Redes de Comunicaciones I – Prácticas 2020
Práctica 3: Análisis de tráfico

Turno y pareja: 1302, P05

Integrantes:

Junco de las Heras Valenzuela

Marta Vaquerizo Núñez

Fecha de entrega:

Contenido

1 Introducción

En esta práctica se ha realizado un análisis de una *traza.pcap*, generada con el generador facilitado, mediante monitorización pasiva. Para realizar el análisis se realiza los siguientes cálculos:

- De porcentajes del tipo de protocolo encapsulado en el nivel de enlace y de red, para tener una idea de cuáles son los protocolos que predominan en la traza.
- De los siguientes distintos tipos de gráficas: las de forma de tarta, para saber cuáles son el top 5 de las direcciones IP que más bytes/paquetes envían o reciben, y de los puertos TCP y UDP más usados; ECDF, para saber la distribución asintótica del tamaño de paquete a nivel dos, y de los tiempos entre llegadas para la IP de flujo de TCP y para el puerto UDP; y por último, se van a calcular series temporales, para saber cuantos bits por segundo se capturan (ancho de banda).

Las gráficas se calculan en ambos sentidos, es decir, por cada dirección IP o dirección MAC o puerto TCP o UDP, se consideran dos gráficas, una como origen, y otra como destino.

Para realizar estos cálculos, se ha completado el script *practica3.py*. Las gráficas resultantes, se han guardado en la carpeta "resultados". En este informe se mostrarán y comentarán los resultados obtenidos.

2 Realización de la práctica

1. Análisis de protocolos.

En la carpeta "resultados" se ha añadido una captura de la ejecución del cálculo de los porcentajes, que se llama *porcentajes.png*.

Obtener los porcentajes de paquetes IP y NO IP (entendemos como NO-IP aquellos paquetes que no son ni ETH|IP ni ETH|VLAN|IP).

% Paquetes IP	**% Paquetes NO-IP**
99.05	0.95

Para sacar estos porcentajes, se ha usado el siguiente comando:

tshark -r "traza.pcap" -T fields -e eth.type –e vlan.etype

Se ha elegido este comando, ya que muestra los protocolos del nivel de red encapsulados en el nivel de enlace (con eth.type y vlan.type).

Se puede concluir que prácticamente todos los paquetes del tráfico capturado son IP.

Obtener los porcentajes de paquetes UDP, TCP y OTROS sobre los que son IP (igualmente entienda, un paquete IP como aquel que cumpla la pila ETH|IP o ETH|VLAN|IP).

% Paquetes TCP	**% Paquetes UDP**	**% Paquetes OTROS**
85.499	13.17	1.332

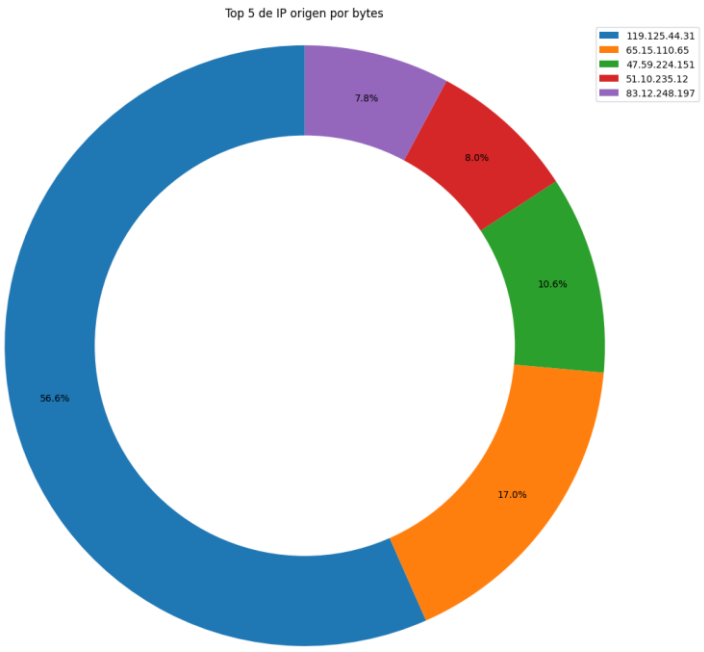
Para sacar estos porcentajes, se ha utilizado el siguiente comando:

tshark -r “traza.pcap” -T fields -e ip.proto -Y “ip”

Se ha elegido este comando, ya que muestra los protocolos encapsulados, filtrados por IP.

En este caso, los porcentajes están más repartidos que en el caso anterior, pero se observa que hay bastantes más paquetes IP con protocolo TCP que con protocolo UDP.

2. Obtención de top 5 de direcciones IP

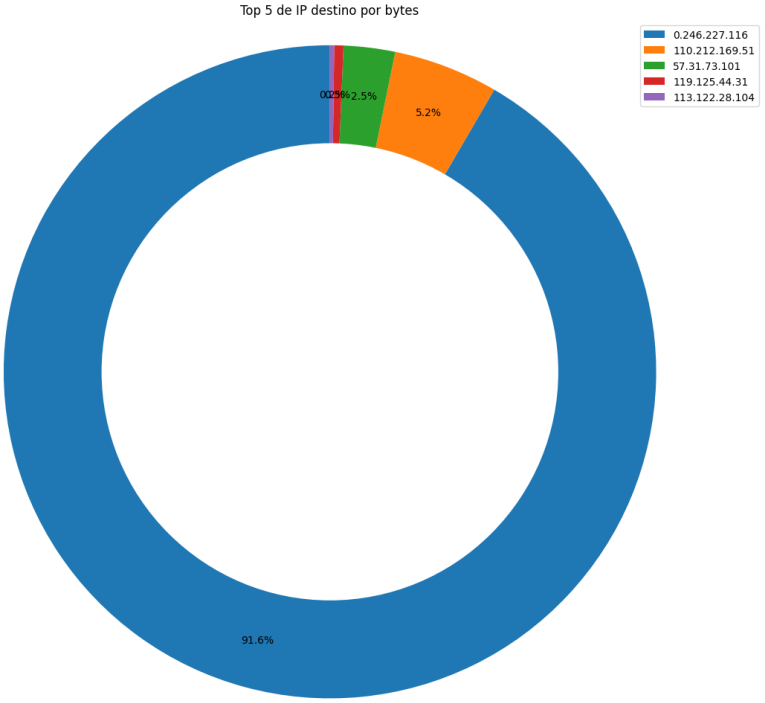


El comando usado para obtener los datos de la gráfica es el siguiente:

tshark -r “traza.pcap” -T fields -e ip.src -e frame.len

Con ip.src se obtienen las direcciones IP origen, y con frame.len se obtiene el tamaño del paquete. Con esta información, se puede obtener el número de bytes que la IP envía sumando el tamaño del paquete cada vez que aparezca la IP. Por ejemplo, la IP 0.0.0.0 envía un paquete de 100 Bytes, entonces se guarda ese dato, y luego envía otro paquete de 200 Bytes, entonces se suma al dato anterior, obteniendo que esa IP ha enviado 300 Bytes.

En la gráfica se puede observar que la dirección IP 119.125.44.31 es la que más bytes ha enviado, ocupando más de la mitad de la gráfica; seguida por las IP 65.15.110.65, 47.59.224.151, 51.10.235.12 y 83.12.248.197, donde la primera de ellas ocupa un 17% de la gráfica, mientras que las demás ocupan aproximadamente lo mismo(~10%).



El comando usado para obtener los datos de la gráfica es el siguiente:

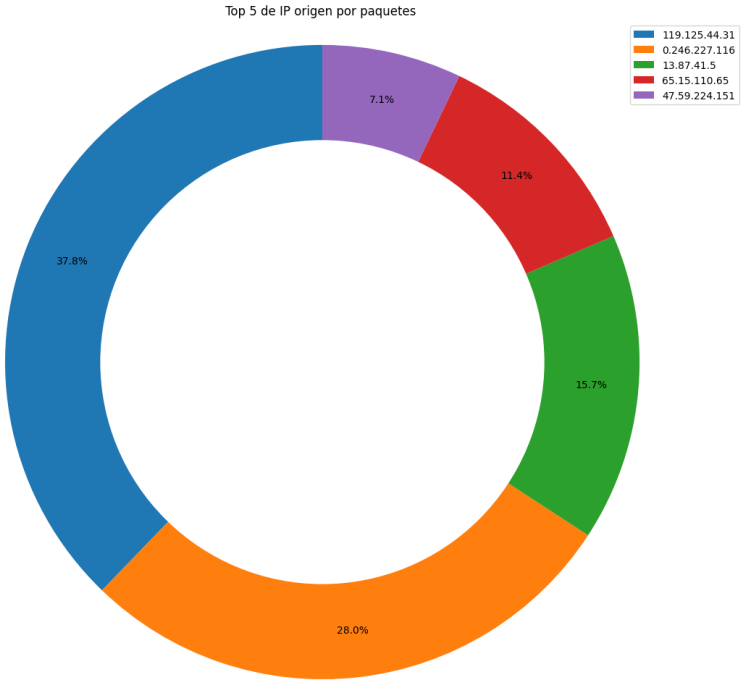
tshark -r “traza.pcap” -T fields -e ip.dst -e frame.len

Con ip.dst se obtienen las direcciones IP destino, y con frame.len se obtiene el tamaño del paquete. Con estos datos, se puede obtener el número de bytes que la IP envía sumando el tamaño del paquete cada vez que aparezca la IP. Por ejemplo, la IP 0.0.0.0 recibe un paquete de 100 Bytes, entonces se guarda ese dato, y luego recibe otro paquete de 200 Bytes, entonces se suma al dato anterior, obteniendo que esa IP ha recibido 300 Bytes.

En esta gráfica, en comparación con la anterior, se puede observar que hay una IP (0.246.227.116) que recibe una gran cantidad de bytes, ya que ocupa el 90% de la gráfica, y destaca por mucho respecto a las demás, que ocupan entre el 0.05% y el 5% de la gráfica. Esto puede implicar dos cosas, que la primera IP mencionada:

- Ha recibido bastante más paquetes que las otras direcciones IP.
- Los paquetes que ha recibido contenían muchos bytes, y los paquetes que han recibido las demás direcciones IP, no eran muy pesados (pocos bytes).

Cabe destacar que la dirección IP 119.125.44.31, que es el top 1 de la gráfica del apartado anterior, es decir, que envió muchos bytes, en esta gráfica ocupa un 0.05%, aproximadamente, lo que indica que no recibe tantos bytes respecto al total.



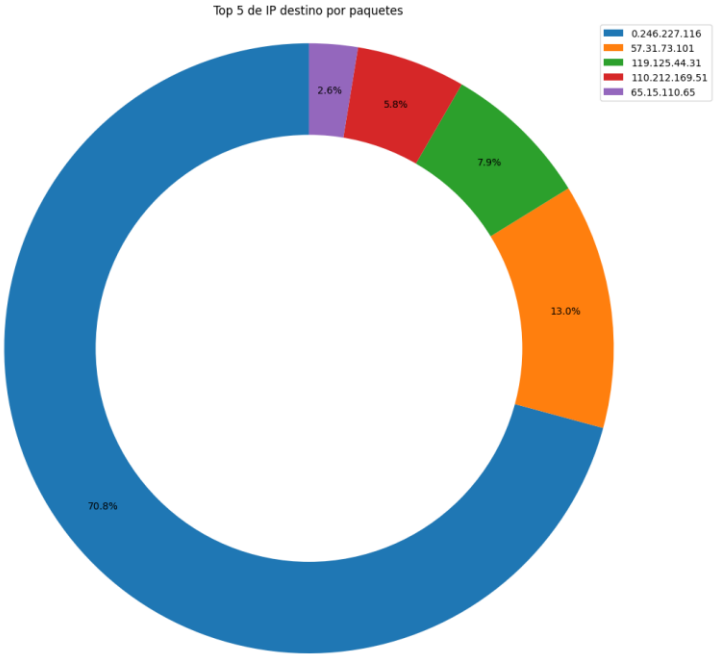
El comando usado para obtener los datos de la gráfica es el siguiente:

tshark -r “traza.pcap” -T fields -e ip.src

Se ha elegido este comando, ya que con ip.src se obtienen las direcciones IP origen. A partir de esta información, se puede sacar el número de paquetes que ha enviado la dirección contando las veces que sale la dirección.

En esta gráfica se puede observar que la dirección IP que envía más paquetes es la misma que la que envía más bytes en la primera gráfica. A diferencia de la primera gráfica, que ocupaba un 56% de la gráfica, en esta gráfica ocupa un 37%. La segunda dirección IP que más paquetes ha enviado es 0.246.227.116, que es la que más bytes ha recibido de acuerdo a la gráfica del apartado anterior. Curiosamente, esta dirección IP no aparece en el top 5 de direcciones IP por bytes, esto significa que aunque haya enviado muchos paquetes, éstos no contienen tanto bytes.

En relación con la primera gráfica, se puede destacar que la dirección IP top 2 65.15.110.65 (17% de la gráfica), en esta gráfica ocupa un 11%. Se puede observar que no hay una gran diferencia entre el porcentaje de la cantidad de bytes y el porcentaje del número de paquetes enviados, pero hay que tener en cuenta que estos porcentajes dependen del total de los bytes y los paquetes del top 5.



El comando usado para obtener los datos de la gráfica es el siguiente:

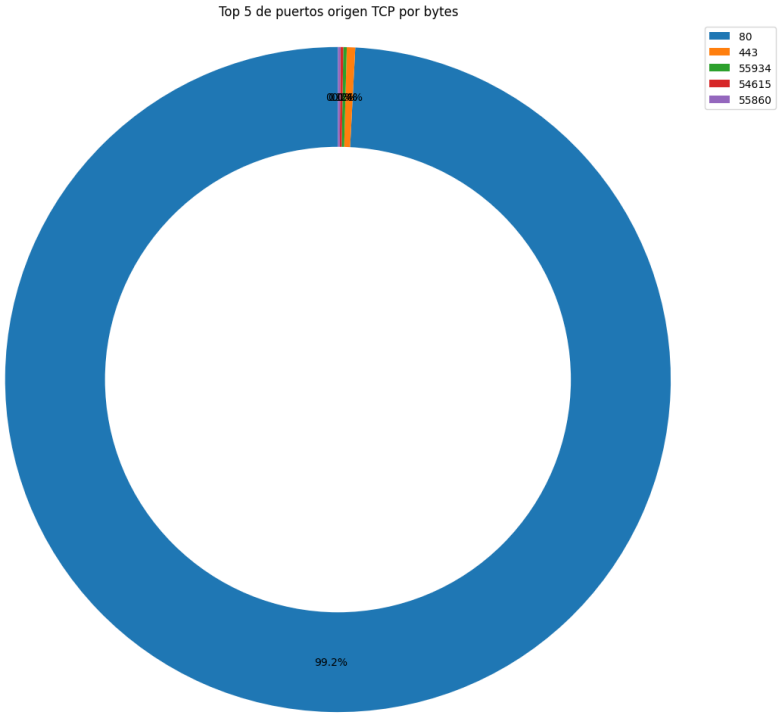
tshark -r “traza.pcap” -T fields -e ip.dst

Se ha elegido este comando ya que con ip.dst se obtienen las direcciones IP destino. A partir de esta información, se puede sacar el número de paquetes que ha recibido la dirección contando las veces que sale la dirección.

Se puede observar en la gráfica que la dirección IP en el top 1 es la misma que en el top 1 de la gráfica top 5 de bytes destino, lo que implica que 0.246.227.116 recibe muchos paquetes respecto al total, lo que produce que reciba muchos bytes respecto al total. Otro hecho destacable es que la IP 119.125.44.31 que recibía muy pocos bytes respecto al total (0.05% de la gráfica), en la gráfica top 5 de bytes destino, en esta gráfica ocupa casi un 8%, lo que implica que recibe en proporción más paquetes que bytes en la otra gráfica.

3. Obtención de top 5 de puertos:

TCP:

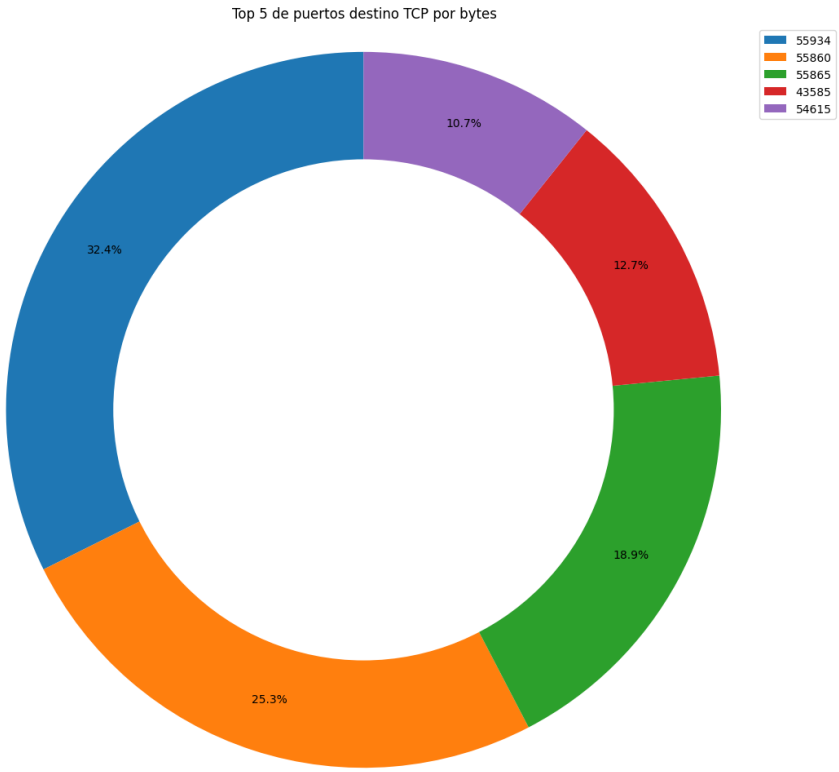


El comando usado para obtener los datos de la gráfica es el siguiente:

tshark -r “traza.pcap” -T fields -e tcp.srcport –e frame.len

Se ha elegido este comando ya que con tcp.srcport se obtienen los puertos origen del flujo TCP, y con frame.len se obtiene el tamaño del paquete. Para obtener el número de bytes que salen de cada puerto, se realiza el mismo cálculo que en una de las anteriores gráficas top 5 por bytes.

En esta gráfica se puede observar que por el puerto TCP 80 pasan prácticamente todos los paquetes del tráfico capturado, ocupando un 99.2% de la gráfica. Esto implica que por los otros cuatro puertos no pasan casi paquetes.

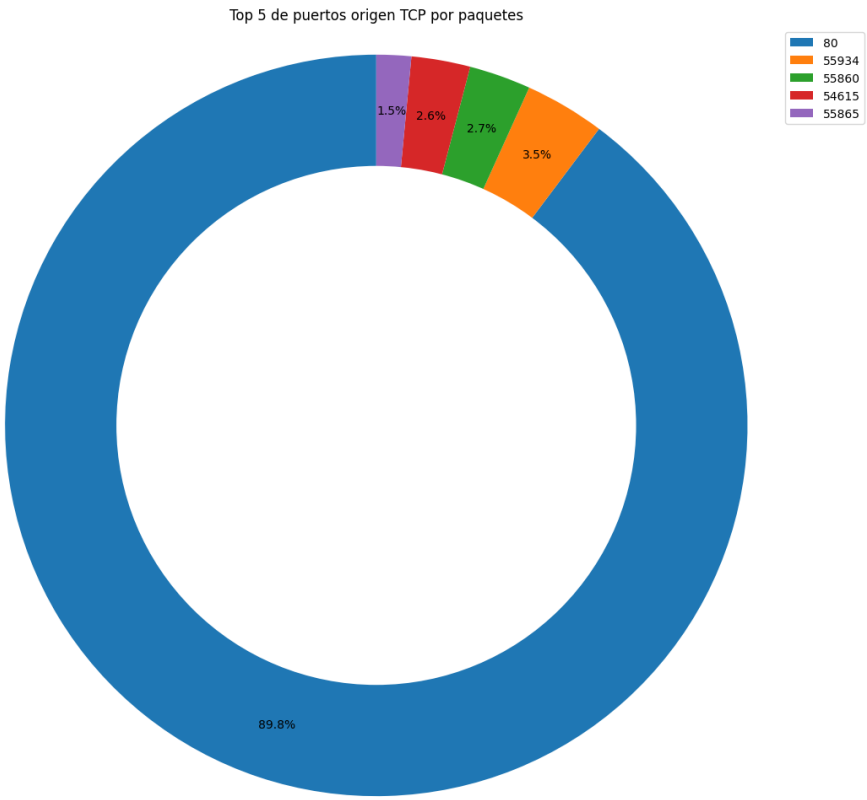


El comando usado para obtener los datos de la gráfica es el siguiente:

tshark -r “traza.pcap” -T fields -e tcp.dstport –e frame.len

Se ha elegido este comando ya que con tcp.dstport se obtienen los puertos destino del flujo TCP, y con frame.len se obtiene el tamaño del paquete. Para obtener el número de bytes que llegan a cada puerto, se realiza el mismo cálculo que en una de las anteriores gráficas top 5 por bytes.

En esta gráfica se puede observar que, a diferencia de las anteriores, está más repartida, no hay un puerto TCP que reciba muchos bytes respecto a los demás puertos. El puerto que recibe más Bytes es el 55934.

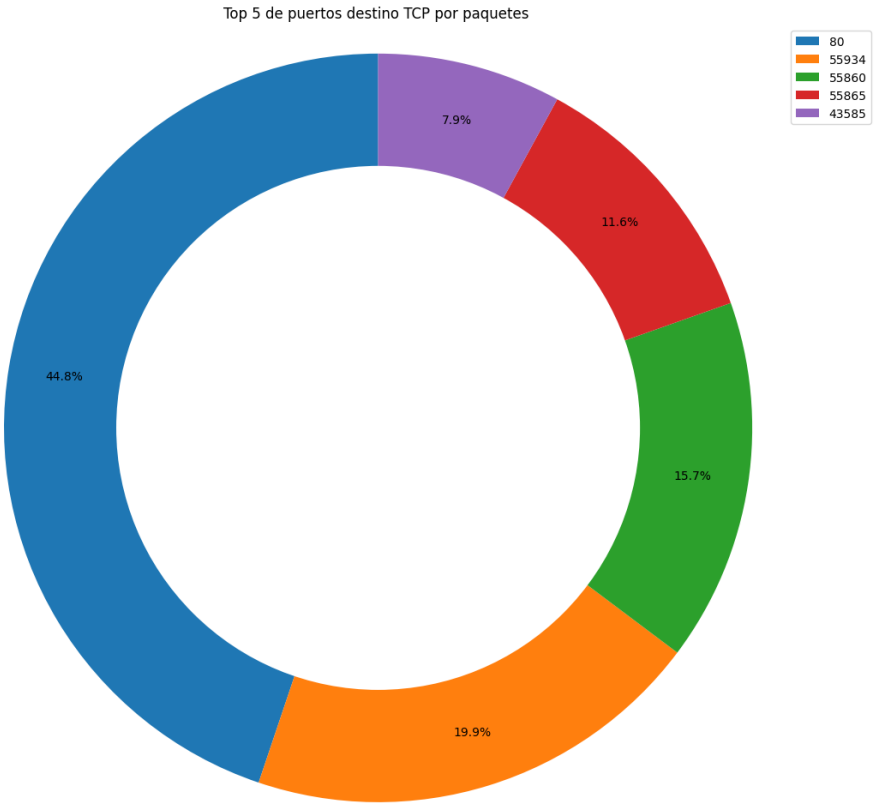


El comando usado para obtener los datos de la gráfica es el siguiente:

tshark -r “traza.pcap” -T fields -e tcp.srcport

Se ha elegido este comando ya que con tcp.srcport se obtienen los puertos origen del flujo TCP. Para obtener el número de paquetes que salen de cada puerto, se realiza el mismo cálculo que en una de las anteriores gráficas top 5 por paquetes.

En esta gráfica, se observa que el puerto origen 80, desde el que se enviaban más Bytes (99.2% de la otra gráfica), es desde el que se envían más paquetes (90% de la gráfica). Desde los demás puertos se envían más o menos el número de paquetes (porcentajes parecidos en la gráfica).



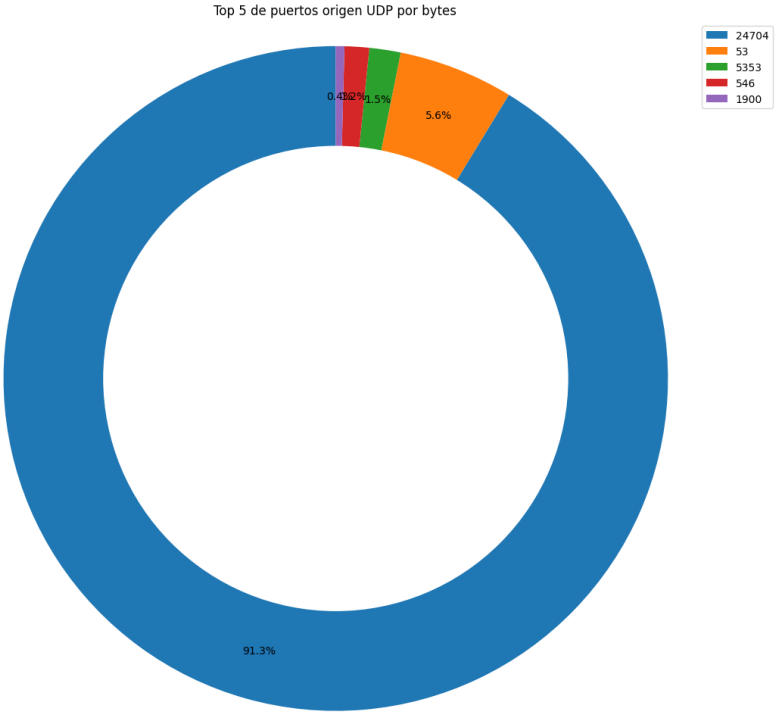
El comando usado para obtener los datos de la gráfica es el siguiente:

tshark -r “traza.pcap” -T fields -e tcp.dstport

Se ha elegido este comando ya que con tcp.dsrport se obtienen los puertos origen del flujo TCP. Para obtener el número de paquetes que llegan a cada puerto, se realiza el mismo cálculo que en una de las anteriores gráficas top 5 por paquetes.

En esta gráfica, se puede destacar que el puerto 80 es al que más paquetes llegan, pero este puerto no aparecía en el top 5 por bytes, es decir, que a pesar de que a él llegan más paquetes que a ningún otro, esos paquetes no contienen tal cantidad de Bytes para entrar en el top 5. El segundo puerto al que llegan más paquetes es el 55934 (20% en la gráfica), que es el top 1 al que llegan más bytes (32% en la otra gráfica – top 5 TCP destino por bytes). A los demás puertos llegan un número parecido de paquetes (porcentajes en la gráfica entre 8% y 16%).

UDP:

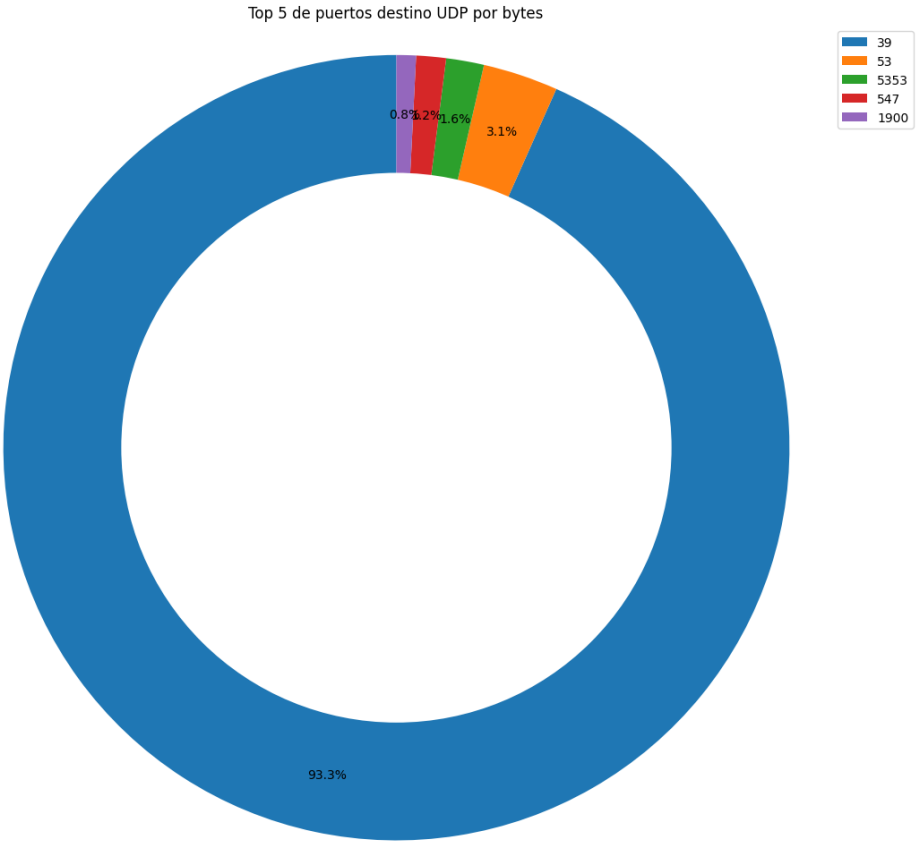


El comando usado para obtener los datos de la gráfica es el siguiente:

tshark -r “traza.pcap” -T fields -e udp.srcport –e frame.len

Se ha elegido este comando ya que con udp.srcport se obtienen los puertos origen del flujo UDP, y con frame.len se obtiene el tamaño del paquete. Para obtener el número de bytes que salen de cada puerto, se realiza el mismo cálculo que en una de las anteriores gráficas top 5 por bytes.

En la gráfica, se observa que el puerto 24704 es desde donde se envía más Bytes, ocupando un 91.3% de la gráfica. Con un 5.6% le sigue el puerto 53, y los demás puertos no ocupan casi nada de la gráfica, por lo que el número de Bytes, que se envían, será despreciable respecto al número de Bytes de los dos primeros.

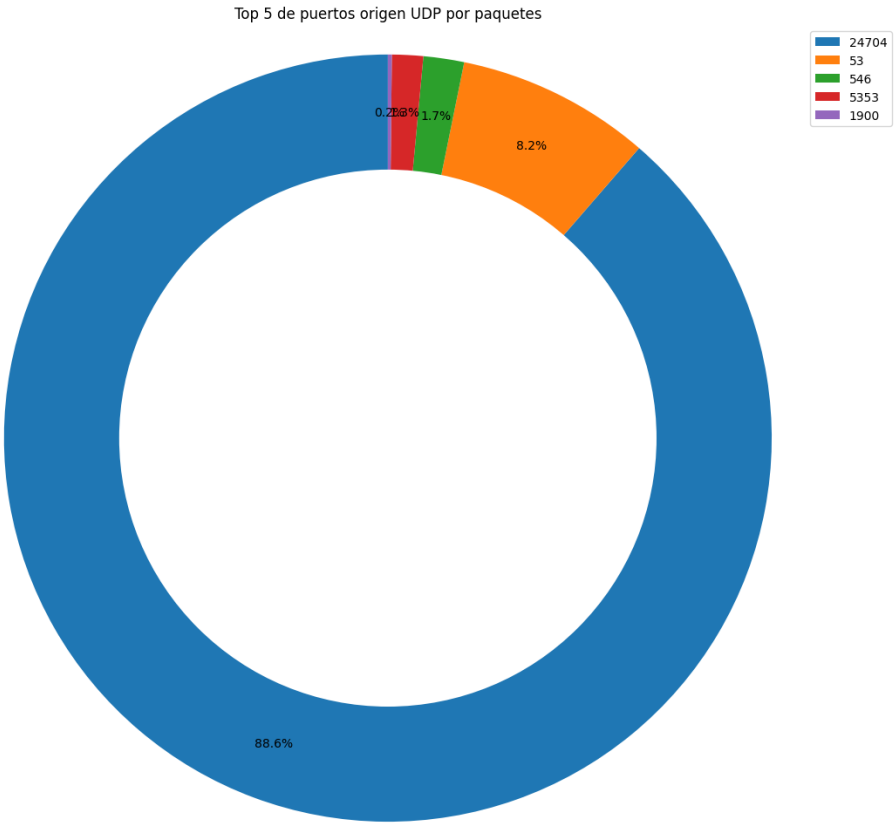


El comando usado para obtener los datos de la gráfica es el siguiente:

tshark -r “traza.pcap” -T fields -e udp.dstport –e frame.len

Se ha elegido este comando ya que con udp.dstport se obtienen los puertos destino del flujo UDP, y con frame.len se obtiene el tamaño del paquete. Para obtener el número de bytes que llegan a cada puerto, se realiza el mismo cálculo que en una de las anteriores gráficas top 5 por bytes.

En esta gráfica, destaca que el puerto al que llegan más bytes es el 39, ocupando un 93.3% de ésta. Por lo que la cantidad de Bytes que llegan a los demás puertos es muy pequeña respecto a la del puerto 39. El segundo puerto con un 3% es el 53.

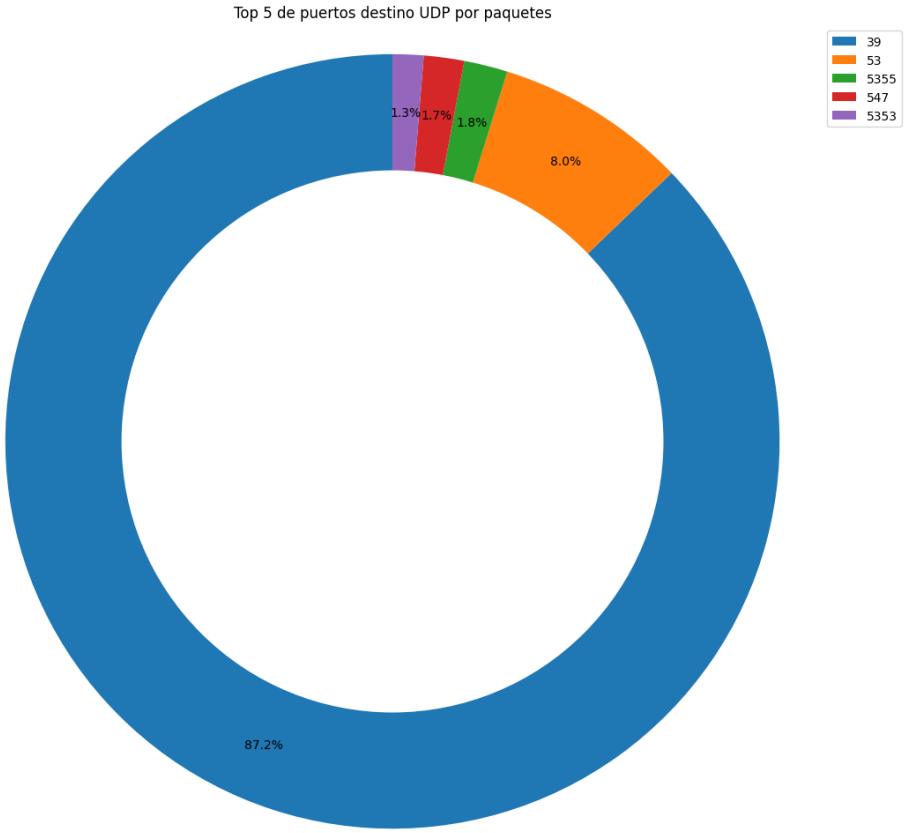


El comando usado para obtener los datos de la gráfica es el siguiente:

tshark -r “traza.pcap” -T fields -e udp.srcport

Se ha elegido este comando ya que con udp.srcport se obtienen los puertos origen del flujo UDP. Para obtener el número de paquetes que salen de cada puerto, se realiza el mismo cálculo que en una de las anteriores gráficas top 5 por paquetes.

De esta gráfica se puede destacar que el puerto desde el que se envían más paquetes (88.6% en la gráfica) es el mismo desde el que se envían más Bytes (91.3% de la otra gráfica). Asimismo, el segundo puerto es el mismo que el top 2 de la gráfica de Bytes. Los demás puertos representan un porcentaje muy pequeño de la gráfica respecto a estos dos últimos.



El comando usado para obtener los datos de la gráfica es el siguiente:

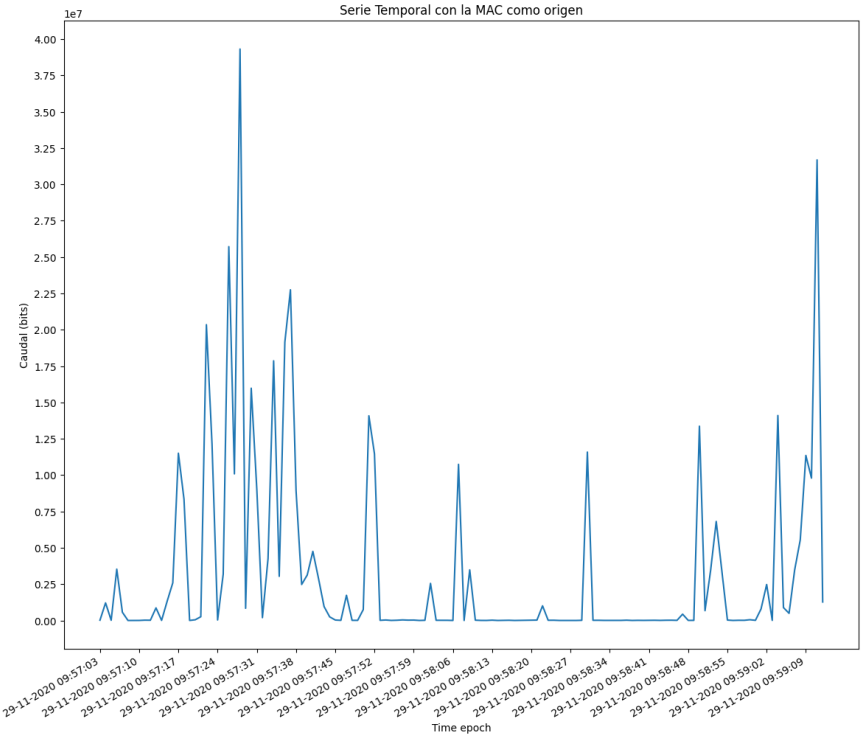
tshark -r “traza.pcap” -T fields -e udp.dstport

Se ha elegido este comando ya que con udp.dstport se obtienen los puertos destino del flujo UDP. Para obtener el número de paquetes que llegan a cada puerto, se realiza el mismo cálculo que en una de las anteriores gráficas top 5 por paquetes.

En esta gráfica, se puede destacar que el puerto al que llegan más paquetes (39) es el mismo puerto que el top 1 de la gráfica de Bytes, así como que el segundo (53) es el top 2 de esa misma gráfica.

4. Series temporales de ancho de banda/tasa/caudal:

La MAC que se está usando es: 00:11:88:CC:33:79, y nos referiremos a ella en la memoria como MAC.

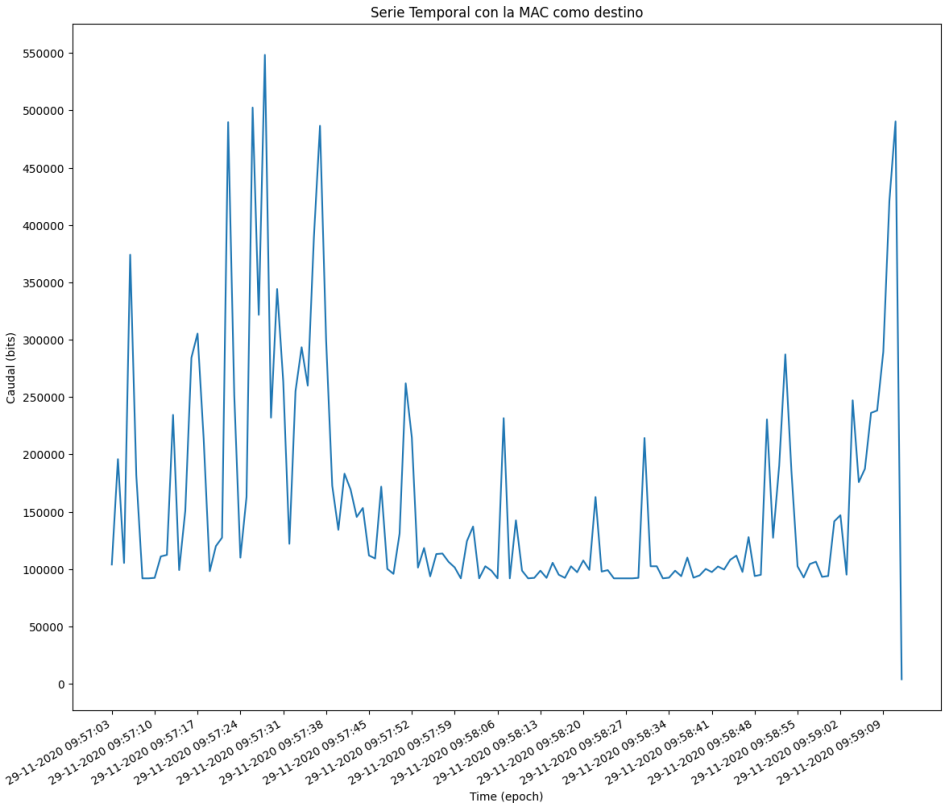


El comando usado para obtener los datos de la gráfica es el siguiente:

tshark -r “traza.pcap” -T fields -e frame.time_epoch -e frame.len -Y "eth and eth.src eq MAC”

Con frame.time_epoch conseguimos el tiempo de captura del paquete, que nos servirá para poder hacer la gráfica; con frame.len conseguimos los bytes del paquete que se pasarán a bits para la gráfica; y con –Y "eth and eth.src eq MAC” filtramos que solo se quieren paquetes a nivel de enlace, y que la dirección Ethernet origen sea la MAC. Con estos datos se puede calcular en cada segundo cuantos bits se han capturado, y si entre segundos, no se capturan paquetes, se añaden 0’s.

En la gráfica, se puede observar que hay valores del tiempo (eje de abscisas) que toman el valor 0 bits, esto implica que en esos tiempos no se capturan paquetes. También se puede destacar que hay un máximo absoluto toma el valor de $4e^7 bits$ ~ 550 Bytes, lo que implica que el máximo de Bytes que se han enviado desde la MAC es 550.



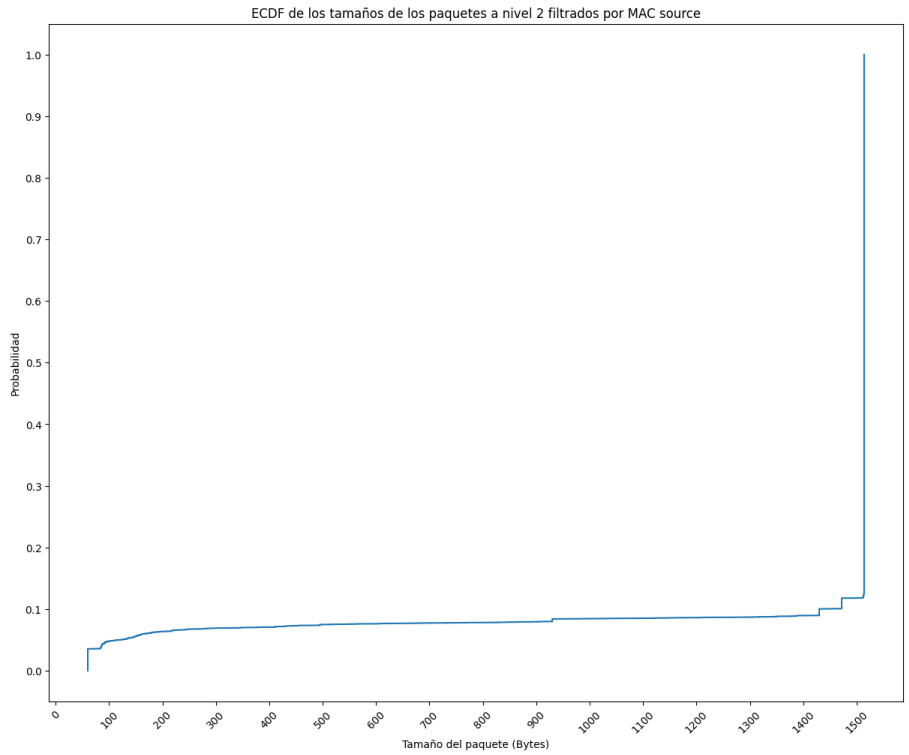
El comando usado para obtener los datos de la gráfica es el siguiente:

tshark -r “traza.pcap” -T fields -e frame.time_epoch -e frame.len -Y "eth and eth.dst eq MAC”

Con frame.time_epoch conseguimos el tiempo de captura del paquete, que nos servirá para poder hacer la gráfica; con frame.len conseguimos los bytes del paquete que se pasarán a bits para la gráfica; y con –Y "eth and eth.dst eq MAC” filtramos que solo se quieren paquetes a nivel de enlace, y que la dirección Ethernet destino sea la MAC. Los datos de la gráfica se calculan como en el apartado anterior, pero con la MAC como destino.

En esta gráfica, a diferencia de la anterior, no se observa que en algún segundo, se tome el valor de 0 bits, es decir, que en cada segundo se captura al menos un paquete. Otra diferencia es que la cantidad de bits que recibe la MAC es mucho mayor que los que enviaba (gráfica anterior). El máximo absoluto en esta gráfica se alcanza en 550000 bits ~ 68750 Bytes, y el mínimo absoluto, 100000 bits ~ 12500 Bytes. Se puede concluir que en cada segundo, la cantidad de bits oscila entre los 100000 y 550000 (12500 y 68750 pasado a Bytes).

5. ECDFs de los tamaños de los paquetes

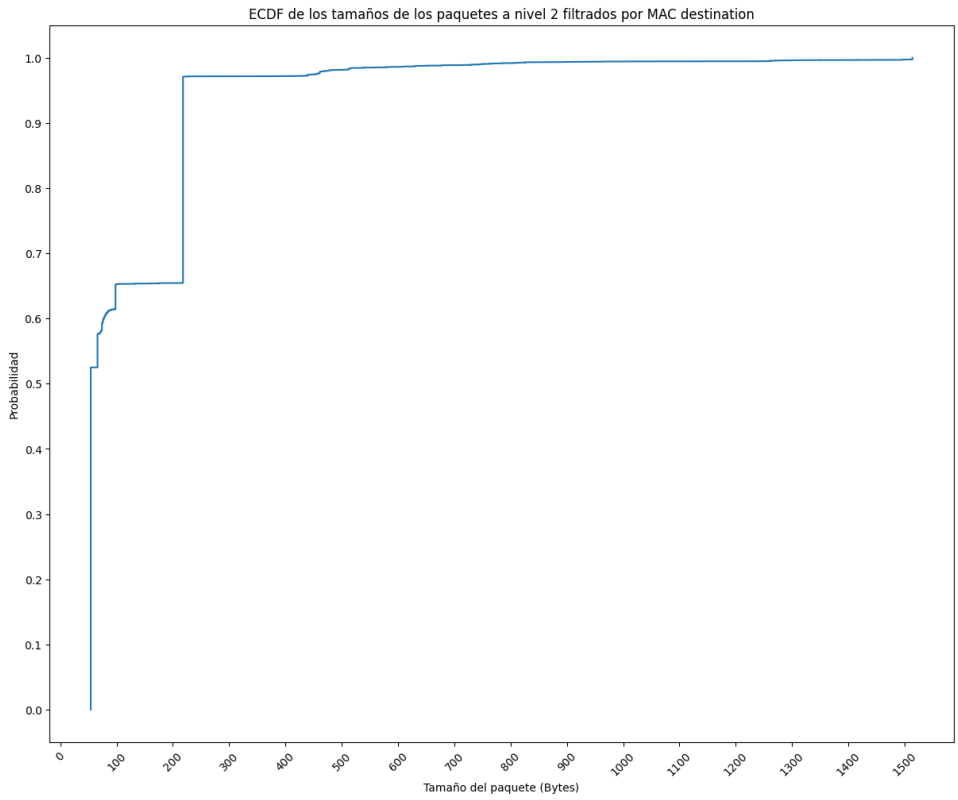


El comando usado para obtener los datos de la gráfica es el siguiente:

tshark -r “traza.pcap” -T fields -e frame.len -Y "eth and eth.src eq MAC”

Con frame.len conseguimos los bytes del paquete, y con –Y "eth and eth.src eq MAC” filtramos que solo se quieren paquetes a nivel de enlace, y que la dirección Ethernet origen sea la MAC. Con estos datos, se ordenan de menor a mayor, y se calcula la gráfica.

En de la gráfica, se observa que cuando la dirección origen es la MAC, a nivel de enlace, la probabilidad de que el tamaño de paquete sea menor o igual a un valor entre 60 y 1514 B (función de distribución), es como máximo 0.1, es decir, es poco probable que se encuentre un paquete con esos valores, pero realmente, esos valores son los tamaños que un paquete puede tomar en general, por eso se concluye que desde esa dirección MAC no se envían casi paquetes.



El comando usado para obtener los datos de la gráfica es el siguiente:

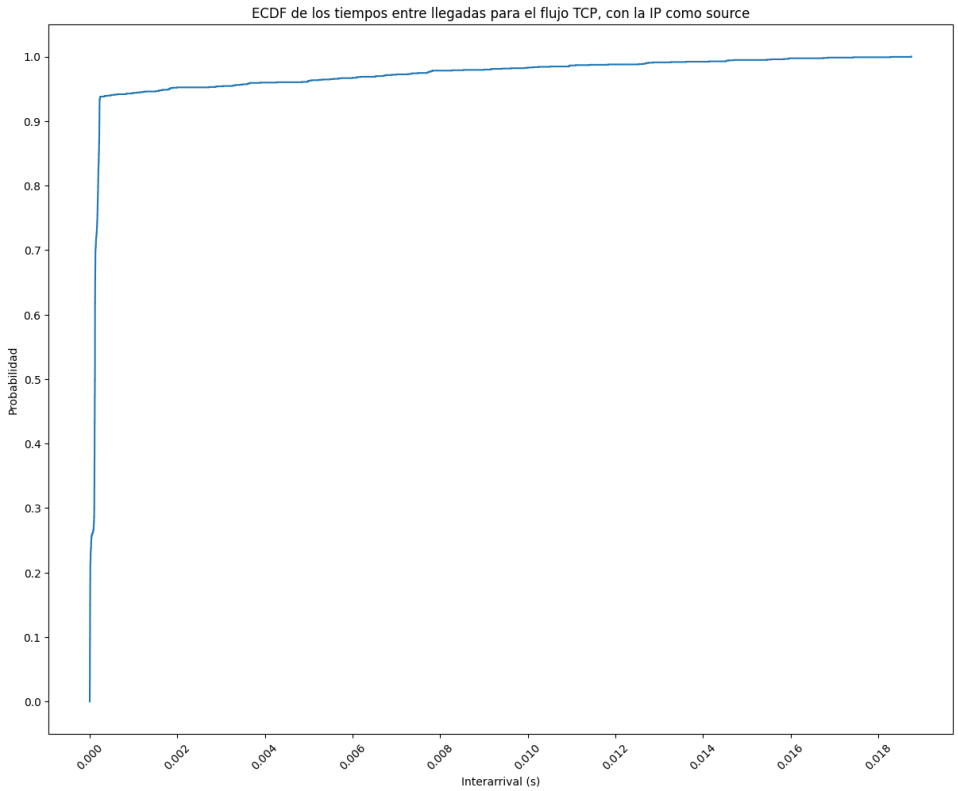
tshark -r “traza.pcap” -T fields -e frame.len -Y "eth and eth.dst eq MAC”

Con frame.len conseguimos los bytes del paquete, y con –Y "eth and eth.dst eq MAC” filtramos que solo se quieren paquetes a nivel de enlace, y que la dirección Ethernet destino sea la MAC. Con estos datos, se ordenan de menor a mayor, y se calcula la gráfica.

En la gráfica, se observa que cuando la dirección destino es la MAC, a nivel de enlace, la probabilidad de que el tamaño de paquete sea mayor de 280 Bytes (aprox) es aproximadamente 0.4, es decir, que casi la mitad de los paquetes que han llegado a esta dirección, tienen un tamaño mayor que 280 Bytes. Por otro lado, los paquetes con un tamaño entre 60 y 70 Bytes tienen una probabilidad de 0.5, por lo que la mitad de paquetes que llegan son de este tamaño. Los demás tamaños no son tan probable como los mencionados.

6. ECDF tiempos entre paquetes

En este apartado, se va a hacer referencia a la dirección IP del flujo TCP (89.223.72.193), a la nos referiremos como IPT, y el puerto UDP (39), al que nos referiremos como PORT. Estos nombres de referencia solo se usan en la memoria.

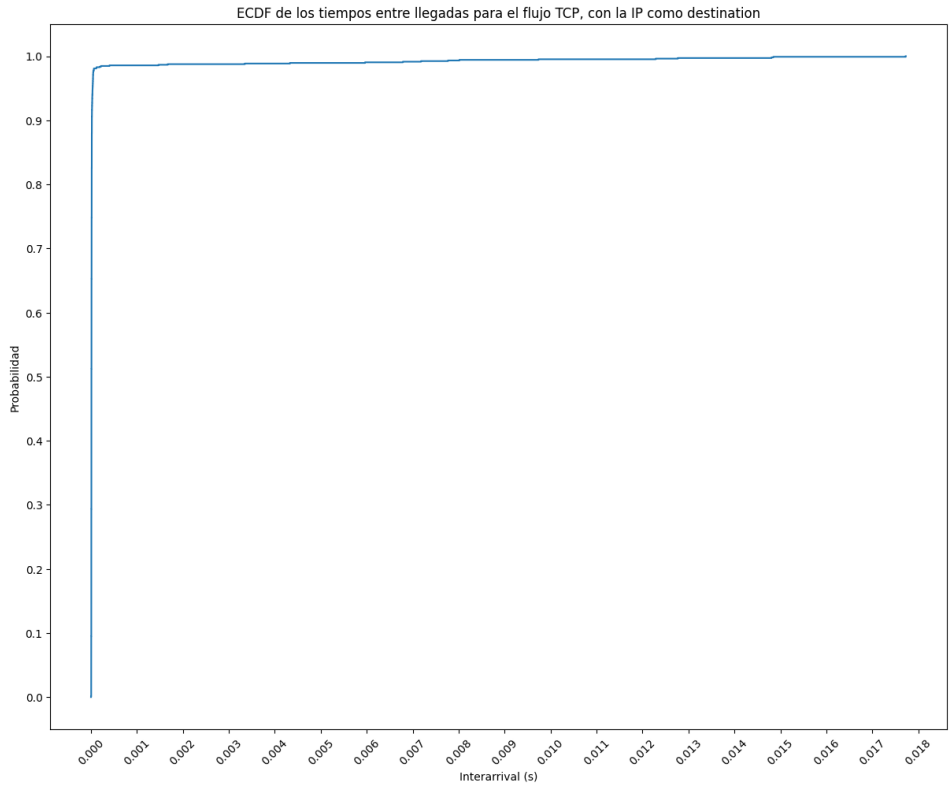


El comando usado para obtener los datos de la gráfica es el siguiente:

tshark -r “traza.pcap” -T fields -e frame.time_delta-Y "tcp and ip.src eq IP_T”

Con frame.time_delta se consigue el intervalo de tiempo desde el anterior paquete capturado, y con –Y "tcp and ip.src eq IP_T” se filtra que solo se quieren paquetes del flujo TCP, y que la dirección IP origen sea la IP_T. Con estos datos, se ordenan de menor a mayor, y se calcula la gráfica.

En la gráfica se observa que la mayoría de intervalos de tiempo entre llegadas de paquetes es menor que 0.001 segundos, y luego hay pocos intervalos que tardar entre 0.001 y 0.018 segundos.



El comando usado para obtener los datos de la gráfica es el siguiente:

tshark -r “traza.pcap” -T fields -e frame.time_delta-Y "tcp and ip.dst eq IP_T”

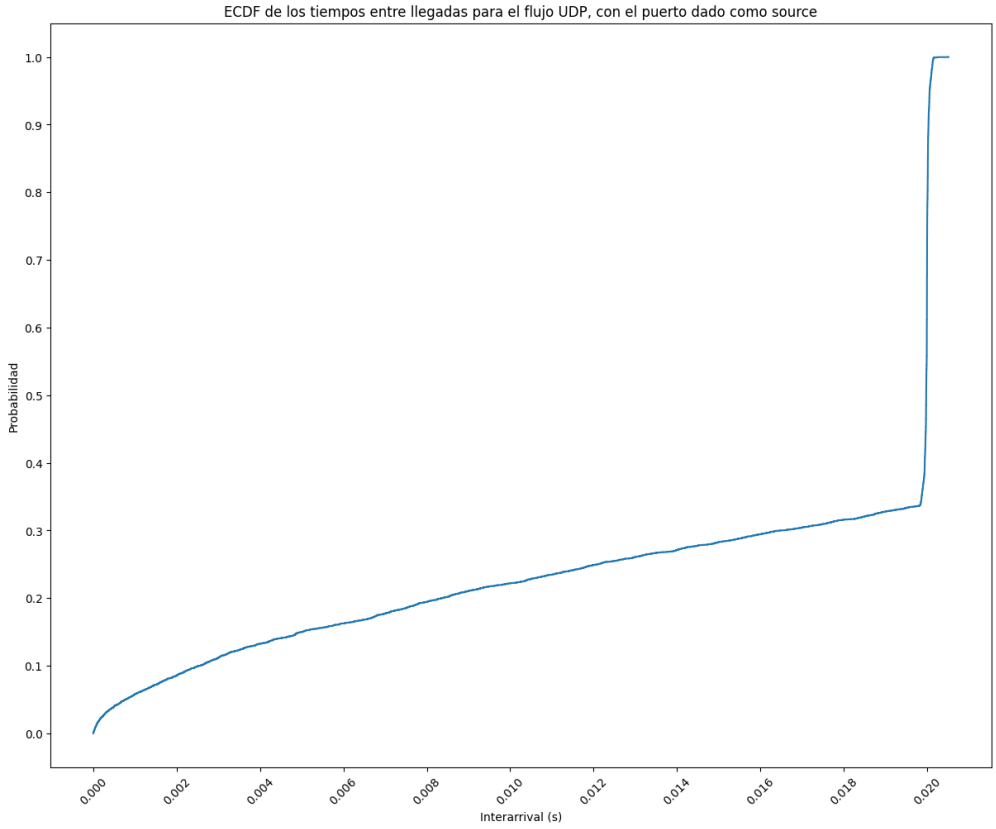
Con frame.time_delta se consigue el intervalo de tiempo desde el anterior paquete capturado, y con –Y "tcp and ip.dst eq IP_T” se filtra que solo se quieren paquetes del flujo TCP, y que la dirección IP destino sea la IP_T. Con estos datos, se ordenan de menor a mayor, y se calcula la gráfica.

En esta gráfica se ve claramente que prácticamente todos los paquetes llegan microsegundos después del anterior paquete capturado. Lo que implica que hay muy pocos paquetes que llegan milisengundos después del anterior.

En esta celda iría la ECDF correspondiente a los tiempos entre llegadas del flujo UDP con la IP como origen, pero al ejecutarlo no grafica nada, es decir, que no hay ningún paquete que pase por ese puerto UDP. Pero se usa el siguiente comando:

tshark -r “traza.pcap” -T fields -e frame.time_delta -Y "udp and udp.srcport eq PORT”

Con frame.time_delta se consigue el intervalo de tiempo desde el anterior paquete capturado, y con –Y " udp and udp.srcport eq PORT” se filtra que solo se quieren paquetes del flujo UDP, y que el puerto UDP origen sea la PORT. Con estos datos, se ordenan de menor a mayor, y se calcula la gráfica.



El comando usado para obtener los datos de la gráfica es el siguiente:

tshark -r “traza.pcap” -T fields -e frame.time_delta -Y "udp and udp.dstport eq PORT”

Con frame.time_delta se consigue el intervalo de tiempo desde el anterior paquete capturado, y con –Y " udp and udp.dstport eq PORT” se filtra que solo se quieren paquetes del flujo UDP, y que el puerto UDP destino sea la PORT. Con estos datos, se ordenan de menor a mayor, y se calcula la gráfica.

En esta gráfica, se puede observar que más de la mitad (0.7 de probabilidad en la gráfica) de los paquetes llegan 0.002 o más segundos después del anterior paquete capturado. Los demás intervalos de tiempo entre llegadas de paquetes oscilan entre 0.0001 y 0.019 segundos. La probabilidad de que el intervalo de tiempo sea menor o igual que 0.019 segundos es de 0.3.

3 Conclusiones

Para desarrollar la práctica hemos aprendido a usar la herramienta tshark, una especie de versión de terminal de wireshark. Además, ha mejorado nuestro conocimiento de procesamiento de paquetes de red, analizando los atributos del paquete y comprendiendo los campos de los mismos, así como sus cabeceras. También hemos mejorado nuestro nivel de Python y de Bash gracias al tutorial de Moodle.

En nuestra traza, casi todos los paquetes eran IP, y, en casi todas las gráficas tipo tarta, la porción más alta predominaba, en la mayoría de casos, por mucho sobre las demás porciones. Esto implica que no hay homogeneidad en el envío y la recepción de paquetes. Por ejemplo, el puerto TCP desde el que salían más paquetes era el 80. El hecho de que un puerto que sea mucho más usado que los otros puede significar que normalmente, si tomas un intervalo de tiempo en la red, o una dirección IP ha enviado muchos paquetes a un servidor/terminal con ese puerto, o no todos los equipos están igualmente distribuidos.

Con las series temporales, hemos podido comprobar que aunque parece que la red presente una regularidad, se observan picos de bits/s no esperados en las gráficas, así que si se está administrando o manteniendo la red, habrá que estar preparado para que la red soporte esas subidas de demanda.

Las ECDFs han servido para ver la probabilidad de los intervalos de tiempo entre llegadas de paquetes de distintos flujos (TCP y UDP), así como de su tamaño cuando son a nivel de enlace. Cuanto mayor es la pendiente en un punto, mayor es la probabilidad de que la variable tome ese valor.

En general, ha sido una práctica que nos ha ayudado a comprender mejor el análisis de los paquetes que navegan por la red, así como la cantidad de bytes que circula por la red, es decir, la congestión de la misma.