

Utilizando Layer7

Para Filter y QoS

Sobre nosotros



Cursos oficiales

- Dictamos entrenamientos oficiales en modalidad presencial y virtual de las marcas MikroTik, Ubiquiti, EigoWave y redes TCP/IP.

[Ver catálogo](#)



Consultoría

Contamos con un grupo de consultores profesionales con conocimiento y experiencia en las áreas de redes de datos y TI.

[Más información](#)



Soporte

A través de nuestro Centro de Soporte, alineamos los recursos necesarios para ofrecer asistencia a redes ISP y empresariales.

[Consultar planes](#)

Objetivo de esta presentación

- Poder detectar y marcar tráfico en base al contenido de alguno de los paquetes de una conexión.
- Aplicar reglas de **Filter**.
- Aplicar políticas de **QoS**.
- En esta presentación se darán ejemplos con sitios web.

Agenda

- **Introducción**
- **Comunicación con un sitio web**
 - Comunicación DNS
 - Comunicación HTTP
 - Comunicación HTTPs
 - Comunicación QUIC
- **Layer7 en acción**
 - Detección con Layer7
 - Ejemplos con Filter y QoS
- **Consideraciones especiales**
 - Compatibilidad con IPv6
 - DNS gestionado

Tipo de comunicaciones

Introducción

Algunos mitos y verdades:

“Layer7 no funciona con tráfico HTTPs”

Cierto, aunque de todos modos algo puede hacerse.

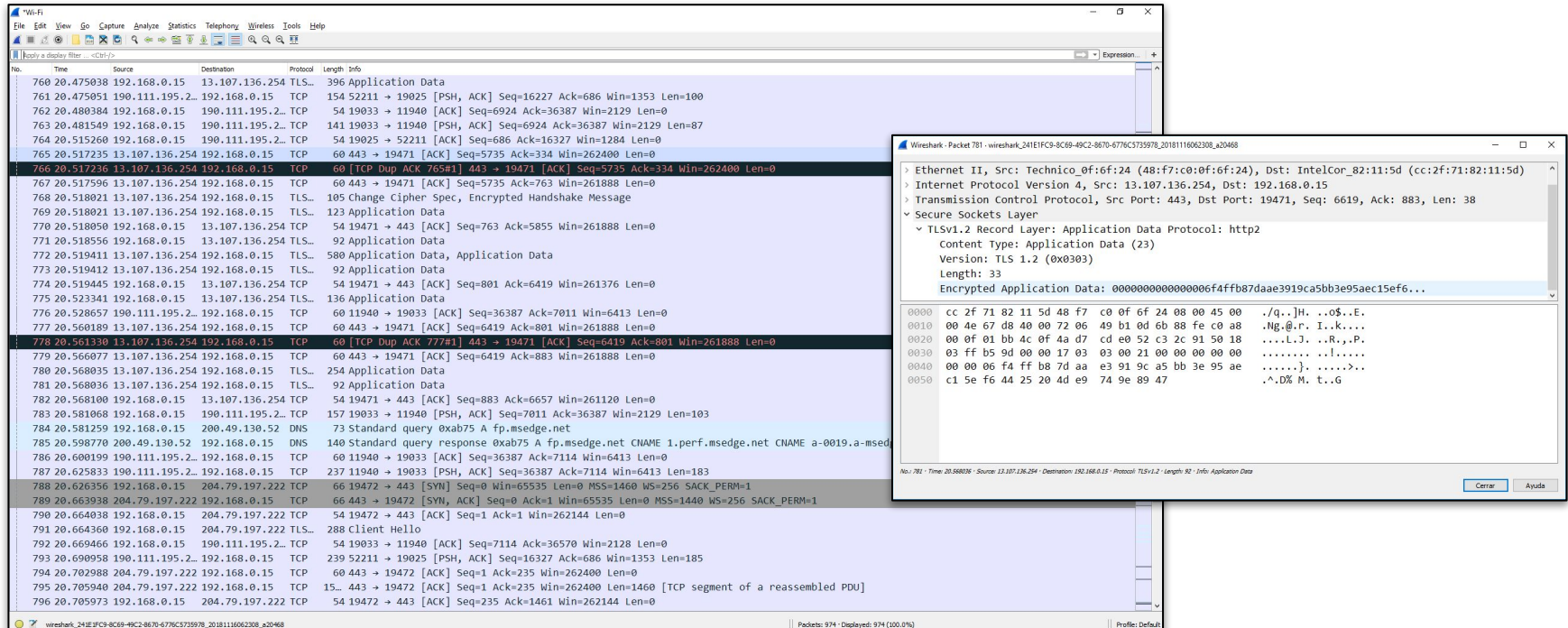
“Layer7 no funciona con servicios de Google”

Falso, el problema viene de asociado al protocolo QUIC.

“Layer7 me carga mucho el CPU”

Cierto, pero bien configurado podemos evitar el problema.

- Para conocer cómo funcionan estas comunicaciones utilizamos Wireshark:



Comunicación con un sitio Web

- Sitio común HTTP/HTTPS

1) Consulta DNS
(UDP 53)

2) Conexión HTTP/HTTPS
(TCP 80 o TCP 443)

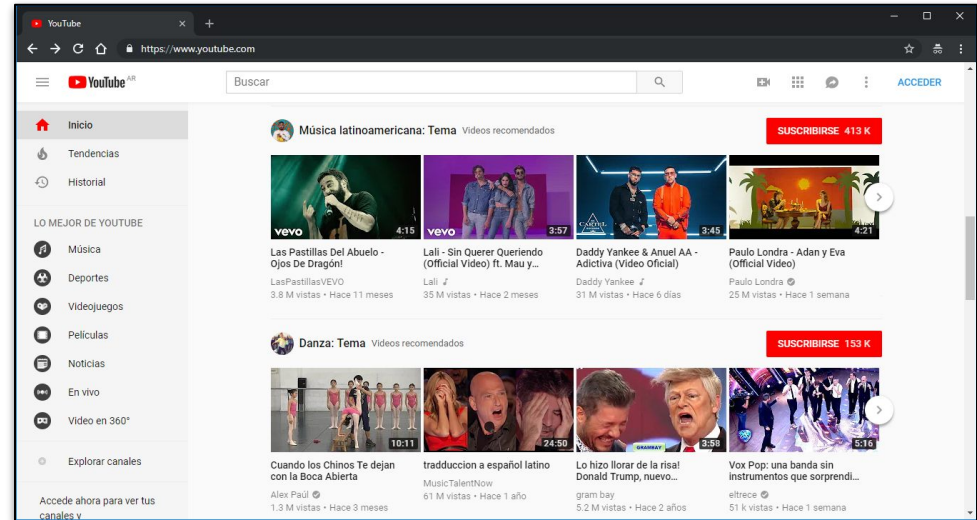


Comunicación con un sitio Web

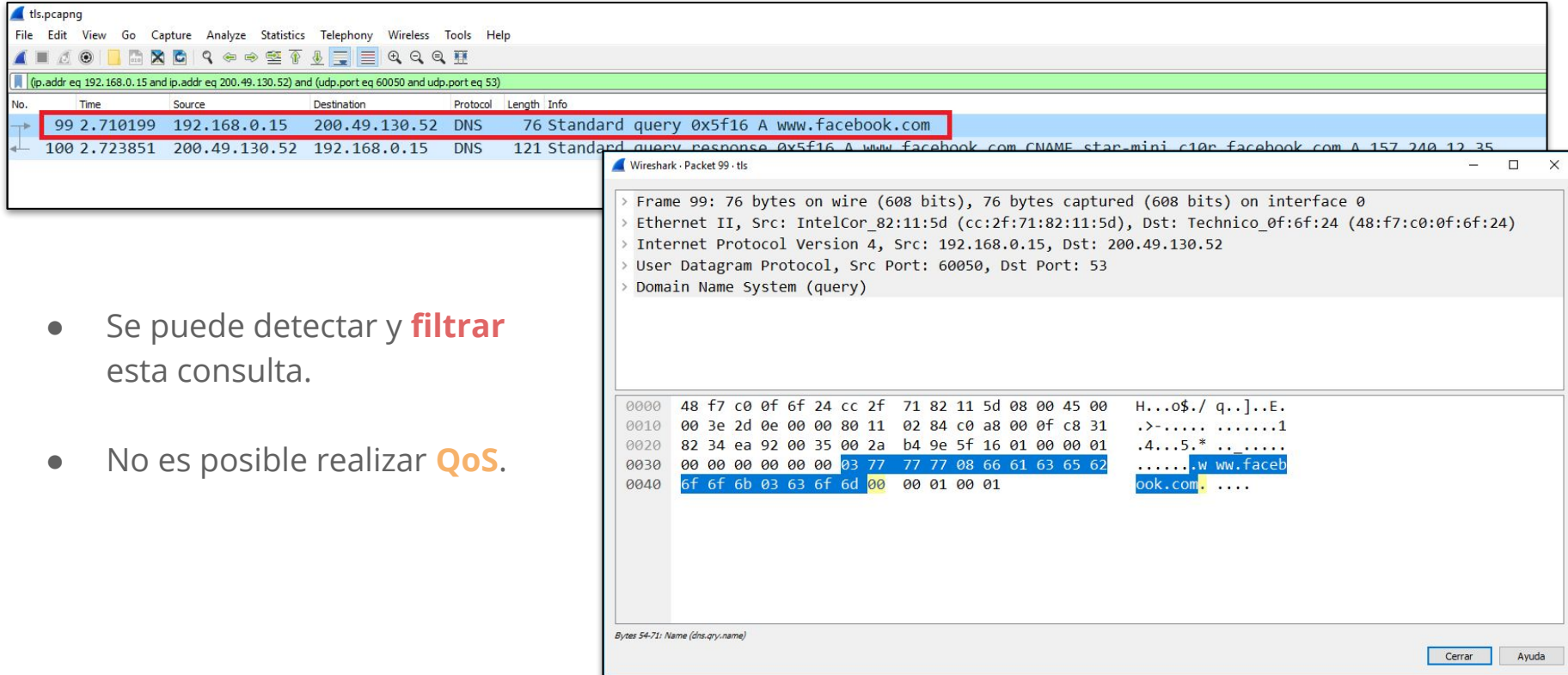
- Sitio de Google Inc. desde Chrome

1) Consulta DNS
(UDP 53)

2) Conexión QUIC
(UDP 80 o UDP 443)



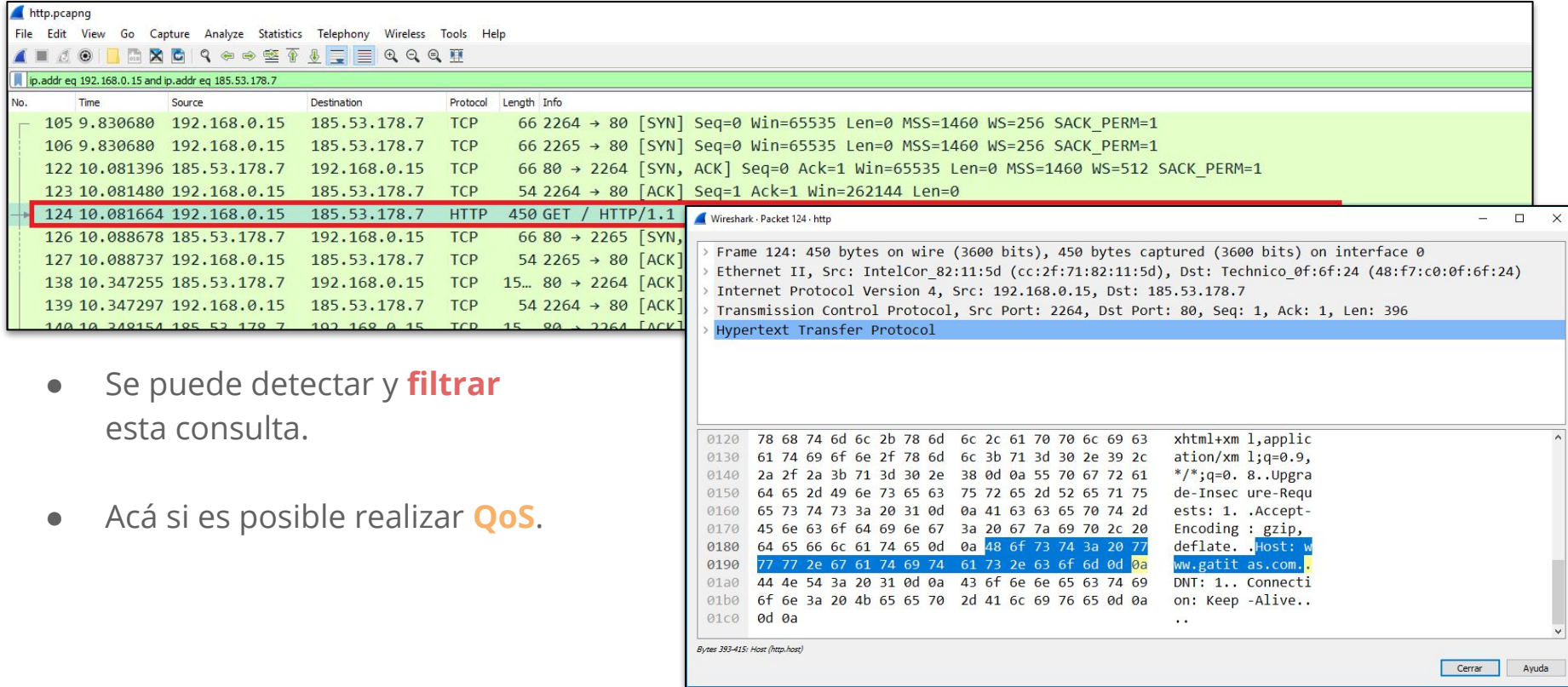
Comunicación DNS



The image shows a Wireshark capture of a DNS query. The main window displays a list of packets with a filter applied: `(p.addr eq 192.168.0.15 and ip.addr eq 200.49.130.52) and (udp.port eq 60050 and udp.port eq 53)`. Packet 99 is highlighted, showing a DNS Standard query from 192.168.0.15 to 200.49.130.52 for `www.facebook.com`. A detailed view of packet 99 is shown in the bottom pane, displaying the raw bytes and their hexadecimal representation. The query is for `www.facebook.com`.

- Se puede detectar y **filtrar** esta consulta.
- No es posible realizar **QoS**.

Comunicación HTTP

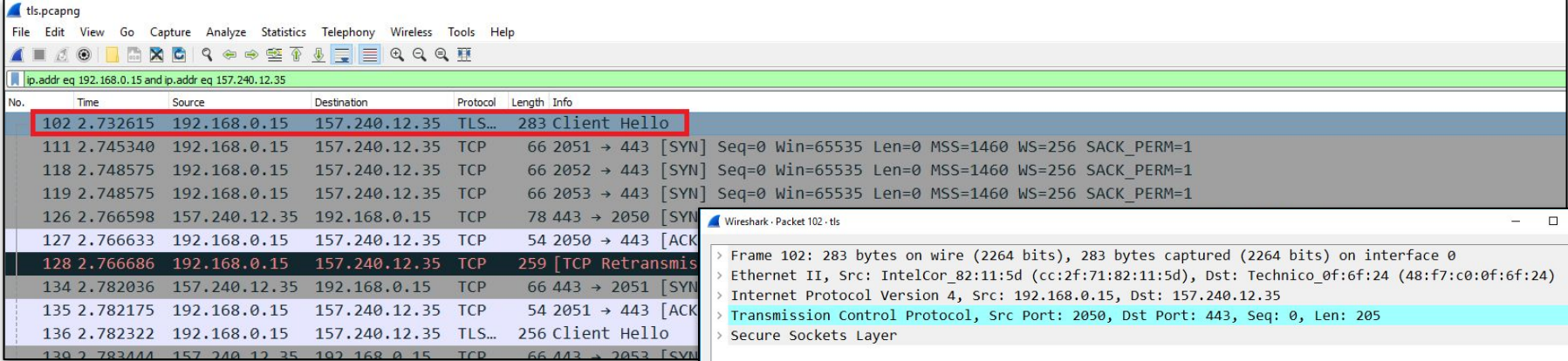


The image shows a Wireshark network traffic capture. The main packet list on the left shows several TCP and HTTP packets. Packet 124 is highlighted in red and is an HTTP GET request from 192.168.0.15 to 185.53.178.7. A detailed view of this packet is shown on the right, displaying the raw bytes and their hexadecimal representation. The packet is an HTTP GET request for the root path (/) using HTTP/1.1. The raw bytes show the request structure, including the method, path, version, host, and user-agent.

Se puede detectar y **filtrar** esta consulta.

Acá si es posible realizar **QoS**.

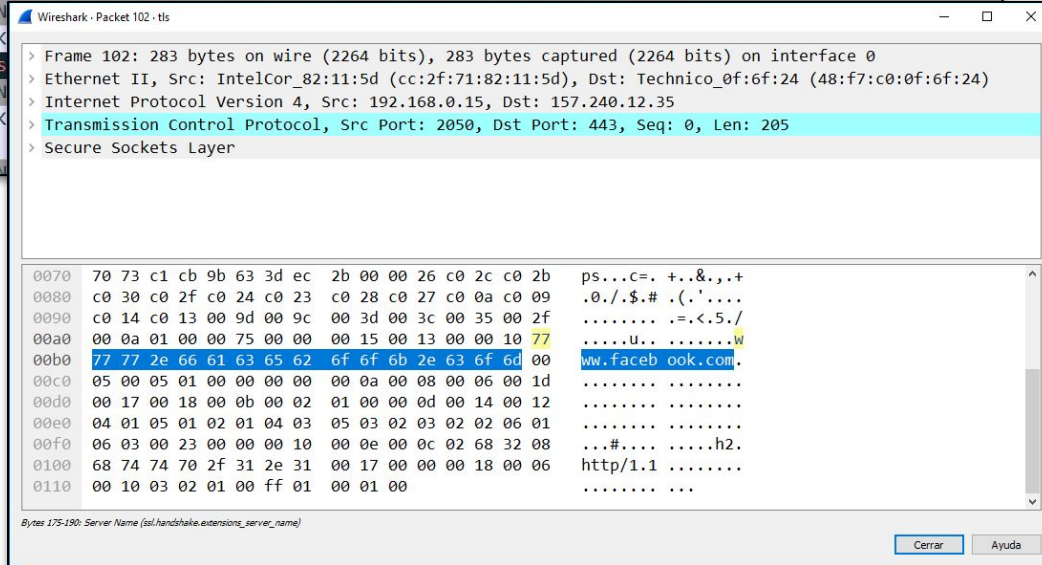
Comunicación HTTPs



Wireshark capture showing TLS traffic. The packet list on the left shows several packets. Packet 102 is highlighted with a red box, showing a TLS Client Hello from 192.168.0.15 to 157.240.12.35. Packet 128 is also highlighted, showing a TCP Retransmission. The packet details pane on the right shows the structure of packet 102: Ethernet II, Internet Protocol Version 4, Transmission Control Protocol, and Secure Sockets Layer.

No.	Time	Source	Destination	Protocol	Length	Info
102	2.732615	192.168.0.15	157.240.12.35	TLS...	283	Client Hello
111	2.745340	192.168.0.15	157.240.12.35	TCP	66	2051 → 443 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM=1
118	2.748575	192.168.0.15	157.240.12.35	TCP	66	2052 → 443 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM=1
119	2.748575	192.168.0.15	157.240.12.35	TCP	66	2053 → 443 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM=1
126	2.766598	157.240.12.35	192.168.0.15	TCP	78	443 → 2050 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM=1
127	2.766633	192.168.0.15	157.240.12.35	TCP	54	2050 → 443 [ACK] Seq=0 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM=1
128	2.766686	192.168.0.15	157.240.12.35	TCP	259	[TCP Retransmission] Seq=0 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM=1
134	2.782036	157.240.12.35	192.168.0.15	TCP	66	443 → 2051 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM=1
135	2.782175	192.168.0.15	157.240.12.35	TCP	54	2051 → 443 [ACK] Seq=0 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM=1
136	2.782322	192.168.0.15	157.240.12.35	TLS...	256	Client Hello

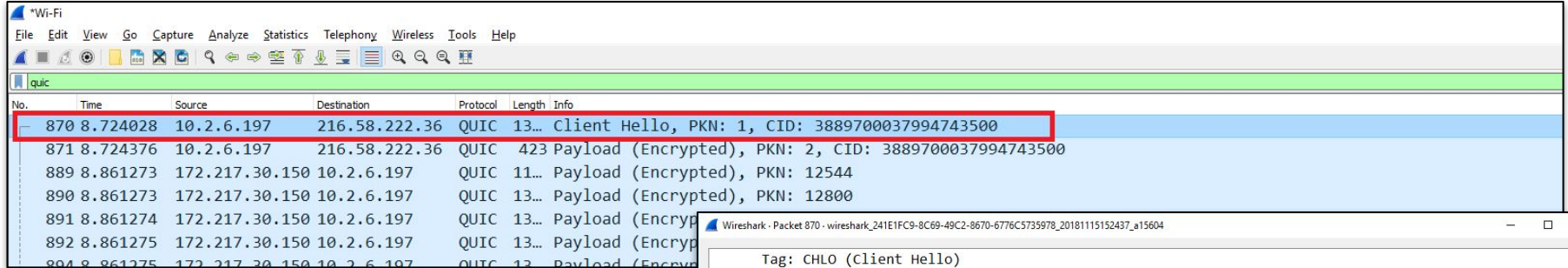
- HTTPs es encriptado, pero antes de establecerse utiliza **ClientHello** que tiene un parámetro llamado **Server Name**.
- Se puede detectar y **filtrar** esta consulta.
- Acá si es posible realizar **QoS**.



Wireshark packet details for packet 102 (283 bytes on wire, 283 bytes captured). The details show the structure of the TLS Client Hello message, including the Ethernet II header, Internet Protocol Version 4 header, Transmission Control Protocol header, and the Secure Sockets Layer (SSL) handshake extension. The SSL handshake extension contains the Server Name field, which is highlighted in blue and shows the value "www.facebook.com".

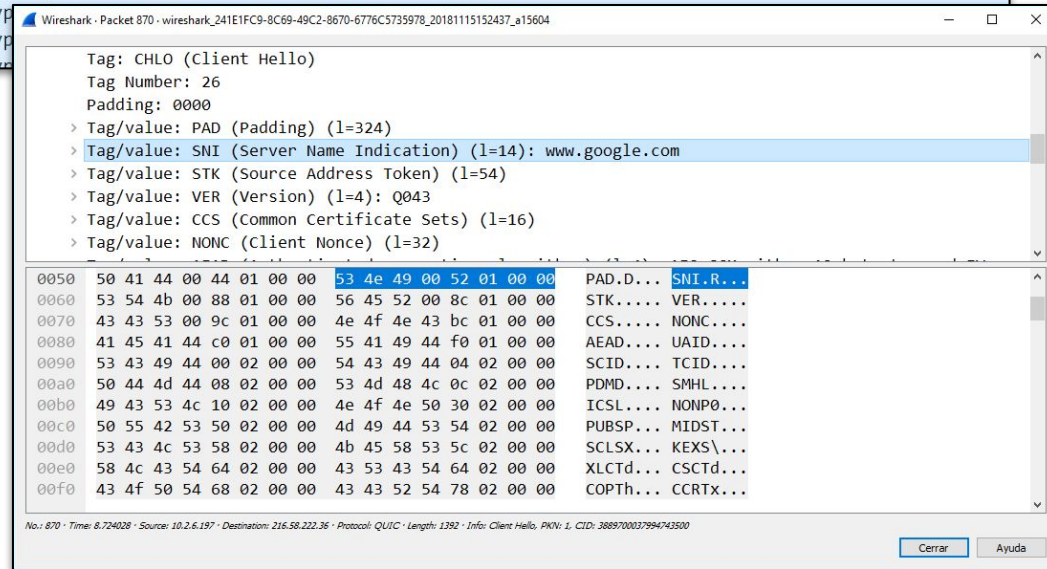
Bytes 175-190: Server Name (ssl.handshake.extensions_server_name)

Comunicación QUIC



No.	Time	Source	Destination	Protocol	Length	Info
870	8.724028	10.2.6.197	216.58.222.36	QUIC	13...	Client Hello, PKN: 1, CID: 3889700037994743500
871	8.724376	10.2.6.197	216.58.222.36	QUIC	423	Payload (Encrypted), PKN: 2, CID: 3889700037994743500
889	8.861273	172.217.30.150	10.2.6.197	QUIC	11...	Payload (Encrypted), PKN: 12544
890	8.861273	172.217.30.150	10.2.6.197	QUIC	13...	Payload (Encrypted), PKN: 12800
891	8.861274	172.217.30.150	10.2.6.197	QUIC	13...	Payload (Encrypted)
892	8.861275	172.217.30.150	10.2.6.197	QUIC	13...	Payload (Encrypted)
894	8.861275	172.217.30.150	10.2.6.197	QUIC	13...	Payload (Encrypted)

- QUIC corre sobre UDP y es encriptado, pero antes de establecerse también utiliza **ClientHello** que tiene un parámetro llamado **Server Name** (igual que TLS).
- Se puede detectar y **filtrar** esta consulta.
- Acá si es posible realizar **QoS**.



Wireshark - Packet 870 - wireshark_241E1FC9-8C69-49C2-8670-6776C5735978_20181115152437_e15604

Tag: CHLO (Client Hello)
 Tag Number: 26
 Padding: 0000

- > Tag/value: PAD (Padding) (l=324)
- > Tag/value: SNI (Server Name Indication) (l=14): www.google.com
- > Tag/value: STK (Source Address Token) (l=54)
- > Tag/value: VER (Version) (l=4): Q043
- > Tag/value: CCS (Common Certificate Sets) (l=16)
- > Tag/value: NONC (Client Nonce) (l=32)

0050	50 41 44 00 44 01 00 00	53 4e 49 00 52 01 00 00	PAD.D...	SNI.R...
0060	53 54 4b 00 88 01 00 00	56 45 52 00 8c 01 00 00	STK....	VER....
0070	43 43 53 00 9c 01 00 00	4e 4f 4e 43 bc 01 00 00	CCS....	NONC...
0080	41 45 41 44 c0 01 00 00	55 41 49 44 f0 01 00 00	AEAD...	UAID...
0090	53 43 49 44 00 02 00 00	54 43 49 44 04 02 00 00	SCID...	TCID...
00a0	50 44 4d 44 08 02 00 00	53 4d 48 4c 0c 02 00 00	PDMD...	SMHL...
00b0	49 43 53 4c 10 02 00 00	4e 4f 4e 50 30 02 00 00	ICSL...	NONP...
00c0	50 55 42 53 50 02 00 00	4d 49 44 53 54 02 00 00	PUBSP...	MIDST...
00d0	53 43 4c 53 58 02 00 00	4b 45 58 53 5c 02 00 00	SCLSX...	KEXS...
00e0	58 4c 43 54 64 02 00 00	43 53 43 54 64 02 00 00	XLCTd...	CSCTd...
00f0	43 4f 50 54 68 02 00 00	43 43 52 54 78 02 00 00	COPTh...	CCRTx...

No.: 870 • Time: 8.724028 • Source: 10.2.6.197 • Destination: 216.58.222.36 • Protocol: QUIC • Length: 1392 • Info: Client Hello, PKN: 1, CID: 3889700037994743500

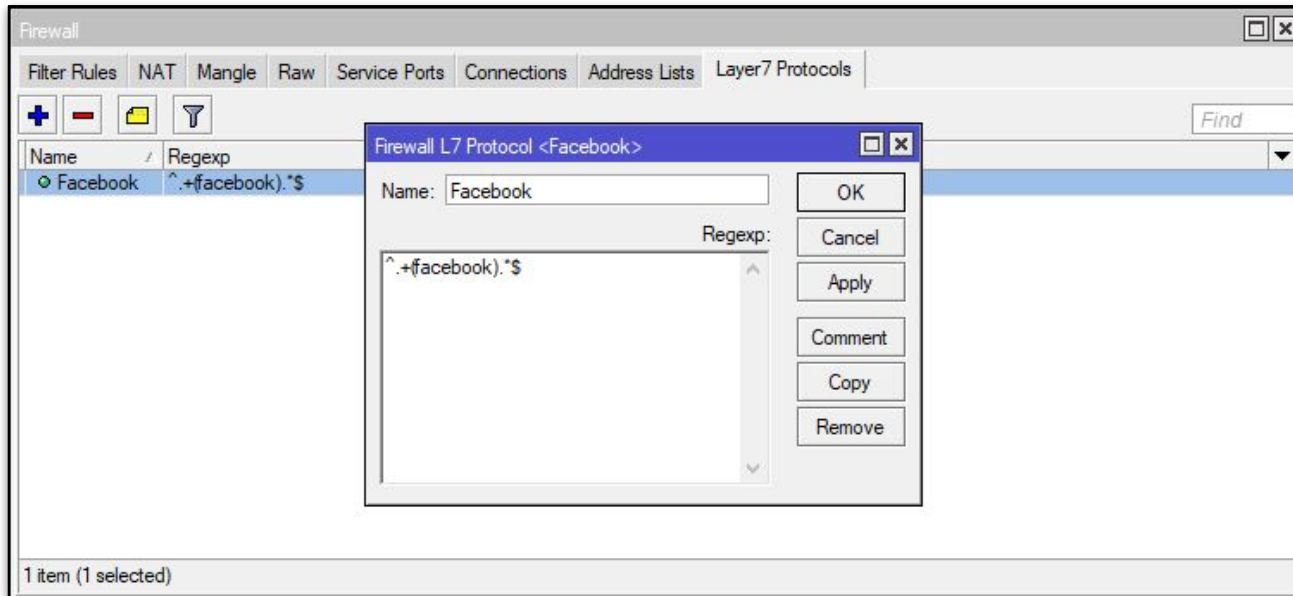
Cerrar Ayuda

Layer7 en acción

Ejemplo con Filter - Detección con Layer7

/ip firewall layer7-protocol

add name=Facebook regexp="^.+(facebook).*\$"



Ejemplo con Filter (HTTP/HTTPS)

```
/ip firewall filter
```

```
add comment="Analisis TCP" \
```

```
chain=forward \
```

```
protocol=tcp \
```

```
dst-port=80,443 \
```

← Acotar la regla a conexiones TCP 80 y 443.

```
connection-bytes=0-100000 \
```

← Conexiones de hasta 100k de transferencia.

```
action=jump jump-target=analisis_layer7
```

← Salto a otra cadena.

Ejemplo con Filter (HTTP/HTTPS)

```
/ip firewall filter
```

```
add comment="Bloquear Facebook" \
```

```
chain=analysis_layer7 \
```

```
protocol=tcp \
```

```
layer7-protocol=Facebook \
```

← Registro Layer7.

```
action=reject reject-with=tcp-reset
```

← Rechazo instantáneo.

Ejemplo con Filter (QUIC)

```
/ip firewall filter
```

```
add comment="Análisis UDP" \
```

```
chain=forward \
```

```
protocol=udp \
```

← QUIC funciona en UDP.

```
dst-port=80,443 \
```

← Acotar la regla a conexiones UDP 80 y 443.

```
connection-bytes=0-100000 \
```

```
action=jump jump-target=analisis_layer7
```

← Salto a otra cadena.

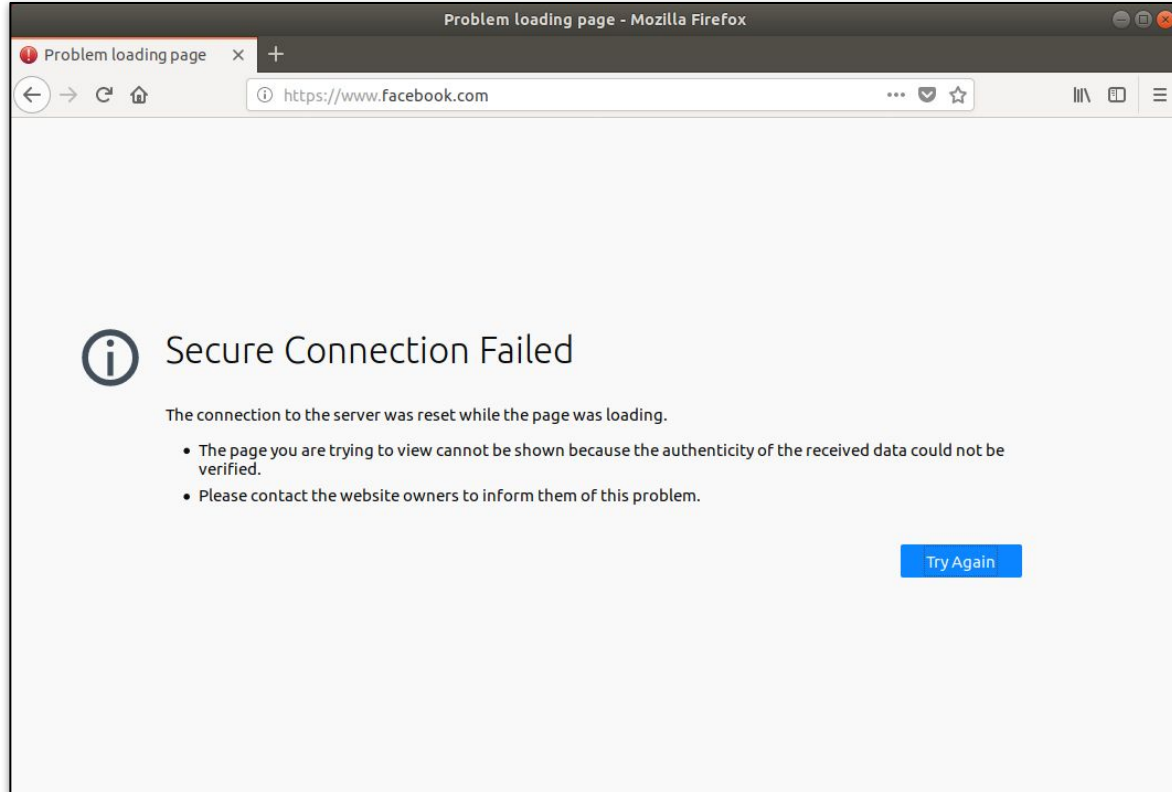
Ejemplo con Filter (QUIC)

```
/ip firewall filter  
add comment="Bloquear YouTube" \  
chain=analysis_layer7 \  
layer7-protocol=YouTube \  
action=reject reject-with=icmp-admin-prohibited
```

← Registro Layer7.

↑ Rechazo casi instantáneo.

Ejemplo con Filter (HTTP/HTTPS/QUIC)

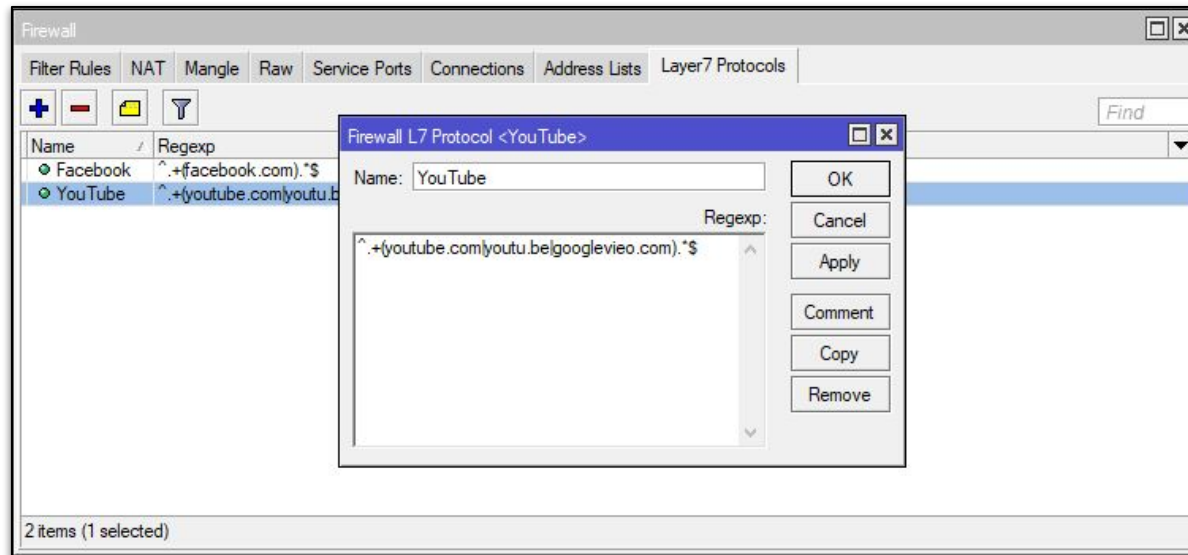


Ejemplo con QoS - Detección con Layer7

```
/ip firewall layer7-protocol
```

```
add name=YouTube \
```

```
regexp="^.+(youtube.com|youtu.be|googlevideo.com).*\$"
```



Ejemplo con QoS

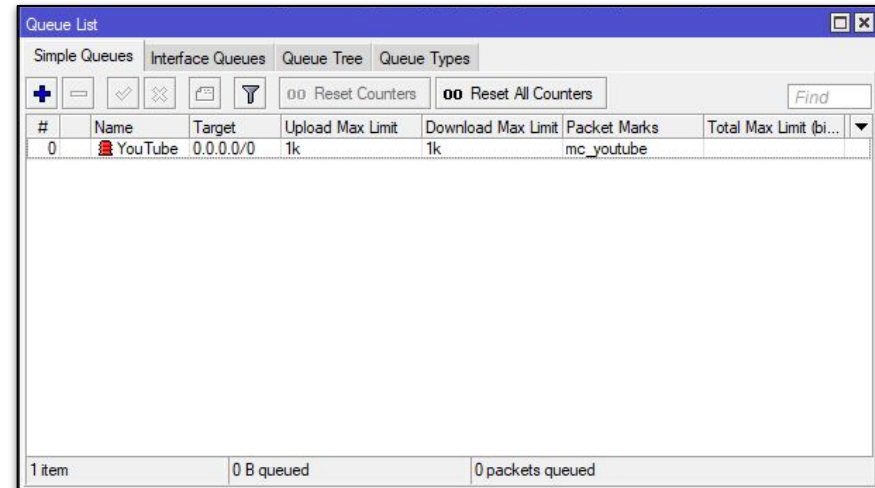
```
/ip firewall mangle  
add comment="Marcar conexiones de YouTube" \  
chain=forward \  
connection-mark=no-mark \  
layer7-protocol=YouTube \  
action=mark-connection \  
new-connection-mark=mc_youtube \  
passthrough=yes
```

Ejemplo con QoS

```
/ip firewall mangle  
add comment="Marcar paquetes de YouTube" \  
chain=forward \  
connection-mark=mc_youtube \  
action=mark-packet \  
new-packet-mark=mc_youtube \  
passthrough=no
```

Ejemplo con QoS

```
/queue simple  
add name="No Vas A Ver YouTube, No" \  
target="" \  
packet-marks=mc_youtube \  
max-limit=1k/1k
```



#	Name	Target	Upload Max Limit	Download Max Limit	Packet Marks	Total Max Limit (bi...
0	 YouTube	0.0.0.0/0	1k	1k	mc_youtube	

1 item 0 B queued 0 packets queued

Consideraciones especiales

Consideraciones especiales

- Para QoS por sitio web, de momento esta es la única técnica que se puede utilizar sobre cierto patrón de tráfico*.
- Para filtrado, esta técnica es poco escalable y es recomendable utilizar un **DNS gestionado**.
- En IPv6 aún no está disponible el comparador Layer7, pero puede utilizarse el comparador ***content***.

¿Preguntas?

Utilizando Layer7 para Filter y QoS