Packet Sniffer

Corso di Gestione di Rete 2009

di **Salvatore Iodice** matricola **301188**

Indice

1.0 Introduzione

Sviluppo di un semplice packet sniffer che salva il flusso di dati catturato in un array e in un database rrd di cui stampa anche il grafico relativo.
Lo sniffer inoltre, stampa periodicamente alcuni parametri di ogni flusso.

2.0 Funzionamento del progetto

Lo sniffer all'avvio permette di selezionare un device da cui iniziare l'analisi.
Successivamente attiva due thread :
- uno che si occupera' degli aggiornamenti al database e delle stampe del flusso
- uno che gestira' la terminazione del programma.

A questo punto e' aperto il device, creato il database rrd e fatto partire il loop di cattura.
Per ogni pacchetto sniffato la funzione callback del loop si occupa di incrementare i contatori dei pacchetti e di salvare, a seconda del tipo di pacchetto (tcp o udp), le porte e gli indirizzi IP sorgente e destinazione, il timestamp del pacchetto e la sua dimensione.
Il flusso e' monodirezionale, quindi e' la stessa funzione callback che crea nuovi flussi e aggiorna quelli gia' presenti.
Ogni flusso e' rappresentato da una struct con i relativi campi (vedi sezione files), tutti i flussi sono contenuti in un array globale acceduto dal gestore dei pachetti e dal gestore degli aggiornamenti.
Ho optato per la soluzione single-threaded in quanto e' lo steso gestore degli aggiornamenti che stampa a video i flussi.
Ogni sessanta secondi il thread gestore del database riceve un segnale di SIGALRM, segnale che lo abilita ad aggiornare il database.
Il thread aggiorna dunque per prima cosa il database rrd, aggiorna il grafico relativo e infine stampa i flussi catturati su stdout.
Lo sniffer puo' essere interrotto con un segnale di SIGTERM o SIGINT, alla ricezione dei quali sono stampati i flussi per l'ultima volta e liberata la memoria usata dal programma.

3.0 Elenco e descrizione files sviluppati

Per la realizzazione del progetto ho implementato i files iodiceSniffer.h e iodiceSniffer.c
Non ho usato ulteriori files in quanto il progettino consta di sole 500 righe di codice quindi, a mio avviso, una ulteriore modularizzazione, almeno per adesso, sembrava superflua.

3.1 iodiceSniffer.h

Contenitore di macro utili al programma e definizioni di funzioni. E' qui definito il prototipo di un singolo flusso di dati :

```
struct pktFlusso {

      int srcPort;              /* porta sorgente (chiave) */
      char* srcIp;              /* Ip sorgente (chiave) */
      int dstPort;              /* porta destinazione (chiave) */
      char* dstIp;              /* Ip destinazione (chiave) */
      int totalPkts;            /* numero pacchetti flusso */
      int totalSize;            /* size totale flusso */
      char strInittime[MAXLEN]; /* timestamp inizio flusso */
      char strFinaltime[MAXLEN];/* timestamp fine flusso */
};
```

3.2 iodiceSniffer.c

Il file in cui e' implementato lo sniffer, le specifiche delle funzioni sono contenute nel relativo file “.h”.