

VERIFICAR LA CAPACIDAD DEL CANAL EN UNA RED (ÓPTICA) DE COMUNICACIONES

Escuela Politécnica Nacional
Facultad de Ingeniería Eléctrica y Electrónica
Xavier Chasi, Cristian Gallo, Alejandra Silva, Francisco Valdez
Quito, Ecuador

ricardo.chasi@epn.edu.ec, cristian.gallo.@epn.edu.ec, alejandra.silva@epn.edu francisco.valdez@epn.edu.ec

Resumen – En el siguiente documento se presentará el uso y el manejo de distintos softwares para conocer acerca de la red que se está usando y además de conocer cuál es la capacidad máxima de dicha red.

Palabras clave – UDP, TCP, SIPp.

I. INTRODUCCION

El monitoreo de trafico de paquetes nos permite obtener diferentes características de la red que se este monitoreando. Para este fin se utilizando softwares especializados que mediante sensores nos permiten captar varios tipos de paquetes esto con el fin de observar el comportamiento de la red para distintos paquetes, además de observar las características de los diferentes paquetes.

Para un análisis mas especifico, se hace uso de software de generación de trafico controlado como es el caso de jperf/iperf que nos permite generar trafico UDP, TCP de manera controlada en intervalos de tiempo y que posean diferentes características esto con el fin de medir la capacidad máxima de nuestra red seteando valores cada vez mayores. Además, se puede hacer uso de este software como analizador de tráfico, pero su potencia es bastante limitada permitiendo censar tráfico únicamente TCP y UDP, por lo cual para el análisis de trafico mas exhaustivo se hace uso de softwares como Cacti, Nagios o PRTG que poseen una variada librería con sensores para diferente tipo de tráfico.

Así también, se utilizará flujos de tráfico no tan común como es el caso de trafico SNMP y trafico SIPp de lo cual se tratará a medida que se avance en este informe.

II. OBJETIVOS

- Conocer el funcionamiento de los softwares de generación y censado de tráfico a través de la red.
- Conocer con qué tipo de paquetes se puede conocer de mejor manera la capacidad de una red.

III. INFORME

1. Generar tráfico UDP y TCP entre dos hosts utilizando jperf y iperf.

- Trafico UDP

Para generar tráfico UDP se hizo uso de un modelo cliente/servidor, en donde el servidor puede ser Windows, IOS

o Linux, para este caso se utilizó el sistema operativo Windows como servidor y para el caso del cliente se utilizó una máquina virtual, siendo especifico Kali Linux. Al ejecutar se debe de colocar los parámetros necesarios en el programa jperf indicando que es cada uno. Se uso paquetes UDP debido a que no son orientados a la conexión por lo cual no hace un acuse de recibo como lo haría TCP, por tal motivo este protocolo es el más adecuado para constatar la capacidad máxima de la red. A continuación, se muestran las interfaces configuradas para el cliente y el servidor.

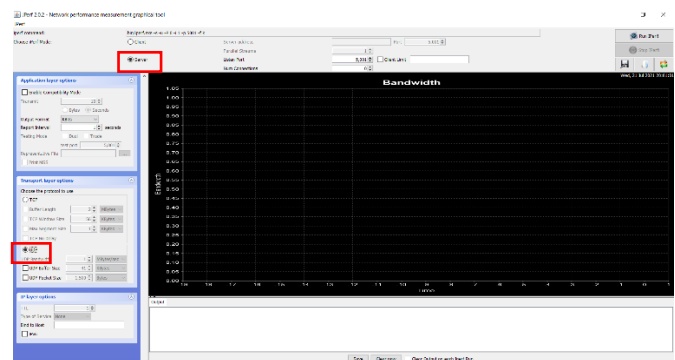


Figura 1. Jperf configurado en el servidor de Windows para captura de paquetes UDP.

En el equipo que se utilizara como servidor no se necesita hacer ninguna configuración adicional más que seleccionar servidor y el tipo de paquetes que va a recibir, siendo el primer caso UDP. Es recomendable utilizar la configuración que viene por defecto con excepción del tipo de paquetes según se necesite.

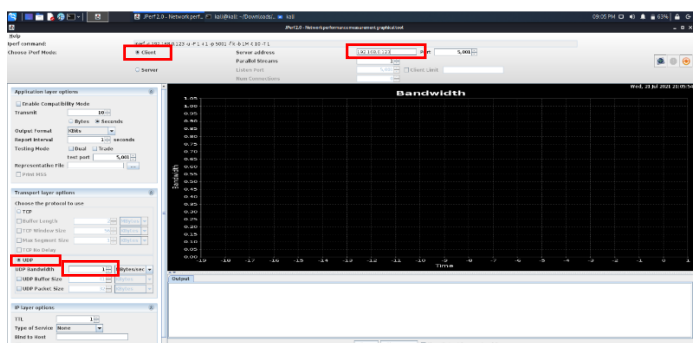


Figura 2. Jperf configurado en el cliente de Kali Linux para captura de paquetes UDP.

Para la maquina cliente se utilizó una máquina virtual con distribución Debian de Linux [1], para configurar el modo cliente se debe conocer la IP del servidor que en este caso fue nuestra maquina física con Windows, una vez configurado estos

parámetros como se ven en figura 2 se elige el ancho de banda que se quiere enviar y analizar.

Como primer paso se comenzó a configurar la parte del cliente a enviar un ancho de banda específico, este fue de 1Mbyte/s, para capturarlo en el servidor y poder observar que sucede al aumentar dicho ancho de banda y así al final, saber cuál es valor máximo que se tendrán registros en las gráficas.

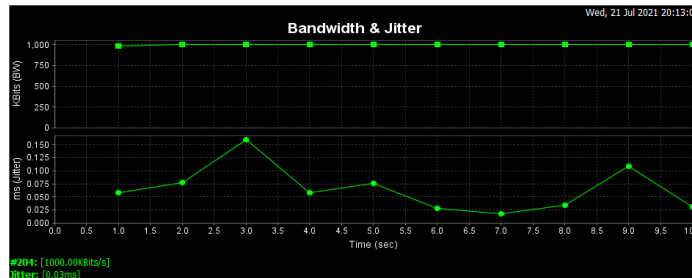


Figura 3. Ancho de banda obtenido en el servidor.

[ID]	Interval	Transfer	Bandwidth	Jitter	Lost/Total	Datagrams
[204]	0.0- 1.0 sec	121 KBytes	988 Kbits/sec	0.058 ms	0/ 84	(0%)
[204]	1.0- 2.0 sec	122 KBytes	1000 Kbits/sec	0.078 ms	0/ 85	(0%)
[204]	2.0- 3.0 sec	122 KBytes	1000 Kbits/sec	0.159 ms	0/ 85	(0%)
[204]	3.0- 4.0 sec	122 KBytes	1000 Kbits/sec	0.058 ms	0/ 85	(0%)
[204]	4.0- 5.0 sec	122 KBytes	1000 Kbits/sec	0.076 ms	0/ 85	(0%)
[204]	5.0- 6.0 sec	122 KBytes	1000 Kbits/sec	0.028 ms	0/ 85	(0%)
[204]	6.0- 7.0 sec	122 KBytes	1000 Kbits/sec	0.018 ms	0/ 85	(0%)
[204]	7.0- 8.0 sec	122 KBytes	1000 Kbits/sec	0.034 ms	0/ 85	(0%)
[204]	8.0- 9.0 sec	122 KBytes	1000 Kbits/sec	0.108 ms	0/ 85	(0%)
[204]	9.0-10.0 sec	122 KBytes	1000 Kbits/sec	0.031 ms	0/ 85	(0%)
[204]	0.0-10.0 sec	1223 KBytes	1000 Kbits/sec	0.027 ms	0/ 852	(0%)

Figura 4. Datos recibidos en el servidor.

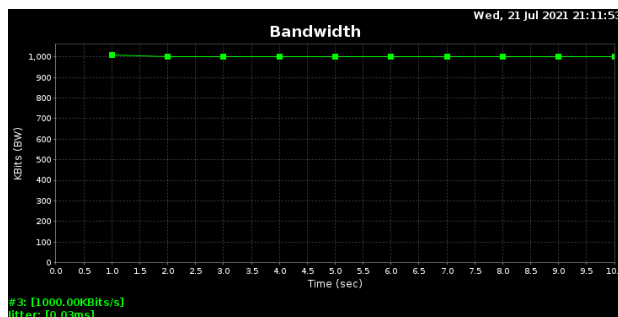


Figura 5. Ancho de banda obtenido en el cliente.

```
[ 3] local 192.168.0.129 port 59368 connected with 192.168.0.123 port 5001
[ 3] 0.0-1.0 sec 123 KBytes 1011 Kbits/sec
[ 3] 1.0-2.0 sec 122 KBytes 1000 Kbits/sec
[ 3] 2.0-3.0 sec 122 KBytes 1000 Kbits/sec
[ 3] 3.0-4.0 sec 122 KBytes 1000 Kbits/sec
[ 3] 4.0-5.0 sec 122 KBytes 1000 Kbits/sec
[ 3] 5.0-6.0 sec 122 KBytes 1000 Kbits/sec
[ 3] 6.0-7.0 sec 122 KBytes 1000 Kbits/sec
[ 3] 7.0-8.0 sec 122 KBytes 1000 Kbits/sec
[ 3] 8.0-9.0 sec 122 KBytes 1000 Kbits/sec
[ 3] 9.0-10.0 sec 122 KBytes 1000 Kbits/sec
[ 3] 0.0-10.0 sec 1223 KBytes 1000 Kbits/sec
[ 3] Sent 852 datagrams
[ 3] Server Report:
[ 3] 0.0-10.0 sec 1223 KBytes 1000 Kbits/sec 0.026 ms 0/ 852 (0%)
Done.
```

Figura 6. Datos enviados de paquetes UDP desde el cliente al servidor.

Luego de varias pruebas enviando diferentes anchos de banda desde el cliente hacia el servidor se pudo observar que al enviar un valor de 117 Kbytes/s, es su valor máximo de capacidad o de resolución en el cual se puede obtener una

gráfica en el lado del servidor, pasado el valor de 117 se comprueba que se envían paquetes, pero ya no se puede observar ninguna gráfica.

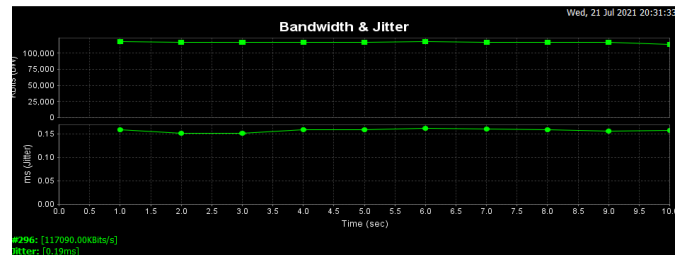


Figura 7. Ancho de banda y Jitter del lado del servidor a un AB = 117Kbytes/s.

[ID]	Interval	Transfer	Bandwidth	Jitter	Lost/Total	Datagrams
[296]	0.0- 1.0 sec	14348 KBytes	117541 Kbits/sec	0.158 ms	0/ 9995	(0%)
[296]	1.0- 2.0 sec	14334 KBytes	117424 Kbits/sec	0.151 ms	0/ 9985	(0%)
[296]	2.0- 3.0 sec	14301 KBytes	117153 Kbits/sec	0.151 ms	0/ 9962	(0%)
[296]	3.0- 4.0 sec	14345 KBytes	117518 Kbits/sec	0.159 ms	0/ 9993	(0%)
[296]	4.0- 5.0 sec	14344 KBytes	117506 Kbits/sec	0.158 ms	0/ 9992	(0%)
[296]	5.0- 6.0 sec	14353 KBytes	117576 Kbits/sec	0.161 ms	0/ 9998	(0%)
[296]	6.0- 7.0 sec	14337 KBytes	117447 Kbits/sec	0.160 ms	0/ 9987	(0%)
[296]	7.0- 8.0 sec	14343 KBytes	117494 Kbits/sec	0.158 ms	0/ 9991	(0%)
[296]	8.0- 9.0 sec	14317 KBytes	117282 Kbits/sec	0.156 ms	0/ 9973	(0%)
[296]	9.0-10.0 sec	13912 KBytes	113966 Kbits/sec	0.157 ms	0/ 9691	(0%)
[296]	0.0-10.0 sec	143003 KBytes	117090 Kbits/sec	0.192 ms	-1/99576	(-0.001%)
[296]	0.0-10.0 sec	1 datagrams	received out-of-order			

Figura 8. Detalles obtenidos de los datagramas enviados del cliente a un AB = 117 Kbytes/s.

Como se pudo observar en la primera prueba a 1Kbyte/s, en el lado del cliente solamente se grafican los 10 paquetes al ancho de banda configurado.

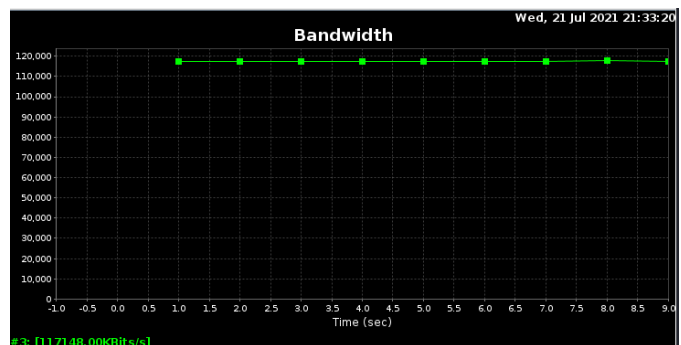


Figura 9. Ancho de banda en la parte del cliente a 117Kbytes/s.

```
[ 3] local 192.168.0.129 port 40970 connected with 192.168.0.123 port 5001
[ 3] 0.0-1.0 sec 14344 KBytes 117506 Kbits/sec
[ 3] 1.0-2.0 sec 14343 KBytes 117494 Kbits/sec
[ 3] 2.0-3.0 sec 14315 KBytes 117271 Kbits/sec
[ 3] 3.0-4.0 sec 14355 KBytes 117600 Kbits/sec
[ 3] 4.0-5.0 sec 14347 KBytes 117529 Kbits/sec
[ 3] 5.0-6.0 sec 14355 KBytes 117600 Kbits/sec
[ 3] 6.0-7.0 sec 14330 KBytes 117388 Kbits/sec
[ 3] 7.0-8.0 sec 14360 KBytes 117635 Kbits/sec
[ 3] 8.0-9.0 sec 14328 KBytes 117377 Kbits/sec
[ 3] 0.0-10.0 sec 143003 KBytes 117148 Kbits/sec
[ 3] Sent 99616 datagrams
[ 3] Server Report:
[ 3] 0.0-10.0 sec 143003 KBytes 117090 Kbits/sec 0.191 ms 0/99576 (0%)
[ 3] 0.0-10.0 sec 1 datagrams received out-of-order
Done.
```

Figura 10. Datagramas enviados desde la parte del cliente a 117 Kbytes/s.

Un parámetro que se debe de tomar en cuenta al enviar un distinto ancho de banda de la parte del cliente con paquetes UDP es que el programa su máxima capacidad es de

aproximadamente de 379Mbytes/s, debido a que se realizó pruebas de 500Mbytes/s y superior pero el valor máximo que llegaba a transmitir es de 379Mbytes/s como ya se mencionó anteriormente.

- Tráfico TCP

Para la generación de tráfico TCP igualmente se utilizó como maquina cliente, la máquina virtual Kali Linux y como servidor se utilizó la maquina física Windows. La configuración tanto de cliente como de servidor es la misma para el tráfico UDP, pero en el apartado de las opciones de la capa de transporte se debe seleccionar tráfico TCP como se muestra en la siguiente figura.

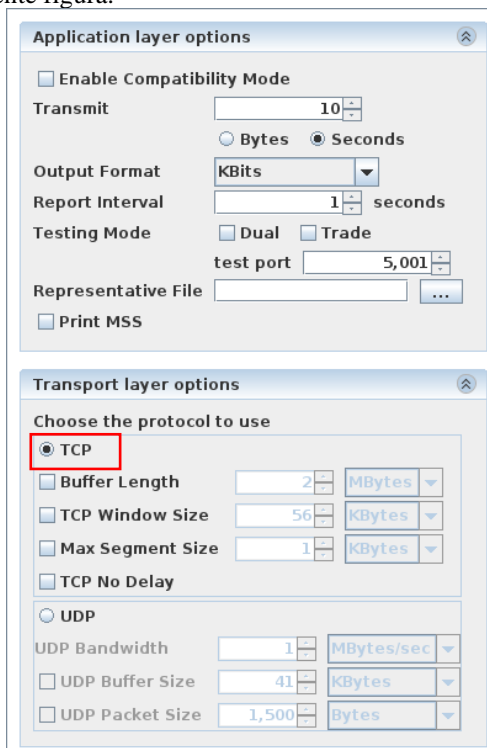


Figura 11. Selección del tipo de tráfico.

Dentro la configuración del tráfico TCP podemos configurar cuatro parámetros:

- Longitud del Buffer
- Tamaño de la ventana TCP
- Tamaño máximo del segmento
- TCP sin retardo

Primeramente, se hizo pruebas con los parámetros por defecto para observar el comportamiento de este tipo de tráfico y los resultados obtenido de acuerdo con esto.

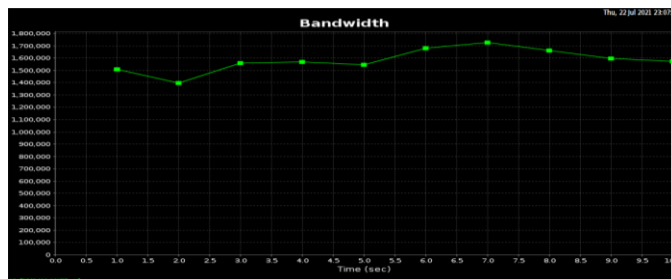


Figura 12. Generación de tráfico TCP.

```
Client connecting to 192.168.100.9, TCP port 5001
TCP window size: 561 KByte (default)

[ 3] local 192.168.100.106 port 52082 connected with 192.168.100.9 port 5001
[ 3] 0.0- 1.0 sec 184056 KBytes 1507787 Kbits/sec
[ 3] 1.0- 2.0 sec 170824 KBytes 1399390 Kbits/sec
[ 3] 2.0- 3.0 sec 190144 KBytes 1557660 Kbits/sec
[ 3] 3.0- 4.0 sec 191408 KBytes 1568014 Kbits/sec
[ 3] 4.0- 5.0 sec 188880 KBytes 1547305 Kbits/sec
[ 3] 5.0- 6.0 sec 204880 KBytes 1678377 Kbits/sec
[ 3] 6.0- 7.0 sec 210800 KBytes 1726874 Kbits/sec
[ 3] 7.0- 8.0 sec 202688 KBytes 1660420 Kbits/sec
[ 3] 8.0- 9.0 sec 194672 KBytes 1594753 Kbits/sec
[ 3] 9.0-10.0 sec 192104 KBytes 1573716 Kbits/sec
[ 3] 0.0-10.0 sec 1930464 KBytes 1581409 Kbits/sec
Done.
```

Figura 13. Información de los paquetes generados.

Como se puede observar tanto en la figura 12 como en la figura 13, el tráfico generado tiene una ventana TCP de 561 Kbyte por defecto y el enlace tiene un ancho de banda de 1.5 Gbps aproximadamente.

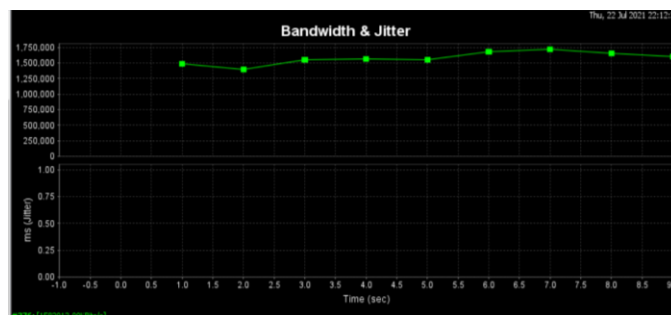


Figura 14. Captura de tráfico TCP.

```
Server listening on TCP port 5001
TCP window size: 64.0 KByte (default)

OpenSCManager failed - Access is denied. (0x5)
[376] local 192.168.100.9 port 5001 connected with 192.168.100.106 port 52082
[ ID] Interval      Transfer      Bandwidth
[376] 0.0- 1.0 sec 181191 KBytes 1484318 Kbits/sec
[376] 1.0- 2.0 sec 171022 KBytes 1401016 Kbits/sec
[376] 2.0- 3.0 sec 190256 KBytes 1558579 Kbits/sec
[376] 3.0- 4.0 sec 190568 KBytes 1561137 Kbits/sec
[376] 4.0- 5.0 sec 189085 KBytes 1548984 Kbits/sec
[376] 5.0- 6.0 sec 205814 KBytes 1686029 Kbits/sec
[376] 6.0- 7.0 sec 210444 KBytes 1723957 Kbits/sec
[376] 7.0- 8.0 sec 202874 KBytes 1661947 Kbits/sec
[376] 8.0- 9.0 sec 195231 KBytes 1599331 Kbits/sec
[376] 9.0-10.0 sec 1930464 KBytes 1582012 Kbits/sec
```

Figura 15. Información del tráfico capturado.

Como se puede observar en las figuras 14 y 15 tenemos el tráfico capturado por el servidor en la maquina física Windows, de donde podemos comparar los paquetes recibidos con los enviados y el ancho de banda del enlace. Teniendo

aproximadamente los mismos resultados que los vistos en la generación de tráfico.

Variando los parámetros del cliente y servidor podemos modificar el tamaño de los paquetes a tal punto de saturar en enlace. Como tamaño máximo de tamaño de paquete tenemos 10240, es decir 10 Kbyte, por tal motivo generamos una mayor cantidad de tráfico para observar el efecto que tiene.

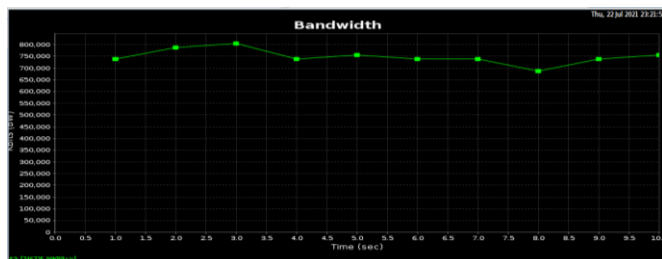


Figura 16. Generación de tráfico TCP, paquetes de mayor tamaño.

```
Client connecting to 192.168.100.9, TCP port 5001
TCP window size: 200 KByte (WARNING: requested 100 KByte)

[ 3] local 192.168.100.106 port 52106 connected with 192.168.100.9 port 5001
[ 3] 0.0- 1.0 sec 90112 KBytes 738198 Kbits/sec
[ 3] 1.0- 2.0 sec 96256 KBytes 788529 Kbits/sec
[ 3] 2.0- 3.0 sec 98304 KBytes 805306 Kbits/sec
[ 3] 3.0- 4.0 sec 90112 KBytes 738198 Kbits/sec
[ 3] 4.0- 5.0 sec 92160 KBytes 754975 Kbits/sec
[ 3] 5.0- 6.0 sec 90112 KBytes 738198 Kbits/sec
[ 3] 6.0- 7.0 sec 90112 KBytes 738198 Kbits/sec
[ 3] 7.0- 8.0 sec 83968 KBytes 687866 Kbits/sec
[ 3] 8.0- 9.0 sec 90112 KBytes 738198 Kbits/sec
[ 3] 9.0-10.0 sec 92160 KBytes 754975 Kbits/sec
[ 3] 0.0-10.0 sec 915456 KBytes 746725 Kbits/sec
WARNING: attempt to set TCP maximum segment size to 10240, but got 536
Done.
```

Figura 17. Información del tráfico generado.

Como se puede observar en las figuras 16 y 17, el tráfico generado para colapsar la red a hecho que el ancho de banda original disminuya y generando una alerta de generación de paquetes dentro del rango máximo.

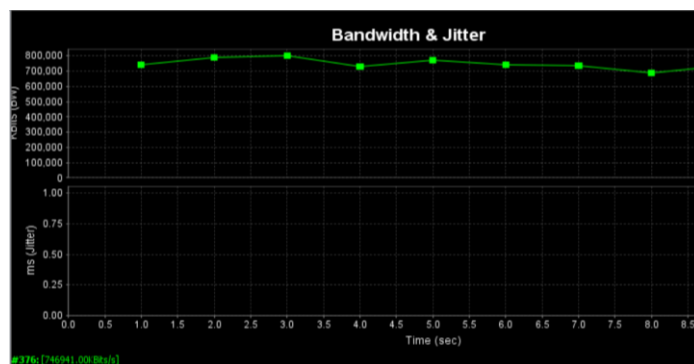


Figura 18. Captura de tráfico TCP.

```
Server listening on TCP port 5001
TCP window size: 20.0 KByte

-----
OpenSCManager failed - Access is denied. (0x5)
[376] local 192.168.100.9 port 5001 connected with 192.168.100.106 port 52106
[ ID] Interval Transfer Bandwidth
[376] 0.0- 1.0 sec 90443 KBytes 740910 Kbits/sec
[376] 1.0- 2.0 sec 96663 KBytes 791860 Kbits/sec
[376] 2.0- 3.0 sec 97888 KBytes 801902 Kbits/sec
[376] 3.0- 4.0 sec 88863 KBytes 727968 Kbits/sec
[376] 4.0- 5.0 sec 93942 KBytes 769574 Kbits/sec
[376] 5.0- 6.0 sec 90194 KBytes 738865 Kbits/sec
[376] 6.0- 7.0 sec 89783 KBytes 735502 Kbits/sec
[376] 7.0- 8.0 sec 84158 KBytes 689424 Kbits/sec
[376] 8.0- 9.0 sec 89814 KBytes 735760 Kbits/sec
[376] 9.0-10.0 sec 91309 KBytes 748000 Kbits/sec
[376] 0.0-10.0 sec 915456 KBytes 746941 Kbits/sec
```

Figura 19. Información de los paquetes capturados.

De la misma manera como se observó en el tráfico generado, en la parte del servidor podemos observar como el ancho de banda igualmente bajo con respecto al ejemplo por defecto y esto es debido al asuramiento del enlace que se generó en el lado del cliente con el fin de observar que efectos tenía y se pudo observar que uno de ellos fue la disminución del ancho de banda y de la cantidad de paquetes transferidos que de igual manera bajo con respecto al ejemplo por defecto.

- Tráfico UDP con IPv6

Para generar tráfico UDP se hizo uso de un modelo cliente/servidor, en donde el servidor puede ser Windows, IOS o Linux, para este caso se utilizó el sistema operativo Windows como servidor y para el caso del cliente Windows de uno de los integrantes del grupo que posea IPv6. Al ejecutar se debe de colocar los parámetros necesarios en el programa jperf indicando que es cada uno. Se uso paquetes UDP debido a que no son orientados a la conexión por lo cual no hace un acuse de recibo como lo haría TCP, por tal motivo este protocolo es el más adecuado para constatar la capacidad máxima de la red. A continuación, se muestran las interfaces configuradas para el cliente y el servidor.

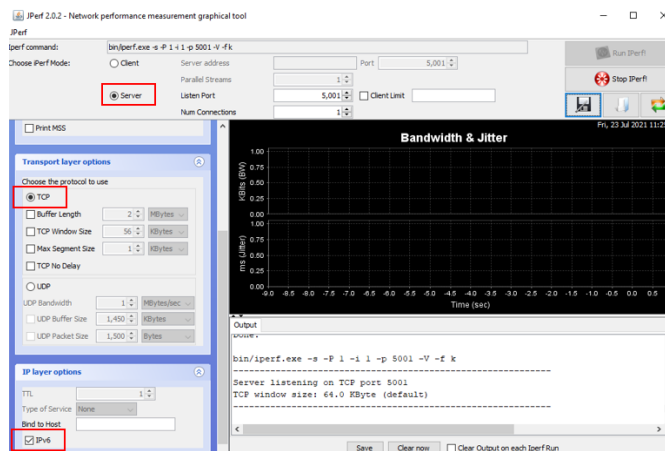


Figura 20. Jperf configurado en el servidor de Windows para captura de paquetes UDP con IPv6.

Para la maquina cliente se utilizó una máquina Windows de un integrante del grupo para configurar el modo cliente se debe conocer la IPv6 del servidor, una vez configurado estos

parámetros como se ven en figura 21 se elige el ancho de banda que se quiere enviar y analizar.

Como primer paso se comenzó a configurar la parte del cliente a enviar un ancho de banda específico, este fue de 1Mbyte/s, para capturarlo en el servidor y poder observar que sucede al aumentar dicho ancho de banda y así al final, saber cuál es valor máximo que se tendrán registros en las gráficas.

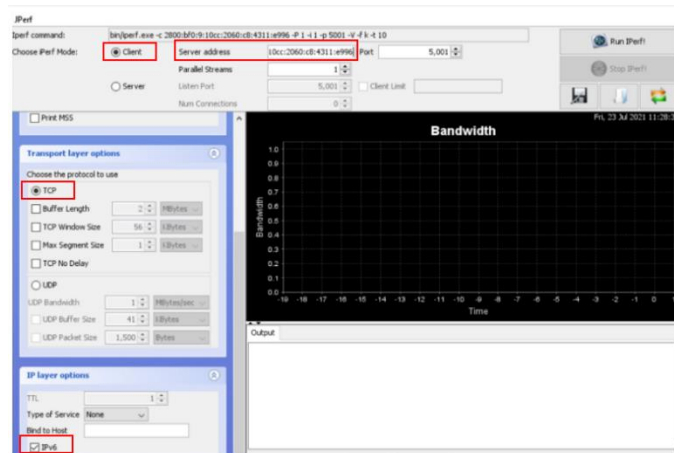


Figura 21. Jperf configurado en el cliente de Windows para captura de paquetes TCP con IPv6.

Ahora se puede visualizar desde la parte del servidor la banda ancha.

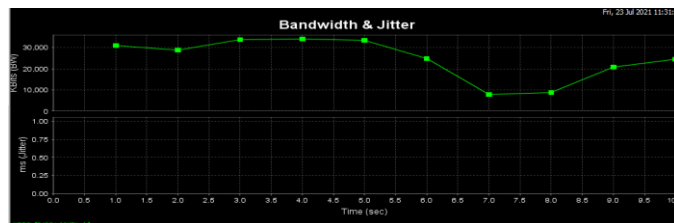


Figura 22. Generación de tráfico TCP con IPv6.

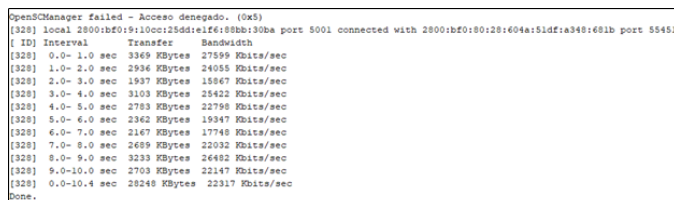


Figura 23. Información de los paquetes TCP con IPv6.

Mientras que la visualización en el cliente es la siguiente:

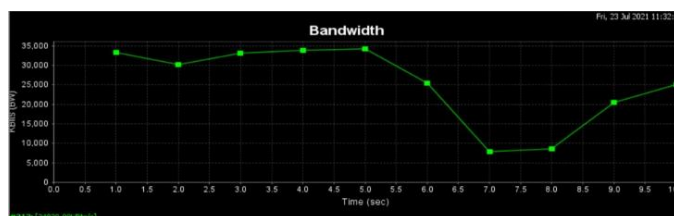


Figura 24. Captura de tráfico con IPv6.

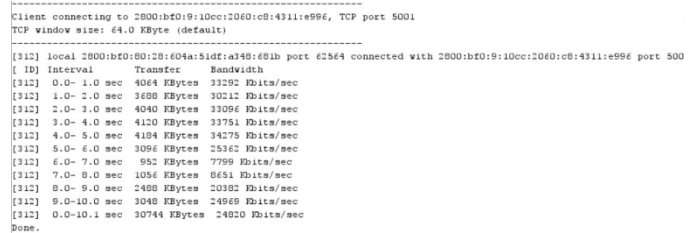


Figura 25. Información de los paquetes TCP capturados con IPv6.

2. Sensado/monitoreo de tráfico usando PRTG.

Para el monitoreo utilizando distintos sensores, se eligió el software PRTG disponible para distintos sistemas operativos, para este caso en Windows se lo descargo de la página oficial y se lo instaló según el manual, que se encuentra en su página oficial.

Al momento de ingresar al PRTG se debe configurar la IP de la máquina que se desea monitorear el tráfico en este caso la maquina actual verificando la ip como se indica.

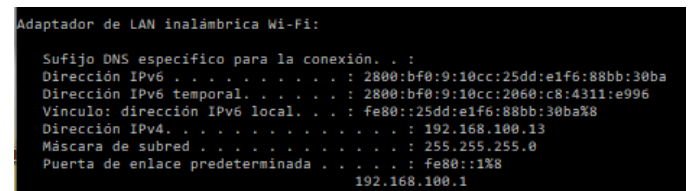


Figura 26. Verificación de la IP.

Después se debe ingresar en la URL de un browser la IP que se detectó para ir a la página principal de PRTG.



Figura 27. Ingreso de la IP a la URL.

Posteriormente se visualiza todos los componentes que posee la página y se dirige hacia resultados



Figura 28. Servicios de PRTG.

El cual nos dirige al resumen de todos los dispositivos y los sensores que estos utilizan para la captura de tráfico como SNMP, Ping, HTTP, IMAP, etc.

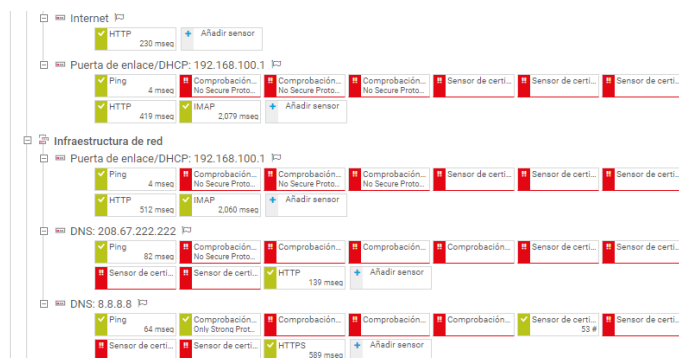


Figura 29. Resumen de Dispositivos y sensores.

El grupo que se va a seleccionar es la puerta de enlace/DHCP: 192.168.100.1, seleccionando el dispositivo que tiene este será HTTP para monitorear el tráfico.

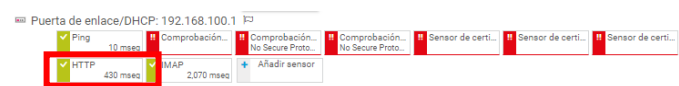


Figura 30. Grupo para seleccionar.

Finalmente se dirige en la sección de datos en vivo y se obtiene el siguiente resultado, monitoreando el tráfico de HTTP.

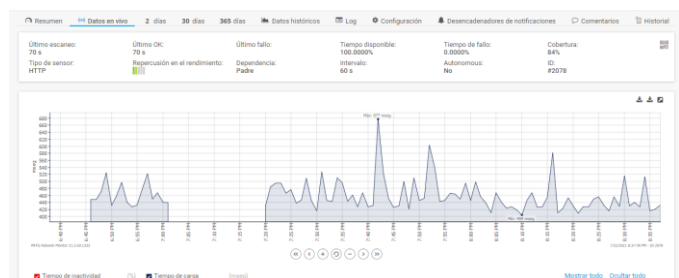


Figura 31. Trafico HTTP.

Se sensor se tiene a tráfico mediante IMAP.



Figura 32. Trafico IMAP.

También se creó un cliente con la IP de la maquina física en este caso 192.168.100.13.

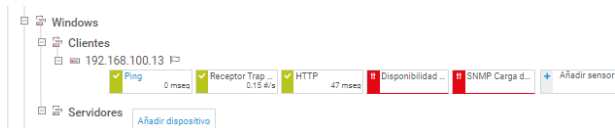


Figura 33. Creación de un cliente.

Ahora se generó un sensor en este caso Receptor Trap SNMP.



Figura 34. Sensor Receptor Trap SNMP.

Y finalmente en las opciones de este sensor se debe dirigir a datos en vivo y se puede observar las gráficas en tiempo real como se indica a continuación.



Figura 35. Trafico SNMP.

3. Tráfico SIPp.

SIP, (Session Initiation Protocol o Protocolo de iniciación de sesión por sus siglas en inglés), es un protocolo de señalización utilizado para establecer una “sesión” entre 2 o más participantes, modificar esa sesión y eventualmente terminar esa sesión. Ha encontrado su mayor uso en el mundo de la Telefonía IP. El hecho de que SIP sea un estándar abierto ha despertado un enorme interés en el mercado de las Centralitas Telefónicas IP. Los mensajes SIP describen la identidad de los participantes en una llamada y cómo los participantes pueden ser alcanzados sobre una red IP. Encapsulado dentro de los mensajes SIP [2].

Algunos fabricantes de teléfonos basados en SIP han tenido un crecimiento exponencial en este sector.

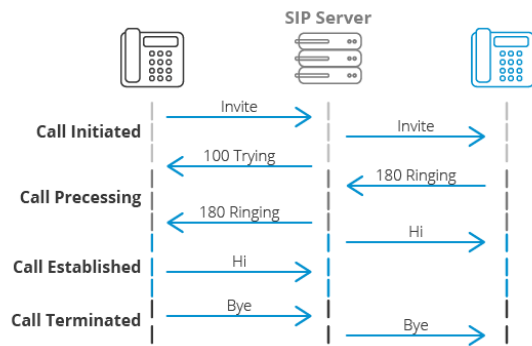


Figura 36. Modelo de Servicio SIP.

Para la generación de tráfico SIPp se necesitará:

- Una máquina virtual con distribución Debian (Ubuntu, Kali, etc.)
- Celular.
- Servidor Asterisk (Servidor de Comunicaciones).

Instalación:

Dentro del ambiente virtual, tenemos que configurar el adaptador de red en modo: Adaptador Puente. Para tener a la máquina virtual en la misma red del celular en este caso la dirección de red es 192.168.200.0

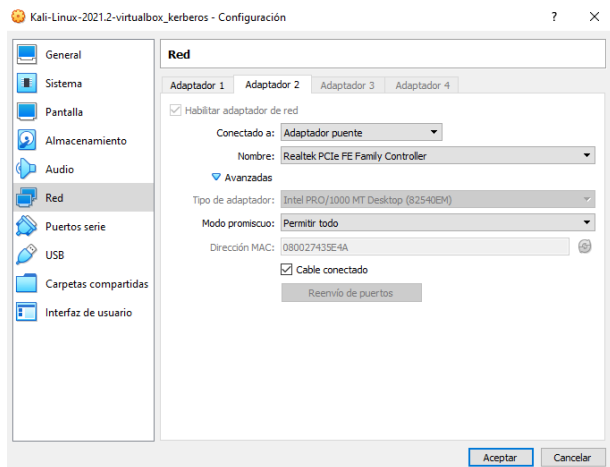


Figura 37. Adaptador Puente-Máquina virtual Kali.

Dentro de la consola de comandos de la máquina virtual en este caso la máquina Kali, comprobaremos que pertenezca a la dirección de red local (192.168.200.0)

```

3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state
link/ether 08:00:27:43:5e:4a brd ff:ff:ff:ff:ff:ff
inet 192.168.200.19/24 brd 192.168.200.255 scope global dynamic eth1
valid_lft 86363sec preferred_lft 86363sec
inet6 fe80::a00:27ff:fe43:5e4a/64 scope link
valid_lft forever preferred_lft forever
  
```

Figura 38. Dirección IP de la máquina Kali.

Con la comprobación de la red, Debemos loguearnos como root, se procede a instalar al servidor Asterisk con los siguientes comandos:

- apt-get install update (Actualiza el Repositorio)
- apt-get install Asterisk (Instalación)
- apt-get install asterisk-prompt-es asterisk-core-sounds-es asterisk-core-sounds-es-gsm asterisk-core-sounds-es-wav asterisk-core-sounds-es-g722 (Paquetes necesarios)

Los comandos de administración del archivo son: service asterisk start, service asterisk stop, service asterisk restart, service asterisk status.

Para empezar a configurar en el archivo /etc/asterisk/cel.conf y en /etc/asterisk/cdr.conf debemos agregar al final del archivo la línea radiuscfg => /etc/radcli/radiusclient.conf (Dirección correcta para que exista clientes).

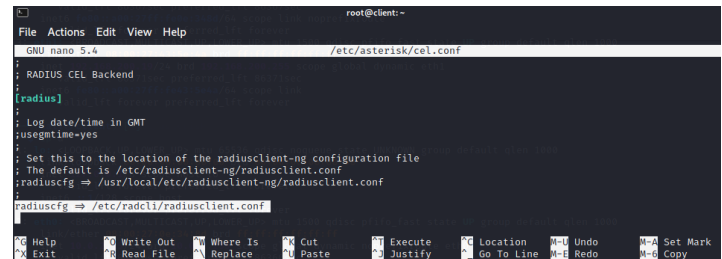


Figura 39. Archivo /etc/asterisk/cel.conf

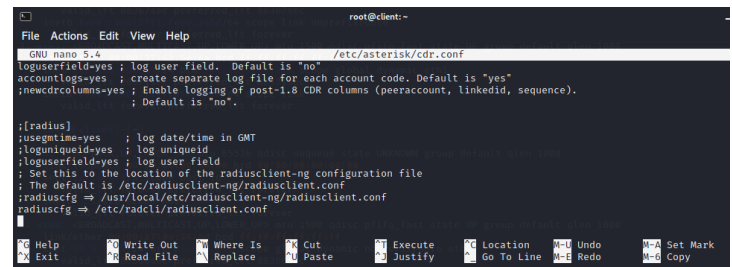


Figura 40. Archivo /etc/asterisk/cdr.conf

Los dos ficheros más importantes para la configuración de Asterisk son:

Sip.conf: que permite definir los canales SIP (peers), tanto para llamadas entrantes como salientes.

Extensions.conf: El que define el comportamiento que va a tener una llamada en nuestra central [3].

Archivo sip.conf

```

GNU nano 5.4 sip.conf *
[general]
context=public
allowoverlap=no
; Permite definir las opciones generales
; Default context for incoming calls. Defaults to 'default'
; Desactiva soporte para marcación superpuesta

udpbindaddr=0.0.0.0
; Dirección de escucha SIP para el protocolo UDP
; (0.0.0.0 escucha por todos los interfaces del servidor)
; Por defecto las conexiones UDP están habilitadas
; y las TCP desactivadas en el servidor
tcpbindaddr=0.0.0.0
; Dirección de escucha SIP para el protocolo TCP
; Protocolo de transporte por defecto
; Activa el enrutamiento de llamadas salientes en base a nombresDNS
transport=udp
rvoicemail=yes
; Permite monitorear la conexión con los teléfonos VoIP
language=es
disallow=all
allow=ulaw
; Idioma por defecto para todos los usuarios
; Desactivar todos los codificadores
; Permitir codificadores en orden de preferencia Asterisk

[usuario1]
type=friend
host=dynamic
context=traficospip

[ext101]
username=celu
secret=1234
port=5061

[ext102]
username=cris
secret=1234
port=5061

```

Figura 41. Archivo /etc/asterisk/sip.conf

Se procede a iniciar el servicio con `service asterisk start`, `service asterisk reload`.

Archivo extensions.conf

```

GNU nano 5.4 extensions.conf
[traficospip]
exten => 101,1,Dial(SIP/ext101)
exten => 102,1,Dial(SIP/ext102)

```

Figura 42. Archivo /etc/asterisk/extensions.conf

Una vez realizado estos cambios para estos archivos, entramos al prompt de Asterisk con comando `Asterisk -rvvvv`, una vez en esta consola ejecutar el comando `dialplan reload`.

Ya tenemos configurado nuestro servidor Asterisk, ahora debemos tener usuarios para ello utilizaremos Zoiper 5, para la versión de celular solo se necesita descargarse la Play Store de nuestro teléfono.

Para la computadora tenemos un proceso más extenso de la página de Zoiper: <https://www.zoiper.com/en/voip-softphone/download/current>

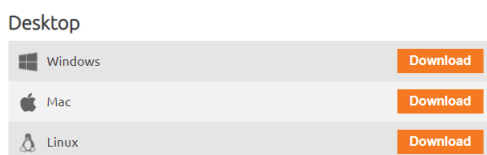


Figura 43. Descarga Zoiper.

Descargaremos la versión en Linux (Free) para la versión Debian.

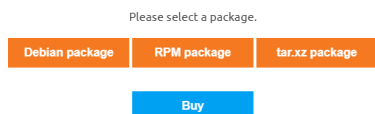


Figura 44. Paquete Debian.

Una vez descargado

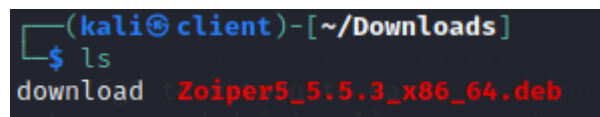


Figura 45. Archivo descargado.

Ejecutamos los comandos:

`dpkg -i Zoiper5_5.3_x86_64.deb`

`apt install -f`

Configuración de Usuarios:

Esta creación depende del archivo sip.conf (Usuario Cris)

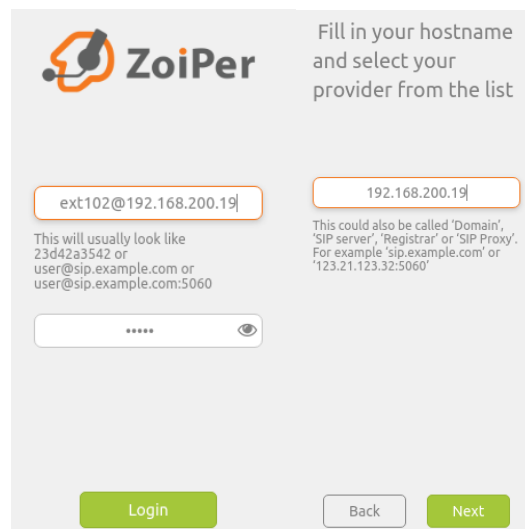


Figura 46. Configuración con extensión 102.

En la siguiente ventana:

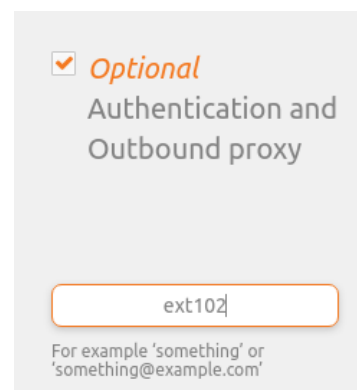


Figura 47. Configuración ext102.

En la ventana final veremos cómo se encuentra SIP UDP lo que se configuro en sip.conf:



Figura 48. SIP UDP.

Y ya podremos realizar llamadas, para el celular es el mismo procedimiento solo cambia la extensión 102 por la 101.

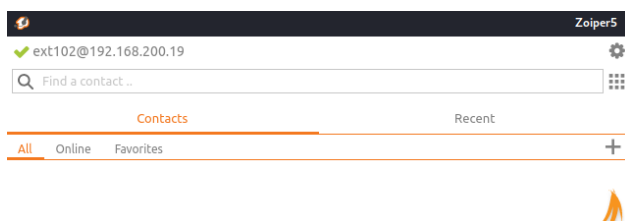


Figura 49. Usuario ext102.

Prueba de Funcionamiento:

Llamada saliente de la ext 102 a la 101.

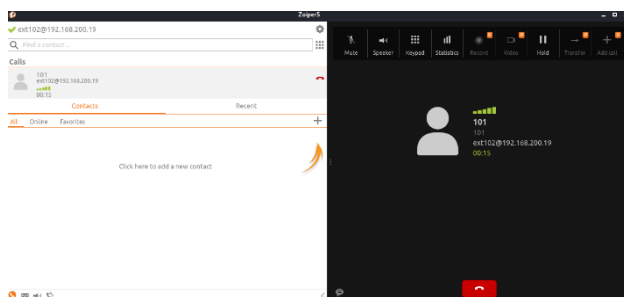


Figura 50. Llamada saliente.

Llamada entrante a la ext 101, de la ext 102.

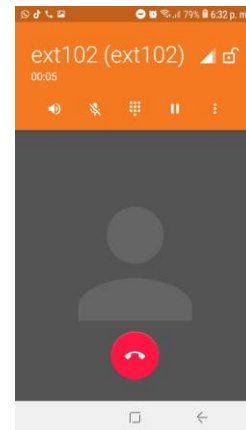


Figura 51. Llamada Entrante

Prompt de Asterisk:

```

client*CLI> sip show users
Username      Secret      Accountcode  Def.Context  ACL  Forced
ext101        s1234       Accountcode  traficosipp  No   No
ext102        s1234       Accountcode  traficosipp  No   No
client*CLI> sip show peers
Name/username  Host      Dyn  Forcerport  Comedia
ption
ext101/celu    192.168.200.3  D  Auto (No)  No
ext102/cris    192.168.200.19 D  Auto (No)  No

2 sip peers [Monitored: 2 online, 0 offline Unmonitored: 0 online, 0 offline]
client*CLI>

```

Figura 52. Prompt Asterisk.

Tráfico generado:

No.	Time	Source	Destination	Protocol	Length	Info
24	0.95443361	192.168.200.19	192.168.200.3	TCP	577	Request: INVITE tcp=192.168.200.19:55442->192.168.200.3:50482
25	0.95443361	192.168.200.19	192.168.200.3	TCP	70	Status: 200 OK
26	1.11470752	192.168.200.3	192.168.200.19	SIP	54	Status: 100 Trying
27	1.11470752	192.168.200.3	192.168.200.19	SIP	54	Status: 180 Ringing
28	1.52739383	192.168.200.3	192.168.200.19	SIP	54	Status: 180 Ringing
29	1.52739383	192.168.200.3	192.168.200.19	SIP	767	Request: ACK sip=192.168.200.3:50482->192.168.200.19:55442
30	1.52739383	192.168.200.19	192.168.200.3	SIP	490	Request: ACK sip=192.168.200.3:50482
31	1.52739383	192.168.200.19	192.168.200.3	SIP	911	Request: INVITE sip=192.168.200.3:50482->192.168.200.19:55442
32	1.52739383	192.168.200.19	192.168.200.3	TCP	70	Status: 200 OK
33	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
34	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
35	1.52739383	192.168.200.19	192.168.200.3	TCP	767	Status: 200 OK
36	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
37	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
38	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
39	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
40	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
41	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
42	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
43	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
44	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
45	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
46	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
47	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
48	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
49	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
50	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
51	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
52	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
53	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
54	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
55	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
56	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
57	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
58	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
59	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
60	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
61	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
62	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
63	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
64	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
65	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
66	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
67	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
68	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
69	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
70	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
71	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
72	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
73	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
74	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
75	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
76	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
77	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
78	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
79	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
80	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
81	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
82	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
83	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
84	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
85	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
86	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
87	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
88	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
89	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
90	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
91	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
92	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
93	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
94	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
95	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
96	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
97	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
98	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
99	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482
100	1.52739383	192.168.200.19	192.168.200.3	TCP	490	Request: ACK sip=192.168.200.3:50482




Figura 53. Tráfico SIP.

IV. CONCLUSIONES

- Se pudo comprobar que para obtener la capacidad máxima de la red utilizando los programas jperf y iperf, se tuvo que enviar paquetes UDP debido a que este es un protocolo no orientado a la conexión por lo cual se lo puede usar para inundar la red y observar su capacidad.
- El uso de un software de monitoreo como es PRTG es de gran ayuda no solo para observar el tráfico de la red, si no de los dispositivos aledaños a esta, como son una impresora, fax, y así observar con distintos sensores como esta varía según su uso.
- Mediante el software jperf para poder sondear y modificar el tráfico de direcciones IPv6 se hace uso de la IPv6 temporal que existe en la máquina de dos de los integrantes del grupo con tipo de paquetes TCP en la que se evidencia él envió mediante el servidor-cliente en el ancho de banda y dejando el campo de jitter sin utilizar.

- Existe tipo de tráfico que no es tan común y generalmente en las redes no se puede captar directamente, por ejemplo, de redes como internet, pero se puede generar de manera manual haciendo uso de máquinas virtuales y en el caso del tráfico SNMP se hace consultas sobre la información de la máquina de interés.

V. BIBLIOGRAFIA

- [1] P. Vouzis, «How to Use JPerf», *NetBeez*, ago. 22, 2018. <https://netbeez.net/blog/how-to-use-jperf/> (accedido jul. 22, 2021).
- [2] «¿Qué es SIP?», *3CX.es*. <https://www.3cx.es/voip-sip/sip/> (accedido jul. 26, 2021).
- [3] Redes Plus,  **ASTERISK TUTORIAL 01**  **INSTALAR y Configurar Servidor VoIP**  *Ubuntu*. Accedido: jul. 26, 2021. [En línea Video]. Disponible en: <https://www.youtube.com/watch?v=yCpPo6aeKU4&t=2743s>