



## Tarea 3: Monitoreo Red LAN

31 de Octubre del 2018

Manuel Becker 13622986 - Sebastián Cruz 14635909

---

### 1. Diagrama de la red

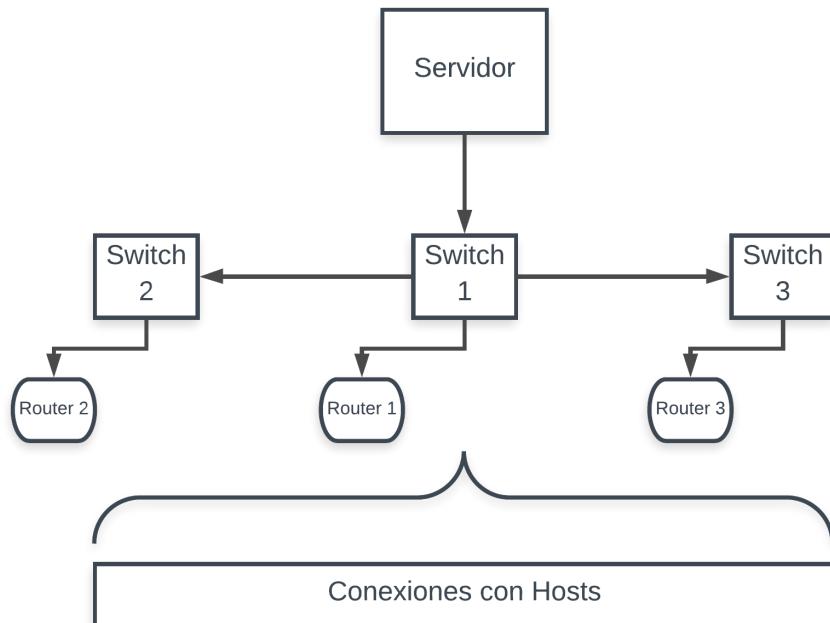


Figura 1: Red LAN establecida durante la actividad

La red establecida en la sala de clases contaba con un servidor el cual estaba conectado a switches. El servidor correspondía a un servidor Ruby on Rails. Directamente, el servidor estaba conectado a uno de estos switches, mientras que los otros 2 estaban conectados al switch inicial. Cada uno de estos contaba con una conexión a un router. Desde cada computador, con nuestra propia dirección IP estática, logramos establecer una conexión al servidor a través de los switches, quienes dirigían a la IP del servidor quien tiene acceso a todas las

transferencias realizadas entre todas las conexiones.

Las conexiones con Hosts involucran a todos los computadores/notebooks conectados en la actividad mediante cable de red o wifi.

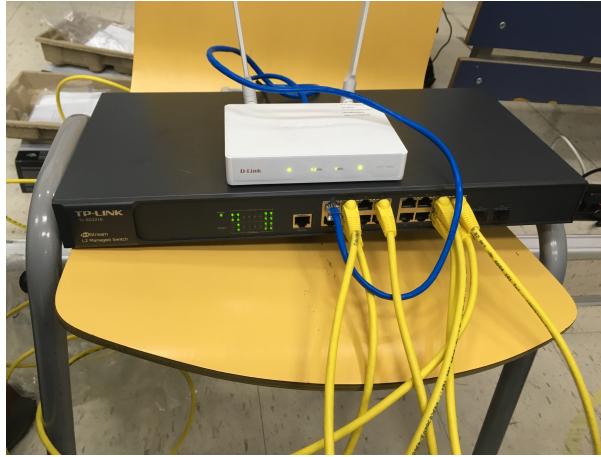


Figura 2: Swith 1 con router 1 y conexiones de hosts

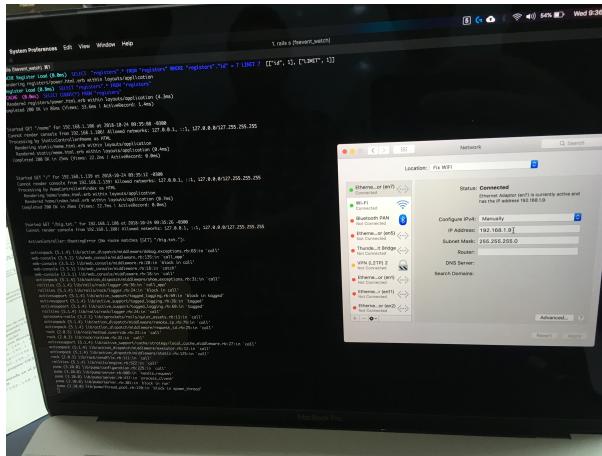


Figura 3: Imagen del servidor con todas las conexiones realizándose durante la actividad

## **2. Parte a**

### **2.1. ¿Qué browser hace la solicitud?**

Utilizamos Mozilla Firefox Quantum (Ubuntu) para realizar las solicitudes correspondientes durante la actividad.

User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86\_64; rv:62.0) Gecko/20100101 Firefox/62.0

### **2.2. Para cada acceso, ¿en qué formato se transfieren los datos?**

Para cada acceso, el formato en que se transfieren los datos fue:

1. http://192.168.1.9:3000/register TCP
2. http://192.168.1.9:3000/ TCP
3. http://192.168.1.9:3000/ TCP
4. http://192.168.1.9/big.txt TCP
5. http://192.168.1.9/win98.mp3 TCP
6. http://192.168.1.9/meme TCP
7. http://192.168.1.9/power TCP

### **2.3. Para cada acceso, ¿cuál es el código HTTP de respuesta?**

Para cada acceso, el código HTTP de respuesta fue el 200.

### **2.4. ¿Cuál es la dirección obtenida de la imagen de la primera vista?**

La dirección obtenida de la imagen de la primera vista es /comando/ultra/secreto/germy.jpg

Dado que no dejaron claro al momento de entregar la tarea cuál era la que debíamos colocar, se recibieron otros formatos de imagen que muestro a continuación.

1. ./comando/ultra/secreto/unpixeltransparente.gif
2. ./comando/ultra/secreto/ingenieriaucsiding.png
3. ./comando/ultra/secreto/logouc\_bottom.png
4. ./comando/ultra/secreto/germy.jpg

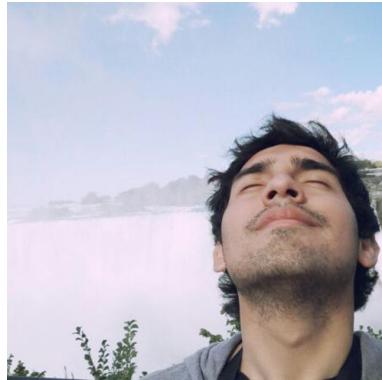


Figura 4: Imagen obtenida a partir de TCP

- 2.5. ¿Qué tipo de paquetes se registraron cuando abrieron la dirección reemplazando el nombre de la imagen por win98.mp3? ¿Por qué existen múltiples paquetes de este tipo luego de acceder? ¿Cuántos bytes se transmitieron del archivo?**

Una vez realizada la GET request, se registraron paquetes de tipo TCP para recibir la data correspondiente al audio. Existen múltiples paquetes, dado que es un archivo más grande, además de que de esta manera, mediante el uso de TCP, se reciben todos los elementos necesarios para que el archivo no sea corrompido al llegar a destino. Fueron transmitidos en total 129310 bytes en total

- 2.6. Para cada acceso de la captura, ¿cuántos byte retorna el browser en cada acceso?**

1. <http://192.168.1.9:3000/register> 91931 bytes (considerando todos los GET que realiza)  
En caso de que sólo quieran la cantidad de bytes que se reciben del acceso a register, y no de todo lo que desencadena, el número de bytes es 3176
2. <http://192.168.1.9:3000/> 104808 bytes
3. <http://192.168.1.9:3000/> 17596 bytes
4. <http://192.168.1.9/big.txt> 6488782 bytes
5. <http://192.168.1.9/win98.mp3> 129310 bytes
6. <http://192.168.1.9/meme> 161503 bytes
7. <http://192.168.1.9/power> 4971 bytes

## 2.7. Para cada acceso, ¿cuántos GET se efectúan en cada caso y por qué?

1. http://192.168.1.9:3000/register 1 GET de register y 15 GET's de lo que desencadena.  
Total 16 GET's
2. http://192.168.1.9:3000/ 16 GET's
3. http://192.168.1.9:3000/ 1 GET's
4. http://192.168.1.9/big.txt 1 GET's
5. http://192.168.1.9/win98.mp3 1 GET's
6. http://192.168.1.9/meme 3 GET's
7. http://192.168.1.9/power 1 GET's

Algunos accesos requieren obtener más información que otros, tales como el 1er, 2ndo y 6to acceso que hace GET a varias imágenes para ser enviadas a través del protocolo TCP. El acceso 3 (repetición del acceso 2) sólo se realiza un GET por el hecho de que se guardan los resultados en caché. Mientras que otras, como el acceso 4, 5 y 7 sólo corresponden a un archivo, lo que hace necesario solo un GET para gatillar el traspaso de información.

## 2.8. ¿Podría indicar si las imágenes de http://192.168.1.9/meme se descargan en forma serial o si esta operación se realiza en paralelo?

Sí se podría indicar. Se realizan en paralelo las descargas de las imágenes. La operación se genera a partir del GET /meme lo que gatilla 2 nuevos GET a cada imagen respectivamente. En la figura 5 se ve el lugar donde se generan ambos GET.

HTTP	73	HTTP/1.1	200	OK	{text/html}
TCP	66	57812 → 3000	[ACK]	Seq=640	A
HTTP	661	GET	/meme_os.jpg	HTTP/1.1	
TCP	74	57814 → 3000	[SYN]	Seq=0	Wi
TCP	66	3000 → 57812	[ACK]	Seq=3157	
TCP	78	3000 → 57814	[SYN, ACK]	Seq=1	
TCP	66	57814 → 3000	[ACK]	Seq=1	Ack
HTTP	663	GET	/unix_meme.jpg	HTTP/1.1	
TCP	66	[TCP Window Update]	3000		

Figura 5: Inicialmente se realizan 2 GET's tal de iniciar la transferencia de información

Posterior a esto, se procede a través del protocolo TCP a recibir los bytes correspondientes a cada imagen y se realiza de manera paralela para ambas. Se puede ver en la figura 5 la recepción de la información a través del TCP.

HTTP	661 GET /meme_os.jpg HTTP/1.1
TCP	74 57814 → 3000 [SYN] Seq=0 Win=2920
TCP	66 3000 → 57812 [ACK] Seq=3157 Ack=1
TCP	78 3000 → 57814 [SYN, ACK] Seq=0 Ack=1
TCP	66 57814 → 3000 [ACK] Seq=1 Ack=1 Wi
HTTP	663 GET /unix_meme.jpg HTTP/1.1
TCP	66 [TCP Window Update] 3000 → 57814
TCP	66 3000 → 57814 [ACK] Seq=1 Ack=598
TCP	180 3000 → 57812 [PSH, ACK] Seq=3157
TCP	180 3000 → 57814 [PSH, ACK] Seq=1 Ack
TCP	66 57814 → 3000 [ACK] Seq=598 Ack=11
TCP	4410 3000 → 57814 [ACK] Seq=115 Ack=59
TCP	66 57814 → 3000 [ACK] Seq=598 Ack=44
TCP	2962 3000 → 57814 [ACK] Seq=4459 Ack=5
TCP	66 57814 → 3000 [ACK] Seq=598 Ack=73
TCP	1018 3000 → 57814 [PSH, ACK] Seq=7355
TCP	66 57814 → 3000 [ACK] Seq=598 Ack=83
TCP	1514 3000 → 57812 [ACK] Seq=3271 Ack=1
TCP	66 57812 → 3000 [ACK] Seq=1235 Ack=4

Figura 6: Luego de ambos GET, comienza a recibirse la información a través de TCP

Luego de recibida la información, en la figura 7 se muestra como ambos GET terminan en código 200 OK.

TCP	1514 3000 → 57812 [ACK] Seq=73647 Ack=1235 Wi
TCP	1514 3000 → 57812 [ACK] Seq=75095 Ack=1235 Wi
HTTP	2938 HTTP/1.1 200 OK (image/jpeg)
TCP	66 57814 → 3000 [ACK] Seq=598 Ack=79235 Wi
TCP	1514 3000 → 57812 [ACK] Seq=76543 Ack=1235 Wi
HTTP	4345 HTTP/1.1 200 OK (JPEG JFIF image)
TCP	66 57812 → 3000 [ACK] Seq=1235 Ack=82270 Wi
ARP	66 Who has 192.168.1.12 Tell 192.168.1.136

Figura 7: Luego de terminada la transmisión, se responde 200 OK

## 2.9. ¿Qué método (de HTTP) se usa en el caso de la request http://192.168.1.9/power y por qué? ¿Qué inconvenientes podría provocar el no usar ese método?

Se hace el método GET, ya que se necesita recibir información. Esta información muestra todas las direcciones IP a las que después había que elegir algunas y se debía hacer ping. Si no se usa GET para que el servidor entregue información, no se podría lograr mostrar el resultado en la página.

### **3. Parte b**

#### **3.1. ¿Cuántos segmentos TCP se transmiten en cada acceso?**

1. http://192.168.1.9:3000/register 41 TCP segments
2. http://192.168.1.9:3000/ 47 TCP segments
3. http://192.168.1.9:3000/ 3 TCP segments
4. http://192.168.1.9/big.txt 658 TCP segments
5. http://192.168.1.9/win98.mp3 29 TCP segments
6. http://192.168.1.9/meme 75 TCP segments
7. http://192.168.1.9/power 3 TCP segments

#### **3.2. ¿Cuáles son los rangos de segmentos TCP que corresponden a cada mensaje HTTP?**

Por rango de segmento creemos que se refieren al máximo y mínimo entre el número recibido por cada GET del HTTP.

1. http://192.168.1.9:3000/register 2 y 10
2. http://192.168.1.9:3000/ 2 y 9
3. http://192.168.1.9:3000/ 3
4. http://192.168.1.9/big.txt 658
5. http://192.168.1.9/win98.mp3 29
6. http://192.168.1.9/meme 3 y 36
7. http://192.168.1.9/power 3

#### **3.3. ¿Hubo paquetes perdidos, dañados, o duplicados? Indique cuántos hubo de cada caso y cómo los identificó.**

Utilizando los comandos disponibles en Wireshark, obtuvimos la siguiente información:

1. tcp.analysis.lost\_segment utilizando este filtro, no nos arroja ningún paquete perdido
2. tcp.analysis.duplicate\_ack utilizando este filtro, no nos arroja ningún paquete duplicado.

Identificamos a partir de estos comandos, que no tuvimos ningún duplicado o pérdida de segmentos TCP en ningún mensaje HTTP realizado.

**3.4. Identifique una secuencia de handshake. Indique en qué paquetes se efectúa y los números de secuencia de cada lado.**

TCP	66 3000 → 57810	[FIN, ACK]	Seq=129311 Ack=667 Win=13
TCP	66 57810 → 3000	[FIN, ACK]	Seq=667 Ack=129312 Win=12
TCP	66 3000 → 57810	[ACK]	Seq=129312 Ack=668 Win=131072
TCP	74 57812 → 3000	[SYN]	Seq=0 Win=29200 Len=0 MSS=1460
TCP	78 3000 → 57812	[SYN, ACK]	Seq=0 Ack=1 Win=65535 Len=0
TCP	66 57812 → 3000	[ACK]	Seq=1 Ack=1 Win=29312 Len=0 TS
HTTP	705 GET /meme HTTP/1.1		
TCP	66 [TCP Window Update] 3000 → 57812	[ACK]	Seq=1 Ack=640 Win=131104 Len=0
TCP	66 3000 → 57812	[ACK]	Seq=1 Ack=640 Win=131104 Len=0
TCP	750 3000 → 57812	[PSH, ACK]	Seq=1 Ack=640 Win=131104
TCP	66 57812 → 3000	[ACK]	Seq=640 Ack=685 Win=30592 Len=0
TCP	2531 3000 → 57812	[PSH, ACK]	Seq=685 Ack=640 Win=131104
TCP	66 57812 → 3000	[ACK]	Seq=640 Ack=3150 Win=35584 Len=0
HTTP	73 HTTP/1.1 200 OK (text/html)		
TCP	66 57812 → 3000	[ACK]	Seq=640 Ack=3157 Win=35584 Len=0

Figura 8: Handshake realizado

Se puede identificar claramente como se realiza un handshake a partir del envío de un ACK, respuesta de SYN y luego ACK SYN para establecer una comunicación.

## 4. Parte c

- 4.1. Construya una lista que incluya los miembros observados en la red. cada entrada de la lista debe incluir: dirección MAC, dirección IP y fabricante de tarjeta de red.

Se construye una lista a partir del filtro `ip.src host == 192.168.1.115` :

MAC	IP	Tarjeta Red
74:e5:f9:a2:2c:95	192.168.1.101	IntelCor
88:e9:fe:6c:06:2e	192.168.1.102	Apple
2c:56:dc:37:7a:f9	192.168.1.103	AsustekC
40:6c:8f:14:e8:57	192.168.1.105	Apple
48:bf:6b:e9:9b:b4	192.168.1.109	Apple
3c:95:09:89:d4:6b	192.168.1.122	LiteonTe
38:63:bb:aa:b8:85	192.168.1.123	HewlettP
f0:76:1c:bd:cd:3a	192.168.1.126	CompaqIn
38:c9:86:05:c8:0d	192.168.1.127	Apple
94:53:30:a6:69:81	192.168.1.131	HonHaiPr
10:02:b5:77:4f:9d	192.168.1.138	IntelCor
18:67:b0:38:82:44	192.168.1.140	SamsungE
68:5b:35:9b:8a:79	192.168.1.142	Apple
3c:a0:67:05:e8:c4	192.168.1.144	LiteonTe
14:2d:27:dd:2a:cd	192.168.1.145	HonHaiPr
28:c2:dd:22:40:a3	192.168.1.147	Azurewave

- 4.2. Explique por qué podrían existir direcciones IP sin información dentro de la tabla ARP de su interfaz de red.

Tabla ARP:

- ? (10.200.0.1) at 00:08:e3:ff:fc:28 [ether] on wlp2s0
- ? (192.168.1.142) at 68:5b:35:9b:8a:79 [ether] on eno1
- ? (192.168.1.138) at 10:02:b5:77:4f:9d [ether] on eno1
- ? (192.168.1.127) at 38:c9:86:05:c8:0d [ether] on eno1
- ? (192.168.1.123) at 38:63:bb:aa:b8:85 [ether] on eno1
- ? (192.168.1.103) at 2c:56:dc:37:7a:f9 [ether] on eno1

- ? (192.168.1.147) at 28:c2:dd:22:40:a3 [ether] on eno1
- ? (192.168.1.1) at incomplete on eno1
- ? (192.168.1.131) at 94:53:30:a6:69:81 [ether] on eno1
- ? (192.168.1.9) at 00:e0:4c:6c:16:c8 [ether] on eno1
- ? (192.168.1.144) at 3c:a0:67:05:e8:c4 [ether] on eno1
- ? (192.168.1.140) at 18:67:b0:38:82:44 [ether] on eno1
- ? (192.168.1.105) at 40:6c:8f:14:e8:57 [ether] on eno1
- ? (192.168.1.101) at 74:e5:f9:a2:2c:95 [ether] on eno1
- ? (192.168.1.109) at 48:bf:6b:e9:9b:b4 [ether] on eno1
- ? (192.168.1.145) at 14:2d:27:dd:2a:cd [ether] on eno1
- ? (192.168.1.122) at 3c:95:09:89:d4:6b [ether] on eno1
- ? (192.168.1.126) at f0:76:1c:bd:cd:3a [ether] on eno1
- ? (192.168.1.126) at f0:76:1c:bd:cd:3a [ether] on eno1

Hay una entrada con incomplete. Esto puede ser por las siguientes razones:

- No respondió al ping.
- No existe.
- Está offline.
- Bloqueó la comunicación.

En este caso, descartamos que esté offline, que no exista y que se bloquee la comunicación por la actividad realizada. Queda que no haya respondido al ping.

#### **4.3. Para una de las direcciones obtenidas luego del ping identifique los paquetes que se envían por la red y que permitan descubrir la dirección MAC de ese miembro de la red.**

Se encuentra un reply al ping realizado por nosotros. Eso significa que ellos son el source y nosotros el destination (ICMP).

En la siguiente entrada del protocolo ICMP se encuentra lo siguiente:  
Ethernet II, Src: LiteonTe\_05:e8:c4 (3c:a0:67:05:e8:c4), Dst: Dell\_1c:05:3b (34:e6:d7:1c:05:3b).

Por lo tanto la MAC de quien pingeamos es 3c:a0:67:05:e8:c4.