Normativa de la evaluación de prácticas en la convocatoria de junio

1. A la convocatoria de prácticas de **junio** se pueden presentar todos los estudiantes que hayan suspendido en la convocatoria de enero.
2. Las prácticas tendrán **dos componentes**:
   a) **Prácticas del curso** (50%):
      Se deberán presentar las tres prácticas del curso, cuyos enunciados están publicados en la web de la asignatura. La ponderación de las prácticas, esto es: P1: 35%, P2: 45%, P3: 20%.
   b) **Práctica final** (50%):
      Se deberá presentar una práctica adicional cuyo enunciado se encuentra en este documento.
3. La nota de cada una de las dos partes (a y b) debe ser **superior o igual a 5**. La nota final será la **media aritmética** de la nota **de ambas partes**.
4. Para la parte a) se realizará un **examen** con preguntas relacionadas con las prácticas implementadas por el alumno (similar a los realizados durante el curso).
5. Para la parte b) se realizará un **examen práctico** que consistirá en hacer una serie de modificaciones sobre la práctica entregada.
6. Las prácticas se podrán entregar **hasta el día 22 de junio a las 23:59**.
7. Los exámenes tendrán lugar en uno de los laboratorios durante la semana **del 23 de junio**. La fecha, hora y lugar será anunciada con antelación.
8. La calificación de los exámenes es APTO/NO APTO. En caso de ser **no apto**, la práctica correspondiente se considerará **suspensa**.
9. Para poder optar a la evaluación de junio es **obligatorio inscribirse** mandando un mail a javier.ramos@uam.es **antes del día 31 de mayo**.
10. La práctica, a diferencia de las del modo de evaluación continua, es **individual**.

Práctica final convocatoria de junio: TCP-Tracker

Introducción

El análisis de tráfico de red es necesario en distintos contextos: monitorización, diagnóstico de problemas de red, análisis forense, seguridad, etc.

En esta práctica vamos a aprender el manejo básico de la librería libpcap y, con la ayuda de esta librería, vamos a implementar una aplicación, TCP-Tracker, que permita seguir el rastro y darnos estadísticas de una sesión TCP, dentro de una traza de tráfico agregado.

TCP-Tracker

Con la ayuda de la librería libpcap, vamos a implementar un analizador de tráfico que nos permita seguir el rastro de una sesión TCP dentro de una captura de tráfico.

Para identificar una sesión TCP necesitamos conocer la llamada 5-tupla: dirección IP del extremo que inicia la conexión (cliente), dirección IP del otro extremo (servidor), protocolo de transporte (6 para TCP), puerto TCP del cliente y puerto TCP del servidor.
Las direcciones y puertos de la conexión que queremos tracear se leerán de un fichero de entrada que tiene los 4 datos cada uno en una línea (IPcliente, IP servidor, puerto cliente, puerto servidor).

De la sesión TCP a rastrear nos interesa obtener estadísticas globales como son la duración (diferencia de tiempo entre el primer paquete del cliente y el último paquete, que puede ser del cliente o del servidor), el número total de paquetes en cada uno de los sentidos (del cliente al servidor y del servidor al cliente), el número total de bytes en cada uno de los sentidos a nivel

Ethernet (todo el paquete) y el número total de bytes a nivel TCP (obviando las cabeceras Ethernet, IP y TCP).
Además, es interesante (por ejemplo, para medir retardos) tener la marca de tiempo de los paquetes que forman la apertura y el cierre de la conexión. En particular, estamos interesados, por un lado, en el SYN del cliente al servidor y el SYN/ACK del servidor al cliente (*handshake* de apertura) y, por otro lado, en el FIN del cliente y el FIN del servidor (*handshake* de cierre).

Para procesar sólo aquellos paquetes que nos interesan tenemos que ser capaces de ir leyendo las distintas cabeceras (Ethernet, IP, TCP) e identificar los paquetes de nuestra sesión, descartando el resto. Para ello, para cada paquete de la captura:

1. Extraemos el campo Ethertype de la cabecera Ethernet y comprobamos que es del tipo IP (0x0800).
2. Extraemos los campos versión y protocolo de la cabecera IP, y comprobamos que la versión es 4 y el protocolo es 6.
3. Extraemos las direcciones IP origen y destino de la cabecera IP, y comprobamos que son las del cliente y servidor (o viceversa).
4. Extraemos los puertos origen y destino de la cabecera TCP, y comprobamos que son los del cliente y servidor (o viceversa).
5. Extraemos las banderas de la cabecera TCP, para poder detectar los paquetes de los *handshake* de apertura y cierre de la conexión.

En resumen, el funcionamiento de nuestro programa TCP-Tracker debe ser como sigue:

- En primer lugar, deberá leer de un fichero de configuración (llamado tcptracker.cfg) la 4-tupla de nuestra sesión (IP cliente/servidor, puerto cliente/servidor) y el nombre del fichero de captura de donde vamos a leer el tráfico. El fichero tcptracker.cfg tiene 5 líneas y en cada una uno de los parámetros. Por ejemplo:

```
1.1.1.1
2.2.2.2
3333
80
/home/user/capture1.pcap
```

- En segundo lugar, se debe abrir el fichero de captura con la ayuda de la librería pcap.
- Debemos leer paquete a paquete la captura, identificando los paquetes de nuestra sesión y diferenciando cada uno de los sentidos (cliente->servidor, servidor->cliente) y actualizando las estadísticas (contadores de bytes y paquetes) o los contadores de tiempo (marcas de tiempo de los paquetes de los *handshake*).
- Al finalizar el fichero, se cierra y se vuelcan por pantalla las estadísticas obtenidas de la sesión: 4-tupla, duración, contadores de paquetes y bytes totales por cada sentido, bytes a nivel TCP por cada sentido, timestamp del SYN, timestamp del SYN/ACK, timestamp del FIN del cliente y timestamp del FIN del servidor.

Criterios de evaluación

- Lectura del fichero de entrada (5-tupla): 0.5 puntos
- Manejo correcto de la traza (apertura, lectura, cierre): 0.5 puntos.
- Lectura correcta de la cabecera Ethernet (ethertype): 0.5 puntos.
- Lectura correcta de la cabecera IP (versión, direcciones, protocolo): 2 puntos.
- Lectura correcta de la cabecera TCP (puertos, banderas): 2 puntos.
- Estadísticas globales de la sesión: 1.5 puntos
  - Duración (0.5 puntos)
  - Paquetes/bytes a nivel Ethernet del cliente/servidor (0.5 puntos)
  - Bytes a nivel TCP del cliente/servidor (0.5 puntos)
- Tiempos de handshake de apertura (SYN del cliente, SYN/ACK del servidor): 1.5 puntos.
- Tiempos de handshake de cierre (FIN del cliente, FIN del servidor): 1.5 puntos.

Apéndice 1. libpcap

Libpcap es una librería *open-source* que provee al usuario una API en C para la captura, lectura, almacenamiento... de paquetes de red.

Para usarla desde nuestro programa C, hay que incluir "#include <pcap.h>" y compilar en gcc con la bandera -lpcap.

Funciones básicas (para más información consultar el man de las mismas)

Abrir un fichero pcap

Para abrir un fichero previamente capturado:

```
pcap_t *pcap_open_offline(const char *fname, char *errbuf);
```

Donde

- fname es el nombre del fichero pcap que se desea abrir.
- En errbuf se guarda el mensaje de error, si procede.
- La función nos devuelve el puntero al descriptor de fichero pcap.

Ejemplo

p=pcap_open_offline("traza.pcap", errbuf);

Abre para lectura el fichero traza.pcap. En caso de error, guarda el mensaje en la cadena errbuf.

Leer fichero o interfaz

```
const u_char *pcap_next(pcap_t *p, struct pcap_pkthdr *h);
```

Donde

- p es el puntero a descriptor de fichero del que queramos leer (que anteriormente hemos abierto con pcap_open_offline).
- h es el puntero a la cabecera pcap del paquete. Esta cabecera es un struct con cuatro campos:
  - h->ts.tv_sec, timestamp del paquete en segundos.
  - h->ts.tv_usec, timestamp del paquete en microsegundos.
  - h->len: longitud real del paquete
  - h->caplen: longitud capturada del paquete. Esto es, el puntero que nos devuelve sólo contiene h->caplen bytes.

La función nos devuelve el puntero al inicio del paquete.

Ejemplo

bp = (u_int8_t *) pcap_next (p, &h);

Cerrar Fichero

Los ficheros abiertos con pcap_open_offline se cierran con pcap_close.

```
void pcap_close(pcap_t *p);
```

Donde

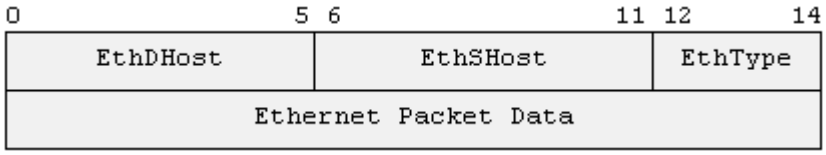
- p es el fichero a cerrar

Ejemplo

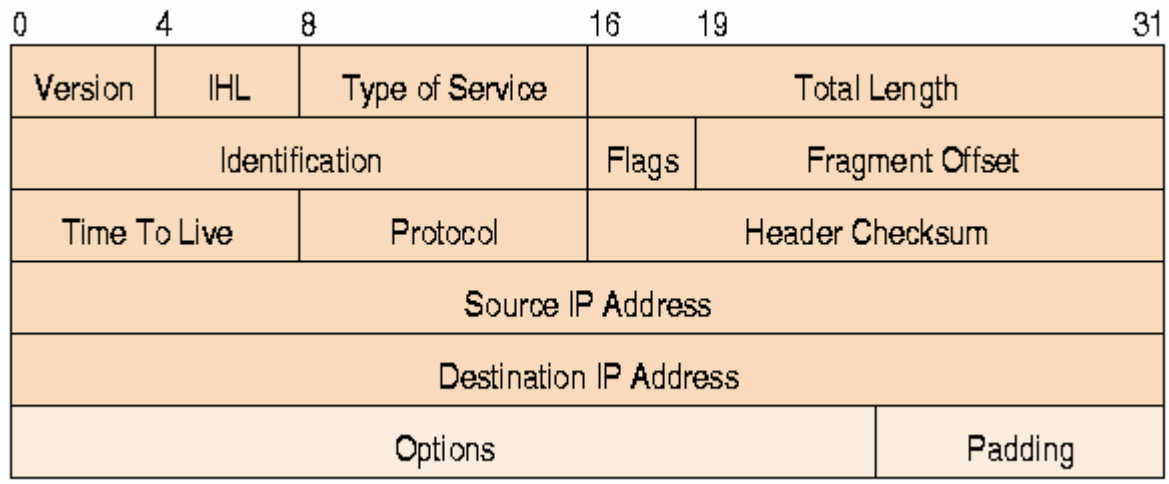
pcap_close (p);

Apéndice 2. Cabeceras de protocolos

Ethernet



IP



TCP