Trabajo Práctico Final Integrador: Configuración de Redes y Análisis de protocolos

Fecha de entrega: 25/06/25

Autor: Abril Babino
Legajo: 199147

En el laboratorio

1. Descargue el labo base e implemente en el laboratorio de Kathara las redes de acuerdo al plano de topología de la Figura fig:topologia.

2. Configure las interfaces, ruteadores/rutas y resolvers. El servidor DNS maneja la zona tpfinal-tyr.com (ya se encuentra configurado). Revise qué servicios debe iniciar y los puertos en los que operan los servidores web y el proxy. El equipo de usuario (Usuario) accede al exterior a través del servidor proxy (revise la configuración para reconocer cómo es en este caso).

PcUsuario

```
source /root/.bashrc
ip address add 10.10.0.10/24 dev eth0
ip route add default via 10.10.0.1
echo "nameserver 8.8.8.8" >> /etc/resolv.conf
```

Proxy

```
echo "visible_hostname proxy.tpfinal-tyr.com" >> /etc/squid/squid.conf
sed -i 's/\#http_access allow localnet/http_access allow all/'
/etc/squid/squid.conf
#rm -r /var/spool/squid/*
#squid -z
ip address add 10.10.0.1/24 dev eth0
ip route add default via 10.10.0.100
echo "nameserver 8.8.8.8" >> /etc/resolv.conf
service squid start
```

Servidor WWW

```
sed -i 's/\KeepAlive On/KeepAlive Off/' /etc/apache2/apache2.conf
ip address add 12.12.0.11/24 dev eth0
ip route add default via 12.12.0.1
echo "nameserver 8.8.8.8" >> /etc/resolv.conf
service apache2 start
```

Servidor Datos

```
chown www-data:www-data /usr/lib/cgi-bin/datos.pl
chmod 750 /usr/lib/cgi-bin/datos.pl
sed -i 's/\KeepAlive On/KeepAlive Off/' /etc/apache2/apache2.conf
a2enmod cgid
ip address add 12.12.0.12/24 dev eth0
ip route add default via 12.12.0.1
echo "nameserver 8.8.8.8" >> /etc/resolv.conf
service apache2 start
```

Servidor DNS

```
chmod 755 /etc
chmod 755 /etc/bind
```

```
ip address add 8.8.8.8/28 dev eth0
ip route add default via 8.8.8.1
service bind start
```

Router1

```
ip address add 190.7.231.40/24 dev eth0
ip address add 10.10.0.100/24 dev eth1
ip route add 12.12.0.0/24 via 190.7.231.20
ip route add 8.8.8.0/28 via 190.7.231.30
```

Router2

```
ip address add 190.7.231.30/24 dev eth0
ip address add 8.8.8.1/28 dev eth1
ip route add 10.10.0.0/24 via 190.7.231.40
ip route add 12.12.0.0/24 via 190.7.231.20
```

Router3

```
ip address add 190.7.231.20/24 dev eth0
ip address add 12.12.0.1/24 dev eth1
ip route add 10.10.0.0/24 via 190.7.231.40
ip route add 8.8.8.0/28 via 190.7.231.30
```

3. Copie la plantilla con sus datos al directorio público del servidor datos.tpfinal-tyr.com.

4. Luego, debe realizar la captura de tráfico de una transferencia completa de su página web usando el comando w3c (las instrucciones se encuentran dentro de los fuentes del laboratorio) y todo su contenido asociado, utilizando el mnemónico del servidor web. La captura debe comenzar con el equipo cliente apagado (tablas ARP de ruteadores y hosts

1. Descarte los mensajes que no pertenecen específicamente a esta transferencia. Ordene las PDUs por su marca de tiempo y numérelas (no use el número de secuencia asignado durante la captura). Agregue en su interface de Wireshark las direcciones de capa 2 para facilitar la identificación real de origen/destino de los mensajes.

2. Confeccione cuadros con la distribución de mensajes por protocolo por capa (Indique claramente qué protocolo/s ubica en cada capa y por qué).

CAPA	PROTOCOLO	DISTRIBUCION	EXPLICACION
ENLACE	ETHERNET	100,00%	Es responsable de la transmisión de datos en redes locales y del control de acceso al medio.
	ARP	4,76%	Se encarga de la resolución de direcciones IP a direcciones MAC dentro de una red local., ARP recibe una dirección IP desde la capa de red, la encapsula en una trama y se envía en broadcast, todos los nodos la reciben, pero solo el correcto responde con un mensaje ARP de respuesta (unicast) que contiene su MAC.
RED	IPv4	95,24%	Es responsable del direccionamiento y enrutamiento de paquetes de datos entre dispositivos en diferentes redes.
	ICMPv4	2,86%	Se utiliza para enviar mensajes de control y error relacionados con la entrega de paquetes IP. En esto caso, ICMP se utiliza el comando ping para verificar la conectividad entre dispositivos en una red.
TRANSPORTE	TCP	58,10%	Proporciona un servicio de comunicación confiable y orientado a la conexión entre dispositivos finales. En la captura se puede observar cómo TCP maneja el establecimiento, la transferencia y el cierre de conexiones
	UDP	28,57%	Es responsable de la entrega de datagramas entre aplicaciones en dispositivos finales. Su estructura simple y sus características lo convierten en una opción ideal para aplicaciones que requieren velocidad y pueden tolerar la pérdida de datos.
APLICACIÓN	DNS	28,57%	Es un servicio que permite a las aplicaciones resolver nombres de dominio en direcciones IP
	HTTP	5,71%	Permite que los navegadores web soliciten y reciban contenido de servidores web, facilitando la comunicación entre el cliente y el servidor.

3. Identifique todas las conexiones TCP. Por cada una, indique: finalidad, sockets de cliente y servidor, segmentos de apertura y cierre. Muestre los parámetros intercambiados durante la fase de apertura.

Nro conexión	Finalidad	Socket cliente	Socket servidor	Segmentos apertura	Parametros intercambiados	Segmentos cierre
1	El host PcUsuario se conecta con el servidor proxy para obtener el contenido de www.tpfinal-tyr.com	<10.10.0.10:39030>	<10.10.0.1:3128>	23,24,25	S23: MSS- 1460 bytes, SACK permitted, timestamp, No-operation, Window scale: 7 S24:MSS- 1460 bytes, SACK permitted, timestamp, No-operation, Window scale: 7	80,82,87
2	El servidor proxy se conecta con el servidor web para obtener el contenido de www.tpfinal-tyr.com	<10.10.0.1:51084>	<12.12.0.11:80>	39,44,47	S39: MSS: 1460 bytes, SACK permitted, Timestamps, No-Operation, Window scale: 7 S46: MSS: 1460 bytes, SACK permitted, Timestamps, No-Operation, Window scale: 7	68,61,86
3	El host PcUsuario se conecta con el servidor proxy para obtener el contenido html de datos.tpfinal-tyr.com/index.html	<10.10.0.10:39032>	<10.10.0.1:3128>	102,103,104	S102:MSS: 1460 bytes, SACK permitted, Timestamps, No-Operation, Window scale: 7 S103: MSS: 1460 bytes, SACK permitted, Timestamps, No-Operation, Window scale: 7	147,152,154
4	El servidor proxy se conecta al servidor datos para obtener el contenido html	<10.10.0.1:37676>	<12.12.0.12:80>	117,114,121	S117:MSS: 1460 bytes, SACK permitted, Timestamps, No-Operation, Window scale: 7 S118: MSS: 1460 bytes, SACK permitted, Timestamps, No-Operation, Window scale: 7	141,125,150
5	El host PcUsuario se conecta con el proxy para solicitar la ejecucion del script datos.pl y devuelva su salida como respuesta HTTP	<10.10.0.10:39046>	<10.10.0.1: 3128>	169,170,171	S169:MSS: 1460 bytes, SACK permitted, Timestamps, No-Operation, Window scale: 7 S170: MSS: 1460 bytes, SACK permitted, Timestamps, No-Operation, Window scale: 7	202,203,204
6	El servidor proxy se conecta al servidor datos para ejecutar el script datos.pl y devolver su salida como respuesta HTTP	<10.10.0.1:37680>	<12.12.0.12: 80>	178,176,182	S178:MSS: 1460 bytes, SACK permitted, Timestamps, No-Operation, Window scale: 7 S179: MSS: 1460 bytes, SACK permitted, Timestamps, No-Operation, Window scale: 7	197,205,210

4. Para la conexión TCP utilizada para recuperar el archivo .html entre usuario y proxy indique la finalidad de cada PDU intercambiada a nivel de transporte y aplicación.

Segmento 102:

Nivel transporte
-**socket cliente** <10.10.0.10:39032> → **socket servidor** <10.10.0.1:3128>
-Flags: **[Syn]**, primer segmento de apertura de la conexión
-ISN=2247794913 (0 relativo)
-Header Length: 40 bytes (tiene opciones)
→ MSS: 1460 bytes
→ SACK permitted El cliente acepta SACK (Selective ACK).
→ Timestamps
→ NOP
→ Window Scale: 7 (indica que la ventana se multiplica ×128).
-Window Size: 64240 (Tamaño de la ventana de recepción inicial).
Finalidad: El cliente TCP inicia activamente una conexión con el proxy, para obtener el contenido HTML de datos.www.tpfinal-tyr.com/index.html.

Segmento 103:
Nivel transporte:
-**socket origen** <10.10.0.1:3128> → **socket destino** <10.10.0.10:39032>
-Flags: **[Syn][Ack]**,
-ISN= 4229258019. Número inicial de secuencia que el servidor usará para esta conexión
-ACK= 2247794914. Reconoce el ISN del cliente + 1
-Header Length: 40 bytes (tiene opciones)
→ MSS: 1460 bytes
→ SACK permitted El servidor acepta SACK (Selective ACK).
→ Timestamps
→ NOP
→ Window Scale: 7 (indica que la ventana se multiplica ×128).
-Window Size: 65160 (Tamaño de la ventana de recepción inicial del servidor).
Finalidad: El servidor confirma al cliente que su solicitud de conexión fue recibida correctamente (ACK) y, a la vez, inicia la numeración del servidor (SYN).

Segmento 104:
Nivel transporte:
-**socket origen** <10.10.0.10:39032> → **socket destino** <10.10.0.1:3128>
-Flags: **[Ack]**,
-ISN= 2247794914.
-ACK= 4229258020. (Confirma recepción del ISN del servidor)
-Header Length: 32 bytes (tiene opciones)
→ Timestamps
→ NOP
-Window Size: 502 (escala aplicada: 128 → ventana real 64256)
Finalidad: Confirma la recepción del segmento SYN+ACK enviado por el servidor.
De esta forma, queda establecida la conexión TCP entre cliente y servidor.

Segmento 105:
Nivel transporte:
-**socket origen** <10.10.0.10:39032> → **socket destino** <10.10.0.1:3128>
-Flags: **[PSH][Ack]**,

-ISN= 2247794914.
-ACK= 4229258020.
-Header Length: 32 bytes (tiene opciones)
→ Timestamps
→ NOP
-Window Size: 502 (escala aplicada: 128 → ventana real 64256)
-TCP Segment Length: 297 bytes (indica que contiene datos)
Finalidad: Transporta datos de la capa de aplicación (HTTP) desde el cliente hacia el servidor a través de una conexión ya establecida.
La bandera PSH solicita al receptor procesar inmediatamente los datos recibidos sin esperar más segmentos.

Nivel aplicación:
-Método: GET
-URI solicitada: http://datos.tpfinal-tyr.com/index.html
-Versión: HTTP/1.0
Cabeceras incluidas:
-User-Agent: w3m/0.5.3+git20230121
-Accept: text/html, text/*;q=0.5, image/*, application/*
-Accept-Encoding: gzip, compress, bzip, bzip2, deflate
-Accept-Language: en;q=1.0
-Host: datos.tpfinal-tyr.com
-Referer: http://www.tpfinal-tyr.com/
Finalidad: El cliente solicita al servidor (proxy) la página web ubicada en http://datos.tpfinal-tyr.com/index.html, especificando qué tipos de contenido acepta y con qué configuraciones de cliente.

Segmento 106:
Nivel transporte:
-**socket origen** <10.10.0.1:3128> → **socket destino** <10.10.0.10:39032>
-Flags: **[Ack]**,
-ISN= 4229258020
-ACK= 2247795211. (relativo a 298)
-Header Length: 32 bytes (tiene opciones)
→ Timestamps
→ NOP
-Window Size: 507 (escalada → 64896)
Finalidad: Confirma la recepción del segmento anterior enviado por el cliente que contenía el mensaje HTTP GET.

Segmento 143:
Nivel transporte:
-**socket origen** <10.10.0.1:3128> → **socket destino** <10.10.0.10:39032>
-Flags: **[Psh][Ack]**,
-ISN= 4229258020
-ACK= 2247795211. (relativo a 298)
-Header Length: 32 bytes (tiene opciones)

→ Timestamps
→ NOP
-Window Size: 507 (escalada → 64896)
-TCP Segment Length: 440 bytes
Finalidad: Contiene los primeros 440 bytes de la respuesta HTTP a la solicitud GET /index.html, como parte del contenido del sitio solicitado. Iniciar el envío de la respuesta HTTP correspondiente desde el proxy hacia el cliente.

Segmento 145:
Nivel transporte:
-**socket origen** <10.10.0.10:39032>→ **socket destino** <10.10.0.1:3128>
-Flags: **[Ack]**,
-ISN= 2247795211.
-ACK= 4229258460 (relativo a 411)
-Header Length: 32 bytes (tiene opciones)
→ Timestamps
→ NOP
-Window Size: 501 (escalada → 64128)
Finalidad: Confirma la recepción de los 440 bytes enviados anteriormente por el proxy e indica que el cliente está listo para recibir más datos si fuera necesario.

Segmento 146:
Nivel transporte:
-**socket origen** <10.10.0.1:3128> → **socket destino** <10.10.0.10:39032>
-Flags: **[Psh][Ack]**
-ISN= 4229258460 (relativo a 411)
-ACK= 2247795211 (relativo a 298)
-Header Length: 32 bytes (tiene opciones)
→ Timestamps
→ NOP
-Window Size: 507 (escalada → 64896)
-TCP Segment Length: 666 bytes (indica que contiene datos)
Finalidad: Transporta la respuesta HTTP completa, contiene el contenido comprimido de la página web solicitada (index.html).

Nivel aplicación:
Protocolo: HTTP/1.1
Código de respuesta: 200 OK → la solicitud se procesó con éxito.
Encabezados relevantes:
-Content-Encoding: gzip → el cuerpo está comprimido.
-Content-Length: 666 → tamaño comprimido en bytes.
-Content-Type: text/html → es una página web.
-Connection: close → se cierra la conexión una vez entregado el contenido.
-X-Cache: MISS from proxy... → indica que el contenido no estaba cacheado.
Cuerpo del mensaje HTTP:
-Contenido comprimido (gzip) de 666 bytes, que una vez descomprimido pesa 1050 bytes.
-Es una página web HTML con 47 líneas de texto.

Finalidad: Respuesta a la solicitud HTTP que contiene el archivo html.

Segmento 147:
Nivel transporte:
-**socket origen** <10.10.0.1:3128> → **socket destino** <10.10.0.10:39032>
-Flags: **[Fin][Ack]**
-ISN= 4229259126 (relativo a 1107)
-ACK= 2247795211 (relativo a 298)
-Header Length: 32 bytes (tiene opciones)
→ Timestamps
→ NOP
-Window Size: 507 (escalada → 64896)
Finalidad: Marca el inicio del cierre de la conexión por parte del proxy, que indica que ha terminado de enviar datos hacia el cliente y solicita cerrar su flujo de envío.

Segmento 152:
Nivel transporte:
-**socket origen** <10.10.0.10:39032> → **socket destino** <10.10.0.1:3128>
-Flags: **[Fin][Ack]**
-ISN= 2247795211 (relativo a 298)
-ACK= 4229259127 (relativo a 1108)
-Header Length: 32 bytes (tiene opciones)
→ Timestamps
→ NOP
-Window Size: 501 (escalada → 64128)
Finalidad: el cliente informa que no tiene más datos que enviar y responde al FIN enviado por el servidor.

Segmento 154:
Nivel transporte:
-**socket origen** <10.10.0.1:3128> → **socket destino** <10.10.0.10:39032>
-Flags: **[Ack]**
-ISN= 4229259127 (relativo a 1108)
-ACK= 2247795212 (relativo a 229)
-Header Length: 32 bytes (tiene opciones)
→ Timestamps
→ NOP
-Window Size: 507 (escalada → 64896)
Finalidad: Es el último segmento del cierre de conexión TCP. Corresponde al ACK enviado por el servidor en respuesta al FIN, ACK del cliente.

5. Genere un diagrama de intercambios en el tiempo que muestre cómo sucedieron los mensajes, incluyendo TODOS los dispositivos involucrados. Por cada mensaje, identifique los principales parámetros que hacen a su función. En su gráfico, utilice diferentes colores para las diferentes conexiones (incluya las referencias correspondientes).

6. Compare los headers HTTP del requerimiento del cliente al proxy con respecto del realizado por el proxy al servidor web. Muestre y explique las diferencias.

PcUsuario

-Solicitud para recuperar una página del servidor www
GET http://www.tpfinal-tyr.com/ HTTP/1.0\r\n
User-Agent: w3m/0.5.3+git20230121\r\n
Accept: text/html, text/*;q=0.5, image/*, application/*\r\n
Accept-Encoding: gzip, compress, bzip, bzip2, deflate\r\n
Accept-Language: en;q=1.0\r\n
Host: www.tpfinal-tyr.com\r\n

-Solicitud para recuperar el archivo html del servidor datos
GET http://datos.tpfinal-tyr.com/index.html HTTP/1.0\r\n
User-Agent: w3m/0.5.3+git20230121\r\n
Accept: text/html, text/*;q=0.5, image/*, application/*\r\n
Accept-Encoding: gzip, compress, bzip, bzip2, deflate\r\n
Accept-Language: en;q=1.0\r\n
Host: datos.tpfinal-tyr.com\r\n
Referer: http://www.tpfinal-tyr.com/\r\n

-Solicitud para recuperar una página dinámica del servidor datos
GET http://datos.tpfinal-tyr.com/cgi-bin/datos.pl HTTP/1.0\r\n
User-Agent: w3m/0.5.3+git20230121\r\n
Accept: text/html, text/*;q=0.5, image/*, application/*\r\n
Accept-Encoding: gzip, compress, bzip, bzip2, deflate\r\n
Accept-Language: en;q=1.0\r\n
Host: datos.tpfinal-tyr.com\r\n
Referer: http://www.tpfinal-tyr.com/\r\n

Servidor Proxy

-Solicitud para recuperar una página del servidor www
GET / HTTP/1.1\r\n
User-Agent: w3m/0.5.3+git20230121\r\n
Accept: text/html, text/*;q=0.5, image/*, application/*\r\n
Accept-Encoding: gzip, compress, bzip, bzip2, deflate\r\n
Accept-Language: en;q=1.0\r\n
Host: www.tpfinal-tyr.com\r\n
Via: 1.0 proxy.tpfinal-tyr.com (squid/5.7)\r\n
X-Forwarded-For: 10.10.0.10\r\n
Cache-Control: max-age=259200\r\n
Connection: keep-alive\r\n

-Solicitud para recuperar el archivo html del servidor datos

GET /index.html HTTP/1.1\r\n
User-Agent: w3m/0.5.3+git20230121\r\n
Accept: text/html, text/*;q=0.5, image/*, application/*\r\n
Accept-Encoding: gzip, compress, bzip, bzip2, deflate\r\n
Accept-Language: en;q=1.0\r\n
Referer: http://www.tpfinal-tyr.com/\r\n
Host: datos.tpfinal-tyr.com\r\n
Via: 1.0 proxy.tpfinal-tyr.com (squid/5.7)\r\n
X-Forwarded-For: 10.10.0.10\r\n
Cache-Control: max-age=259200\r\n
Connection: keep-alive\r\n

-Solicitud para recuperar una página dinámica del servidor datos
GET /cgi-bin/datos.pl HTTP/1.1\r\n
User-Agent: w3m/0.5.3+git20230121\r\n
Accept: text/html, text/*;q=0.5, image/*, application/*\r\n
Accept-Encoding: gzip, compress, bzip, bzip2, deflate\r\n
Accept-Language: en;q=1.0\r\n
Referer: http://www.tpfinal-tyr.com/\r\n
Host: datos.tpfinal-tyr.com\r\n
Via: 1.0 proxy.tpfinal-tyr.com (squid/5.7)\r\n
X-Forwarded-For: 10.10.0.10\r\n
Cache-Control: max-age=0\r\n
Connection: keep-alive\r\n

Diferencias Clave

Versión del Protocolo: El cliente del host usuario utiliza la version 1.0 de HTTP, mientras que el proxy utiliza la versión 1.1 que permite conexiones persistentes y manejo de caché.

URI vs. Path: El cliente del host usuario envía la URI completa http://www.tpfinal-tyr.com/, mientras que el proxy envía solo el path /, ya que en HTTP/1.1, el proxy necesita solo el path, ya que el encabezado Host especifica el dominio.

Encabezados Adicionales: El proxy incluye los encabezados adicionales:

-Via: Indica que la solicitud ha pasado por un proxy y proporciona información sobre el proxy utilizado.

-X-Forwarded-For: Proporciona la dirección IP original del cliente (10.10.0.10).

-Cache-Control: Indica si el proxy puede almacenar la respuesta en caché (Cuando el max-age>0).

-Connection: Indica si la conexión es persistente (keep-alive), lo que permite múltiples solicitudes a través de la misma conexión TCP. En este caso todas las conexiones que realice el proxy van a ser persistentes.

Las diferencias entre los encabezados HTTP de las solicitudes del cliente y del proxy muestra como el proxy actúa como intermediario, añadiendo información adicional y mejorando la comunicación entre el cliente y el servidor web.

7. Confeccione una tabla con los diferentes protocolos involucrados, cantidad de PDUs, total en headers y total en datos. De allí, calcule el overhead ($Overhead = T\ otal\_Datos\_Control\ T\ otal\_Datos\_T\ x$) total y por protocolo generado para lograr la transferencia. Se sugiere armar una tabla en la cual cada fila Fi corresponde a una PDU y cada columna a un protocolo Pj (de diferentes capas). Luego, en cada celda i, j consigne el tamaño en bytes que corresponde a esa Fi y Pj. Genere una gráfica acorde para mostrar los resultados.

OVERHEAD (Cálculos realizados en el Excel)

ETH	ARP	IPv4	ICMPv4	TCP	UDP	DNS	HTTP	TOTAL
13,57%	1,01%	14,51%	0,17%	16,13%	1,72%	2,59%	10,71%	60,41%