

### PRACTICA 3.8

#### NIVEL DE TRANSPORTE: TCP – UDP

1. Abre con el programa wireshark la captura udp.cap en la que se muestra una comunicacion UDP. Contesta a las siguientes preguntas:

a) .Cuales son las direcciones IP y los puertos asociados a la comunicacion?

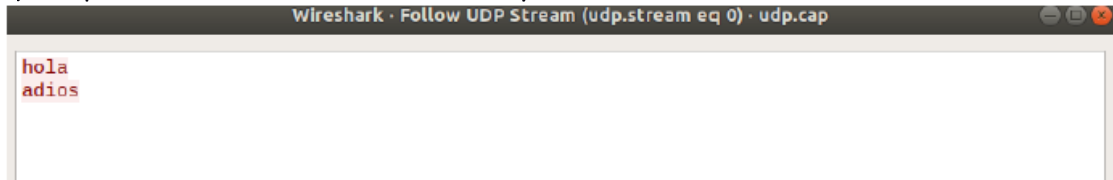
Src: 10.0.0.2 Dst: 11.0.0.2

Src Port: 32768 Dst Port: 33000

b) .En que categoria se encuentran los puertos utilizados?

Se encuentra en la categoría Registrados porque se encuentra entre 1024-49151

c) Comprueba los datos enviados desde Analyze → Follow UDP Stream.



d) Indica que campos tiene una cabecera UDP

0	16	31
Puerto origen UDP		Puerto destino UDP
Longitud de mensaje		Suma de comprobación
Datos (opcional)		

2. Abre con el programa wireshark la captura tcp.cap en la que se muestra una comunicacion TCP. Contesta a las siguientes preguntas:

a) .Cual es la direccion IP y el puerto del cliente TCP y la direccion IP y el puerto del servidor TCP?

Cliente:

IP: 10.0.0.2 Puerto: 60709

Servidor:

IP: 11.0.0.2 Puerto: 34000

b) .En que categoria se encuentran los puertos utilizados?

El cliente se encuentra en el rango Dinámico/Privado (1024-49151)

El servidor se encuentra en el rango Registrados (49152-65535)

c) Comprueba los datos enviados desde Analyze → Follow TCP Stream.



3. Segun lo observado en las dos preguntas anteriores explica resumidamente que diferencias existen entre TCP y UDP.

TCP:

-Establece conexión entre ordenadores antes de transmitir datos.

-Está orientado a conexión.

-Es lento.

-Tiene una confiabilidad altamente fiable.

-Requiere reconocimiento de datos y si el usuario quiere, se puede volver a transmitir.

UDP:

- Envía los datos directamente al destino sin confirmar si está listo para recibir.
- No orientado a conexión.
- Confiabilidad no fidedigna.
- Es rápido.
- No requiere conocimientos ni retransmitir datos perdidos.

4. Utilizando wireshark accede a una pagina web, analiza la primera conexion HTTP. Rellena los siguientes datos:

- Puerto de origen TCP:  
50730
- Clasificación del puerto de origen:  
Registrado
- Puerto de destino TCP:  
443
- Clasificación del puerto de destino:  
Bien conocidos
- ¿Cual es el numero de secuencia relativa establecido?

```
[TCP Segment Len: 0]
Sequence number: 0 (relative sequence number)
[Next sequence number: 0 (relative sequence number)]
```

5. Selecciona la respuesta a la petición HTTP analizada en el ejercicio anterior, en wireshark selecciona "Go → Next Packet in Conversation"

- ¿Cuales son los valores de los puertos de origen y destino?

```
Source Port: 43662
Destination Port: 443
```

- ¿Cuales son los numeros de acuse de recibo y de secuencia relativa establecidos?

```
[Next sequence number: 2124
Acknowledgment number: 63213]
```

6. Utilizando wireshark accede a una pagina web, analiza las consultas y respuestas DNS. Rellena los siguientes datos:

- Dirección MAC de origen y de destino:  
▼ Ethernet II, Src: PcsCompu\_21:45:70 (08:00:27:21:45:70), Dst: RealtekU\_12:35:03 (52:54:00:12:35:03)
- Dirección IP de origen y de destino:  
► Internet Protocol Version 4, Src: 10.0.2.15, Dst: 10.0.2.3
- Puerto de origen y de destino:  
► User Datagram Protocol, Src Port: 58889, Dst Port: 53

7. Vamos a crear una aplicación cliente/servidor que utiliza sockets. Crea el siguiente fichero llamado "servidor.py"

```
#!/usr/bin/python3
import socket
IP_SERVIDOR = "192.168.25.122" /* IP del usuario Axel, donde se aloja el servidor */
PUERTO_SERVIDOR = 4444
#Crea el socket
s = socket.socket()
#Espera conexiones en la IP y puerto establecidos
s.bind((IP_SERVIDOR, PUERTO_SERVIDOR))
print ("Esperando conexiones en el puerto "+ str(PUERTO_SERVIDOR)+"...")
# Espera una conexión y envía confirmación
s.listen(1)
```

```

socket_cliente, addr = s.accept()
print("Conexión recibida de: " + str(addr))
envia = "Conexion con el servidor establecida."
socket_cliente.send(bytes(envia,"utf-8"))
while True:
    recibido = socket_cliente.recv(1024)
    if (recibido.decode() == "quit"):
        print ("Conexión cerrada.")
        break
    print("Recibido: "+recibido.decode())
socket_cliente.close()
s.close()

```

**Ahora crea el siguiente fichero llamado "cliente.py":**

```

#!/usr/bin/python3
import socket
IP_SERVIDOR = "192.168.25.122"
PUERTO_SERVIDOR = 4444
#Crea el socket
s = socket.socket()
print ("Intentando conectar "+str(PUERTO_SERVIDOR)) con "+IP_SERVIDOR+" en el
puerto "+str(PUERTO_SERVIDOR))
# Conexión a la IP y puerto del servidor
servidor = (IP_SERVIDOR, PUERTO_SERVIDOR)
s.connect(servidor)
# Recibe hasta 1024 bytes
recibido = s.recv(1024)
print(recibido.decode())
print ("Enviar mensajes al servidor (escribe 'quit' para terminar): ")
mensaje = ""
while (mensaje != "quit"):
    mensaje = input("> ")
    s.send(bytes(mensaje, 'utf-8'))
s.close()

```

Establece permiso de ejecucion a los ficheros creados "chmod +x \*.py" y ejecuta cada uno en un terminal distinto. Comprueba que funciona. Modifica los ficheros para establecer conexion con el equipo de algun companero. .Que es un socket? .Que informacion necesitan los sockets para establecer una conexion? .Que puertos se usan en el cliente para establecer la conexion?

```

kevmar@PC04:~/Asignaturas/Sistemas Informaticos/TEMA3/Practica 3.8$ ./cliente.py
Intentando conectar con 192.168.25.122 en el puerto 4444
Conexion con el servidor establecida.
Enviar mensajes al servidor (escribe 'quit' para terminar):
> hola

```

```

axegas@PC03:~/Documentos/SI/prac3$ chmod +x *.py
axegas@PC03:~/Documentos/SI/prac3$ ./servidor.py
Esperando conexiones en el puerto 4444...
Conexión recibida de: ('192.168.25.121', 56196)
Recibido: hola

```

Socket: Es un segmento TCP. Queda definido por un protocolo de transporte, un nº de puerto y una dirección IP. Para establecer una conexión entre dos equipos, tiene que conocer el puerto

de origen, (56196 en nuestro caso) el puerto destino (4444 en nuestro caso), la dirección IP destino y la dirección IP origen.

8. Con la aplicación cliente/servidor del ejercicio anterior en ejecución obten una captura de pantalla de todas las conexiones establecidas por tu equipo con el comando “netstat”. Busca las conexiones relativas a tu aplicación. ¿Que protocolos de la capa de transporte se están utilizando? ¿Que números de puertos se están usando en las conexiones? ¿Como se clasifican esos puertos?.

- Se está usando el protocolo TCP
- Se utilizan los puertos 4444(servidor-puerto registrado) y 56212(cliente-puerto dinámico/privado)

**Servidor:**

```
tcp        0      0 192.168.25.122:4444 0.0.0.0:*      ESCUCHAR
12041/python3
```

**Cliente:**

```
tcp        0      0 192.168.25.121:60310 192.168.25.11:789 ESTABLECIDO -
tcp        0      0 192.168.25.121:56212 192.168.25.122:4444 ESTABLECIDO 11836/python3
tcp        56      0 192.168.25.121:60894 192.168.25.11:139 ESTABLECIDO 4208/gvfsd-smb-b
```