Práctica 4

Desarrollo de un cliente y un servidor básico sobre UDP en Java. El servidor ofrecerá un servicio básico de extracción de textos ocultos.

Tarea 1: Especificación del Cliente (clase ClienteUDP.java)

Las especificaciones del cliente:

- La dirección IP y puerto del servidor al que debe el cliente debe enviar se le pasará como argumento en la línea de comandos. Por ejemplo: java ClienteUDP 127.0.0.1 12345
- Una vez conectado el cliente, deberá pedir de forma continua al usuario un texto (un dígito y una línea, por ejemplo: 3hola, ¿qué tal?) que debe ser enviado al servidor.
- Tras cada envío, el cliente deberá esperar la respuesta del servidor que contendrá el texto extraído.
- Cuando el usuario quiera terminar escribirá por teclado el texto que no empiece por un dígito.
- Cuando el cliente detecte que el usuario desea terminar, acabará el programa (sin enviar nada).
- Durante toda la ejecución el cliente debe informar al usuario (escribiendo por pantalla) su estado (por ejemplo: Conectado a 192.168.1.2:12345, Esperando la respuesta...).

Tarea 2: Especificación del Servidor (clase ServidorUDP.java)

Las especificaciones del servidor:

- El puerto por el que recibirá peticiones será pasado como argumento en la línea de comando. Por ejemplo: java ServidorUDP 12345
- El servidor estará esperando constantemente datagramas que incluyen como datos una cadena de texto que debe analizar.
- Use un buffer de 800 Bytes para la recepción.
- El servidor enviará una respuesta al cliente del que recibió el datagrama.
- La respuesta del servidor será el texto extraído. El texto a devolver estará formado por algunas letras del texto original, de forma que cogemos una y luego nos saltamos X, y así sucesivamente, donde X es el dígito que viene al inicio del mensaje. Por ejemplo, el texto 3hola, ¿qué tal? generaría h,ua.
- El servidor deberá informar por la salida estándar (pantalla) de su estado en cada momento.

Tarea 3: Captura de trazas

Simule los siguientes comportamientos tomando con Wireshark la traza del tráfico generado. Si el cliente y el servidor están en la misma máquina, use como IP del servidor la de loopback (127.0.0.1). Como interfaz para capturar debe usar el interfaz denominado Adapter for loopback traffic capture.

Comportamiento UDP1 (traza 1 – **p4e1-3.pcapng**):

- Inicie el servidor y posteriormente dos clientes.
- Envíe varios mensajes en cada cliente y de forma alternada.
- En alguno de los clientes envíe un mensaje con tildes.
- Finalmente finalice los clientes escribiendo un texto que no empiece por un dígito.

Comportamiento UDP2 (traza 2 – **p4e4.pcapng**):

- Sin tener activo ningún servidor intente iniciar el cliente.
- Envíe un mensaje al servidor y posteriormente corte (interrumpa) la ejecución del cliente.

Tarea 4: Análisis de nuestro protocolo en UDP

Responda a las siguientes preguntas usando las trazas capturadas anteriormente.

Usando la traza UDP1 (**p4e1-3.pcapng**):

Ejercicio 1. ¿Cuál es el puerto que usa el cliente? ¿Y el servidor? ¿Qué tipo de puerto es cada uno de ellos?

Ejercicio 2. Wireshark ofrece la opción de “Follow UDP stream”, pero en UDP no existe tal concepto. ¿Cómo es capaz Wireshark de decidir de un mensaje pertenece a un “flujo” u a otro?

Ejercicio 3. Examine un mensaje que lleve tildes, ¿coincide el tamaño indicado en el campo longitud de la cabecera de UDP con la cantidad de letras enviadas? ¿Por qué?

Usando la traza UDP2 (**p4e4.pcapng**):

Ejercicio 4. ¿Por qué consigue enviar si no hay ningún servidor activo? ¿Recibe alguna respuesta? En caso afirmativo, indique qué significa esa respuesta y si es tratada o no.

Sin traza:

Ejercicio 5. Asegure que captura la excepción de la creación del socket UDP y que muestra (método `getMessage()`) el error que se produce (modifique el código si no lo hacía). Intente abrir dos veces el servidor con los mismos parámetros, ¿qué error indica que se produce? ¿Qué debería hacer para solucionar ese error y tener dos servidores del mismo tipo en su equipo?