en WhatsApp y Telegram vemos que estos están encriptados, como se esperaba, y el largo del mensaje es consistente con lo que se dice sobre la seguridad de cada uno. En cuanto al largo del mensaje enviado con el recibido en WhatsApp, este tuvo una pequeña variación, en donde el mensaje no fue intervenido, sino su metadata.

En Telegram hubo una mayor diferencia en los tamaños de los paquetes, esto es ya que se ocupa una cadena de bytes de largo aleatorio.

Referencias

- `Documentación Whatsapp:` https://www.whatsapp.com/security/
- `Documentación Whatsapp:` https://faq.whatsapp.com/es/general/28030015/
- `Documentación Telegram:` https://core.telegram.org/mtproto
- `Documentación Telegram:` https://core.telegram.org/api/end-to-end
- `Documentación Telegram:` https://telegram.org/faq#security

Anexo

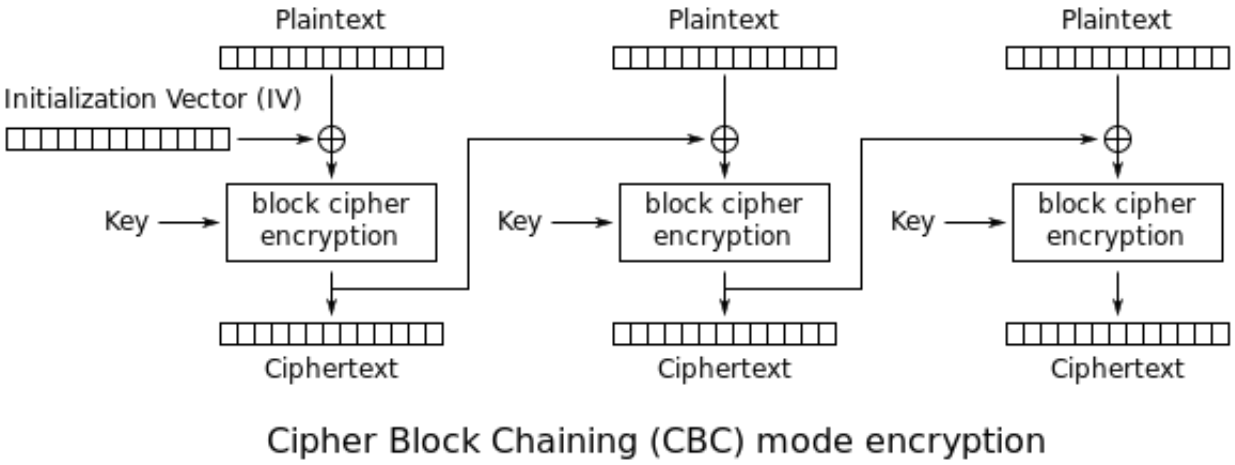

Figure 1: cifrar

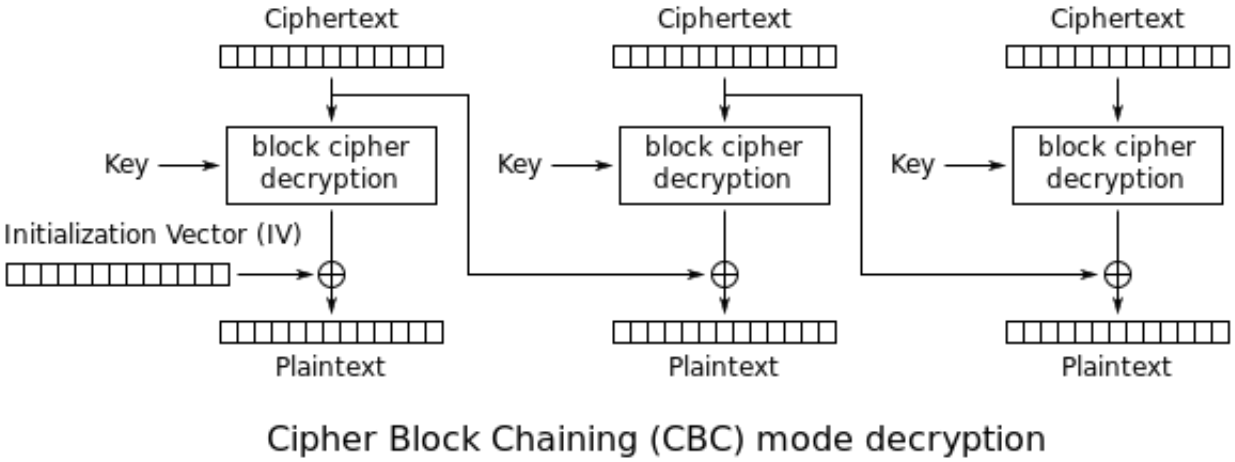

Figure 2: descifrar

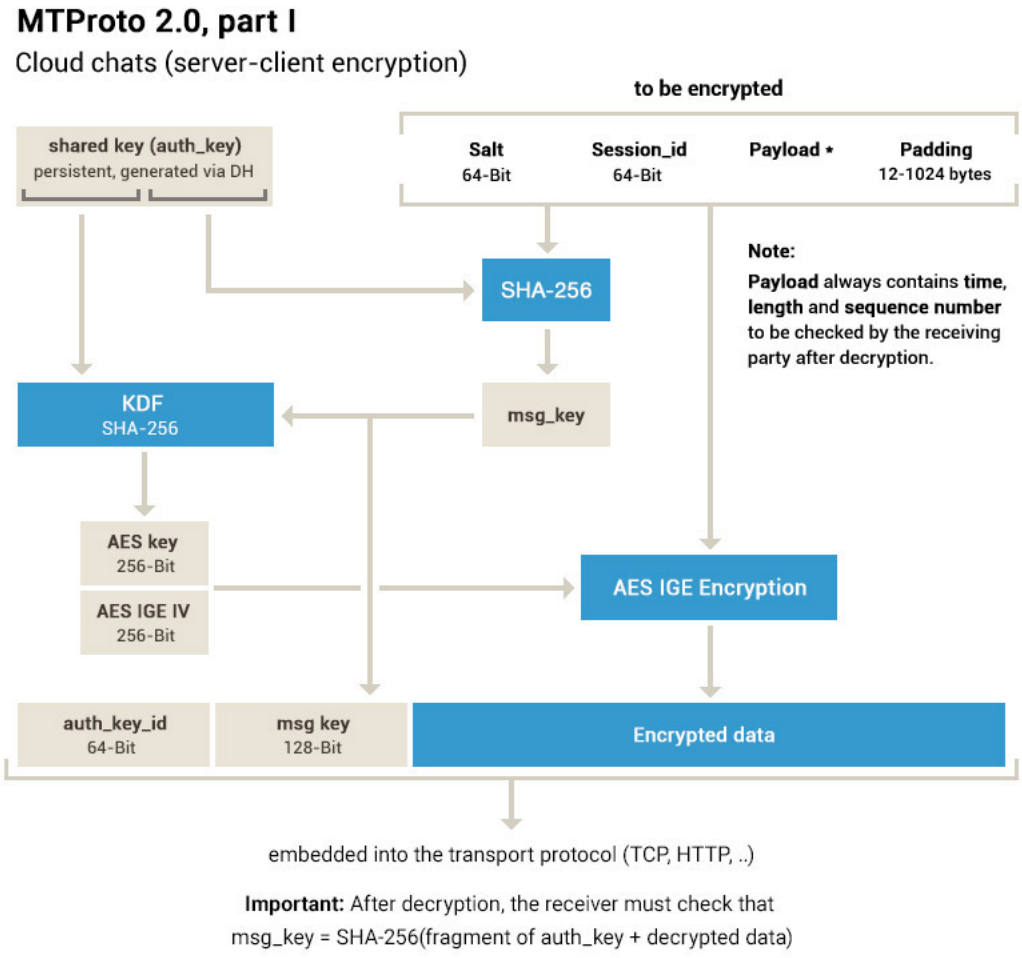

Figure 3: Protocolo usado por Telegram

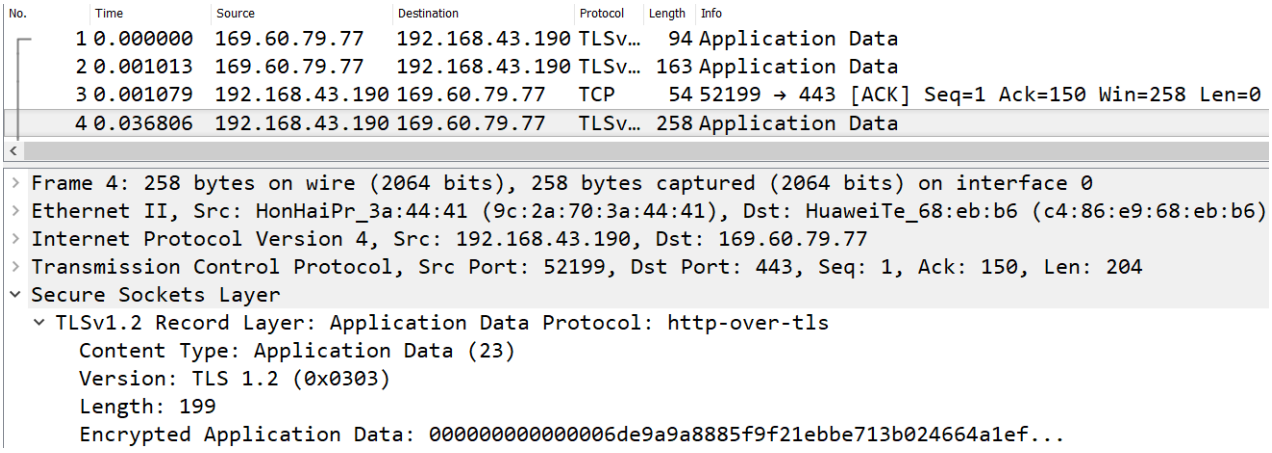

Figure 4: Captura envío Whatsapp

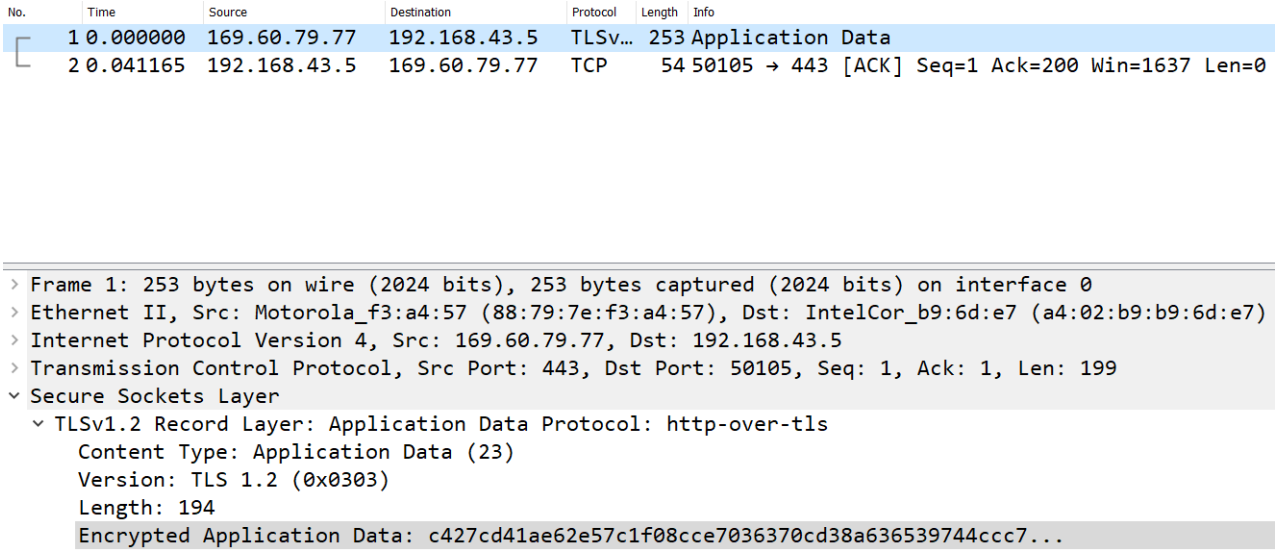

Figure 5: Captura recepción Whatsapp

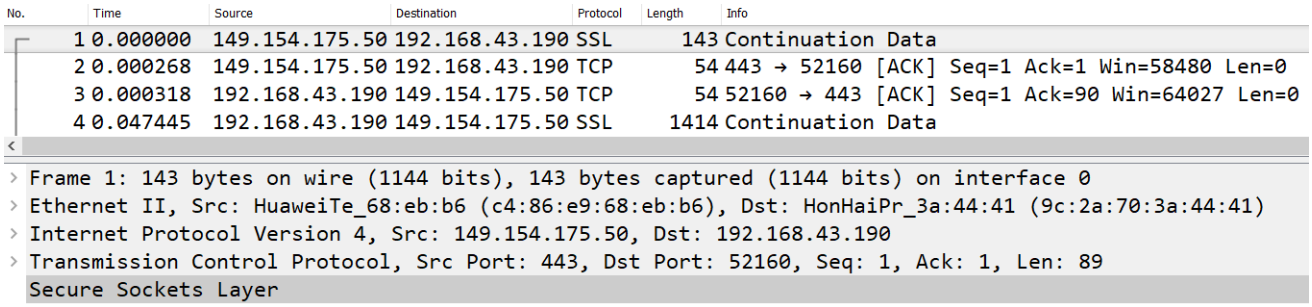

Figure 6: Captura envío Telegram

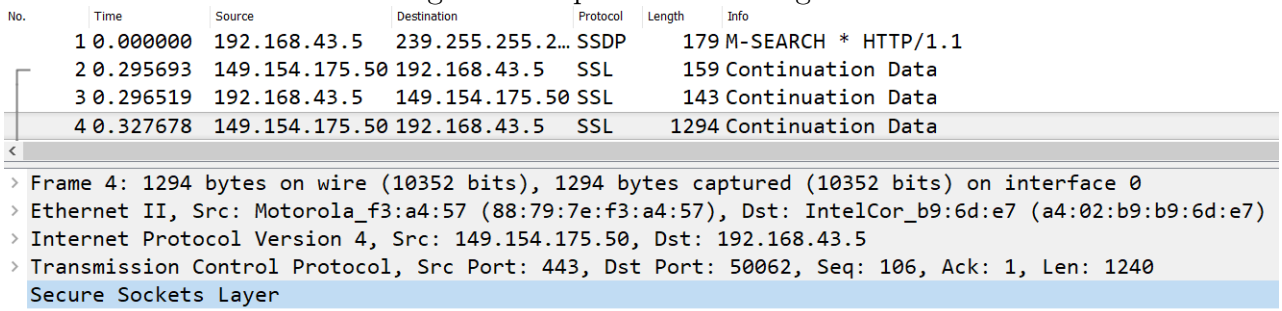

Figure 7: Captura recepción Telegram