

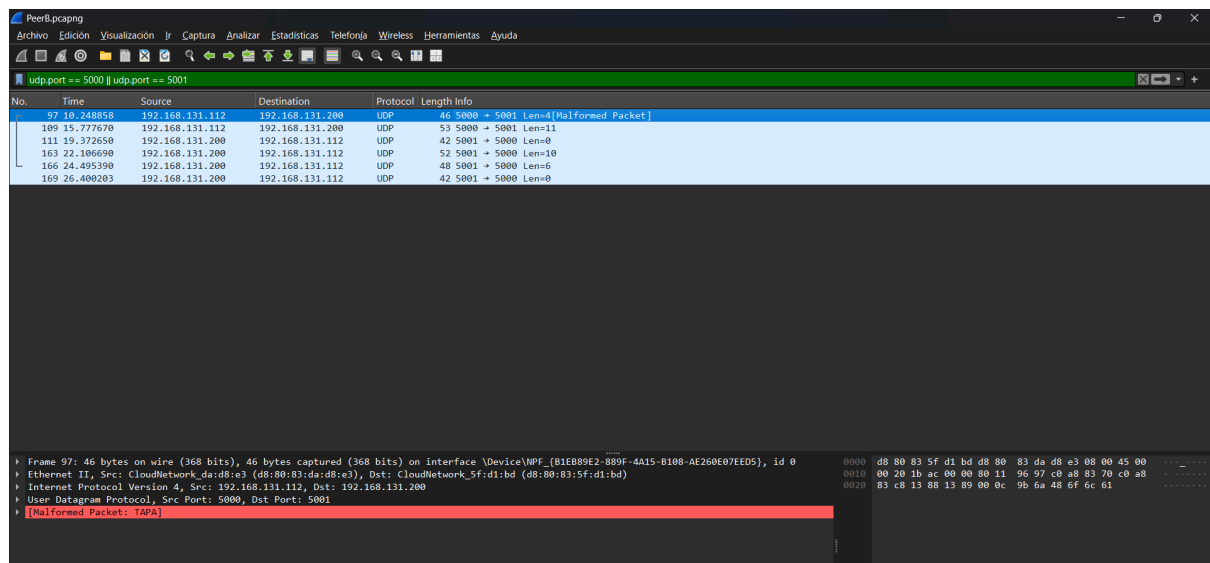
Melissa Dayana Mora Ballesteros

Repositorio: https://github.com/Meli-Ballesteros/Compunet_peer

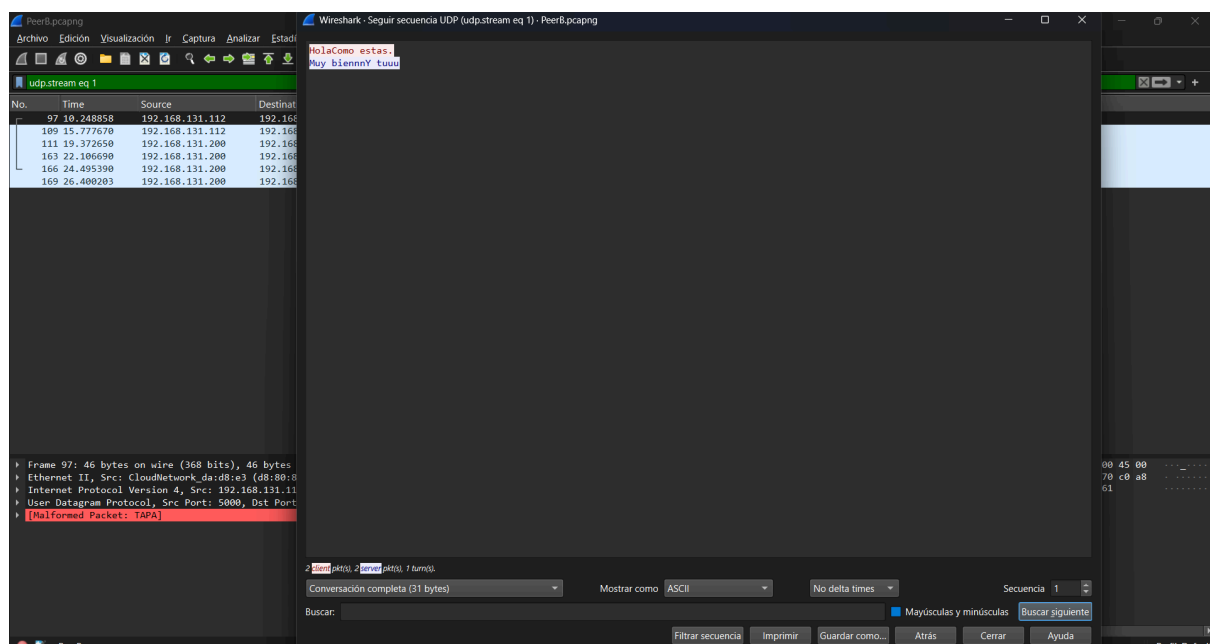
¿Es posible ver en la captura de Wireshark el contenido del mensaje enviado?

R//:

Sí, es completamente posible ver el contenido de los mensajes UDP en Wireshark. Debido a que UDP no cifra ni protege los datos de aplicación. Esto significa que cualquier mensaje que se envíe queda visible en el paquete UDP.



Dandole en click derecho y Seguir se puede ver el contenido.



¿Cuál es el checksum de la captura? ¿Explique/investiguen porque este checksum?

R//:

The image shows a Wireshark packet capture of a UDP stream. The top pane displays a list of packets. Packet 169 is selected, showing details of a User Datagram Protocol (UDP) packet. The packet length is 42 bytes. The source IP is 192.168.131.200 and the destination IP is 192.168.131.112. The source port is 5001 and the destination port is 5000. The checksum is 0x5043, which is highlighted in blue. The bottom pane shows the raw data of the packet in hexadecimal and ASCII format.

No.	Time	Source	Destination	Protocol	Length	Info
97	10.248858	192.168.131.112	192.168.131.200	UDP	46	5000 → 5001 Len=4 [Malformed Packet]
109	15.777670	192.168.131.112	192.168.131.200	UDP	53	5000 → 5001 Len=11
111	19.372650	192.168.131.200	192.168.131.112	UDP	42	5001 → 5000 Len=0
163	22.106690	192.168.131.200	192.168.131.112	UDP	52	5001 → 5000 Len=10
166	24.495390	192.168.131.200	192.168.131.112	UDP	48	5001 → 5000 Len=6
169	26.400203	192.168.131.200	192.168.131.112	UDP	42	5001 → 5000 Len=0

Frame 169: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface \Device\NPF_{B1E889E2-889F-4A15-B108-AE260E07EED5}, id 0
Ethernet II, Src: CloudNetwork_5f:d1:bd (d8:80:83:5f:d1:bd), Dst: CloudNetwork_da:d8:e3 (d8:80:83:da:d8:e3)
Internet Protocol Version 4, Src: 192.168.131.200, Dst: 192.168.131.112
User Datagram Protocol, Src Port: 5001, Dst Port: 5000
Source Port: 5001
Destination Port: 5000
Length: 8
Checksum: 0x5043 [unverified]
[Checksum Status: Unverified]
[Stream Index: 1]
[Stream Packet Number: 6]
[Timestamps]
[Time since first frame: 16.151345000 seconds]

El checksum 0x5043 en hexadecimal es un valor de 16 bits que verifica la integridad de los datos enviados en un datagrama UDP. Este checksum permite que el receptor valide si el contenido fue alterado durante la transmisión.

Wireshark calcula este valor a partir de:

- IP de origen: 192.168.131.200
- IP de destino: 192.168.131.112
- Protocolo: UDP
- Longitud total del mensaje

¿Qué patrones de diseño/arquitectura aplicará al desarrollo de un programa basado en red como este?

- Singleton
- Thread Multihilo
- Single Responsibility
- Mejoraría aplicar Observer Pattern para notificar eventos de llegada de mensaje a una GUI, dividiendo en capas MVC si se implementa JavaFX. Con su UI, eventos y conexión UDP.

Melissa Dayana Mora Ballesteros

Repositorio: https://github.com/Meli-Ballesteros/Compunet_peer

Investiguen que modificaciones son necesarias para implementar este mismo sistema pero para la comunicación TCP en java

R//: Se necesita una gestión explícita de conexiones, como usar Sockets, establecer conexiones reales si no, no funciona. Algo así:

“Servidor”

```
ServerSocket serverSocket = new ServerSocket(5000);  
Socket clientSocket = serverSocket.accept();  
BufferedReader in = new BufferedReader(new  
InputStreamReader(clientSocket.getInputStream()));  
System.out.println("Mensaje recibido: " + in.readLine());
```

“Cliente”

```
Socket socket = new Socket("192.168.131.112", 5000);  
PrintWriter out = new PrintWriter(socket.getOutputStream(), true);  
out.println("Hola desde TCP");
```

¿Qué utilidades de codificación o seguridad agregarías al código?

R//: Aplicaría Advanced Encryption Standard en modo simétrico. Al igual, siento que compartiría una clave secreta entre peers. También, permitiría solo recibir paquetes de IPs autorizadas, rechazando inmediatamente IPs ajenas a mi conocimiento. Igualmente, limitaría el tamaño de los mensajes que se envían, teniendo asimismo un filtro para ver los mensajes que se envían clasificándolos entre buenos y malos según mi criterio. Por último, agregaría una autenticación ya sea con password o token.