Informe

Victor Navajas, Micaela Oliva y Exequiel Muñoz

Informe de Tecnología Digital 4

Información General

- **Curso:** Tecnología Digital 4
- **Fecha:** [29 de octubre de 2023]
- **Profesores:** [Emmanuel Iarussi y Lucio Emilio Santi]
- **Estudiantes:** [Victor Navajas, Micaela Oliva y Exequiel Munoz]

Resumen

En este informe, se abordan tres aspectos del análisis de redes. En la primera parte, implementamos Traceroute utilizando Python y Scapy para rastrear la ruta y los tiempos de respuesta entre un origen y un destino. En la segunda parte, desarrollamos un Port Scanner que detecta el estado de los puertos en un host. Este escáner puede considerar puertos abiertos, cerrados o filtrados y se extiende para verificar la conexión TCP completa. Finalmente, en la tercera parte, realizamos experimentos con estas herramientas en universidades de diferentes continentes y se analizan los resultados, incluyendo un análisis comparativo con la herramienta nmap, adicionalmente vamos a presentar experimentos realizados por nosotros. El informe busca proporcionar una visión integral de estas herramientas y sus aplicaciones en el análisis de redes.

Índice

1. Introducción

Traceroute... Portscanner...

2. Explicación de la Implementación

2.1 Traceroute

Creamos una funcion llamada Traceroute donde pasamos como parametros al host destino y la cantidad maxima de TTL (Time to Live) que tendra ese paquete. Usando la libreria Scapy para cada uno de los TTLs creamos un paquete IP, con su TTL correspondiente hasta llegar a la cantidad maxima. En nuestro caso decidimos definir un TTL igual a 45, que es un poco mayor a lo que esta definido en el Traceroute de Windows (TTL=30),esto de debe a que en algunos casos requerimos un numero de TTL mayor a 30 porque los host de destino estaban muy lejos. Luego enviamos los paquetes. Esperamos las respuestas por 1 segundo para evitar demoras ante una pérdida ya que al hacer varias pruebas observamos que las respuestas no demoraban mas de 500 ms. Suponemos que si se supera el tiempo, no hubo respuesta. Adicionalemte tomamos el tiempo con la libreria Time para los experimentos. En el caso de que el TTL expira en transito(ICMP de tipo 11 y codigo 0), imprimimos por pantalla la IP del router con el tiempo correspondiente. Por otro lado, tambien contemplamos el caso en el que si llega al destino. Adicionalemte, en el caso donde no reciabamos respuesta, aparece un "*".

2.2 Port Scanner I

Para implementar el primer Port Scanner, nos basamos en un SYN scan, que envia paquetes SYN a los priemros 1000 puertos de un host destino y chequea si responde con un SYN-ACK. Para eso usamos la libreria Scapy para crear un paquete IP con un flag SYN de TCP, luego los enviamos. Definimos un tiempo de espera de las respuestas de 500 ms, llegamos a este tiempo lugo de hacer varias pruebas y observamos que recibiamos un numero de respuestas satisfactorio para una cantidad de tiempo razonable (no muy larga y sin mucha diferencia contra tiempos mas altos). Para cada paquete enviado, chqueamos si la respuesta tiene los flags SYN y ACK prendidos, en el caso de que se cumpla, consideramos al puerto como "open" o abierto. En el caso contrario, es decir, que no recibamos respuesta o que este cerrado (la respuesta es negativa) los consideramos "filtered" o filtrado.

2.3 Port Scanner II

A diferencia del primer Port Scanner, para el segundo Port Scanner lo basamos en un CONNECT scan, este funciona de manera similar al priemro pero al recibir una respuesta positiva se envia adicionalmente otro paquete ACK con un payload en espera de un ACK. Por lo cual creamos un paquete IP con un flag SYN, lo enviamos en espera de una respuesta con un tiempo de espera de 500 ms por la misma razon que elegimos este tiempo en el primer Port Scanner. En el caso de recibir una respuesta positiva, es decir un paquete SYN-ACK,procedemos a crear un paquete con un ACK y un payload, en el caso contrario consideramos a ese puerto como "filtered". En nustro caso decidimos que el payload sea un string de "Hello, [IP del destino!]". Luego esperamos la respuesta con una espera de 500ms, por lo cual si esta llega correctamente, consideramos al puerto escaneado como "open", en el caso contrario lo consideramos como "filtered". Tambien implementamos una funcion llamada Scan Ports que guarda los resultados de ambos Port Scanners en archivos de texto.

3. Experimentación

3.1 Experimentaciones de traceroute

a Para el primer experimento elegimos 6 universidades de distintos continentes y son:

- Universidad de Massachusets-Amherst (America del Norte) - http://gaia.cs.umas.edu/ (GAIA)
- Universidad China de Hong Kong (Asia) - http://www.cuhk.edu.hk/ (CUHK)
- Universidad Africana de Ciencia y Tecnologia (Africa) - http://aust.edu.ng/ (AUST)
- Universidad de Sorbonne (Europa) - http://www.sorbonne-universite.fr/ (SORBONNE)
- Universidad de San Pablo (America del Sur) - http://www5.usp.br/ (USP)
- Universidad Torcuato di Tella (America del Sur) - http://www.utdt.edu/ (UTDT)

Para cada una de las universidades elegidas ejecutamos el traceroute con sus respectivas URLs, implementando el codigo de `traceroute` tal que nos muestra el camino hecho del paquete por cada router hasta llegar a

su destino y que nos devuelva el porcentaje de paquetes con `ttl-zero-during-transit`, el porcentaje de paquetes de `no-response` y si el paquete termina llegando a su destino o no con `destination-reached`.

Implementamos a `ttl-zero-during-transit` que a partir de un contador que cuenta la cantidad de saltos de las cuales recibimos respuesta exitosa de un router intermedio, a ese valor del contador lo dividimos por la cantidad de hops totales que terminó haciendo ese paquete, en el caso de que haya llegado a su destino podría llegar a ser menor que los 45 hops que habíamos puesto como el máximo, en el otro caso, si no llegaba al destino contaría los 45 hops. Por lo cuál, para calcular el porcentaje de `ttl-zero-during-transit`:

$$\%\ ttl - zero - during - transit\ = \frac{\#casos\ exitosos\ de\ losroutersintermedios}{\#hops\ totales\ del\ paquete}$$

Por el otro lado, para calcular el porcentaje de hosts intermedios que devolvieron una respuesta negativa para paquetes de `no-response` o no respuesta, es decir, de la cuál no obtuvimos una respuesta del router, sea porque se haya perdido algun paquete o que el router haya decidido de no responder, también inicializamos otro contador que cuente esos casos y lo dividimos por la canntidad total de hops que terminó haciendo el paquete:

$$\%\ de\ no\ respuesta = \frac{\#casos\ no\ exitosos}{\#hops\ totales\ del\ paquete}$$

Adicionalmente, si `destination-reached` vale 0 el paquete no terminó llegando al host destino, en cambio si vale 1 es que sí llegó al destino.

Por lo cual, nos pareció interesante arrancar el experimento corriendo la implementación de `traceroute` desde la misma computadora en dos lugares distintos para ver los resultados que podría arrojar, y en qué se podrían diferenciar. Es por eso, que decidimos correrlo desde la universidad con el WiFi de ALUMNOSUTDT, que nos proporcionó estos resultados para cada una de las seis universidades elegidas:

Host destino	# de hops que hizo	% de ttl-zero-during-transit	% de no recibir respuesta	Llega al destino
GAIA	43	90,70	6,98	1
UTDT	45	0,00	100,00	0
AUST	20	70,00	25,00	1
SORBONNE	13	69,23	23,08	1
CUHK	17	64,71	29,41	1
USP	19	68,42	26,32	1

Figura 1: Resultados de traceroute corrido desde la universidad

Pudimos observar que corriendo desde la universidad, la tasa de excito era bastante alta, pues de las 6 universidades, los paquetes de 5 de ellas pudieron llegar al host destino, que es un 83,3% de éxito. El único paquete que no llegó a su destino fue el de www.utdt.edu, ya que en todos los 45 hops obtuvimos de respuesta '*', es decir, que tuvo un 100% de no respuesta. Eso podría significar que los routers sobre el path no responden al paquete ICMP o también supusimos que la universidad tiene configurada la firewall para no mostrar esa información en particular por medidas de seguridad, ya que es una página web muy frecuentado por toda la comunidad de UTDT.

Asimismo, observamos que gaia.cs.umass.edu por mas que obtuvo la mayor cantidad de hops que necesito para que llegue al destino, es el host que tuvo el mayor porcentaje de ttl-zero-during-transit con un 90%, mientras que el resto llegaron también a su destino pero cerca de un 70%.

Por el otro lado, lo corrimos también desde la casa de Micaela para ver que resultados nos podría arrojar:

Host destino	# de hops que hizo	% de ttl-zero-during-transit	% de no recibir respuesta	Llega al destino
GAIA	45	6,67	93,33	0
UTDT	45	6,67	93,33	0
AUST	45	4,44	95,56	0
SORBONNE	45	6,67	93,33	0
CUHK	45	6,67	93,33	0
USP	45	6,67	93,33	0

Figura 2: Resultados de traceroute corrido desde la casa de Micaela

En la *Figura 2* podemos ver que los resultados son bastante distintos a los de la *Figura 1*. Una de las razones, es que ninguno de los paquetes llegó a su host destino, tal que hubo 0% de éxito. Aunque, mirando un poco más el path que hace cada uno de los paquetes, vimos que los paquetes en un cierto punto muestran los routers que atravesaron, pero después solamente aparecen los "*". Lo que nos lleva a especular, que los host intermedios y de destino no se quieren comunicar con el host de Micaela por correrlo desde un WiFi personal, en comparación con uno institucional, ya que por ahí no lo ven como un host de origen confiable.

Asimismo, decidimos comparar ambos casos con el tracert de Windows para ver que resultados arroja. Para esto mostraremos que sucedió con las páginas de GAIA:

traceroute.py	tracert
Tracing route to gaia.cs.umass.edu	Tracing route to gaia.cs.umass.edu [128.119.245.12] over a maximum of 30 hops:
1: 10.9.255.254 (Time: 12.67 ms)	1 3 ms 1 ms 1 ms 10.9.255.254
2: 200.89.140.129 (Time: 9.00 ms)	2 4 ms 2 ms 2 ms 129-140-89-200.fibertel.com.ar [200.89.140.129]
3: 10.104.63.169 (Time: 6.53 ms)	3 1 ms 6 ms 1 ms 10.104.63.169
4: *	4 * * * Request timed out.
5: 10.100.1.213 (Time: 13.06 ms)	5 3 ms 2 ms 6 ms 10.100.1.213
6: 200.89.164.230 (Time: 15.00 ms)	6 4 ms 1 ms 2 ms 230-164-89-200.fibertel.com.ar [200.89.164.230]
7: 10.110.144.2 (Time: 10.01 ms)	7 2 ms 2 ms 9 ms 10.110.144.2
8: 10.110.144.1 (Time: 6.99 ms)	8 2 ms 2 ms 2 ms 10.110.144.1
9: 200.89.164.229 (Time: 13.00 ms)	9 3 ms 3 ms 5 ms 229-164-89-200.fibertel.com.ar [200.89.164.229]
10: *	10 * * * Request timed out.
11: *	11 361 ms 3 ms * host234.181-96-113.telecom.net.ar [181.96.113.234]
12: 195.22.220.56 (Time: 11.94 ms)	12 3 ms 3 ms 3 ms ae25.baires1.bai.seabone.net [195.22.220.56]
13: 89.221.41.187 (Time: 147.94 ms)	13 130 ms 131 ms 130 ms et2-1-0.miami19.mia.seabone.net [89.221.41.187]
14: 96.87.9.145 (Time: 180.60 ms)	14 129 ms 132 ms 130 ms be-207-pe01.nota.fl.ibone.comcast.net [96.87.9.145]
15: 96.110.36.81 (Time: 136.91 ms)	15 129 ms 128 ms 129 ms be-3101-cs01.miami.fl.ibone.comcast.net [96.110.36.81]
16: 96.110.44.166 (Time: 136.28 ms)	16 130 ms 130 ms 130 ms be-1112-cr12.miami.fl.ibone.comcast.net [96.110.44.166]
17: 96.110.39.249 (Time: 149.10 ms)	17 135 ms 137 ms 135 ms be-303-cr11.jacksonville.fl.ibone.comcast.net [96.110.39.249]
18: 96.110.47.97 (Time: 159.00 ms)	18 140 ms 140 ms 140 ms be-1311-cs03.jacksonville.fl.ibone.comcast.net [96.110.47.97]
19: 96.110.47.110 (Time: 156.83 ms)	19 137 ms 135 ms 136 ms be-1314-cr14.jacksonville.fl.ibone.comcast.net [96.110.47.110]
20: 96.110.39.193 (Time: 163.37 ms)	20 149 ms 149 ms 148 ms be-303-cr11.56marietta.ga.ibone.comcast.net [96.110.39.193]
21: 96.110.32.13 (Time: 155.07 ms)	21 147 ms 147 ms 151 ms be-1311-cs03.56marietta.ga.ibone.comcast.net [96.110.32.13]
22: 96.110.34.234 (Time: 158.37 ms)	22 148 ms 148 ms 150 ms be-1313-cr13.56marietta.ga.ibone.comcast.net [96.110.34.234]
23: 96.110.39.50 (Time: 157.89 ms)	23 150 ms 150 ms 149 ms be-302-cr11.doraville.ga.ibone.comcast.net [96.110.39.50]
24: 96.110.34.173 (Time: 168.43 ms)	24 149 ms 150 ms 154 ms be-1411-cs04.doraville.ga.ibone.comcast.net [96.110.34.173]
25: 96.110.34.206 (Time: 158.14 ms)	25 150 ms 150 ms 150 ms be-1413-cr13.doraville.ga.ibone.comcast.net [96.110.34.206]
26: 96.110.32.2 (Time: 170.30 ms)	26 149 ms 149 ms 149 ms be-301-cr11.beaumeade.va.ibone.comcast.net [96.110.32.2]
27: 68.86.84.29 (Time: 166.25 ms)	27 148 ms 147 ms 147 ms be-1411-cs04.beaumeade.va.ibone.comcast.net [68.86.84.29]
28: 68.86.85.78 (Time: 165.97 ms)	28 149 ms 149 ms 149 ms be-1413-cr13.beaumeade.va.ibone.comcast.net [68.86.85.78]
29: 96.110.36.118 (Time: 169.05 ms)	29 154 ms 152 ms 150 ms be-302-cr12.newark.nj.ibone.comcast.net [96.110.36.118]
30: 96.110.35.81 (Time: 162.77 ms)	30 150 ms 150 ms 150 ms be-1112-cs01.newark.nj.ibone.comcast.net [96.110.35.81]
31: 96.110.42.194 (Time: 164.57 ms)	
32: 162.151.52.226 (Time: 171.01 ms)	
33: 96.108.47.146 (Time: 165.00 ms)	
34: 50.222.38.42 (Time: 160.28 ms)	
35: 192.80.83.109 (Time: 165.59 ms)	
36: 128.119.0.216 (Time: 166.19 ms)	
37: 128.119.7.74 (Time: 161.36 ms)	
38: 128.119.7.66 (Time: 161.61 ms)	
39: 128.119.0.217 (Time: 160.00 ms)	
40: 128.119.3.33 (Time: 168.29 ms)	
41: 128.119.3.32 (Time: 182.00 ms)	
42: 128.119.240.253 (Time: 161.00 ms)	
43: 128.119.245.12 (Destination Reached) (Time: 164.46 ms)	

Figura 3: Resultados de traceroute y de tracert de GAIA corrido desde la universidad

El tracert de Windows utiliza un TTL máximo de tamaño 30, por lo cual al llegar a 30 termina el trace. Sin embargo, los caminos que hace son idénticos, con la única excepción que el tracert además muestra los nombres de los routers, como por ejemplo el de ae25.baires1.bai.seabone.net, que puede pertenecer al sistema de cableado que pasa bajo el oceano referenciando al que se encuentra en Buenos Aires, ya que Seabonne es una de las empresas que se encarga de eso. Lo explicaremos en el siguiente punto en más detalle qué podrían significar los saltos en el tiempo que puede tardar en devolver una respuesta.

traceroute.py	tracert
Tracing route to gaia.cs.umass.edu	Tracing route to gaia.cs.umass.edu [128.119.245.12] over a maximum of 30 hops:
1: 192.168.0.1 (Time: 55.52 ms)	1 9 ms 1 ms 20 ms 192.168.0.1
2: *	2 * * * Request timed out.
3: *	3 * * * Request timed out.
4: *	4 * * * Request timed out.
5: *	5 * * * Request timed out.
6: *	6 * * * Request timed out.
7: 195.22.220.56 (Time: 23.36 ms)	7 13 ms 59 ms 58 ms ae25.baires1.bai.seabone.net [195.22.220.56]
8: 89.221.41.161 (Time: 169.33 ms)	8 148 ms 140 ms 140 ms et9-1-5.miami19.mia.seabone.net [89.221.41.161]
9: *	10 218 ms 142 ms 141 ms be-3401-cs04.miami.fl.ibone.comcast.net [96.110.36.93]
10: *	11 134 ms 136 ms 133 ms be-1412-cr12.miami.fl.ibone.comcast.net [96.110.44.202]
11: *	12 142 ms 143 ms 146 ms be-302-cr11.jacksonville.fl.ibone.comcast.net [96.110.39.245]
12: *	13 146 ms 149 ms 147 ms be-1311-cs03.jacksonville.fl.ibone.comcast.net [96.110.47.97]
13: *	15 157 ms 185 ms 161 ms be-304-cr12.doraville.ga.ibone.comcast.net [96.110.36.253]
14: *	16 149 ms 149 ms 150 ms be-1112-cs01.doraville.ga.ibone.comcast.net [96.110.34.177]
15: *	17 156 ms 169 ms 155 ms be-1113-cr13.doraville.ga.ibone.comcast.net [96.110.34.194]
16: *	18 173 ms 171 ms 281 ms be-303-cr11.beaumeade.va.ibone.comcast.net [96.110.32.73]
17: *	19 180 ms 172 ms 174 ms be-1211-cs02.beaumeade.va.ibone.comcast.net [68.86.84.21]
18: *	20 173 ms 171 ms 177 ms be-1213-cr13.beaumeade.va.ibone.comcast.net [68.86.85.70]
19: *	21 163 ms 156 ms 156 ms be-302-cr12.newark.nj.ibone.comcast.net [96.110.36.118]
20: *	22 156 ms 158 ms 156 ms be-1412-cs04.newark.nj.ibone.comcast.net [96.110.35.93]
21: *	23 162 ms 162 ms 286 ms be-32041-ar01.woburn.ma.boston.comcast.net [96.110.42.206]
22: *	24 164 ms 168 ms 322 ms 162.151.52.226
23: *	25 165 ms 165 ms 168 ms be-12-sur01.amherst.ma.boston.comcast.net [96.108.47.146]
24: *	26 172 ms 169 ms 168 ms 50.222.38.42
25: *	27 168 ms 172 ms 170 ms core1-rt-et-8-3-0.gw.umass.edu [192.80.83.109]
26: *	28 164 ms 170 ms 165 ms n1-rt-1-1-et-0-0-0.gw.umass.edu [128.119.0.216]
27: *	29 165 ms 171 ms 165 ms 128.119.7.74
28: *	30 166 ms 178 ms 164 ms 128.119.7.66
29: *	
30: *	
31: *	
32: *	
33: *	
34: *	
35: *	
36: *	
37: *	
38: *	
39: *	
40: *	
41: *	
42: *	
43: *	
44: *	
45: *	

Figura 4: Resultados de traceroute y de tracert de GAIA corrido desde la casa de Micaela

Podemos observar que los hops 1, 7 y 8 otra vez son idénticos en este caso (vale aclarar que no siempre pueden llegar a ser idénticos como en estos casos, ya que en cualquier momento uno de los routers puede llegar a decidir a tomar otro camino, por ejemplo, debido a la congestión), por lo cual nosotros asumimos que el hecho de que nuestra implementación de `traceroute` haya tenido tantos "*" después del hop 8, es que los host intermedios y de destino no nos quieren proporcionar la información por seguridad y es por eso que no obtenemos una respuesta de si llegó el paquete o no al host destino de la universidad.

Este patrón lo pudimos observar en todos lo paquetes que mandamos desde la computadora y casa de Micaela, lo que no sucedía en la universidad, donde sí llegaban los paquetes mandados a los host destino, donde UTDT era una excepción.

b Para el traceroute de la universidad de EEUU, el salto mas significativo, es decir el que tardo mas comparado con el anterior, tiene una IP de un servidor Italiano, el anterior tambien, esto puede deberse a que los paquetes de datos pueden seguir rutas inesperadas debido a la forma en que se establecen las conexiones entre proveedores de servicios de Internet (ISP) y redes globales. A veces, el enrutamiento puede llevar paquetes a través de ubicaciones geográficas inesperadas.

Para el traceroute de la Universidad China de Hong Kong, el salto mas significativo, es decir el que tardo mas comparado con el anterior, tiene una IP de un servidor privado, Las redes privadas a menudo implementan políticas de tráfico y filtrado de paquetes. Esto puede causar retrasos en la transmisión de datos y, en consecuencia, un aumento en el RTT.

Para el traceroute de la Africana de Ciencia y Tecnologia ,el salto mas significativo, es decir el que tardo mas comparado con el anterior, tiene una IP de un servidor de Reino Unido, el anterior es de Argentina,los paquetes deben atravesar conexiones internacionales y pasar por infraestructuras de enrutamiento específicas que a veces implican procedimientos de seguridad y verificación adicionales. Estos procesos pueden agregar tiempo al RTT.

Para el traceroute de la Universidad de Sorbonne, el salto mas significativo, es decir el que tardo mas comparado con el anterior, tiene una IP de un servidor privado, Las redes privadas a menudo implementan políticas de tráfico y filtrado de paquetes. Esto puede causar retrasos en la transmisión de datos y, en consecuencia, un aumento en el RTT.

Para el traceroute a la Universidad de San Pablo, el salto mas largo (numero 16) se observa justo despues de tres hops sin respuesta, el aumento en el RTT en el hop número 16 podría deberse a varios factores, como enrutamiento lento, congestión de red o problemas técnicos en ese punto de la ruta. La falta de respuesta en los hops 12, 13 y 14 podría ser el resultado de dispositivos que no responden a las solicitudes de Traceroute.

Para el traceroute de Universidad Torcuato di Tella no se obtuvieron respuestas de ningún host intermedio a lo largo de la ruta hacia el host destino. Esto podría deberse a configuraciones de red que bloquean o no responden a las solicitudes de traceroute, medidas de seguridad para proteger la infraestructura de red o a la inaccesibilidad de la ruta desde la ubicación donde se realizó el traceroute.

3.2 Experimentaciones de los port scan (SYN scan y Connect scan)

a Realizar un análisis estadístico sobre el porcentaje de puertos cerrados, abiertos y filtrados. Para realizar el análisis sugerimos graficar e intentar explicar los resultados obtenidos

Para este experimento de los port scanners, decidimos utilizar las mismas 6 universidades de los 5 continentes.

Para cada una de las universidades elegidas ejecutamos el `port_scanner -h` o SYN scan y `port_scanner -f` o Connect scan con sus respectivas URLs, tal que nos muestre los puertos abiertos de cada universidad para cada scan. Comenzando con el SYN scan, nuestra implementacion devuelve la cantidad total de puertos abiertos, cantidad de puertos filtrados (asumimos que cualquier puerto que no sea abierto es filtrado) y todos los estados de los puertos, de tal modo podríamos calcular el `open-percentage` o el porcentaje de puertos abiertos y el `filtered-percentage` o el porcenetaje de puertos filtrados. Con los estados de los puertos armamos los archivos de texto solicitados.

Por lo cuál, para calcular el porcentaje de `open-percentage`, considerando que hicimos el scan a los 1000 primero puertos:

$$\%\ open - percentage\ = \frac{\#puertos\ abiertos}{\#puertos\ totales} = \frac{\#puertos\ abiertos}{1000}$$

Asimismo, para calcular el porcentaje de `filtered-percentage`, considerando que hicimos el scan a los 1000 primeros puertos:

$$\%\ open - percentage\ = \frac{\#puertos\ filtrados}{\#puertos\ totales} = \frac{\#puertos\ filtrados}{1000}$$

Al haber puesto distintos puertos de origen para ambos port scanners, donde el de SYN scan por default era de 20 y el de Connect scan le pusimos 2002, los corriamos a la vez para ver en qué se podrían llegar a diferenciar o ser similares. Asimismo, nos pareció interesante ver que sucedía si corríamos ambos port scanner desde cada una de nuestras casas y desde la universidad.

Comenzando con el SYN scan, obtuvimos estos resultados para cada una de las seis universidades elegidas:

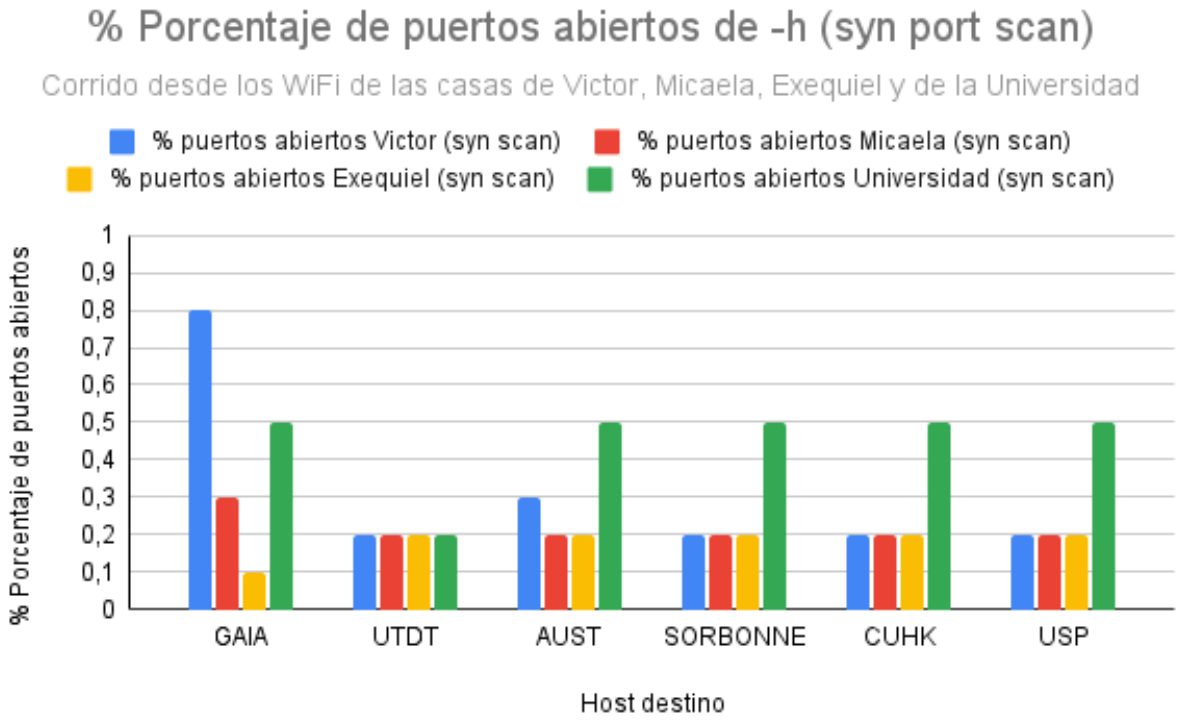

Figura 5: Resultados de SYN port scan de las seis universidades corrido desde la casa de Victor, Micaela, Exequiel y desde la universidad

Lo que observamos fue que con GAIA, Victor obtuvo la mayor cantidad de puertos abiertos, donde obtuvo que habían 8 puertos distintos que estaban abiertos de ellos: $[21, 22, 80, 443, 662, 892, 993, 995]$. Considerando que: - Port 21: tcp/udp - Port 22: tcp/udp - Port 80: HTTP - Port 443: HTTP - Port 662: tcp/udp - Port 892: tcp/udp - Port 993: tcp/udp - Port 995: tcp/udp

- FIGURA 6

Lo que nos provocó bastante curiosidad ver todos esos puertos abiertos, ya que en comparación Micaela obtuvo 3, Exequiel 2 y correrlo desde la universidad 5. Por lo cual, decidimos ver cómo se comparaba con los

resultados de Nmap, ya que es una herramienta ampliamente utilizada para escaneo de redes y para evaluar la seguridad de sistemas informáticos, una de las características claves que tiene es el escaneo de puertos. Por lo cual, para ver los primeros 1000 puertos de un SYN scan usamos el comando `nmap -sS -p 1-1000 gaia.cs.umass.edu`, ya que si no especificamos el rango de los puertos, por default escanea los puertos que Nmap considera los mas importantes. Por lo cuál, arrojó este resultado:

```
Nmap scan report for gaia.cs.umass.edu (128.119.245.12)
Host is up (0.18s latency).
Not shown: 963 filtered tcp ports (no-response), 28 filtered tcp ports
(host-prohibited)
PORT    STATE  SERVICE
21/tcp  open   ftp
22/tcp  open   ssh
80/tcp  open   http
443/tcp open   https
662/tcp open   pftp
875/tcp closed unknown
892/tcp open   unknown
993/tcp open   imaps
995/tcp open   pop3s

Nmap done: 1 IP address (1 host up) scanned in 23.17 seconds
```

Figura 7: Resultados de Nmap SYN port scan de GAIA corrido por Victor

Es decir, que para GAIA Victor obtuvo todos los puertos que al mandar un SYN obtuvo un SYN-ACK, obteniento un 100% de éxito. En cambio, Micaela obtuvo un 37.5%, Exequiel un 12.5% y desde la Universidad un 62.5%. Nosotros suponemos que la diferencia en la cantidad de puertos que cada uno obtuvo, se puede deber a los ISP o las bandas defrecuencia que cada uno tiene, ya que Victor usa IPLAN, Micaela Fibertel de 5.8 GHz, Exequiel Fibertel de 2.4 GHz y la Universidad utiliza servicio de una red diseñada especialmente para establecimientos para soportar la gran cantidad de tráfico que necesita, como lo hacen las empresas, ya que pueden todos tener distintos permisos a la hora de hacer este tipo de pédidos. También podría llegar a ser por dónde nos encontramos en Buenos Aires, ya que Victor es el único que se encuentra en Recoleta, mientras que el resto se ubican en Belgrano. También podría ser por los permisos que tenemos en nuestras computadoras, ya que Victor lo corrió desde una Mac y el resto desde Windows, ya que daba el mismo resultado que en Ubuntu.

Además, nos llamó la atención de que al correrlo desde la universidad todos menos UTDT nos devolvieron que los 5 puertos abiertos que tenían eran: $[25, 80, 110, 143, 443]$.

- No alcanzaban 1%
-

Además, para el caso de Connect scan, queríamos ver qué sucedía si el payload era un mensaje de `Hello, {target_ip}!` o si es un GET de HTTP `GET / HTTP/1.1\r\n Host: {host_destino}\r\n\r\n`, si cambiaban los puertos que se encontraban abiertos.

b [Busca patrones en los estados de los puertos en diferentes servidores y analiza posibles explicaciones para estos patrones.]

c [Compara los resultados del escaneo de servidores con los dos diferentes scanners desarrollados y resalta las diferencias.]

d [Realiza una comparación entre los resultados obtenidos con el Port Scanner desarrollado y la herramienta nmap para el escaneo de servidores.]

4. Conclusiones

[En esta sección, proporciona una conclusión general del trabajo, resumiendo los hallazgos más importantes

y su relevancia en el análisis de redes. También puedes mencionar posibles áreas de mejora o futuras investigaciones en el tema.]

$$

$$