

Universidad de los Andes
Departamento de Ingeniería de Sistemas



**Laboratorio #2: Análisis De Protocolos De La Capa
De Aplicación**

ISIS3204 - Infraestructura de Comunicaciones

Grupo 3:

Juan Esteban Quiroga - 202013216
Juan Manuel Rodriguez - 202013372
Andres Felipe Ortiz - 201727662


2025-10

Contents

Introducción	3
8.1 Prueba ping	3
8.1.1 Prueba de conectividad al servidor DNS	3
8.1.2 Prueba de conectividad al servidor FTP	4
8.2 Análisis de tráfico del Servicio DNS	5
8.2.1 Prueba de conectividad al Servidor Web (IP)	5
8.2.2 Prueba de conectividad al Servidor Web (URL)	6
8.3 Análisis de tráfico del Servicio FTP	6
8.3.1 Conexión al servidor FTP	6
8.3.2 Descarga de archivo (Download)	7
8.3.3 Carga de archivo (Upload)	7
8.4 Análisis de tráfico del Servicio Web	8
8.4.1 Acceso al servidor web mediante HTTP	8
8.5 Análisis del protocolo HTTPS realizando navegación en el sitio de YouTube	9
8.5.1 Navegación en YouTube	9
8.5.2 Navegación en otros sitios HTTPS	9
8.6 Análisis del protocolo VoIP	16
8.6.1 Establecimiento de la llamada	16
8.7 Análisis del protocolo RTMP	16
8.7.1 Inicio de la transmisión	16
9.1 Topología	16

Introducción

En este laboratorio configuramos y probamos algunos servicios de red: DNS, HTTP/HTTPS, FTP, VoIP y RTMP, usando una topología a pequeña escala. Con Wireshark analizamos la conectividad de estos servicios en dicha red. Este laboratorio nos ayudó a entender la interacción de protocolos importantes que soportan la comunicación en redes de computadores. En la figura 1 mostramos las IPs estáticas que usamos para este laboratorio. (Es importante que estas IPs no sean dinámicas por DHCP ya que esto asegura direccionamiento constante y facilidad de configuración estable en la red.) La asignación de direcciones inició en 172.20.10.5 porque el gateway de la red es el hotspot del iPhone en 172.20.10.1, dejando libres las direcciones previas para posibles equipos de infraestructura y evitando conflictos.



VM (Lab 2 role)	Static IP	Notes
BaseServer	172.20.10.4/28	Baseline
server1-web	172.20.10.5/28	Web server
server2-ftp	172.20.10.6/28	FTP server
server3-voip	172.20.10.7/28	VoIP server
server4-dns	172.20.10.8/28	DNS server
server5-rtmp	172.20.10.9/28	RTMP server

Figure 1: Configuración de IPs estáticas

8.1 Prueba ping

En esta sección se verifica la conexión básica entre el cliente y los servidores DNS y FTP mediante pings con el protocolo ICMP, asegurando la conectividad antes de analizar los demás protocolos.

8.1.1 Prueba de conectividad al servidor DNS

Desde el cliente se enviaron pings (echo requests ICMP) al servidor DNS utilizando su dirección IP (172.20.10.8). El tráfico generado se capturó y se guardó en el archivo [Ping_DNS_IP.pcap](#). El archivo fue abierto en Wireshark y se aplicó el filtro icmp para observar únicamente los paquetes de ping. Se registraron la dirección IP de origen, dirección IP de destino, dirección MAC de origen y dirección MAC de destino en las tramas capturadas.

Time	172.20.10.2	172.20.10.8	Comment
0.000000		Echo (ping) request id=0x0fb1, seq=1/256, t...	ICMP: Echo (ping) request id=0x0fb1, seq=1/256.
0.000307		Echo (ping) reply id=0x0fb1, seq=1/256, ttl...	ICMP: Echo (ping) reply id=0x0fb1, seq=1/256, ..
0.930467		Echo (ping) request id=0x0fb1, seq=2/512, t...	ICMP: Echo (ping) request id=0x0fb1, seq=2/512..
0.930934		Echo (ping) reply id=0x0fb1, seq=2/512, ttl...	ICMP: Echo (ping) reply id=0x0fb1, seq=2/512, ..
2.537523		Echo (ping) request id=0x0fb1, seq=3/768, t...	ICMP: Echo (ping) request id=0x0fb1, seq=3/768.
2.538032		Echo (ping) reply id=0x0fb1, seq=3/768, ttl...	ICMP: Echo (ping) reply id=0x0fb1, seq=3/768, .
3.067381		Echo (ping) request id=0x0fb1, seq=4/1024, t...	ICMP: Echo (ping) request id=0x0fb1, seq=4/102.
3.067859		Echo (ping) reply id=0x0fb1, seq=4/1024, t...	ICMP: Echo (ping) reply id=0x0fb1, seq=4/1024..
5.824435		Echo (ping) request id=0x0fb1, seq=6/1536, t...	ICMP: Echo (ping) request id=0x0fb1, seq=6/153.
5.824706		Echo (ping) reply id=0x0fb1, seq=6/1536, t...	ICMP: Echo (ping) reply id=0x0fb1, seq=6/1536.
7.576445		Echo (ping) request id=0x0fb1, seq=8/2048, t...	ICMP: Echo (ping) request id=0x0fb1, seq=8/20...
7.576909		Echo (ping) reply id=0x0fb1, seq=8/2048, t...	ICMP: Echo (ping) reply id=0x0fb1, seq=8/2048.
8.544390		Echo (ping) request id=0x0fb1, seq=9/2304, t...	ICMP: Echo (ping) request id=0x0fb1, seq=9/230.
8.544830		Echo (ping) reply id=0x0fb1, seq=9/2304, t...	ICMP: Echo (ping) reply id=0x0fb1, seq=9/2304.

Figure 2: Flujo de packets en DNS ping

```

> Frame 18: 98 bytes on wire (784 bits), 98 bytes captured (784 bits)
> Ethernet II, Src: Apple_16:40:75 (28:cf:e9:16:40:75), Dst: VMware_7b:b3:c5 (00:0c:29:7b:b3:c5)
> Internet Protocol Version 4, Src: 172.20.10.2 (172.20.10.2), Dst: 172.20.10.8 (172.20.10.8)
> Internet Control Message Protocol

```

Figure 3: Evidencia de IPs y MACs origen/destino

8.1 Prueba ping (DNS)

Direccionamiento IP			Direccionamiento MAC		
Tipo de Paquete	Origen	Destino	Tipo de Paquete	Origen	Destino
Request	172.20.10.2	172.20.10.8	Request	28:CF:E9:16:40:75	00:0C:29:7B:B3:C5
Reply	172.20.10.8	172.20.10.2	Reply	00:0C:29:7B:B3:C5	28:CF:E9:16:40:75

Figure 4: Tabla de IPs y MACs origen/destino

8.1.2 Prueba de conectividad al servidor FTP

Desde el cliente se enviaron pings al servidor FTP utilizando su dirección IP. El tráfico generado se capturó y se guardó en el archivo [Ping_FTP_IP.pcap](#) El archivo fue analizado en Wireshark con el filtro icmp. Se identificaron las direcciones IP y MAC correspondientes a los paquetes de solicitud y respuesta.

Time	172.20.10.2	172.20.10.6	Comment
0.000000		Echo (ping) request id=0x0fc4, seq=1/256...	ICMP: Echo (ping) request id=0x0fc4, seq=1/256.
0.000477		Echo (ping) reply id=0x0fc4, seq=1/256, tt...	ICMP: Echo (ping) reply id=0x0fc4, seq=1/256, .
1.052587		Echo (ping) request id=0x0fc4, seq=2/512...	ICMP: Echo (ping) request id=0x0fc4, seq=2/512.
1.053065		Echo (ping) reply id=0x0fc4, seq=2/512, tt...	ICMP: Echo (ping) reply id=0x0fc4, seq=2/512, ..
12.008779		Echo (ping) request id=0x0fc4, seq=11/281...	ICMP: Echo (ping) request id=0x0fc4, seq=11/28..
12.008781		Echo (ping) request id=0x0fc4, seq=12/307...	ICMP: Echo (ping) request id=0x0fc4, seq=12/30.
12.009323		Echo (ping) reply id=0x0fc4, seq=11/2816...	ICMP: Echo (ping) reply id=0x0fc4, seq=11/281...
12.009368		Echo (ping) reply id=0x0fc4, seq=12/3072...	ICMP: Echo (ping) reply id=0x0fc4, seq=12/307..
14.659701		Echo (ping) request id=0x0fc4, seq=15/384...	ICMP: Echo (ping) request id=0x0fc4, seq=15/38.
14.660001		Echo (ping) reply id=0x0fc4, seq=15/3840...	ICMP: Echo (ping) reply id=0x0fc4, seq=15/384.
26.021105		Echo (ping) request id=0x0fc4, seq=26/665...	ICMP: Echo (ping) request id=0x0fc4, seq=26/6...
26.021603		Echo (ping) reply id=0x0fc4, seq=26/6656...	ICMP: Echo (ping) reply id=0x0fc4, seq=26/66...
26.950529		Echo (ping) request id=0x0fc4, seq=27/691...	ICMP: Echo (ping) request id=0x0fc4, seq=27/6...
26.951204		Echo (ping) reply id=0x0fc4, seq=27/6912...	ICMP: Echo (ping) reply id=0x0fc4, seq=27/691.

Figure 5: Flujo de packets en FTP ping

```

> Frame 52: 98 bytes on wire (784 bits), 98 bytes captured (784 bits)
> Ethernet II, Src: Apple_16:40:75 (28:cf:e9:16:40:75), Dst: VMware_30:c7:64 (00:50:56:30:c7:64)
> Internet Protocol Version 4, Src: 172.20.10.2 (172.20.10.2), Dst: 172.20.10.6 (172.20.10.6)
> Internet Control Message Protocol

```

Figure 6: Evidencia de IPs y MACs origen/destino

8.1 Prueba ping (FTP)

Direccionamiento IP			Direccionamiento MAC		
Tipo de Paquete	Origen	Destino	Tipo de Paquete	Origen	Destino
Request	172.20.10.2	172.20.10.6	Request	28:CF:E9:16:40:75	00:50:56:30:C7:64
Reply	172.20.10.6	172.20.10.2	Reply	00:50:56:30:C7:64	28:CF:E9:16:40:75

Figure 7: Tabla de IPs y MACs origen/destino

⚠ Los controladores de Wi-Fi de macOS no permiten que software de terceros como VMware inyecte direcciones MAC arbitrarias en la tarjeta inalámbrica. Dentro de la máquina virtual, el sistema operativo invitado cree que tiene una tarjeta de red con su propia dirección MAC, pero cuando el paquete sale de la VM y llega a la tarjeta Wi-Fi del MacBook, VMware Fusion la reemplaza por la MAC de la Wi-Fi del MacBook. `[36:08:7F:70:CA:40]`

Para evitar esta limitación, ejecuté el siguiente comando en cada VM e importé los resultados en Wireshark:

```
# En el servidor DNS
sudo tcpdump -i ens33 -nn -e -2 capture-dns-vm.pcap

# En el servidor FTP
sudo tcpdump -i ens33 -nn -e -2 capture-ftp-vm.pcap
```

Por eso existen 2 archivos `.pcap` adicionales para `Ping_DNS(VM)` y `Ping_FTP(VM)` ya a que en los archivos sin el sufijo (VM) aparece la misma MAC del MacBook `[36:08:7F:70:CA:40]` como destino en ambas pruebas!

8.2 Análisis de tráfico del Servicio DNS

En esta sección se analiza el servicio DNS, que traduce nombres de dominio en direcciones IP para facilitar la comunicación en la red. Se genera y captura tráfico con Wireshark, identificando consultas y respuestas, así como el protocolo de transporte y los puertos utilizados al acceder a un servidor web por IP y por nombre de dominio.

Cuando un cliente necesita comunicarse con un dominio, primero envía al servidor DNS una consulta de tipo *A* para obtener su dirección IPv4 y, en paralelo, una consulta de tipo *AAAA* para la dirección IPv6. El servidor responde con los registros correspondientes, que el cliente almacena en caché. Con la IP resuelta, el cliente ya puede establecer la comunicación (ej. enviar un ping) directamente al servidor destino.

8.2.1 Prueba de conectividad al Servidor Web (IP)

Cuando se accedió al servidor escribiendo directamente su dirección IP en el ping request, la conexión se estableció de inmediato ya que no fue necesario consultar al DNS y en la captura se observó únicamente tráfico ICMP entre cliente y servidor. El tráfico generado se capturó y se guardó en el archivo `Ping_WEB_IP.pcap` El archivo fue analizado en Wireshark con el filtro `icmp`.

Time	172.20.10.2	172.20.10.5	Comment
8.761403		Echo (ping) request id=0x475c, seq=1/256, ...	ICMP: Echo (ping) request id=0x475c, seq=1/256...
8.761705		Echo (ping) reply id=0x475c, seq=1/256, tt...	ICMP: Echo (ping) reply id=0x475c, seq=1/256...
10.819604		Echo (ping) request id=0x475c, seq=3/768, ...	ICMP: Echo (ping) request id=0x475c, seq=3/768...
10.819952		Echo (ping) reply id=0x475c, seq=3/768, t...	ICMP: Echo (ping) reply id=0x475c, seq=3/768...
12.873814		Echo (ping) request id=0x475c, seq=5/1280, ...	ICMP: Echo (ping) request id=0x475c, seq=5/128...
12.874236		Echo (ping) reply id=0x475c, seq=5/1280, ...	ICMP: Echo (ping) reply id=0x475c, seq=5/128...
13.910355		Echo (ping) request id=0x475c, seq=6/1536, ...	ICMP: Echo (ping) request id=0x475c, seq=6/15...
13.910696		Echo (ping) reply id=0x475c, seq=6/1536, ...	ICMP: Echo (ping) reply id=0x475c, seq=6/153...

Figure 8: Flujo de packets en WEB IP ping

```

> Frame 4: 98 bytes on wire (784 bits), 98 bytes captured (784 bits)
> Ethernet II, Src: Apple_16:40:75 (28:cf:e9:16:40:75), Dst: 36:08:7f:70:ca:40 (36:08:7f:70:ca:40)
> Internet Protocol Version 4, Src: 172.20.10.2 (172.20.10.2), Dst: 172.20.10.5 (172.20.10.5)
> Internet Control Message Protocol

```

Figure 9: Evidencia de IPs, MACs, y puertos origen/destino

En esta prueba, no hay resolución DNS.

8.2.2 Prueba de conectividad al Servidor Web (URL)

Cuando se accedió al servidor utilizando su nombre de dominio en la solicitud de ping, el cliente primero realizó consultas de tipo A y AAAA al servidor DNS para obtener la dirección IP correspondiente. Una vez resuelta, se estableció la comunicación con el servidor y en la captura se observó inicialmente el tráfico DNS seguido por el intercambio ICMP entre cliente y servidor. El tráfico generado se capturó y se guardó en el archivo [Ping_WEB.pcap](#), el cual fue analizado en Wireshark aplicando filtros para DNS e ICMP.

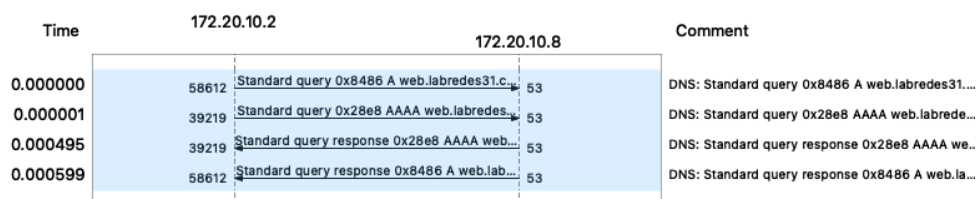


Figure 10: Flujo de packets en WEB Domain ping

```

> Frame 1: 89 bytes on wire (712 bits), 89 bytes captured (712 bits)
> Ethernet II, Src: Apple_16:40:75 (28:cf:e9:16:40:75), Dst: 36:08:7f:70:ca:40 (36:08:7f:70:ca:40)
> Internet Protocol Version 4, Src: 172.20.10.2 (172.20.10.2), Dst: 172.20.10.8 (172.20.10.8)
> User Datagram Protocol, Src Port: 58612 (58612), Dst Port: domain (53)
> Domain Name System (query)

```

Figure 11: Evidencia de IPs, MACs, y puertos origen/destino

8.2 Prueba ping (WEB-IPv4)

Direccionamiento IP		
Tipo de Paquete	Origen	Destino
Request	172.20.10.2	172.20.10.5
Reply	172.20.10.5	172.20.10.2

Direccionamiento MAC		
Tipo de Paquete	Origen	Destino
Request	28:CF:E9:16:40:75	36:08:7F:70:CA:40
Reply	36:08:7F:70:CA:40	28:CF:E9:16:40:75

Puerto		
Tipo de Paquete	Origen	Destino
Request	58612	53
Reply	53	58612

Figure 12: Tabla de IPs, MACs, y puertos origen/destino

8.3 Análisis de tráfico del Servicio FTP

En esta sección se verifica el correcto funcionamiento del servicio FTP realizando una sesión autenticada desde el cliente para descargar y subir un archivo, capturando cada fase en los archivos [FTP_download.pcap](#) y [FTP_upload.pcap](#). Usando Wireshark, se filtra el tráfico FTP para examinar los intercambios de control y de datos, y posteriormente se documentan los detalles de la capa de aplicación, el protocolo de transporte utilizado, y los puertos involucrados.

8.3.1 Conexión al servidor FTP

El servidor responde primero con el mensaje de bienvenida. El cliente intenta establecer una sesión segura con AUTH TLS/SSL, pero el servidor lo rechaza con código **530** y solicita autenticación clásica (Esto se debe a que inicialmente configuré el servidor sde forma segura con vsftpd pero deshabilité la seguridad para poder observar bien el protocolo FTP). Luego el cliente envía **USER hermione** y **PASS test123**, a lo que el servidor responde con **230 Login successful**, confirmando el acceso. Luego, se ejecutan otros

comandos donde el servidor lista sus funcionalidades soportadas (PASV, EPSV, MDTM, etc.). Todo este intercambio ocurre sobre TCP puerto 21 en el canal de control.

Time	ftp.labredes31.com	172.20.10.2	Comment
28.547238	21	Response: 220 Welcome!!Thhis is the Unian...	FTP: Response: 220 Welcome!!Thhis is the Unian...
28.565064	21	Request: AUTH TLS	FTP: Request: AUTH TLS
28.565591	21	Response: 530 Please login with USER and ...	FTP: Response: 530 Please login with USER and ..
28.589602	21	Request: AUTH SSL	FTP: Request: AUTH SSL
28.590154	21	Response: 530 Please login with USER and ...	FTP: Response: 530 Please login with USER and ..
35.812933	21	Request: USER hermione	FTP: Request: USER hermione
35.813294	21	Response: 331 Please specify the password.	FTP: Response: 331 Please specify the password.
35.846624	21	Request: PASS test123	FTP: Request: PASS test123
35.899021	21	Response: 230 Login successful.	FTP: Response: 230 Login successful.
35.911268	21	Request: SYST	FTP: Request: SYST
35.911653	21	Response: 215 UNIX Type: L8	FTP: Response: 215 UNIX Type: L8
35.921806	21	Request: FEAT	FTP: Request: FEAT
35.922207	21	Response: 211-Features:	FTP: Response: 211-Features:
35.922248	21	Response: EPRT	FTP: Response: EPRT
35.922357	21	Response: EPSV	FTP: Response: EPSV
35.922439	21	Response: MDTM	FTP: Response: MDTM
35.922507	21	Response: PASV	FTP: Response: PASV
35.922575	21	Response: REST STREAM	FTP: Response: REST STREAM
35.922649	21	Response: SIZE	FTP: Response: SIZE
35.922739	21	Response: TVFS	FTP: Response: TVFS
35.922824	21	Response: 211 End	FTP: Response: 211 End
35.962356	21	[TCP Fast Retransmission] Response: 211-F...	FTP: [TCP Fast Retransmission] Response: 211-F...

Figure 13: Inicio de sesión FTP

8.3.2 Descarga de archivo (Download)

En esta parte se observa la navegación y transferencia de un archivo en FTP. Tras el mensaje **230 Login successful**, el cliente cambia al directorio **/files** con **CWD** y el servidor confirma con **250**. Luego el cliente consulta el directorio actual y solicita modo pasivo con **8.3-FTP-download-flowPASV**. El servidor responde con la dirección y puerto a usar (**227 Entering Passive Mode**). El cliente pide descargar el archivo **a.txt** con **RETR a.txt**, el servidor abre la conexión de datos (**150 Opening data connection**) y confirma la transferencia exitosa de este archivo con **226 Transfer complete**.

Time	ftp.labredes31.com	172.20.10.2	Comment
44.602779	21	Response: 230 Login successful.	FTP: Response: 230 Login successful.
44.837013	21	Request: CWD /files	FTP: Request: CWD /files
44.837590	21	Response: 250 Directory successfully chang...	FTP: Response: 250 Directory successfully chan...
45.070873	21	Request: PWD	FTP: Request: PWD
45.071307	21	Response: 257 "/files" is the current directory	FTP: Response: 257 "/files" is the current director
45.076366	21	Request: TYPE A	FTP: Request: TYPE A
45.076953	21	Response: 200 Switching to ASCII mode.	FTP: Response: 200 Switching to ASCII mode.
45.089291	21	Request: PASV	FTP: Request: PASV
45.090593	21	Response: 227 Entering Passive Mode (172,...	FTP: Response: 227 Entering Passive Mode (172,...
45.107844	21	Request: RETR a.txt	FTP: Request: RETR a.txt
45.117497	21	Response: 150 Opening BINARY mode data ..	FTP: Response: 150 Opening BINARY mode data ..
45.145126	21	Response: 226 Transfer complete.	FTP: Response: 226 Transfer complete.

Figure 14: Descarga de archivo por usuario "hermione"

8.3.3 Carga de archivo (Upload)

Después del inicio de sesión exitoso (**230 Login successful**), el cliente cambia al directorio **/files** con **CWD /files**, confirmado por el servidor con **250 Directory successfully changed**. Luego verifica la ubicación con

PWD, y el servidor responde con **257 "/files"**. El cliente ajusta el modo de transferencia a ASCII con **TYPE A**, y el servidor responde **200 Switching to ASCII mode**. Con el comando **PASV** se abre un canal de datos en modo pasivo, indicado por la respuesta **227 Entering Passive Mode**. Al final del proceso, el cliente solicita hacer upload del archivo **b.txt** con **STOR b.txt** y el servidor responde **150 Ok to send data** y, al finalizar la transferencia, confirma con **226 Transfer complete**.

En estas capturas no mostramos el hecho de que tenemos dos usuarios: "harry" y "hermione". Los archivos de "hermione" no son visibles desde POV del usuario de "harry" y vice versa. (Esto lo configuramos en `/etc/vsftpd.conf`)

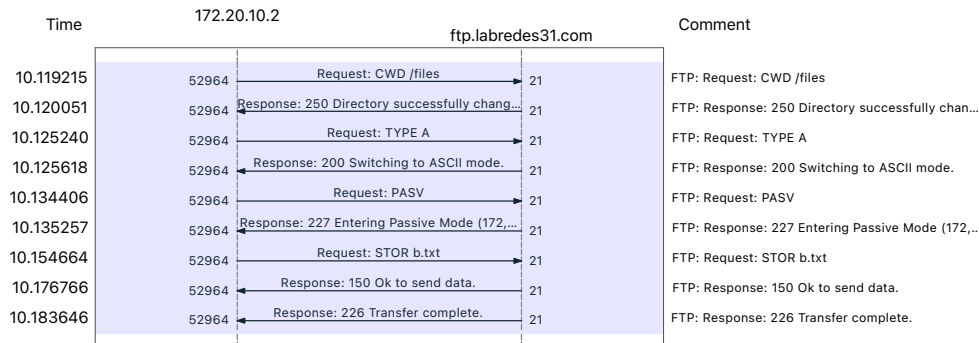


Figure 15: Descarga de archivo por usuario "hermione"

8.4 Análisis de tráfico del Servicio Web

En esta sección se analiza el funcionamiento del protocolo HTTP dentro de la topología configurada. Mediante capturas en Wireshark, se observan las peticiones y respuestas entre el cliente y el servidor web en texto claro, lo que permite identificar directamente los encabezados y contenidos intercambiados, así como los puertos y el protocolo de transporte utilizados en la comunicación. El tráfico generado se capturó y se guardó en el archivo `Ping_WEB_view.pcap`

8.4.1 Acceso al servidor web mediante HTTP

En esta captura se observa primero la resolución DNS de `web.labredes31.com`. A continuación, el cliente establece una conexión TCP con el servidor en el puerto 80, completando el three-way handshake. Luego, el cliente envía una petición **GET / HTTP/1.1** y el servidor responde con **HTTP/1.1 200 OK**, entregando la página en plaintext. Luego el cliente solicita el recurso `/favicon.ico`, que también recibe una respuesta satisfactoria **200 OK**. Finalmente, la comunicación se cierra correctamente mediante el intercambio de mensajes **FIN, ACK**, completando así el ciclo típico de una sesión HTTP.

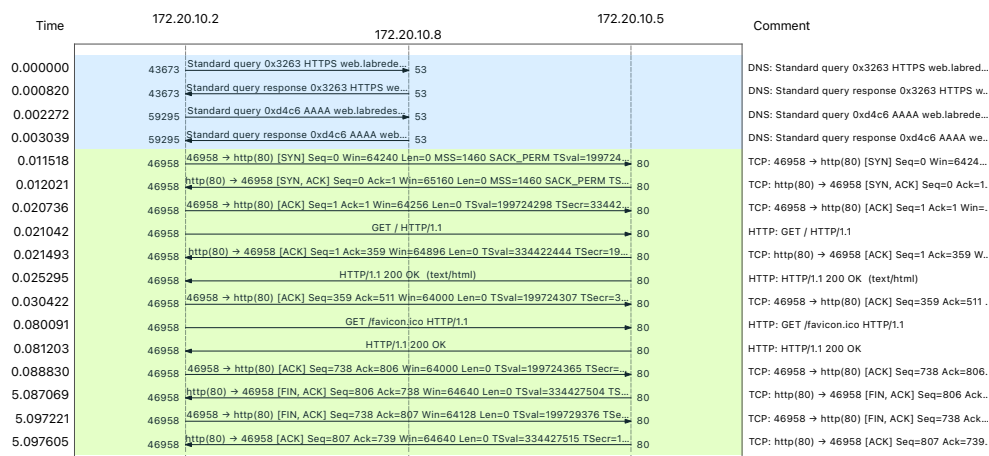


Figure 16: Flow de HTTP

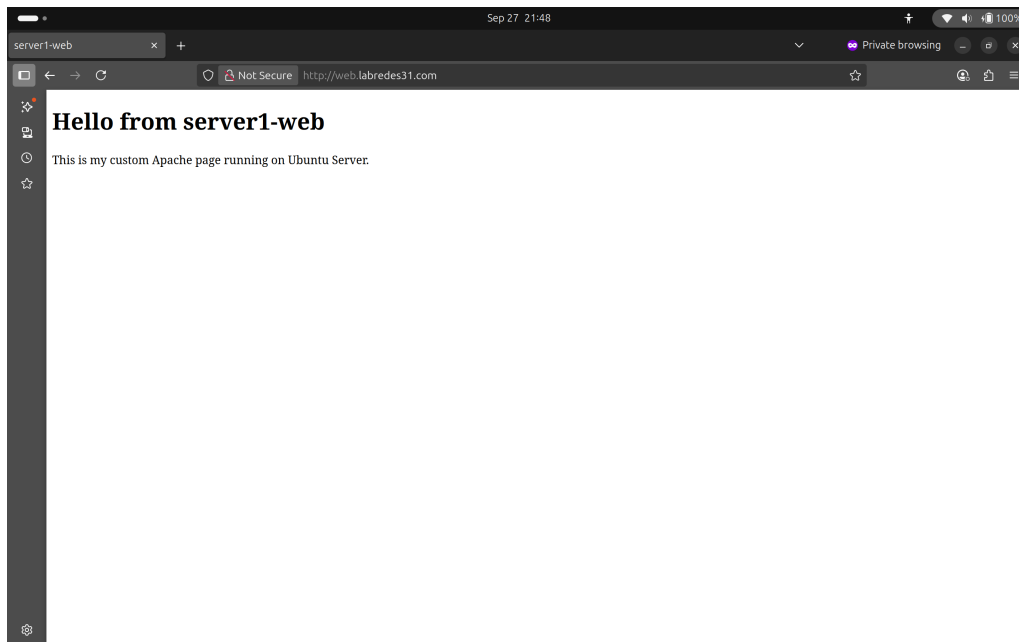


Figure 17: Vista de página web desde buscador en Ubuntu Client

8.5 Análisis del protocolo HTTPS realizando navegación en el sitio de YouTube

8.5.1 Navegación en YouTube

Se realizó la captura de tráfico mientras se navegaba en <https://www.youtube.com/>. Para reducir interferencias, se mantuvo una única pestaña activa en el navegador; se inició sesión con cuenta de Google y se realizaron acciones de navegación/comentarios. La captura se guardó como `YouTube_view.pcapng`.

- **Filtro aplicado en Wireshark:** `tcp.port == 443`.
- **Capa de aplicación (TLS):** Se observan mensajes del *handshake TLS* (*Client Hello*, *Server Hello*, *Certificate*) y posteriormente *Application Data*, que corresponde a tráfico HTTP cifrado.
- **Capa de transporte (TCP):** HTTPS se ejecuta sobre TCP, lo que provee fiabilidad mediante control de flujo, numeración de secuencia y *ACKs*.
- **Puertos:** Destino **443** (HTTPS) y puertos de origen efímeros.
- **SNI/servidores:** `youtube.com`, `accounts.youtube.com` y dominios relacionados.

Conclusión (YouTube): El tráfico está protegido mediante TLS (1.2/1.3). El contenido de peticiones/respuestas HTTP no es visible (aparece como *Application Data*), garantizando confidencialidad e integridad.

8.5.2 Navegación en otros sitios HTTPS

Se realizó una segunda captura visitando: www.elspectador.com, www.eltiempo.com, www.uniandes.edu.co y www.bancolombia.com (`HTTPS_view.pcapng`). En todos los casos se evidenció TLS sobre TCP/443, con presencia de *Client Hello/Server Hello/Application Data*.

Conclusión (otros sitios): Todos emplean HTTPS; se identificaron versiones TLS 1.2/1.3 según el host. El cifrado impide visualizar el contenido HTTP en claro.

Representación gráfica de flujos

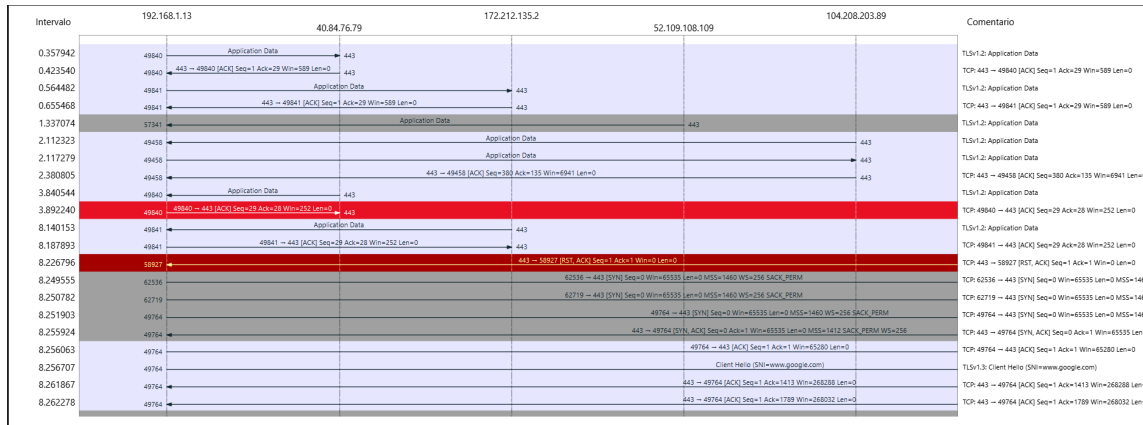


Figure 18: Flujo de comunicación durante la navegación en YouTube. Se aprecian múltiples *Application Data* (tráfico cifrado) y *ACKs* propios de TCP hacia distintos hosts de Google/YouTube.

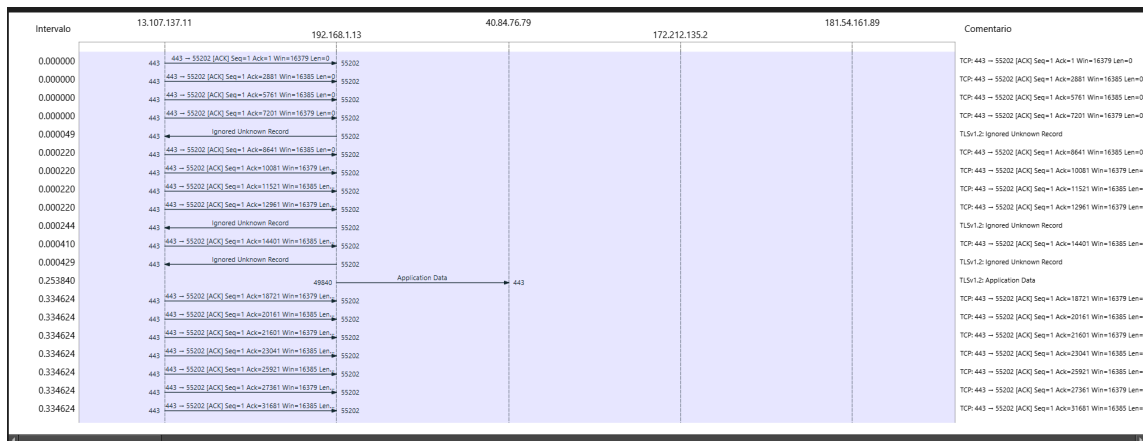


Figure 19: Flujo durante la navegación en sitios HTTPS (El Espectador, El Tiempo, Uniandes y Bancolombia). Se observa el establecimiento de conexión (SYN/SYN-ACK/ACK) y el posterior handshake TLS (por ejemplo, Client Hello).

Análisis de la Capa de Aplicación (TLS)

Durante el *handshake TLS*, el cliente envía un **Client Hello** indicando versiones soportadas (TLS 1.2/1.3), *cipher suites* y el nombre del servidor (*SNI*). El servidor responde con **Server Hello** seleccionando parámetros criptográficos y entrega su *Certificate* (X.509) para autenticación. Tras la negociación de claves, la sesión opera con **Application Data** (HTTP cifrado).

```

▶ Frame 90148: 1880 bytes on wire (15040 bits), 1880 bytes captured (15040 bits) on interface \Device\NPF
▶ Ethernet II, Src: Intel_3d:e4:d3 (ac:19:8e:3d:e4:d3), Dst: zte_74:1a:b4 (98:00:6a:74:1a:b4)
▶ Internet Protocol Version 4, Src: 192.168.1.13, Dst: 104.21.57.147
▶ Transmission Control Protocol, Src Port: 49686, Dst Port: 443, Seq: 1, Ack: 1, Len: 1826
▼ Transport Layer Security
  ▼ TLSv1 Record Layer: Handshake Protocol: Client Hello
    Content Type: Handshake (22)
    Version: TLS 1.0 (0x0301)
    Length: 1821
    ▼ Handshake Protocol: Client Hello
      Handshake Type: Client Hello (1)
      Length: 1817
      ▶ Version: TLS 1.2 (0x0303)
      ▶ Random: a84c8eade7850600c5bfd0c6d346ab8dca4b98378b9ec2c29075f48191dbf09
      ▶ Session ID Length: 32
      ▶ Session ID: 2baa277ec6d7c88bade302d67b5a5e37d0e295a3dd70148627d3a041bdf48690
      ▶ Cipher Suites Length: 32
      ▶ Cipher Suites (16 suites)
      ▶ Compression Methods Length: 1
      ▶ Compression Methods (1 method)
      ▶ Extensions Length: 1712
      ▶ Extension: Reserved (GREASE) (len=0)
      ▶ Extension: session_ticket (len=0)
      ▶ Extension: supported_groups (len=12)
      ▶ Extension: supported_versions (len=7) TLS 1.3, TLS 1.2
      ▶ Extension: psk_key_exchange_modes (len=2)
      ▶ Extension: signature_algorithms (len=18)
      ▶ Extension: key_share (len=1263) X25519MLKEM768, x25519
      ▶ Extension: encrypted_client_hello (len=282)
      ▶ Extension: compress_certificate (len=3)
      ▶ Extension: extended_master_secret (len=0)
      ▶ Extension: status_request (len=5)
      ▶ Extension: ec_point_formats (len=2)
      ▶ Extension: server_name (len=25) name=www.uniandes.edu.com
      ▶ Extension: application_layer_protocol_negotiation (len=14)
      ▶ Extension: Unknown type 17613 (len=5)
      ▶ Extension: signed_certificate_timestamp (len=0)
      ▶ Extension: renegotiation_info (len=1)
      ▶ Extension: Reserved (GREASE) (len=1)

```

Figure 20: *Client Hello* hacia *www.uniandes.edu.co*. Se observan extensiones TLS (p. ej., *supported_versions*, *server_name*).

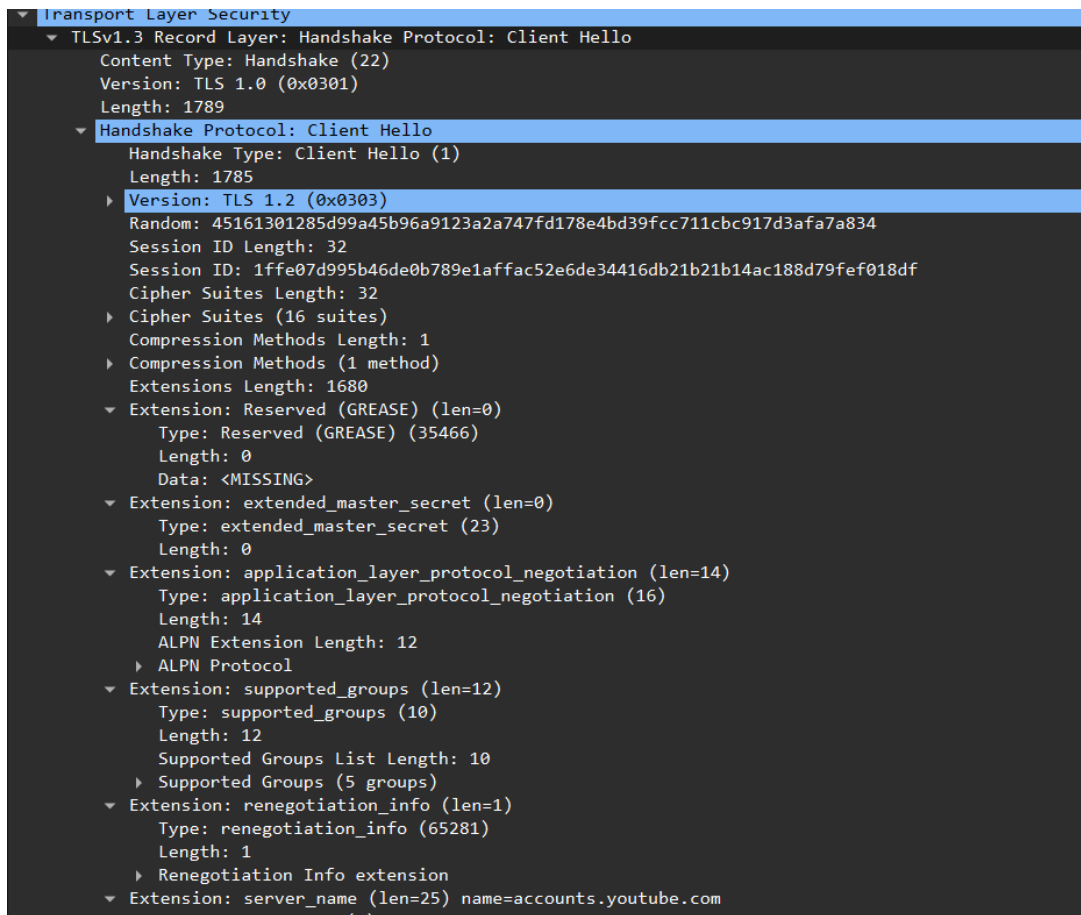


Figure 21: *Client Hello* hacia *accounts.youtube.com*. Se evidencia el uso de TLS 1.2/1.3, cipher suites ofrecidas y el SNI del host.

Implicación: la capa de aplicación expone la negociación TLS, pero el contenido HTTP queda cifrado.

Análisis de la Capa de Transporte (TCP)

HTTPS requiere de TCP para asegurar fiabilidad (entrega ordenada, control de congestión) y se inicia con el *three-way handshake* (SYN, SYN-ACK, ACK). En las capturas se aprecian campos como puertos origen/destino, números de secuencia, ventana y flags (PSH, ACK), además del *payload* que TLS cifra.

No.	Time	Source	Destination	Protocol	Length	Info
1048...	180.091295	192.168.1.13	45.60.247.99	TCP	54	55373 → 443 [ACK] Seq=1 Ack=1 Win=65280
1048...	180.091314	192.168.1.13	45.60.247.99	TCP	54	65023 → 443 [ACK] Seq=1 Ack=1 Win=65280
1048...	180.091323	192.168.1.13	45.60.247.99	TCP	54	58958 → 443 [ACK] Seq=1 Ack=1 Win=65280
1048...	180.091910	192.168.1.13	45.60.247.99	TLSv1.3	2118	Client Hello (SNI=www.bancolombia.com)
1048...	180.092351	192.168.1.13	45.60.247.99	TLSv1.3	2054	Client Hello (SNI=www.bancolombia.com)
1048...	180.092437	45.60.247.99	192.168.1.13	TCP	60	443 → 53020 [ACK] Seq=23065 Ack=5242 Wi
1048...	180.092437	45.60.247.99	192.168.1.13	TCP	60	443 → 53020 [ACK] Seq=23065 Ack=5581 Wi
1048...	180.092729	192.168.1.13	45.60.247.99	TLSv1.3	1879	Client Hello (SNI=www.bancolombia.com)
1048...	180.093769	3.174.238.20	192.168.1.13	TLSv1.3	203	Application Data, Application Data
1048...	180.095609	192.168.1.13	3.174.238.20	TLSv1.3	118	Change Cipher Spec, Application Data
1048...	180.097513	18.155.252.59	192.168.1.13	TLSv1.3	85	Application Data

▶ Frame 104834: 2118 bytes on wire (16944 bits), 2118 bytes captured (16944 bits) on interface \Device\NPF	0020	f7 6
▶ Ethernet II, Src: Intel_3d:e4:d3 (ac:19:8e:3d:e4:d3), Dst: zte_74:1a:b4 (98:00:6a:74:1a:b4)	0030	00 f
▶ Internet Protocol Version 4, Src: 192.168.1.13, Dst: 45.60.247.99	0040	03 7
▼ Transmission Control Protocol, Src Port: 55373, Dst Port: 443, Seq: 1, Ack: 1, Len: 2064	0050	00 2
Source Port: 55373	0060	bd 2
Destination Port: 443	0070	c6 7
[Stream index: 3459]	0080	ed 9
[Stream Packet Number: 4]	0090	c0 2
▶ [Conversation completeness: Complete, WITH_DATA (63)]	00a0	00 2
[TCP Segment Len: 2064]	00b0	00 2
Sequence Number: 1 (relative sequence number)	00c0	68 7
Sequence Number (raw): 3345126372	00d0	02 0
[Next Sequence Number: 2065 (relative sequence number)]	00e0	00 1
Acknowledgment Number: 1 (relative ack number)	00f0	6d 6
Acknowledgment number (raw): 4255340720	0100	11 e
0101 = Header Length: 20 bytes (5)	0110	32 0
▶ Flags: 0x018 (PSH, ACK)	0120	03 0
Window: 255	0130	c0 5
[Calculated window size: 65280]	0140	43 5
[Window size scaling factor: 256]	0150	81 1
Checksum: 0xe65b [unverified]	0160	79 2
[Checksum Status: Unverified]	0170	05 c
Urgent Pointer: 0	0180	03 1
▶ [Timestamps]	0190	58 8
▶ [SEQ/ACK analysis]	01a0	c8 7
TCP payload (2064 bytes)	01b0	04 9
▶ Transport Layer Security	01c0	63 3
	01d0	39 f
	01e0	a7 8

Figure 22: Detalle TCP hacia www.bancolombia.com: puertos, números de secuencia/ack, ventana y TCP payload (transportado por TLS).

▼ Transmission Control Protocol, Src Port: 56970, Dst Port: 443, Seq: 1, Ack: 1, Len: 1794
Source Port: 56970
Destination Port: 443
[Stream index: 19]
[Stream Packet Number: 4]
▶ [Conversation completeness: Complete, WITH_DATA (31)]
[TCP Segment Len: 1794]
Sequence Number: 1 (relative sequence number)
Sequence Number (raw): 1709464199
[Next Sequence Number: 1795 (relative sequence number)]
Acknowledgment Number: 1 (relative ack number)
Acknowledgment number (raw): 1053655899
0101 = Header Length: 20 bytes (5)
▶ Flags: 0x018 (PSH, ACK)
Window: 255
[Calculated window size: 65280]
[Window size scaling factor: 256]
Checksum: 0x8d43 [unverified]
[Checksum Status: Unverified]
Urgent Pointer: 0
▶ [Timestamps]
▶ [SEQ/ACK analysis]
TCP payload (1794 bytes)

Figure 23: Paquete TCP en conexión a YouTube mostrando longitud de segmento, flags y TCP payload; el contenido de aplicación va cifrado por TLS.

Evidencia de paquetes HTTPS por sitio web

No.	Time	Source	Destination	Protocol	Length	Info
1048.	180.087518	192.168.1.13	13.227.26.13	TCP	54	58944 → 443 [ACK] Seq=1794 Ack=5910 Win=65280 Len=0
1048.	180.087531	192.168.1.13	18.155.252.59	TCP	54	57903 → 443 [ACK] Seq=2388 Ack=6151 Win=65280 Len=0
1048.	180.087538	192.168.1.13	3.174.238.20	TCP	54	51997 → 443 [ACK] Seq=1828 Ack=5761 Win=65280 Len=0
1048.	180.087763	192.168.1.13	18.155.252.59	TLSv1.3	85	Application Data
1048.	180.089290	192.168.1.13	13.227.26.13	TLSv1.3	118	Change Cipher Spec, Application Data
1048.	180.091179	45.60.247.99	192.168.1.13	TCP	60	443 → 51172 [ACK] Seq=53228 Ack=3926 Win=64128 Len=0
1048.	180.091179	45.60.247.99	192.168.1.13	TCP	66	443 → 55933 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 SACK_PERM WS=128
1048.	180.091179	45.60.247.99	192.168.1.13	TCP	66	443 → 65023 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 SACK_PERM WS=128
1048.	180.091179	45.60.247.99	192.168.1.13	TCP	66	443 → 58958 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 SACK_PERM WS=128
1048.	180.091295	192.168.1.13	45.60.247.99	TCP	54	55373 → 443 [ACK] Seq=1 Ack=1 Win=65280 Len=0
1048.	180.091314	192.168.1.13	45.60.247.99	TCP	54	65023 → 443 [ACK] Seq=1 Ack=1 Win=65280 Len=0
1048.	180.091323	192.168.1.13	45.60.247.99	TCP	54	58958 → 443 [ACK] Seq=1 Ack=1 Win=65280 Len=0
1048.	180.091910	192.168.1.13	45.60.247.99	TLSv1.3	2118	Client Hello (SNI=www.bancolombia.com)
1048.	180.092351	192.168.1.13	45.60.247.99	TLSv1.3	2054	Client Hello (SNI=www.bancolombia.com)
1048.	180.092437	45.60.247.99	192.168.1.13	TCP	60	443 → 53020 [ACK] Seq=23805 Ack=5581 Win=64128 Len=0
1048.	180.092729	192.168.1.13	45.60.247.99	TLSv1.3	1879	Client Hello (SNI=www.bancolombia.com)
1048.	180.093769	3.174.238.20	192.168.1.13	TLSv1.3	203	Application Data, Application Data
1048.	180.095680	192.168.1.13	3.174.238.20	TLSv1.3	118	Change Cipher Spec, Application Data
1048.	180.097513	18.155.252.59	192.168.1.13	TLSv1.3	85	Application Data
1048.	180.097513	18.155.252.59	192.168.1.13	TCP	60	[TCP Previous segment not captured] 443 → 53301 [FIN, ACK] Seq=6089 Ack=1892 Win=69632 Len=0
1048.	180.097603	192.168.1.13	18.155.252.59	TCP	54	[TCP Dup ACK 104789#1] 53301 → 443 [ACK] Seq=1892 Ack=5910 Win=65280 Len=0
1048.	180.098112	18.155.252.59	192.168.1.13	TCP	66	443 → 53301 [ACK] Seq=5910 Ack=1892 Win=69632 Len=0
1048.	180.098112	18.155.252.59	192.168.1.13	TCP	60	443 → 53301 [ACK] Seq=5910 Ack=1892 Win=69632 Len=0
1048.	180.100445	18.155.252.59	192.168.1.13	TCP	233	[TCP Out-Of-Order] 443 → 53301 [PSH, ACK] Seq=5910 Ack=1892 Win=69632 Len=179

Figure 24: *Client Hello* hacia www.bancolombia.com (TLS 1.3). Evidencia del inicio del handshake y establecimiento de un canal cifrado adecuado para datos financieros.

9502	34.393065	192.168.1.13	13.107.137.11	TLSv1.2	2934	Application Data
9503	34.394320	13.107.137.11	192.168.1.13	TCP	60	443 → 55202 [ACK] Seq=1 Ack=12962491 Win=16385 Len=0
9504	34.394320	13.107.137.11	192.168.1.13	TCP	60	443 → 55202 [ACK] Seq=1 Ack=12963931 Win=16379 Len=0
9505	34.394320	13.107.137.11	192.168.1.13	TCP	60	443 → 55202 [ACK] Seq=1 Ack=12965371 Win=16385 Len=0
9506	34.394357	192.168.1.13	13.107.137.11	TCP	7254	55202 → 443 [ACK] Seq=12972571 Ack=1 Win=255 Len=7200 [TCP PDU reassembled in 9545]
9518	34.435023	192.168.1.13	172.217.162.99	TCP	66	62079 → 443 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM
9519	34.435401	192.168.1.13	181.54.160.144	TCP	66	60979 → 443 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM
9520	34.439165	181.54.160.144	192.168.1.13	TCP	66	443 → 60979 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1384 SACK_PERM WS=128
9521	34.439165	172.217.162.99	192.168.1.13	TCP	66	443 → 62079 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1412 SACK_PERM WS=256
9522	34.439274	192.168.1.13	181.54.160.144	TCP	54	60979 → 443 [ACK] Seq=1 Ack=1 Win=65280 Len=0
9523	34.439298	192.168.1.13	172.217.162.99	TCP	54	62079 → 443 [ACK] Seq=1 Ack=1 Win=65280 Len=0
9524	34.439641	192.168.1.13	172.217.162.99	TLSv1.3	2308	Client Hello (SNI=beacons.gcp.gvt2.com)
9525	34.439966	192.168.1.13	181.54.160.144	TLSv1.2	2199	Client Hello (SNI=www.elespectador.com)
9526	34.445356	172.217.162.99	192.168.1.13	TCP	60	443 → 62079 [ACK] Seq=1 Ack=1413 Win=268288 Len=0
9527	34.445356	172.217.162.99	192.168.1.13	TCP	60	443 → 62079 [ACK] Seq=1 Ack=1413 Win=268288 Len=0
9528	34.460351	192.168.1.13	181.54.160.144	TCP	1438	[TCP Retransmission] 60979 → 443 [PSH, ACK] Seq=762 Ack=1 Win=65280 Len=1384
9529	34.468760	192.168.1.13	172.212.135.2	TLSv1.2	85	Application Data
9532	34.691989	13.107.137.11	192.168.1.13	TCP	60	443 → 55202 [ACK] Seq=1 Ack=12966811 Win=16379 Len=0
9533	34.691989	13.107.137.11	192.168.1.13	TCP	60	443 → 55202 [ACK] Seq=1 Ack=12969691 Win=16385 Len=0
9534	34.691989	13.107.137.11	192.168.1.13	TCP	60	443 → 55202 [ACK] Seq=1 Ack=12971131 Win=16379 Len=0
9535	34.691989	13.107.137.11	192.168.1.13	TCP	60	443 → 55202 [ACK] Seq=1 Ack=12978331 Win=16385 Len=0
9536	34.691989	13.107.137.11	192.168.1.13	TCP	60	443 → 55202 [ACK] Seq=1 Ack=12979771 Win=16379 Len=0
9537	34.691989	181.54.160.144	192.168.1.13	TCP	60	[TCP Previous segment not captured] 443 → 60979 [ACK] Seq=265 Ack=2146 Win=64128 Len=0 SFI=762 SRI=2146
9538	34.691989	172.217.162.99	192.168.1.13	TLSv1.3	1466	Server Hello, Change Cipher Spec
9539	34.691989	172.217.162.99	192.168.1.13	TLSv1.3	111	Application Data
9541	34.691989	172.217.162.99	192.168.1.13	TCP	144	[TCP Retransmission] 443 → 60979 [ACK] Seq=1 Ack=1413 Win=268288 Len=0 [TCP PDU reassembled in 9536]

Figure 25: *Conexión a* www.elespectador.com. Se visualiza *Client Hello* (TLS 1.2) y, a continuación, *Application Data* (HTTP sobre TLS).

No.	Time	Source	Destination	Protocol	Length	Info
41132	70.899455	143.244.35.229	192.168.1.13	TLSv1.3	914	Application Data
41134	70.899523	192.168.1.13	143.244.35.229	TCP	54	56749 → 443 [ACK] Seq=2616 Ack=8016 Win=65280 Len=0
41146	70.905457	148.251.178.210	192.168.1.13	TCP	66	443 → 62169 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 SACK_PERM WS=256
41147	70.905535	192.168.1.13	148.251.178.210	TCP	54	62169 → 443 [ACK] Seq=1 Ack=1 Win=65280 Len=0
41149	70.906087	192.168.1.13	148.251.178.210	TLSv1.3	1926	Client Hello (SNI=analytics.smas.com)
41153	70.924913	192.168.1.13	138.199.8.196	TCP	66	55639 → 443 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM
41166	70.922217	192.168.1.13	172.200.168.128	TCP	66	62549 → 443 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM
41172	71.029529	138.199.8.196	192.168.1.13	TCP	66	443 → 55639 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 SACK_PERM WS=2048
41178	71.029726	192.168.1.13	138.199.8.196	TCP	54	55639 → 443 [ACK] Seq=1 Ack=1 Win=65280 Len=0
41179	71.030716	192.168.1.13	138.199.8.196	TLSv1.3	2101	Client Hello (SNI=static.summedia.tv)
41180	71.061928	172.200.168.128	192.168.1.13	TCP	66	443 → 62549 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1440 SACK_PERM WS=1024
41181	71.062167	172.200.168.128	192.168.1.13	TCP	54	62549 → 443 [ACK] Seq=1 Ack=1 Win=65280 Len=0
41182	71.062327	192.168.1.13	172.200.168.128	TLSv1.3	1776	Client Hello (SNI=eltiempo.com)
41183	71.072657	148.251.178.210	192.168.1.13	TCP	60	443 → 62169 [ACK] Seq=1 Ack=1461 Win=68608 Len=0
41184	71.072657	148.251.178.210	192.168.1.13	TCP	60	443 → 62169 [ACK] Seq=1 Ack=1873 Win=71424 Len=0
41185	71.073607	148.251.178.210	192.168.1.13	TLSv1.3	292	Server Hello, Change Cipher Spec, Application Data, Application Data
41186	71.074064	192.168.1.13	148.251.178.210	TLSv1.3	118	Change Cipher Spec, Application Data
41187	71.074323	192.168.1.13	148.251.178.210	TLSv1.3	146	Application Data
41188	71.074828	192.168.1.13	148.251.178.210	TLSv1.3	431	Application Data
41189	71.074876	192.168.1.13	148.251.178.210	TLSv1.3	114	Application Data
41190	71.140932	172.200.168.128	192.168.1.13	TCP	60	443 → 62549 [ACK] Seq=1 Ack=1441 Win=63488 Len=0
41191	71.140932	172.200.168.128	192.168.1.13	TCP	60	443 → 62549 [ACK] Seq=1 Ack=1723 Win=64512 Len=0
41192	71.142222	172.200.168.128	192.168.1.13	TLSv1.3	1514	Server Hello, Change Cipher Spec, Application Data
41193	71.142222	172.200.168.128	192.168.1.13	TCP	1514	443 → 62549 [PSH, ACK] Seq=1461 Ack=1723 Win=64512 Len=1460 [TCP PDU reassembled in 41194]
41194	71.142222	172.200.168.128	192.168.1.13	TLSv1.3	1014	Application Data, Application Data, Application Data
41196	71.142338	192.168.1.13	172.200.168.128	TCP	54	62549 → 443 [ACK] Seq=1785 Ack=7061 Win=65280 Len=0

Figure 26: Tráfico hacia www.eltiempo.com: *Client Hello* en TLS 1.3 seguido de *Server Hello/Change Cipher Spec y Application Data*.

90127	118.155123	142.251.128.14	192.168.1.13	TCP	1466	443 → 52854 [ACK] Seq=4237 Ack=1721 Win=268032 Len=1412 [TCP PDU reassembled in 90338]
90128	118.155123	142.251.128.14	192.168.1.13	TCP	1466	443 → 52854 [PSH, ACK] Seq=5649 Ack=1721 Win=268032 Len=1412 [TCP PDU reassembled in 90338]
90129	118.155123	142.251.128.14	192.168.1.13	TCP	1466	443 → 57525 [ACK] Seq=1413 Ack=1785 Win=268032 Len=1412 [TCP PDU reassembled in 90335]
90130	118.155158	192.168.1.13	142.251.128.14	TCP	54	52854 → 443 [ACK] Seq=1721 Ack=7061 Win=65280 Len=0
90131	118.157007	142.251.128.14	192.168.1.13	TCP	1466	443 → 57525 [PSH, ACK] Seq=4237 Ack=1785 Win=268032 Len=1412 [TCP PDU reassembled in 90335]
90132	118.157007	142.251.128.14	192.168.1.13	TCP	1466	443 → 57525 [ACK] Seq=4237 Ack=1785 Win=268032 Len=1412 [TCP PDU reassembled in 90335]
90133	118.157007	142.251.128.14	192.168.1.13	TCP	1466	443 → 57525 [PSH, ACK] Seq=5649 Ack=1785 Win=268032 Len=1412 [TCP PDU reassembled in 90335]
90135	118.157038	192.168.1.13	142.251.128.14	TCP	54	57525 → 443 [ACK] Seq=1785 Ack=7061 Win=65280 Len=0
90144	118.160875	104.21.57.147	192.168.1.13	TCP	66	443 → 49068 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1400 SACK_PERM WS=8192
90145	118.160875	104.21.57.147	192.168.1.13	TCP	66	443 → 49068 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1400 SACK_PERM WS=8192
90146	118.170130	192.168.1.13	104.21.57.147	TCP	54	49068 → 443 [ACK] Seq=1 Ack=1 Win=65280 Len=0
90147	118.170226	192.168.1.13	104.21.57.147	TCP	54	64008 → 443 [ACK] Seq=1 Ack=1 Win=65280 Len=0
90148	118.170226	192.168.1.13	104.21.57.147	TLSv1.3	1880	Client Hello (SNI=www.uniandes.edu.co)
90149	118.171209	192.168.1.13	104.21.57.147	TLSv1.3	1880	Client Hello (SNI=www.uniandes.edu.co)
90151	118.179507	192.99.55.225	192.168.1.13	TLSv1.3	1514	Application Data, Application Data
90153	118.179507	20.57.95.139	192.168.1.13	TCP	1514	443 → 60379 [ACK] Seq=4308396 Ack=3104 Win=1460 [TCP PDU reassembled in 90204]
90154	118.179507	20.57.95.139	192.168.1.13	TLSv1.3	966	Application Data
90155	118.183000	192.168.1.13	40.84.76.79	TCP	54	49068 → 443 [ACK] Seq=361 Ack=217 Win=254 Len=0
90156	118.183170	192.168.1.13	34.107.254.252	TCP	54	53353 → 443 [ACK] Seq=2084 Ack=831 Win=64512 Len=0
90158	118.187027	20.57.95.139	192.168.1.13	TCP	1514	443 → 55933 [ACK] Seq=5054321 Ack=5074 Win=64512 Len=1460 [TCP PDU reassembled in 90196]
90159	118.187027	20.57.95.139	192.168.1.13	TCP	602	443 → 55933 [PSH, ACK] Seq=5055781 Ack=5074 Win=64512 Len=548 [TCP PDU reassembled in 90196]
90160	118.187027	20.57.95.139	192.168.1.13	TCP	1514	443 → 55933 [ACK] Seq=5056219 Ack=5074 Win=64512 Len=1460 [TCP PDU reassembled in 90196]
90161	118.187134	192.168.1.13	20.57.95.139	TCP	54	55933 → 443 [ACK] Seq=5074 Ack=5057789 Win=261888 Len=0
90162	118.191753	20.57.95.139	192.168.1.13	TCP	1514	443 → 55933 [ACK] Seq=5057789 Ack=5074 Win=64512 Len=1460 [TCP PDU reassembled in 90196]
90163	118.191753	20.57.95.139	192.168.1.13	TCP	1514	443 → 55933 [PSH, ACK] Seq=5059249 Ack=5074 Win=64512 Len=1460 [TCP PDU reassembled in 90196]

Evidencia de paquetes en YouTube

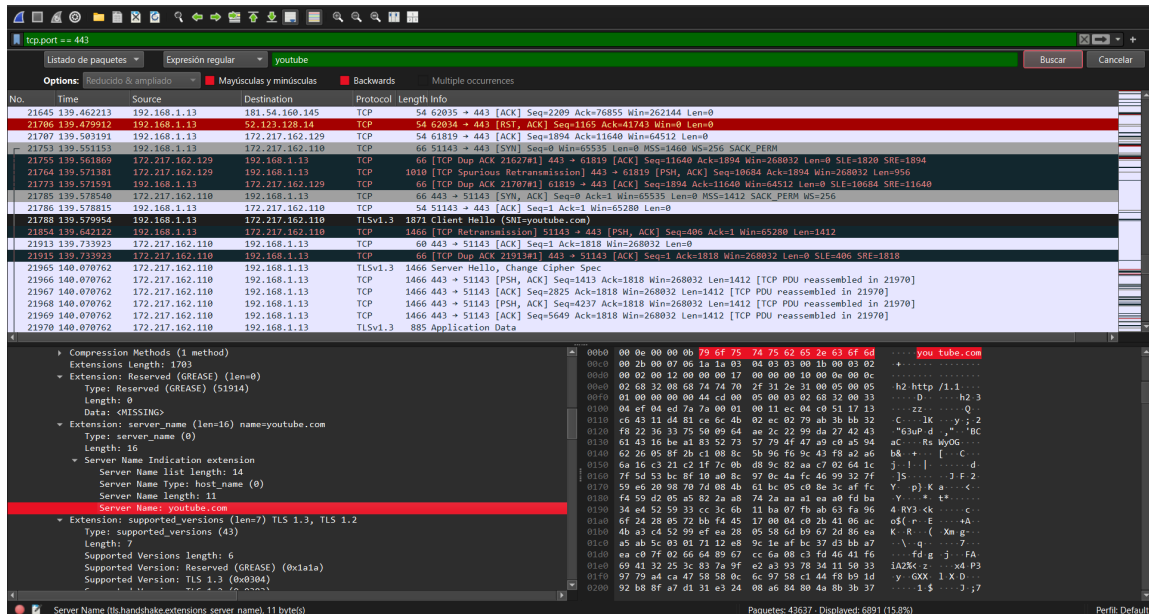


Figure 28: Listado filtrado por `tcp.port == 443` con conexiones a `youtube.com`. Se observan `handshake TLS` y posteriormente `Application Data`.

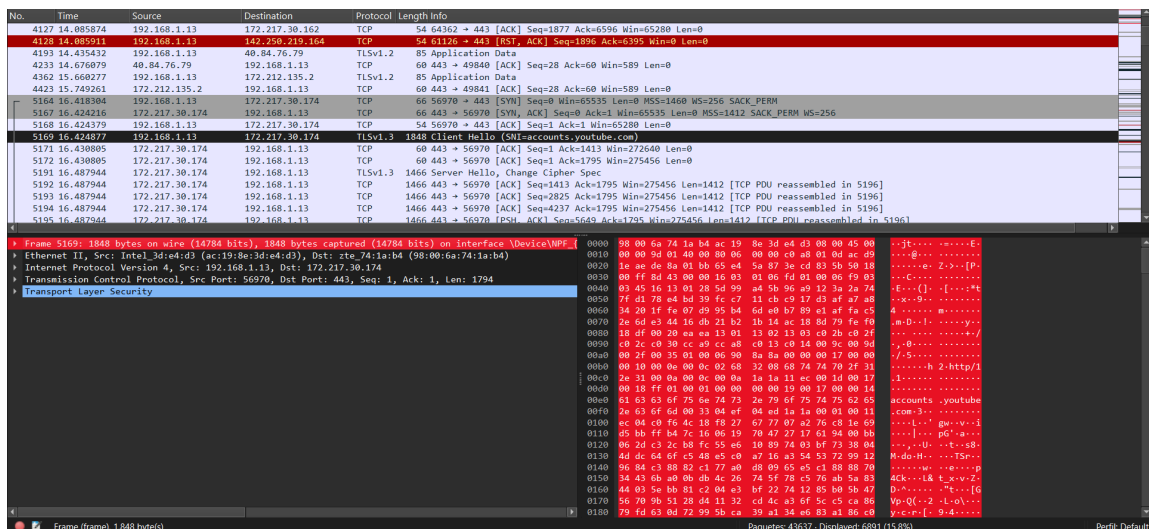


Figure 29: Detalle de un paquete TLS asociado a YouTube: la capa de aplicación aparece como `Application Data`, confirmando el cifrado del contenido HTTP.

Ejemplos de paquetes (tabla)

Tabla 1: Ejemplos representativos de paquetes capturados (YouTube)

Fuente	Destino	Protocolo	Puerto Origen	Puerto Destino
192.168.1.13	YouTube/Google IP	TCP (TLS 1.3)	efímero	443
192.168.1.13	YouTube/Google IP	TCP (TLS 1.2)	efímero	443

Tabla 2: Ejemplos representativos de paquetes (otros sitios HTTPS)

Fuente	Destino	Protocolo	Puerto Origen	Puerto Destino
192.168.1.13	<i>bancolombia.com (IP)</i>	TCP (TLS 1.3)	<i>efímero</i>	443
192.168.1.13	<i>uniandes.edu.co (IP)</i>	TCP (TLS 1.2/1.3)	<i>efímero</i>	443

Conclusiones generales

- HTTPS combina HTTP con TLS sobre TCP/443, garantizando **confidencialidad, integridad y autenticación** del servidor mediante certificados X.509.
- Las capturas muestran el **handshake TLS** (**Client Hello, Server Hello, Certificate**) seguido de **Application Data** (HTTP cifrado), por lo cual el contenido de aplicación no es legible.
- El uso de TCP evidencia control de fiabilidad (SYN/SYN-ACK/ACK, números de secuencia, ventanas y ACKs).
- En YouTube se observan múltiples flujos cifrados (**Application Data**) por la naturaleza de *streaming* y recursos; en los demás sitios se confirma el mismo patrón de seguridad.

8.6 Análisis del protocolo VoIP

8.6.1 Establecimiento de la llamada

8.7 Análisis del protocolo RTMP

8.7.1 Inicio de la transmisión

References

- [1] Computer Networking, a top-down approach. James Kurose, Keith Ross. Addison-Wesley, 6th ed.