UNIVERSIDAD TÉCNICA
FEDERICO SANTA MARIA
DEPARTAMENTO DE ELECTRÓNICA

# Proyecto de Redes de Computadores I

# El funcionamiento de Steam

**Integrantes:** Pilar Arancibia Echeverría
Camila Carrasco Rivera
Freddy Toledo Urrutia
**Profesor:** Agustín González
**Fecha de entrega:** 28-08-2017

# Índice

# Resumen

En este trabajo se estudió las principales funcionalidades de Steam y cómo cada una de estas utiliza distintos protocolos para su conectividad a la red. Principalmente se estudio la comunicación entre el cliente y los servidores de Steam a través de internet. Utilizando Wireshark, se observó que la mayoría de sus conecciones utilizaban protocolo UDP y TCP, pero también se utilizan otros protocolos, como el protocolo de seguridad TLSv1.2, el protocolo HTTP, entre otros. El protocolo TLSv1.2 se investigó con mayor profundidad debido a que se utiliza bastante durante la transacción de dinero durante una compra en Steam. Además se utilizó la página web llamada "Tracemyip" para localizar de donde provenían las IPs que participaban durante la comunicación.

# Introducción

La industria de los videojuegos es el sector económico involucrado en el desarrollo, la distribución, la mercadotecnia, la venta de videojuegos y del hardware asociado. Esta industria ha experimentado en los últimos años altas tasas de crecimiento, debido al desarrollo de la computación, capacidad de procesamiento e imágenes más reales.

La industria de los videojuegos en Chile alcanzó en 2015 su mayor nivel de ganancias, impulsado por el positivo escenario global de este mercado. Pese a tratarse de un mercado sobre el que aún existe desconocimiento, hoy es una de las industrias de mayor crecimiento en el segmento del entretenimiento en el país.

Por esta razón se tiene particular interés por el funcionamiento de la plataforma de entretención llamada Steam, la cual es una plataforma online desarrollada por Valve Corporation de registro gratuito con gran popularidad y éxito, en donde se ofrecen juegos de todo tipo ya sean gratuitos o pagados, y también se les da la posibilidad a los clientes de pagar por accesorios o ayudas en los juegos que ofrece.

# Funcionamiento de Steam

Steam es una plataforma de entretención online de gran popularidad y éxito, es una plataforma que ofrece juegos para computador ya sean gratuitos o pagados, y que además, consta de un servicio de comunicación entre usuarios, el cual ha creado una gran comunidad virtual para juegos multijugador.

Este trabajo está dedicado a la investigación del funcionamiento de Steam, particularmente a la comunicación entre el cliente y los servidores de Steam a través de internet, para esto se utilizará principalmente el programa Wireshark, el cual es un software dedicado al análisis de protocolos de red.

Para comenzar el análisis, inicialmente se estudiará el ingreso del cliente a la plataforma de Steam (Login).

## I. Ingreso a Steam (Login):

Luego de una investigación sobre Steam, se encontró en la sección de soporte de la página oficial de Steam, que este se comunica a través de los protocolos TCP y UDP desde el puerto 27015 al puerto 27030 [1]. Con esta información se procede a analizar el proceso de ingreso (Login) a Steam a través del software Wireshark y una página web llamada “tracemyip” que muestra la ubicación y la empresa a la que pertenece una cierta dirección IP. Entonces se Analizan los mensajes que utilizan un puerto dentro del rango de Steam, dichos mensajes utilizaban protocolo UDP y estaban asociados a distintas IPs, las cuales procedían de distintas empresas:

- Valve Corporation: 162.254.19x.xxx, 208.64.201.xxx y 208.78.164.xxx.
- Centurylink: 72.165.61.xxx.
- NETRONIK: 155.133.254.13x.

Valve Corporation es la empresa desarrolladora de Steam, Centurylink es un proveedor de internet en U.S.A. y NETRONIK es una empresa que se dedica al comercio al por mayor de computadores, equipo periférico y programas de informática.

Durante el proceso de ingreso, se envían una gran cantidad de paquetes usando protocolo UDP, los servidores de Steam utilizan los puertos 27017, 27018, 27019 y 27020 (puertos dentro del rango de Steam), y el computador del usuario utiliza el puerto 53233. Entonces se envían paquetes por turnos, primero el usuario envía varios paquetes de 36 bytes a los servidores de Steam, luego Steam responde al usuario con varios paquetes de 44 bytes, y así sucesivamente. Pero luego los paquetes comienzan a ser más grandes y de distintos tamaños, por ejemplo, 1280 bytes, 1050 bytes, etc.

### II. Descarga de contenido:

Para la descarga de contenido se utiliza protocolo TCP, el servidor utiliza la ip 155.133.249.131, la cual que corresponde a un servidor de Valve Corporation. El computador del usuario utiliza los puerto 56793, 56794 y 56795, y el servidor de Steam utiliza el puerto 80, ya que el puerto 80 es el puerto que un servidor utiliza para "escuchar" al cliente.

Al analizar la comunicación se puede observar que inicialmente el usuario envía un mensaje al servidor de Steam utilizando el protocolo TLSv1.2, este protocolo se utiliza para proporcionar comunicaciones seguras por internet. Luego son enviados 3 mensajes de sincronización a los servidores de Steam, cada uno pidiendo sincronización para cada puerto que se utilizará, a continuación, Steam responde con 3 mensajes de sincronización y ACK, cada uno seguido por la respuesta ACK del usuario. Luego de cada sincronización exitosa el usuario envía un mensaje GET utilizando protocolo HTTP (protocolo de comunicación en World Wide Web) para solicitar la información al servidor, entonces comienza el envío de datos. El servidor envía de a 2 paquetes de datos al puerto 56795, cada 2 paquetes recibidos el usuario envía un ACK, es importante notar que los paquetes tienen un largo de 1514 [bytes], excepto el primer paquete de 260 [bytes]. Luego de varios paquetes enviados, el servidor de Steam envía un mensaje advirtiendo que ya ha enviado todos los segmentos del mensaje TCP, entonces se re-ensamblan los segmentos y el servidor sigue a enviar datos, pero esta vez hacia el puerto 56794, luego ocurre un proceso similar con este puerto y el puerto 56793.

## III. Steam In-Home:

Steam In-Home es un servicio que ofrece al usuario la posibilidad de ejecutar juegos de su cuenta, en diferentes dispositivos, mientras tenga instalado el software en al menos uno de ellos.

Lo que hace la plataforma de Steam es, comenzar una comunicación entre el dispositivo que solicitó la re-transmisión y el dispositivo con el juego instalado, mientras ambos se encuentren en la misma sesión. Mediante pruebas con Wireshark, pudimos darnos cuenta que la comunicación de ambos dispositivos se establece mediante intercambio de datos con protocolos TCP, primeramente, utilizando los puertos 27036 ó 27037 ,y una vez establecida esta conexión, el intercambio de datos se produce mediante protocolos UDP, utilizando los puertos 27031 ó 27036, (si estos puertos están siendo restringidos u ocupados por el usuario, le será imposible lograr una re-transmisión), esto para que se produzca una conexión más fluida, pues lo que se está haciendo con esto es algo muy parecido a un streaming de video, porque el dispositivo con el juego instalado está constantemente enviando datos de audio y video al dispositivo que solicitó la re-transmisión y este a su vez, cada vez que el usuario utiliza un periférico, este enviá la información de vuelta, para que se haga efectivo un manejo de estos, en el dispositivo con el juego instalado.

Lo que logra esta comunicación es simplemente, crear la sensación de estar jugando en un dispositivo que no cuenta con la instalación o los requisitos para correr un juego, haciendo que otro que sí cuente con estas características lo ejecute y lo reproduzca en el dispositivo que no puede, teniendo este la opción de ver los datos de audio y video, e interactuar con el software mediante el control de los periféricos.

## IV. Transacción en tienda:

Steam posee una tienda virtual, a través de la cual los usuarios pueden comprar juegos de computador de una manera más sencilla. El usuario puede comprar en la misma interfaz del cliente, y una vez realizada esta, los juegos adquiridos se asocian permanentemente a la cuenta de Steam del usuario.

Se realizó una captura de wireshark al realizar una transacción y se obtuvo lo mostrado en la **figura iv.i**, desde esta se puede ver como se utiliza el protocolo TLSv1.2.

***Protocolo TLS*** [5]

***(Transport Layer Security, o bien Seguridad en la capa de transporte)***

El protocolo TLS es un protocolo criptográfico que proporciona una comunicación segura por una red. Este protocolo utiliza un conjunto de protocolos Handshake.

Para entablar una conexión entre cliente y servidor, se siguen los siguientes pasos:

1. Cliente envía un mensaje Client Hello, en el que especifica una lista con diversos atributos para la conexión.
2. Cliente recibe un mensaje Server Hello, en donde el servidor eligió los parámetros de conexión a partir de las opciones ofrecidas por el cliente.
3. Se hace un intercambio de certificados
4. Cliente y servidor negocian una clave secreta.

Una vez hecho lo anterior, se puede enviar información de manera segura.

En la **figura iv.i**, se aprecian destacados todos los paquetes utilizados por el protocolo TLS para entablar una conexión. Desde la **figura iv.ii** hasta la i**v.vii**, se pueden ver de manera más detallada cada paso.

# Conclusiones

La industria de los videojuegos es un sector económico que ha tenido una gran tasa de crecimiento en los últimos años a nivel mundial. Es por esto que una plataforma como Steam ha tenido un gran éxito, ya que le brinda la posibilidad al cliente de comprar juegos de forma rápida y desde su casa, sin la necesidad de ir a una tienda o esperar que el juego llegue a su casa.

Para encontrar los paquetes de datos que pertenecen a cierta aplicación, es necesario saber la IP del servidor o el puerto que el servidor está utilizando, en este caso fue posible el análisis gracias a que se averiguó los puertos con los cuales trabajaba Steam. Y mediante el análisis de Wireshark se concluyó que los servidores de Steam se comunican con los computadores de sus clientes utilizando el protocolo UDP para el ingreso y TCP para la descarga de contenido.

Steam puede establecer una conexión entre dos dispositivos, mientras estos tengan la misma cuenta de usuario abierta en ellos. Iniciando una conexión entre ambos y utilizando protocolos dedicados a la transferencia de video y audio (protocolo SKYPE) y otro tipo de información mediante protocolos más especializados (Frame Protocol), estos protocolos son del tipo UDP, y sabemos que se utilizan para obtener un intercambio de datos más fluido que seguro, pues prácticamente es una transmisión de datos muy parecida a un “streaming de video”.

Al realizar una transacción bancaria, se puede apreciar como el cliente de Steam utiliza el protocolo TLSv1.2. Este protocolo se preocupa de encriptar la información, de manera que su tráfico en la red sea más seguro. Este cambio de protocolo es esperable, ya que al manejar información tan sensible como lo son las tarjetas de crédito, el programa debe preocuparse de entregar un producto que sea lo más seguro posible.

En un futuro cercano podría existir una gran plataforma online en la que se pueda conseguir todo tipo de programas, ya sean juegos, procesadores de texto, simuladores y herramientas de todo tipo. De esta forma el usuario solo tendría que escribir el tipo de programa que está buscando y la plataforma le ofrecería todos los programas disponibles, ya sean pagados o gratuitos.

# Referencias

[1] https://support.steampowered.com/kb_article.php?ref=8571-GLVN-8711

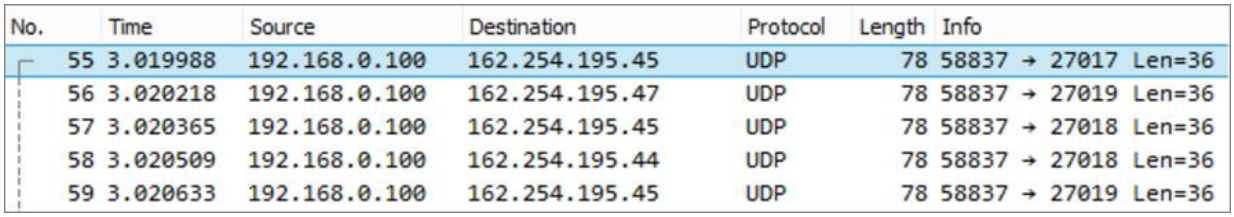
[2] https://tools.tracemyip.org

[3] http://www.latercera.com/noticia/industria-de-videojuegos-en-chile-facturo-13-millones-de-dolares-en-2015/

[4] https://es.wikipedia.org/wiki/Industria_de_los_videojuegos

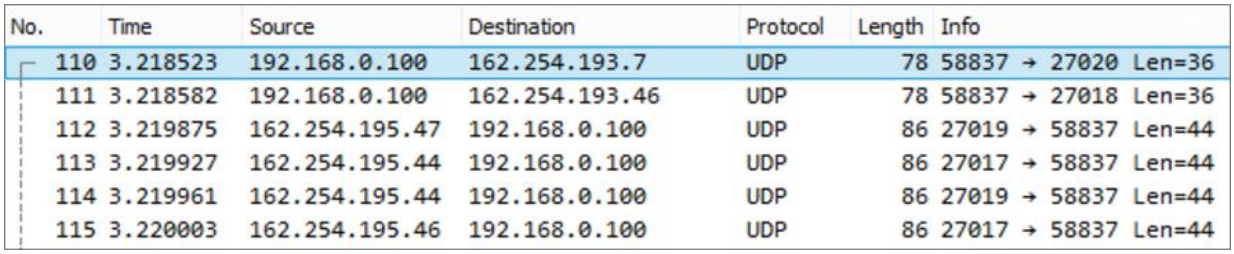
[5] http://deic.uab.es/material/26118-ssl.pdf

# Anexos



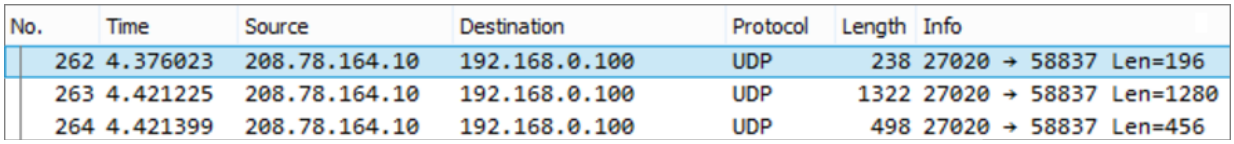
| No. | Time | Source | Destination | Protocol | Length | Info |
|---|---|---|---|---|---|---|
| 55 | 3.019988 | 192.168.0.100 | 162.254.195.45 | UDP | 78 | 58837 → 27017 Len=36 |
| 56 | 3.020218 | 192.168.0.100 | 162.254.195.47 | UDP | 78 | 58837 → 27019 Len=36 |
| 57 | 3.020365 | 192.168.0.100 | 162.254.195.45 | UDP | 78 | 58837 → 27018 Len=36 |
| 58 | 3.020509 | 192.168.0.100 | 162.254.195.44 | UDP | 78 | 58837 → 27018 Len=36 |
| 59 | 3.020633 | 192.168.0.100 | 162.254.195.45 | UDP | 78 | 58837 → 27019 Len=36 |

**Figura i.i** Captura wireshark de ingreso a Steam.



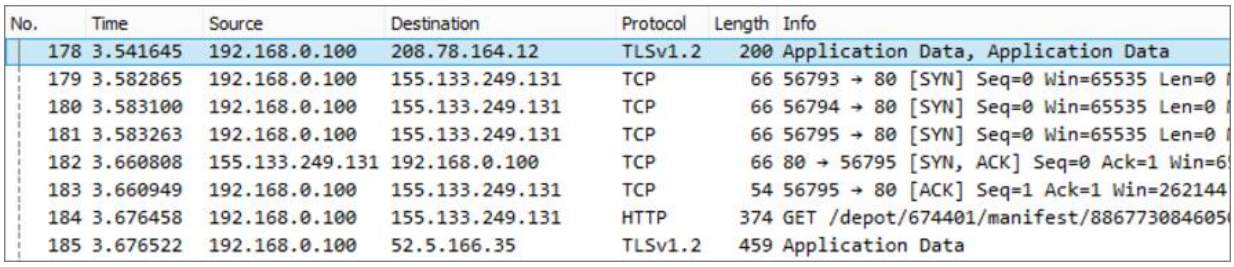
| No. | Time | Source | Destination | Protocol | Length | Info |
|---|---|---|---|---|---|---|
| 110 | 3.218523 | 192.168.0.100 | 162.254.193.7 | UDP | 78 | 58837 → 27020 Len=36 |
| 111 | 3.218582 | 192.168.0.100 | 162.254.193.46 | UDP | 78 | 58837 → 27018 Len=36 |
| 112 | 3.219875 | 162.254.195.47 | 192.168.0.100 | UDP | 86 | 27019 → 58837 Len=44 |
| 113 | 3.219927 | 162.254.195.44 | 192.168.0.100 | UDP | 86 | 27017 → 58837 Len=44 |
| 114 | 3.219961 | 162.254.195.44 | 192.168.0.100 | UDP | 86 | 27019 → 58837 Len=44 |
| 115 | 3.220003 | 162.254.195.46 | 192.168.0.100 | UDP | 86 | 27017 → 58837 Len=44 |

**Figura i.ii** Captura wireshark de ingreso a Steam.



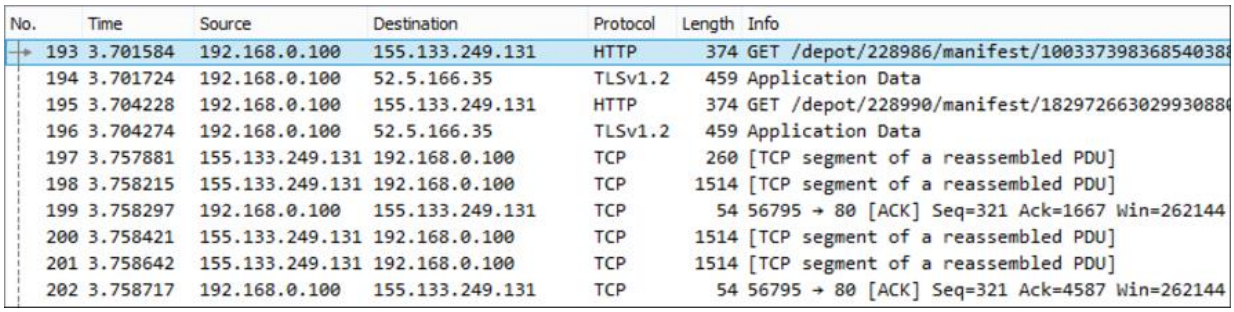
| No. | Time | Source | Destination | Protocol | Length | Info |
|---|---|---|---|---|---|---|
| 262 | 4.376023 | 208.78.164.10 | 192.168.0.100 | UDP | 238 | 27020 → 58837 Len=196 |
| 263 | 4.421225 | 208.78.164.10 | 192.168.0.100 | UDP | 1322 | 27020 → 58837 Len=1280 |
| 264 | 4.421399 | 208.78.164.10 | 192.168.0.100 | UDP | 498 | 27020 → 58837 Len=456 |

**Figura i.iii** Captura wireshark de ingreso a Steam.



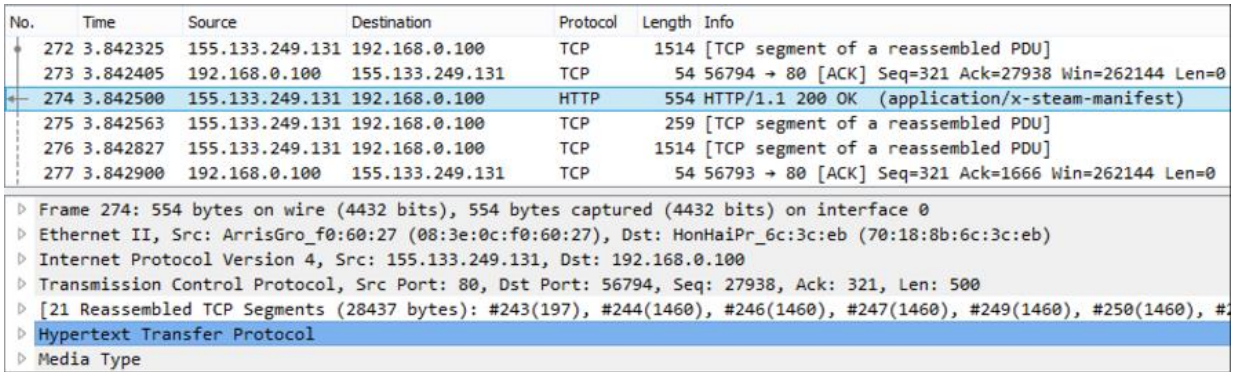
| No. | Time | Source | Destination | Protocol | Length | Info |
|---|---|---|---|---|---|---|
| 178 | 3.541645 | 192.168.0.100 | 208.78.164.12 | TLSv1.2 | 200 | Application Data, Application Data |
| 179 | 3.582865 | 192.168.0.100 | 155.133.249.131 | TCP | 66 | 56793 → 80 [SYN] Seq=0 Win=65535 Len=0 |
| 180 | 3.583100 | 192.168.0.100 | 155.133.249.131 | TCP | 66 | 56794 → 80 [SYN] Seq=0 Win=65535 Len=0 |
| 181 | 3.583263 | 192.168.0.100 | 155.133.249.131 | TCP | 66 | 56795 → 80 [SYN] Seq=0 Win=65535 Len=0 |
| 182 | 3.660808 | 155.133.249.131 | 192.168.0.100 | TCP | 66 | 80 → 56795 [SYN, ACK] Seq=0 Ack=1 Win=6 |
| 183 | 3.660949 | 192.168.0.100 | 155.133.249.131 | TCP | 54 | 56795 → 80 [ACK] Seq=1 Ack=1 Win=262144 |
| 184 | 3.676458 | 192.168.0.100 | 155.133.249.131 | HTTP | 374 | GET /depot/674401/manifest/886773084605 |
| 185 | 3.676522 | 192.168.0.100 | 52.5.166.35 | TLSv1.2 | 459 | Application Data |

**Figura ii.i** Captura wireshark de descarga de contenido, protocolo TCP y TLSv1.2.



| No. | Time | Source | Destination | Protocol | Length | Info |
|---|---|---|---|---|---|---|
| 193 | 3.701584 | 192.168.0.100 | 155.133.249.131 | HTTP | 374 | GET /depot/228986/manifest/100337398368540388 |
| 194 | 3.701724 | 192.168.0.100 | 52.5.166.35 | TLSv1.2 | 459 | Application Data |
| 195 | 3.704228 | 192.168.0.100 | 155.133.249.131 | HTTP | 374 | GET /depot/228990/manifest/182972663029930886 |
| 196 | 3.704274 | 192.168.0.100 | 52.5.166.35 | TLSv1.2 | 459 | Application Data |
| 197 | 3.757881 | 155.133.249.131 | 192.168.0.100 | TCP | 260 | [TCP segment of a reassembled PDU] |
| 198 | 3.758215 | 155.133.249.131 | 192.168.0.100 | TCP | 1514 | [TCP segment of a reassembled PDU] |
| 199 | 3.758297 | 192.168.0.100 | 155.133.249.131 | TCP | 54 | 56795 → 80 [ACK] Seq=321 Ack=1667 Win=262144 |
| 200 | 3.758421 | 155.133.249.131 | 192.168.0.100 | TCP | 1514 | [TCP segment of a reassembled PDU] |
| 201 | 3.758642 | 155.133.249.131 | 192.168.0.100 | TCP | 1514 | [TCP segment of a reassembled PDU] |
| 202 | 3.758717 | 192.168.0.100 | 155.133.249.131 | TCP | 54 | 56795 → 80 [ACK] Seq=321 Ack=4587 Win=262144 |

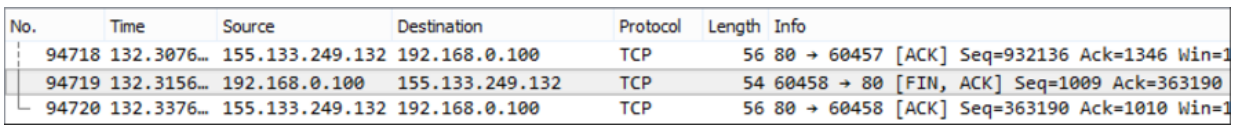
**Figura ii.ii** Captura wireshark de descarga de contenido,
protocolo HTTP: intercambio de información.



| No. | Time | Source | Destination | Protocol | Length | Info |
|---|---|---|---|---|---|---|
| 272 | 3.842325 | 155.133.249.131 | 192.168.0.100 | TCP | 1514 | [TCP segment of a reassembled PDU] |
| 273 | 3.842405 | 192.168.0.100 | 155.133.249.131 | TCP | 54 | 56794 → 80 [ACK] Seq=321 Ack=27938 Win=262144 Len=0 |
| 274 | 3.842500 | 155.133.249.131 | 192.168.0.100 | HTTP | 554 | HTTP/1.1 200 OK (application/x-steam-manifest) |
| 275 | 3.842563 | 155.133.249.131 | 192.168.0.100 | TCP | 259 | [TCP segment of a reassembled PDU] |
| 276 | 3.842827 | 155.133.249.131 | 192.168.0.100 | TCP | 1514 | [TCP segment of a reassembled PDU] |
| 277 | 3.842900 | 192.168.0.100 | 155.133.249.131 | TCP | 54 | 56793 → 80 [ACK] Seq=321 Ack=1666 Win=262144 Len=0 |

```
Frame 274: 554 bytes on wire (4432 bits), 554 bytes captured (4432 bits) on interface 0
Ethernet II, Src: ArrisGro_f0:60:27 (08:3e:0c:f0:60:27), Dst: HonHaiPr_6c:3c:eb (70:18:8b:6c:3c:eb)
Internet Protocol Version 4, Src: 155.133.249.131, Dst: 192.168.0.100
Transmission Control Protocol, Src Port: 80, Dst Port: 56794, Seq: 27938, Ack: 321, Len: 500
[21 Reassembled TCP Segments (28437 bytes): #243(197), #244(1460), #246(1460), #247(1460), #249(1460), #250(1460), #
Hypertext Transfer Protocol
Media Type
```

**Figura ii.iii** Captura wireshark de descarga de contenido, protocolo HTTP: reensamblaje.



| No. | Time | Source | Destination | Protocol | Length | Info |
|---|---|---|---|---|---|---|
| 94718 | 132.3076… | 155.133.249.132 | 192.168.0.100 | TCP | 56 | 80 → 60457 [ACK] Seq=932136 Ack=1346 Win=1 |
| 94719 | 132.3156… | 192.168.0.100 | 155.133.249.132 | TCP | 54 | 60458 → 80 [FIN, ACK] Seq=1009 Ack=363190 |
| 94720 | 132.3376… | 155.133.249.132 | 192.168.0.100 | TCP | 56 | 80 → 60458 [ACK] Seq=363190 Ack=1010 Win=1 |

**Figura ii.iv** Captura wireshark de descarga de contenido 4.

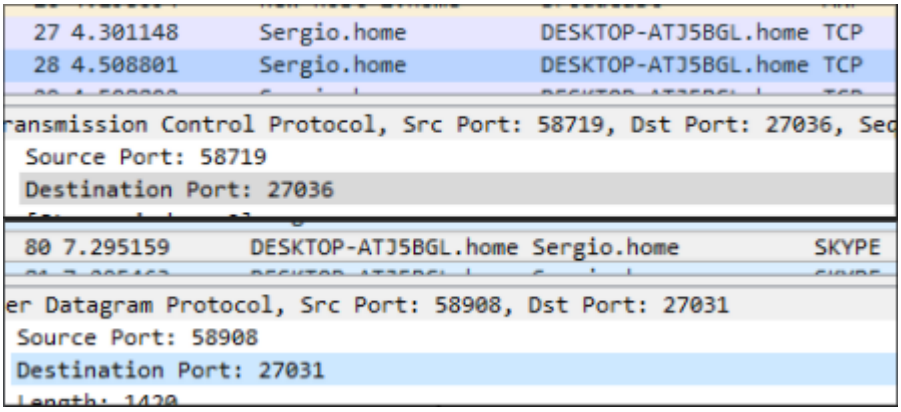



**Figura iii.i** Captura wireshark de Steam In-Home 1.

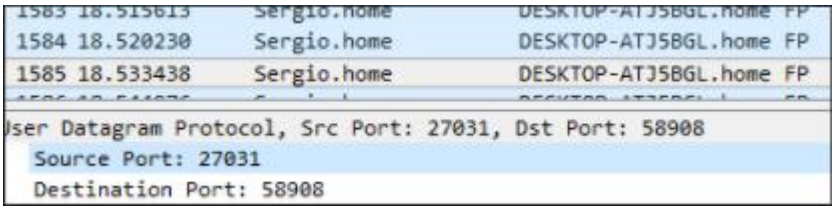



**Figura iii.ii** Captura wireshark de Steam In-Home 2.

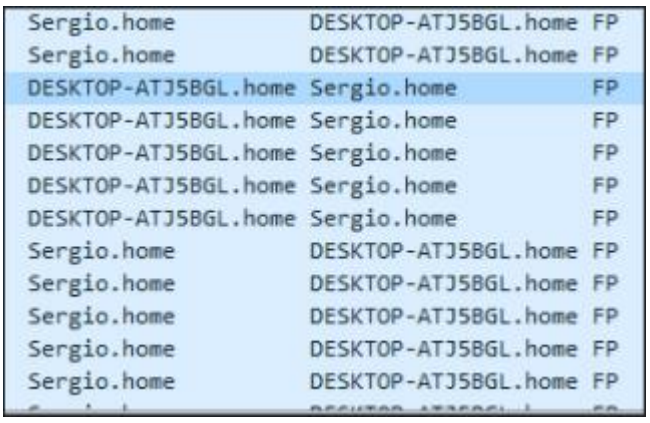
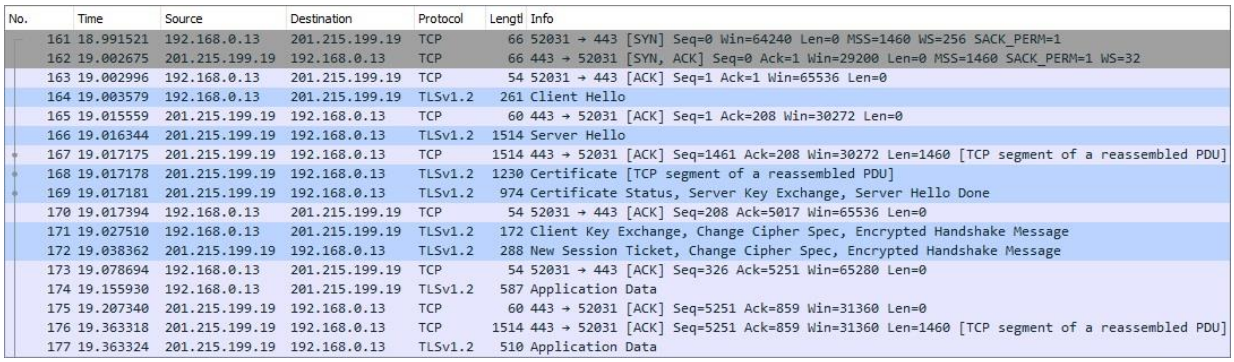



**Figura iii.iii** Captura wireshark de Steam In-Home 3.



| No. | Time | Source | Destination | Protocol | Lengtl | Info |
|---|---|---|---|---|---|---|
| 161 | 18.991521 | 192.168.0.13 | 201.215.199.19 | TCP | 66 | 52031 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1 |
| 162 | 19.002675 | 201.215.199.19 | 192.168.0.13 | TCP | 66 | 443 → 52031 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1460 SACK_PERM=1 WS=32 |
| 163 | 19.002996 | 192.168.0.13 | 201.215.199.19 | TCP | 54 | 52031 → 443 [ACK] Seq=1 Ack=1 Win=65536 Len=0 |
| 164 | 19.003579 | 192.168.0.13 | 201.215.199.19 | TLSv1.2 | 261 | Client Hello |
| 165 | 19.015559 | 201.215.199.19 | 192.168.0.13 | TCP | 60 | 443 → 52031 [ACK] Seq=1 Ack=208 Win=30272 Len=0 |
| 166 | 19.016344 | 201.215.199.19 | 192.168.0.13 | TLSv1.2 | 1514 | Server Hello |
| 167 | 19.017175 | 201.215.199.19 | 192.168.0.13 | TCP | 1514 | 443 → 52031 [ACK] Seq=1461 Ack=208 Win=30272 Len=1460 [TCP segment of a reassembled PDU] |
| 168 | 19.017178 | 201.215.199.19 | 192.168.0.13 | TLSv1.2 | 1230 | Certificate [TCP segment of a reassembled PDU] |
| 169 | 19.017181 | 201.215.199.19 | 192.168.0.13 | TLSv1.2 | 974 | Certificate Status, Server Key Exchange, Server Hello Done |
| 170 | 19.017394 | 192.168.0.13 | 201.215.199.19 | TCP | 54 | 52031 → 443 [ACK] Seq=208 Ack=5017 Win=65536 Len=0 |
| 171 | 19.027510 | 192.168.0.13 | 201.215.199.19 | TLSv1.2 | 172 | Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message |
| 172 | 19.038362 | 201.215.199.19 | 192.168.0.13 | TLSv1.2 | 288 | New Session Ticket, Change Cipher Spec, Encrypted Handshake Message |
| 173 | 19.078694 | 192.168.0.13 | 201.215.199.19 | TCP | 54 | 52031 → 443 [ACK] Seq=326 Ack=5251 Win=65280 Len=0 |
| 174 | 19.155930 | 192.168.0.13 | 201.215.199.19 | TLSv1.2 | 587 | Application Data |
| 175 | 19.207340 | 201.215.199.19 | 192.168.0.13 | TCP | 60 | 443 → 52031 [ACK] Seq=5251 Ack=859 Win=31360 Len=0 |
| 176 | 19.363318 | 201.215.199.19 | 192.168.0.13 | TCP | 1514 | 443 → 52031 [ACK] Seq=5251 Ack=859 Win=31360 Len=1460 [TCP segment of a reassembled PDU] |
| 177 | 19.363324 | 201.215.199.19 | 192.168.0.13 | TLSv1.2 | 510 | Application Data |

**Figura iv.i** Captura wireshark de transacción, headers.

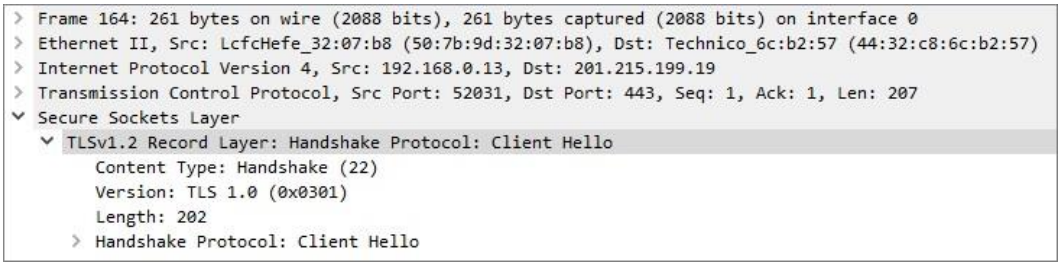



**Figura iv.ii** Captura wireshark de transacción, Client Hello.

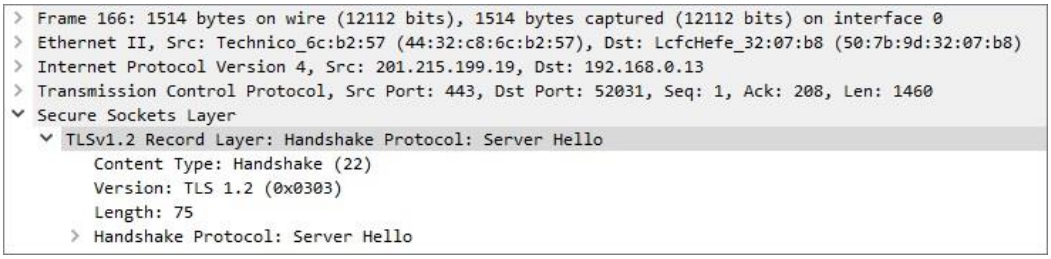



**Figura iv.iii** Captura wireshark de transacción, Server Hello

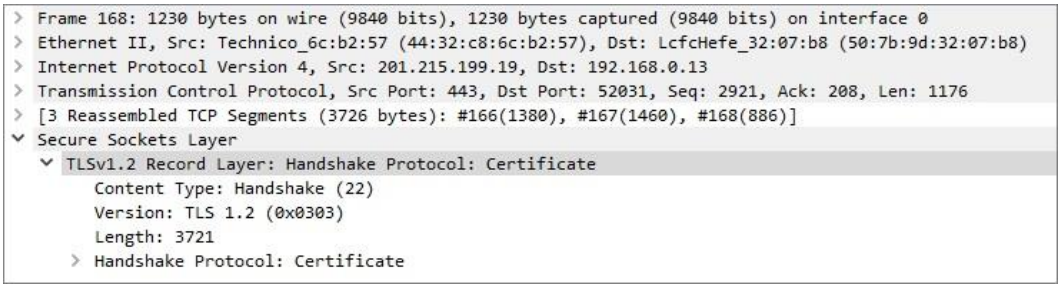



**Figura iv.iv** Captura wireshark de transacción, Certificado.

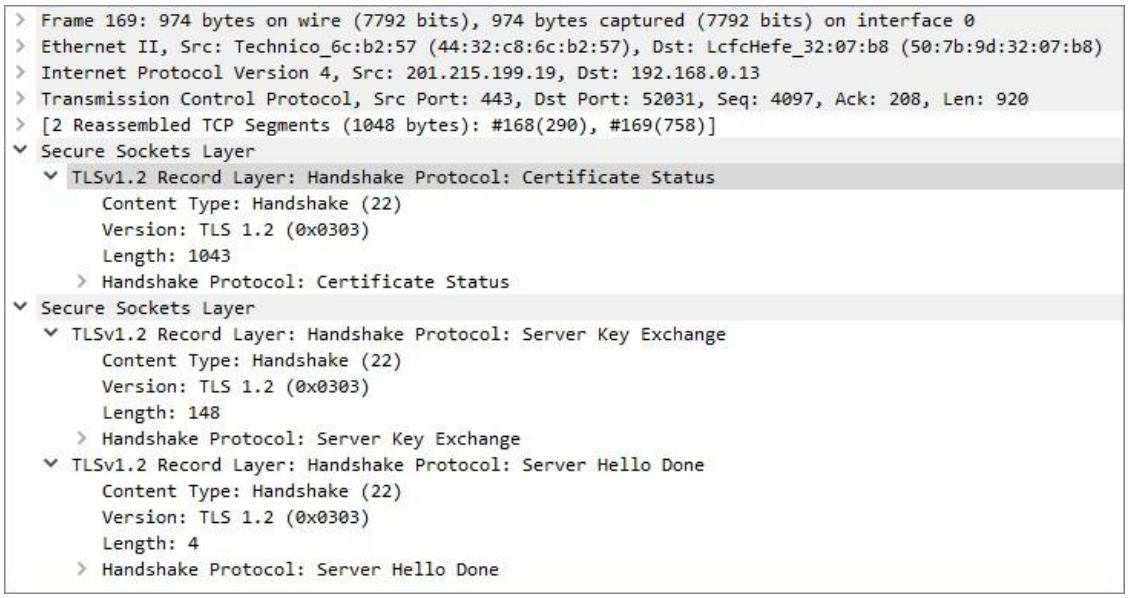



**Figura iv.v** Captura wireshark de transacción, certificado y negociación de clave.

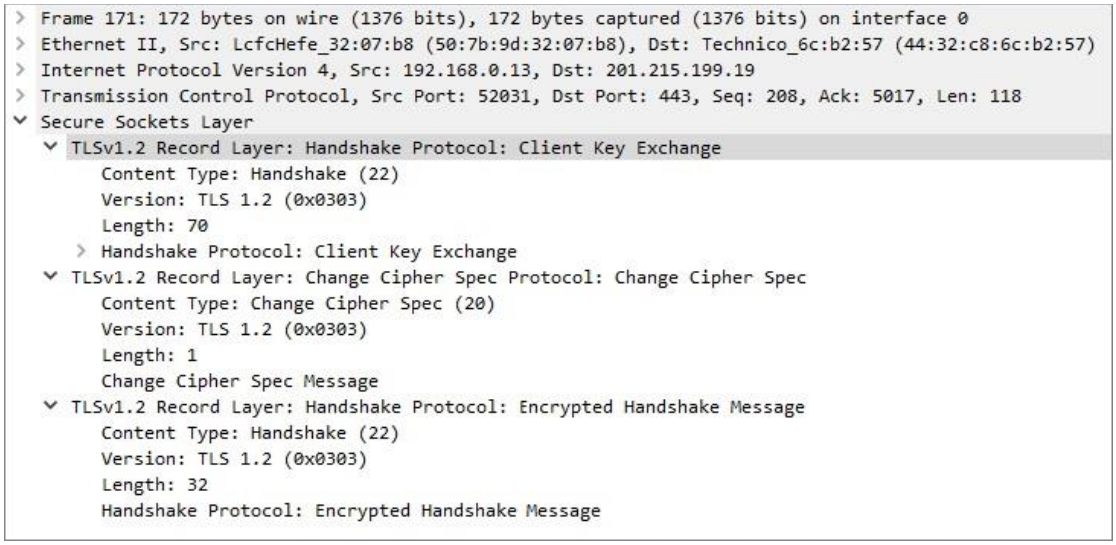



**Figura iv.vi** Captura wireshark de transacción, negociación de clave, y cifrado.


**Figura iv.vii** Captura wireshark de transacción, negociación de clave, y cifrado.