Redes de Computadoras

Curso 2015

Grupo 14

Informe - Obligatorio 1

Autores:
Patricio Vera
Viviane Kunin
Nicolás Alberro

Supervisor:
Martín Giachino

August 20, 2025

Contents

1 Parte 1

1.1 Reporte técnico sobre Tcpdump, Wireshark y TShark

Tcpdump, Wireshark y TShark son tres herramientas similares que permiten capturar y analizar tráfico de datos. Con esto nos referimos al proceso de capturar, inspeccionar e interpretar los paquetes que circulan por una red para entender su contenido, origen, destino y comportamiento, para diagnosticar problemas, optimizar el rendimiento o buscar amenazas.

Tcpdump es una herramienta para utilizar en la línea de comandos de la consola. Se caracteriza por ser una aplicación ligera y no requerir muchos recursos en el dispositivo. Usa filtros BPF ("Berkeley Packet Filter" un lenguaje de filtrado) para seleccionar qué paquetes debe capturar y cuáles ignorar. Así evita registrar todo el tráfico y logra enfocarse solo en lo relevante. La salida de esta herramienta es un texto plano que puede ser interpretado por el usuario. Además de ver la salida en tiempo real se puede almacenar las capturas de los datos de la red en formato pcap, para analizarlos más tarde.

Wireshark cumple un objetivo muy similar, pero cuenta con una interfaz gráfica a diferencia de Tcpdump y TShark. Esto la hace más amigable para el usuario y facilita su uso a quienes no acostumbran interactuar directamente con la consola. Esta aplicación cuenta con filtros de captura al igual que tcpdump que permiten seleccionar qué contenido registrar. Además, cuenta con filtros de visualización que permiten seleccionar qué se ve facilitando hacer un análisis detallado después. Por la visualización con colores y que presenta es más sencillo de interpretar. Los datos para analizar pueden capturarse directamente con WireShark y almacenarlos en formatos .pcap o .pcapng o capturar previamente con alguna de las herramientas de consola y abrir el archivo en WireShark.

TShark también funciona en línea de comandos. Permite realizar los mismos protocolos y aplicar los mismos filtros que con wireshark, pero sin el uso de la interfaz gráfica. Esto le permite ser utilizada en entornos que no tienen este tipo de interfaz. La salida es texto plano que puede ser interpretado por el usuario,

pero no cuenta con filtros de visualización. Esta herramienta también permite guardar capturas de los datos obtenidos en los formatos .pcap o .pcapng.

Tcpdump funciona en los sistemas operativos Unix, Linux y macOS, pero puede utilizarse en Windows con adaptaciones. Wireshark y TShark cuentan con instaladores oficiales para Windows, macOS, Linux y BSD.

Las tres herramientas permiten capturar paquetes en vivo. En el caso de tcpdump, normalmente muestra solo un resumen textual de cada paquete, aunque puede desplegar más detalles con opciones adicionales. Además, todas permiten seleccionar de cuáles interfaces se quiere capturar el tráfico. Usando el comando tcpdump -D o tshark -D podemos ver las interfaces que se encuentran disponibles. Luego para seleccionar una se agrega al comando ejecutado "-i" seguido del identificador de la interfaz, por ejemplo "-i en0". En el caso de WireShark muestra todas las interfaces disponibles en la pantalla inicial al abrir la interfaz gráfica y alcanza con hacer click sobre una para comenzar a capturar en ella.

Por otra parte, las tres herramientas permiten seleccionar en qué interfaces de red capturar tráfico, con algunas diferencias. Para tcpdump, se selecciona la interfaz deseada con un comando y solo puede ser seleccionada una a la vez. Por otro lado, tshark y Wireshark pueden seleccionar varias interfaces a la vez con la diferencia en que en uno se hacen con comandos y en otro a través de la interfaz visual respectivamente.

A pesar de ser 3 herramientas con el mismo fin (analizar el tráfico de paquetes en una red) tienen diferencias en la forma en que lo hacen. Principalmente difieren en qué tan detallada muestran la información y para qué casos de uso están pensadas. Tcdump muestra un resumen de cada paquete directamente en texto y para ver toda la información hay que guardarlo en un archivo y abrirlo con otra herramienta. De esta forma se puede interoperar con las aplicaciones, obteniendo las capturas de forma rápida y analizándolas con los beneficios de la interfaz. La interoperabilidad es posible porque todas pueden guardar y leer capturas en formatos estándar como .pcap o .pcapng. Wireshark y TShark son más similares entre sí, ya que la diferencia está en la visualización de los datos obtenidos, a través

de la interfaz o de la consola. Tcdump y TSarhk están pensados para scripts y automatización mientras que Wireshark se utiliza para un análisis más profundo y visual de los protocolos.

1.2 Análisis del servidor incógnito

Comando que ejecutamos: sudo tcpdump -i lo0 host 127.0.0.1 and port 5001 -w incognito.pcap

Desglose del comando:

- `sudo` da permisos de administrador, que se necesitan para poder capturar paquetes.
- `-i` se usa para indicar la interfaz de la que se quiere capturar, en este caso es `lo0` que refiere a la interfaz de loopback, que es una interfaz virtual dentro del sistema operativo de la computadora. Se usa para enviar y recibir tráfico dentro de la misma máquina, sin salir a la red física.
- Agregando `host 127.0.0.1 and port 5001` estamos filtrando que solo se capturen los mensajes que salen o llegan al servidor que estamos corriendo en la segunda terminal.
- El último fragmento `-w captura_http.pcap` guarda lo capturado en un archivo llamado captura_http en formato pcap que luego podemos abrir en Wireshark.

Análisis de paquetes: Los primeros paquetes que observamos son los que se utilizan para establecer la conexión TCP, el three-way handshake.

El cliente comienza enviando un SYN para iniciar la conexión, el servidor responde con SYN, ACK para aceptar la conexión y luego cliente confirma la aceptación con ACK. Ninguno de esos primeros paquetes contiene datos, por el campo Len que indica el tamaño de los datos es 0.

Luego vienen los paquetes con datos. Los identificamos por tener [PSH, ACK] y Len mayor a 0. En el campo Data, están los bytes con los mensajes que enviamos

(xazf, bb, affhex, zzz) y las respuestas del servidor. Son seguidos por mensajes [ACK] que indican que el mensaje fue recibido correctamente. Esta confirmación es parte de lo que vuelve fiable al mecanismo TCP.

La captura continúa con este tipo de mensajes, hasta los últimos 4 paquetes que corresponden al cierre de conexión. Primero el servidor envía un [FIN, ACK] para indicar que ya no tiene más datos que enviar y que recibió todos los datos del cliente hasta ese momento. El cliente confirma que recibió ese mensaje respondiendo con ACK. Además envía también un [FIN, ACK] para indicar que tampoco tiene más datos para enviar. Por último el servidor confirma la recepción del FIN del cliente con un [ACK] y la conexión queda totalmente cerrada.

Análisis con Follow TCP Stream de la captura obtenida: Usando la funcionalidad "Following Protocol Streams" pudimos observar en rojo los mensajes que enviamos desde el Src Port 56794 al Dst Port 5001 y en azul los mensajes de respuesta que van desde Src Port 5001 a Dst Port 56794, ambos de la IP 127.0.0.1, ya que corren en la misma máquina. Esto nos permitió identificar las siguientes funcionalidades de cada comando:

- `xazf y bb` → Devuelven frases de distintos autores.
- `affhex` → Devuelve la fecha y hora actual en formato YYYY-MM-DD HH:MM:SS.
- `zzz` → actúa como comando de terminación, al recibirlo el servidor no responde contenido y cierra la conexión

1.3 Captura y análisis de tráfico

Captura página we de la fing:

Pasos que seguimos para realizar la captura:
Capturamos el tráfico usando: `sudo tcpdump -i en0 tcp port 443 -w https_web.pcap`
Luego accedimos con el navegador a: `https://www.fing.edu.uy`
Terminamos la captura y aplicamos los filtros: `ip.addr == 164.73.32.20 && tcp.port == 443`

Identificamos los siguientes mensajes:

- Establecimiento de conexión TCP (3-way handshake):. En este caso vemos que se crean dos conexiones TCP paralelas. Hay dos paquetes con el mensaje SYN a comienzo, uno desde el puerto 50804 y otro desde el 50805. Ambos puertos responden a los mensajes con SYN,ACK y terminan recibiendo un ACK. El navegador hace esto para paralelizar.

- Inicio de la conexión TLS: TLS es la conexión de la capa de transporte (Transport Layer Security), que agrega un cifrado a la conexión HTTP. La conexión comienza con el mensaje Client Hello con el url de la página que se quiere visitar, luego el servidor responde con "ServerHelo, Change Cipher Spec". Esto último indica que el servidor empezará a cifrar a partir de este punto. Los siguientes mensajes son datos de la aplicación que aparecen como Application Data porque ya están cifrados.

Al utilizar la función "Follow TCP Stream" vemos una serie de caracteres que no podemos leer directamente. Se diferencia con rojo los mensajes del cliente y con azul los del servidor nuevamente, pero ya no logramos interpretarlos. Lo que Wireshark está mostrando son los datos ya cifrados que viajan entre cliente y servidor.

Esto ocurre porque el protocolo que está siendo usado es HTTPS, no HTTP. Aunque HTTP normalmente es texto plano, en HTTPS, todo el contenido de HTTP (método, URL, encabezados y cuerpo) se cifra antes de ser transmitido.

Captura url de imagen:

Pasos que seguimos para realizar la captura:
Capturamos el tráfico usando: `sudo tcpdump -i en0 tcp port 80 -w http_imagen.pcap`
Luego accedimos con el navegador a: `http://www.columbia.edu/~fdc/family/ssamerica2.jpg`
Terminamos la captura y aplicamos los filtros:`ip.addr == 162.159.138.64 && tcp.port == 80`

La primer parte de stream podemos leerla claramente. Vemos primero el mensaje `GET` con el pedido del servidor a través de `HTTP 1.1`. Podemos identificar los encabezados del mensaje con información como el Host al que se solicita la imagen, y el `User-Agent` que indica el buscador en el cual se accede al url. También

vemos el encabezado `Connection: keep-alive` indica que se puede mantener una conexión TCP persistente si el servidor también la acepta. En el encabezado `Accept` también indica los tipos de contenido que el cliente acepta recibir del servidor, ordenados por preferencia.

En azul vemos la respuesta del servidor que comienza con una parte legible para el usuario, en la que se encuentran los encabezados de la respuesta a HTTP. El primer encabezado dice `200 OK` que es el código de estado de HTTP que indica que la solicitud fue procesada correctamente por el servidor y se devuelve el recurso solicitado.

En el encabezado también se encuentran los campos `Content Type:image/jpg` que indica el tipo de archivo de la respuesta, en este caso se trata de una imagen en formato jpg que era lo que solicitó el cliente y lo que se encuentra en la dirección url accedida. Siguiente a eso se ve el campo `Content Length:222449` que indica la longitud en bytes de los datos de respuesta, que coincide con el tamaño en bytes de la imagen.

Luego la respuesta tiene un salto de línea que marca el final de la cabecera del mensaje e indica que comienza la sección de `DATA`. El resto de la respuesta no puede ser leída directamente ya que consiste de los datos binarios de la imagen, que el dispositivo luego procesa para poder mostrarla.

1.4 Comando ping

Ping es una herramienta de red que se utiliza para verificar si un host está activo y accesible. Permite calcular el RTT (Round Trip Time) de un paquete (tiempo que tarda en ir del cliente al servidor y volver). Además permite detectar si hay una pérdida de paquetes o problemas de conectividad.

Ping utiliza el protocolo ICMP (Internet Control Message Protocol). Este es un protocolo utilizado por hosts y routers para comunicar información a nivel de red. Permite reportar errores, por ejemplo, host unreachable, network unreachable, port unreachable o protocol unreachable.

El comando ping envía un mensaje ICMP “Echo Request” al destino y el host remoto responde con el mensaje ICMP “Echo Reply”. Los mensajes ICMP pertenecen a la capa de red y viajan siempre encapsulados dentro de un datagrama IP, por lo que comparten el encabezado IP estándar cuyos campos principales son:

- Versión de IP
- Header Lengthe.
- Tipo de Servicio
- Datagram Length: largo total del datagrama IP en bytes.
- TTL (Time-to-live)
- Protocolo: Indica el protocolo de capa de transporte utilizado
- Header Checksum
- IP fuente y destino.
- Otras opciones.

Dentro de ese datagrama, el mensaje ICMP tiene su propio encabezado, compuesto por:

- Tipo: identifica el tipo de mensaje ICMP (ej. Echo Request = 8, Echo Reply = 0, Time Exceeded = 11).
- Código (Code): detalla el motivo dentro de ese tipo de mensaje.
- Checksum ICMP: verificación de errores para el propio mensaje ICMP.
- Campos adicionales específicos según el tipo de mensaje.

El comando ping muestra:

- El tiempo que tardó en recibir la respuesta (RTT)
- El número de bytes recibidos

- El TTL del paquete

Además muestra estadísticas:

- porcentaje de pérdida de paquetes
- RTT mínimo / promedio / máximo / desviación estándar

Para calcular el RTT de un paquete ping marca el tiempo exacto de envío de cada paquete y cuando llega una respuesta, resta la marca de tiempo de envío al de recepción. Al final, promedia los tiempos, calcula el mínimo, máximo y desviación.

Realizamos algunas capturas utilizando el comando para observar lo anterior. Mostramos como ejemplo una captura utilizando ping hacia www.google.com.

En una primera terminal ejecutamos: `sudo tcpdump -i en0 icmp -w ping_google.pcap` para guardar la captura con tcpdump.
Luego en otra terminal ejecutamos el comando ping: `ping -c 4 www.google.com`

Al ejecutar el comando se envían 4 mensajes ping y se muestra el RTT de cada uno en la terminal, junto con un informe de porcentaje de pérdida de paquetes y promedios de tiempos. Para calcular los RTT ping guarda la hora exacta en que se envía el mensaje y la hora exacta en que se recibe la respuesta. Estos mensajes pueden distinguirse en la captura tomada en WireShark porque tienen en el campo info el contenido `Echo (ping) reply` o `Echo (ping) request`.

1.5 Comando traceroute

El comando muestra la ruta completa que siguen los paquetes desde el host donde se ejecuta hasta un destino, listando cada router intermedio por el que pasan. Es útil para diagnosticar problemas de red, detectar dónde se corta una conexión, medir el tiempo que tardan en llegar los paquetes a cada router entre el host y el destino.

Funcionamiento: De acuerdo con el RFC 1393 de Traceroute sobre IP “El traceroute existente funciona enviando un paquete con un tiempo de vida (TTL) de 1.

El primer salto devuelve un mensaje de error ICMP [1] que indica que el paquete no se pudo reenviar porque el TTL expiró. El paquete se reenvía con un TTL de 2 y el segundo salto devuelve el TTL expirado. Este proceso continúa hasta que se alcanza el destino. El objetivo es registrar el origen de cada mensaje ICMP de TTL excedido para proporcionar un seguimiento de la ruta que siguió el paquete para llegar a su destino."

Cada router que descarta un paquete responde con el mensaje ICMP Time Exceeded. Cuando el cliente recibe este mensaje puede calcular el RTT del mensaje rechazado, sabiendo cuando fue enviado y cuando se recibió la notificación del rechazo.

Traceroute puede usar diferentes protocolos, dependiendo del sistema operativo o de los flags usados. En linux y MacOs usa por defecto, UDP a puertos altos (que no son utilizados). Agregando la bandera -I puede usar ICMP y agregando -T usa TCP pero es menos utilizado. En Windows se usa tracert en su lugar, que usa directamente ICMP.

Al ir incrementando el TTL de los mensajes, se alcanza una cantidad que equivale a la cantidad de routers entre el origen del mensaje y el destino, por lo que el servidor no recibe el mensaje de "Time Exceeded". La forma en que traceroute reconoce que el mensaje llegó al destino depende del protocolo sobre el que corra. Si traceroute usa UDP el paquete se envía a un puerto UDP muy alto que no es usado por el host de destino. Cuando el paquete llega, como no hay ningún proceso escuchando en ese puerto, el sistema responde con ICMP Port Unreachable y traceroute reconoce que el mensaje llegó a destino. Si se usa ICMP se envían mensajes ICMP Echo Request. Todos los routers intermedios responden con ICMP Time Exceeded y el destino final responde con ICMP Echo Reply, indicando que el destino fue alcanzado. Si se usa TCP, traceroute envía normalmente un SYN a un puerto y el destino final responde con SYN/ACK (si está abierto) o RST (si está cerrado).

Los encabezados también dependen del protocolo sobre el cual corre. Los encabezados del datagrama IP se encuentran siempre presentes, este es el encabezado

que contiene el TTL asignado al paquete. Luego se agregan los encabezados correspondientes al protocolo de la capa superior.

Para calcular el tiempo RTT el comando mide el tiempo desde que envía un paquete hasta que recibe la respuesta (ICMP). Se hacen 3 intentos por TTL y se muestran 3 tiempos por línea. Si no hay respuesta en cierto tiempo (timeout), muestra *.

1.6 Reflexión:

Si solo pudiéramos utilizar una de las herramientas, optaríamos por ping. La razón principal es su simplicidad y eficiencia para realizar un diagnóstico inicial y detectar problemas generales en la red que queremos analizar.

La mayor ventaja de ping frente a traceroute es que nos permite saber de forma rápida y clara si el host con el que queremos comunicarnos está disponible. En caso contrario, recibiremos mensajes ICMP como Destination Host Unreachable o simplemente Request timed out, que nos orientan sobre el tipo de problema y nos ayudan a identificar si se trata de una falta de conectividad o de un bloqueo en la red.

Además, como ping muestra directamente el RTT (Round Trip Time) hasta el host, resulta útil para diagnosticar si los retrasos que percibimos en nuestras aplicaciones provienen de la red o de la propia aplicación.

Si bien traceroute nos brinda información detallada sobre cada salto entre routers, esa información a veces puede ser incompleta, ya que no todos los nodos responden a las sondas. Consideramos que, para la mayoría de los diagnósticos que impactan en nuestras aplicaciones, es más relevante contar con una herramienta que nos diga si tenemos conectividad y latencia aceptables hacia un destino específico, en lugar de conocer cada detalle del camino intermedio, sobre el cual muchas veces no tenemos control directo.

References

[Figueredo and Wolf, 2009] Figueredo, A. J. and Wolf, P. S. A. (2009). Assortative pairing and life history strategy - a cross-cultural study. *Human Nature*, 20:317–330.