RELAZIONE PROGETTO DI GESTIONE DI RETE

“Gestione_pacchetti”

Christian Scarselli 509140

INTRODUZIONE

“Gestione_pacchetti” si occupa di estrapolare informazioni da un file XML contenenti pacchetti catturati per poi lasciare all’ utente la scelta di una visualizzazione desiderata di queste. Specificatamente si occupa di identificare le informazioni dentro i campi desiderati del file XML, di salvarle dentro un database MYSQL e infine fornire un’ interfaccia grafica all’ utente per accedervi e poterle stampare in una tabella di una pagina HTML.

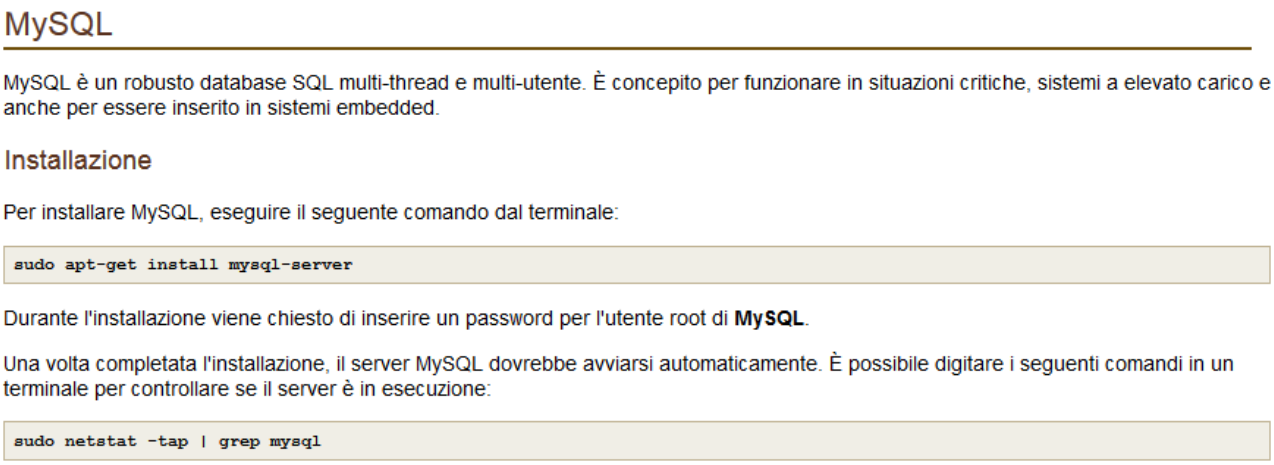
Il software è stato realizzato in JAVA esclusivamente per Linux

Il progetto è essenzialmente diviso in 6 fasi :

1. Fase di installazione e configurazione software necessari al corretto funzionamento (MYSQL SERVER, TSHARK) ;
2. Fase di cattura pacchetti con il software TSHARK e salvataggio di questi in un file XML ;
3. Fase di caricamento del file XML , gestione e identificazione dei dati dei pacchetti e salvataggio di questi come oggetti momentanei ;
4. Fase di creazione di un database denominato “PACCHETTICATTURATI”, di realizzazione di specifiche tabelle interne e di inserimento dati in quest’ ultime ;
5. Fase di realizzazione di una finestra di interfaccia di accesso al database per l’ utente, con la quale inserendo query SQL di selezione sarà possibile estrapolarne i dati desiderati e visualizzarli direttamente ;
6. Fase di stampa dei dati selezionati in una tabella di una pagina HTML.

FASE 1

Per poter utilizzare il software è necessario innanzitutto installare e configurare un MYSQL SERVER, ogni informazione è reperibile dal sito :
http://help.ubuntu-it.org/9.04/ubuntu/serverguide/it/mysql.html



MySQL

MySQL è un robusto database SQL multi-thread e multi-utente. È concepito per funzionare in situazioni critiche, sistemi a elevato carico e anche per essere inserito in sistemi embedded.

Installazione

Per installare MySQL, eseguire il seguente comando dal terminale:

```
sudo apt-get install mysql-server
```

Durante l'installazione viene chiesto di inserire un password per l'utente root di **MySQL**.

Una volta completata l'installazione, il server MySQL dovrebbe avviarsi automaticamente. È possibile digitare i seguenti comandi in un terminale per controllare se il server è in esecuzione:

```
sudo netstat -tap | grep mysql
```

Una volta installato e configurato secondo la guida , non è necessario far altro concernente il server.
Dopo è necessario installare TSHARK, una versione alternativa del software di cattura pacchetti WIRESHARK. Questa versione alternativa mi è stata consigliata dal docente in quanto permette di catturare i pacchetti e poterli salvare direttamente in file XML adoperando un ’ unica riga di codice.

Per poter installare TSHARK ci sono diverse guide online , una utilizzabile è reperibile all' indirizzo http://support.qacafe.com/knowledge-base/how-do-i-install-tshark/

If you would like to obtain your own license for CloudShark, support for CDRouter to CloudShark integration has been added in CDRouter 7.3. You can read more here.

These new features require the applications tshark and capinfos which can be installed on newer Ubuntu systems with the following command:

```
$ sudo apt-get install tshark wireshark
$ /etc/init.d/buddyweb restart
```

Capinfos is normally installed when the wireshark package is installed.

If you are unable to install tshark and capinfos on your Linux distribution, we recommend that you download and compile Wireshark and then install tshark and capinfos as /usr/bin/tshark and /usr/bin/capinfos.

If you are unable to download and compile Wireshark, we have statically linked executables available for 32 and 64 bit x86 systems. Download either the 64 or 32 bit version of the executables below and install as /usr/bin/tshark and /usr/bin/capinfos.

- tshark_x86_64.bin (1.2.10)
- tshark_x86_32.bin (1.2.10)
- capinfos_x86_64.bin (1.2.10)
- capinfos_x86_32.bin (1.2.10)

Una volta installati entrambi la prima fase può dirsi conclusa.

FASE 2

```
netrill@netrill-NC-V5-571G-53334G50MASS:~$ sudo tshark  -i wlan0 -T pdml > filex
ml.xml
[sudo] password for netrill:
tshark: Lua: Error during loading:
 [string "/usr/share/wireshark/init.lua"]:46: dofile has been disabled due to ru
nning Wireshark as superuser. See http://wiki.wireshark.org/CaptureSetup/Capture
Privileges for help in running Wireshark as an unprivileged user.
Running as user "root" and group "root". This could be dangerous.
Capturing on 'wlan0'
1061 ^C
netrill@netrill-NC-V5-571G-53334G50MASS:~$
```

La seconda fase che prevede di catturare i pacchetti unicamente con una riga di codice.

Difatti con il comando da superutente dato nel terminale "**sudo tshark -i wlan0 –T pdml > filexml.xml"** si entra in un ciclo di cattura e tramite l' interruzione di questo (tramite CTRL + c) avviene il salvataggio dei pacchetti in formato XML nel file denominato "filexml.xml".
Abbiamo così il nostro file XML contenente i pacchetti catturati .

FASE 3

Di questa fase si occupa principalmente la classe "XML_LOADER.java". Questa infatti carica il file precedentemente salvato come una sequenza di alberi e i suoi metodi "cercacampiXML() " e "visita ()" si addentrano all' interno alla ricerca dei campi desiderati.
Le informazioni desiderate in questo caso sono :

- Numero pacchetto ;
- Protocollo;
- Mac Address sorgente;
- Mac Address destinatario;
- Tempo della cattura ;
- Dimensione pacchetto ;
- Ip sorgente ;
- Ip destinatario ;
- Udp porta sorgente ;
- Udp porta destinataria ;
- Tcp porta sorgente ;
- Tcp porta Destinataria ;
- Sito internet visitato ;

All' interno di un ciclo for vengono scansionate tutte le radici degli alberi con cui è stato formattato il nostro XML, se queste presentano dei nodi figlio vengono visitati ricorsivamente. Quando "visita()" si trova in corrispondenza di uno dei campi sopra elencati, salva il suo valore in una stringa momentanea, dopodiché quando la lettura di un singolo pacchetto è completata viene creato un oggetto di tipo "pacchetto" in cui saranno salvati i valori appena trovati. Quindi "pacchetto" si unirà a una lista di pacchetti catturati.
E' importante notare che la distinzione da un pacchetto a un altro dentro il file XML avviene tramite il campo "packet" . Per cui per esser sicuri di aver raccolto tutte le informazioni prima di poter salvare il nostro pacchetto definitivamente aspettiamo sempre di leggere il campo successivo ,in modo da saper che la visita del precedente pacchetto si è conclusa e tutte le informazioni che potevano essere catturate sono state catturate.
Inoltre non tutti i pacchetti dentro l' XML sono uguali , ad esempio dentro un pacchetto che non prevede il protocollo DNS non potremo trovare l ' informazione relativa al sito internet. Per cui saranno presenti dei campi nulli dentro ciascun pacchetto.

FASE 4

Per questa fase viene utilizzata principalmente la classe "database.java".
Il costruttore della classe si connetterà automaticamente al server mysql tramite un JDBC (java database connectivity) , creerà il database e le tabelle (se non già presenti), dopodiché farà delle inserzioni su queste se sono presenti pacchetti.

Un JDBC è semplice un connettore che consente l' accesso ai database da un qualsiasi programma scritto in java. Una volta stabilità la connessione è possibile una totale gestione del database e delle sue tabelle tramite query SQL.

Le tabelle in questione sono :

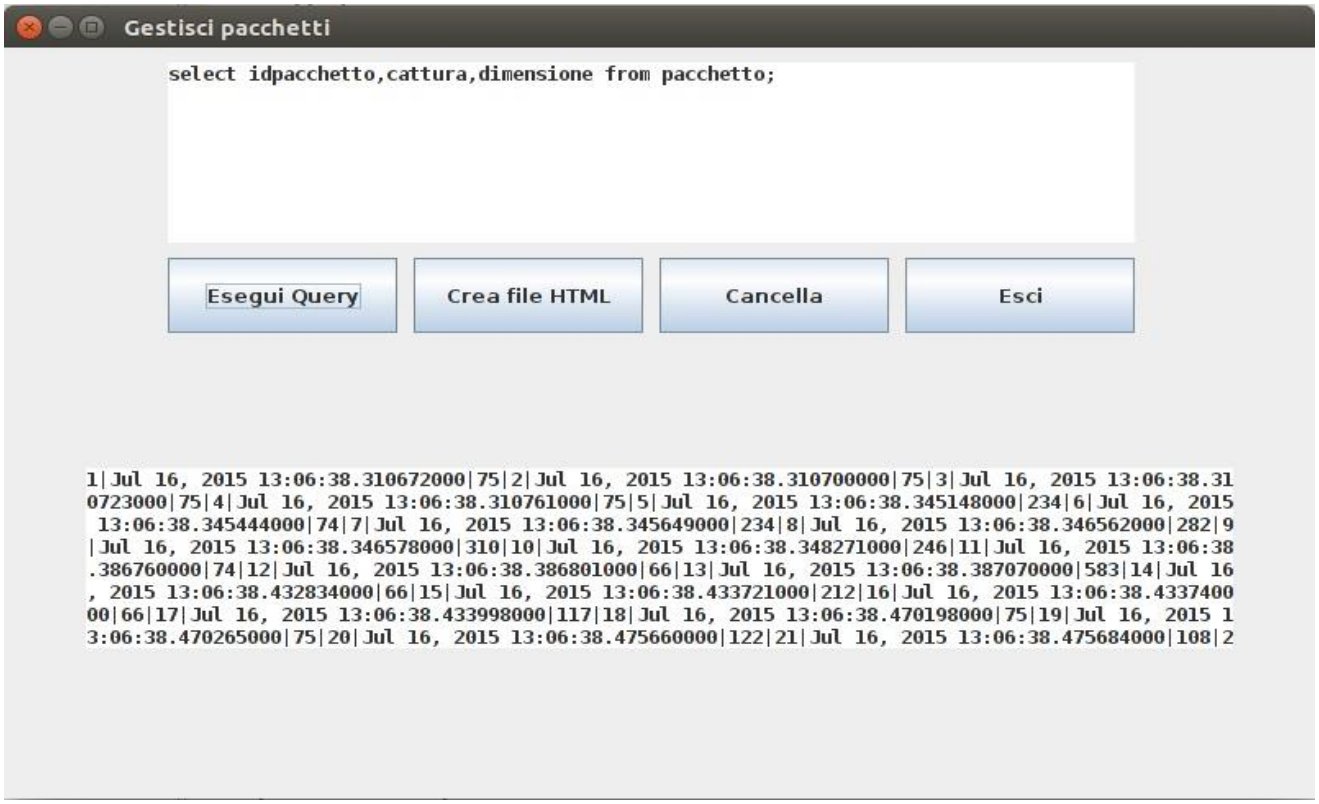
- pacchetto (idpacchetto, numero,cattura,mac_src,mac_dst,protocollo,dimensione);
- ip (idip, ipsrc,ipdst);
- udp (idudp,udpsrc,udpdst) ;
- tcp (idtcp,tcpsrc,tcpdst);
- dns (iddns,sito).

Dove il campo di pacchetto "numero" (numero del pacchetto catturato) sarà la chiave straniera che corrisponderà alla chiave id primarie delle restanti tabelle. Questa relazione permetterà in un secondo momento di effettuare delle JOIN fra tabelle e selezionare più specificatamente le informazioni desiderate. Per esempio possiamo chiedere quale sarà l' ora della cattura dei pacchetti che hanno visitato un sito internet specifico: SELECT cattura,sito FROM pacchetto, dns WHERE pacchetto.numero = dns.iddns and dns.sito ='google.it' . Oppure utilizzare il costrutto SQL "GROUP BY" per aggregare pacchetti in gruppi e poter visualizzare flussi.

Le inserzioni come già detto vengono fatte attraverso delle query, stringhe al cui interno sono concatenati i valori che vengono estrapolati sequenzialmente dai singoli pacchetti. E' stato scelto mi minimizzare gli accessi al database massimizzando i valori da inserire in ciascuna stringa. Questo perché gli accessi sono molto costosi in termini di tempo. Un migliaio di pacchetti possono essere salvati in meno di cinque secondi, con la soluzione opposta precedentemente sviluppata poteva richiedere oltre un minuto nonostante la complessità algoritmica rimanesse identica.
Una volta che tutte le query sono state eseguite , il database è carico dei dati è la fase 4 è ultimata.

FASE 5

Questa fase viene implementata grazie alle classi "interfaccia.java" e "ascolta.java".
Interfaccia, come dal suo nome, crea un' interfaccia grafica contenente 2 JTextArea , 4 JButton e conservando l' accesso al database precedentemente creato. Nei JButton sono implementati degli ascoltatori, in modo che dopo un click di mouse vengono scatenati degli eventi che la classe "ascolta.java" riceverà, identificherà e agirà specificamente per ogni caso.
Nella prima JTextArea dovrà esser scritta la query e una volta premuto il primo tasto manderà in esecuzione e visualizzerà i risultati se presenti nella seconda. L' utente avrà così modo di vedere se il database contiene le informazioni richieste e scegliere in un secondo momento se salvarle in una pagina HTML. Quest' ultima parte viene eseguita dopo l' evento scatenato dal click sul JButton "Crea file HTML", che si occuperà di salvare i dati trovati e i relativi metadati e passarli alla classe "stampahtml.java" come stringa.
E' possibile scrivere solo query di selezione, l' interfaccia non accetta modifiche o inserzioni al database.
Una volta usciti dall' interfaccia avrà termine anche il programma.



FASE 6

L' ultima fase di occupa di creare la pagine HTML finale al cui interno ci saranno i dati selezionati. La classe "stampahtml.java" utilizza una StringBuilder per contenere il codice HTML finale con cui verrà implementata la pagina .
La stringa passata dall' interfaccia, verrà scomposta in singoli dati e stampata nelle celle di appartenenza della tabella.
Infine tramite un buffer di scrittura verrà creato il file finale "tabella_dati.html" e verrà implementato con il codice della StringBuilder.

Conclusioni

Il software estrapola informazioni dai campi di un file XML, le salva in una database e tramite una finestra che opera da interfaccia è possibile visualizzare i dati per poi stamparli in un tabella HTML. Implementando pienamente operazioni SQL di selezione abbiamo notevoli possibilità di analisi dei dati catturati. Ad esempio è possibile visualizzare il singolo pacchetto come in alternativa è possibile aggregarli in flussi (vedi immagini sotto), selezionando ogni qualvolta tramite sintassi SQL il tipo di aggregazione desiderata.
Se si intendesse sviluppare ulteriormente questo software , una possibile scelta di sviluppo potrebbe essere quella di , un primo momento, realizzare del codice che ci consentirebbe di ricavare grafici statistici dalla tabella selezionata e dopo provvedere ad automatizzare tutte le fasi sopra descritte (eccetto la fase di installazione ovviamente) . Ovvero avere una soluzione automatica che provvede ad attivarsi a un' ora stabilita del giorno (o più) e monitorare e stampare i grafici di flusso di quel momento , in modo da poter aver un sistema di monitoraggio che si perpetua nel tempo e poter monitorare e creare statistiche di traffico perfino annuali.

(visione singolo pacchetto)

idpacchetto	numero	cattura	mac_src	mac_dst	protocollo	dimensione
1	1	Jul 16, 2015 13:06:38.310672000	1c:3e:84:8e:b8:5d	00:90:1a:42:28:c2	eth:ip:udp:dns	75
2	2	Jul 16, 2015 13:06:38.310700000	1c:3e:84:8e:b8:5d	00:90:1a:42:28:c2	eth:ip:udp:dns	75
3	3	Jul 16, 2015 13:06:38.310723000	1c:3e:84:8e:b8:5d	00:90:1a:42:28:c2	eth:ip:udp:dns	75
4	4	Jul 16, 2015 13:06:38.310761000	1c:3e:84:8e:b8:5d	00:90:1a:42:28:c2	eth:ip:udp:dns	75
5	5	Jul 16, 2015 13:06:38.345148000	00:90:1a:42:28:c2	1c:3e:84:8e:b8:5d	eth:ip:udp:dns	234
6	6	Jul 16, 2015 13:06:38.345444000	1c:3e:84:8e:b8:5d	00:90:1a:42:28:c2	eth:ip:tcp	74
7	7	Jul 16, 2015 13:06:38.345649000	00:90:1a:42:28:c2	1c:3e:84:8e:b8:5d	eth:ip:udp:dns	234
8	8	Jul 16, 2015 13:06:38.346562000	00:90:1a:42:28:c2	1c:3e:84:8e:b8:5d	eth:ip:udp:dns	282
9	9	Jul 16, 2015 13:06:38.346578000	1c:3e:84:8e:b8:5d	00:90:1a:42:28:c2	eth:ip:icmp:ip:udp:dns	310
10	10	Jul 16, 2015 13:06:38.348271000	00:90:1a:42:28:c2	1c:3e:84:8e:b8:5d	eth:ip:udp:dns	246
11	11	Jul 16, 2015 13:06:38.386760000	00:90:1a:42:28:c2	1c:3e:84:8e:b8:5d	eth:ip:tcp	74
12	12	Jul 16, 2015 13:06:38.386801000	1c:3e:84:8e:b8:5d	00:90:1a:42:28:c2	eth:ip:tcp	66
13	13	Jul 16, 2015 13:06:38.387070000	1c:3e:84:8e:b8:5d	00:90:1a:42:28:c2	eth:ip:tcp:ssl	583
14	14	Jul 16, 2015 13:06:38.432834000	00:90:1a:42:28:c2	1c:3e:84:8e:b8:5d	eth:ip:tcp	66
15	15	Jul 16, 2015 13:06:38.433721000	00:90:1a:42:28:c2	1c:3e:84:8e:b8:5d	eth:ip:tcp:ssl	212
16	16	Jul 16, 2015 13:06:38.433740000	1c:3e:84:8e:b8:5d	00:90:1a:42:28:c2	eth:ip:tcp	66
17	17	Jul 16, 2015 13:06:38.433998000	1c:3e:84:8e:b8:5d	00:90:1a:42:28:c2	eth:ip:tcp:ssl	117

(visione per aggregazione in flussi)

Numero_pacchetti	ipsrc	ipdst	cattura	protocollo	dimensione	mac_src	mac_dst
735	10.196.221.62	224.0.0.251	Jul 17, 2015 11:29:02.056258000	eth:ip:udp:dns	87	1c:3e:84:8e:b8:5d	01:00:5e:00:00:fb
79	103.31.7.184	10.196.221.62	Jul 17, 2015 11:29:23.836706000	eth:ip:tcp	66	00:90:1a:42:28:c2	1c:3e:84:8e:b8:5d
44	104.16.13.8	10.196.221.62	Jul 17, 2015 11:29:24.244724000	eth:ip:tcp	66	00:90:1a:42:28:c2	1c:3e:84:8e:b8:5d
5	131.114.3.24	10.196.221.62	Jul 17, 2015 11:29:06.864386000	eth:ip:tcp	74	00:90:1a:42:28:c2	1c:3e:84:8e:b8:5d
47	131.114.3.27	10.196.221.62	Jul 17, 2015 11:29:10.539684000	eth:ip:tcp	78	00:90:1a:42:28:c2	1c:3e:84:8e:b8:5d
55	141.101.114.59	10.196.221.62	Jul 17, 2015 11:29:23.833705000	eth:ip:tcp	66	00:90:1a:42:28:c2	1c:3e:84:8e:b8:5d
15	173.194.40.64	10.196.221.62	Jul 17, 2015 11:29:09.484117000	eth:ip:tcp	74	00:90:1a:42:28:c2	1c:3e:84:8e:b8:5d
17	198.252.206.16	10.196.221.62	Jul 17, 2015 11:29:23.567945000	eth:ip:tcp	74	00:90:1a:42:28:c2	1c:3e:84:8e:b8:5d
6	198.252.206.17	10.196.221.62	Jul 17, 2015 11:29:25.132471000	eth:ip:tcp	74	00:90:1a:42:28:c2	1c:3e:84:8e:b8:5d
4	2.228.46.105	10.196.221.62	Jul 17, 2015 11:29:25.250361000	eth:ip:tcp	74	00:90:1a:42:28:c2	1c:3e:84:8e:b8:5d
3	2.228.46.113	10.196.221.62	Jul 17, 2015 11:29:24.249446000	eth:ip:tcp	74	00:90:1a:42:28:c2	1c:3e:84:8e:b8:5d
4	2.228.46.121	10.196.221.62	Jul 17, 2015 11:29:24.247964000	eth:ip:tcp	74	00:90:1a:42:28:c2	1c:3e:84:8e:b8:5d
1	216.58.210.234	10.196.221.62	Jul 17, 2015 11:29:23.836831000	eth:ip:tcp	74	00:90:1a:42:28:c2	1c:3e:84:8e:b8:5d
69	216.58.212.110	10.196.221.62	Jul 17, 2015 11:29:10.291504000	eth:ip:tcp	74	00:90:1a:42:28:c2	1c:3e:84:8e:b8:5d
2	216.58.212.67	10.196.221.62	Jul 17, 2015 11:29:04.021561000	eth:ip:tcp	74	00:90:1a:42:28:c2	1c:3e:84:8e:b8:5d
10	216.58.212.78	10.196.221.62	Jul 17, 2015 11:29:09.873110000	eth:ip:tcp	74	00:90:1a:42:28:c2	1c:3e:84:8e:b8:5d
11	23.21.247.182	10.196.221.62	Jul 17, 2015 11:29:24.469633000	eth:ip:tcp	74	00:90:1a:42:28:c2	1c:3e:84:8e:b8:5d
5	23.51.251.27	10.196.221.62	Jul 17, 2015 11:29:25.254042000	eth:ip:tcp	74	00:90:1a:42:28:c2	1c:3e:84:8e:b8:5d
10	54.230.47.231	10.196.221.62	Jul 17, 2015 11:29:10.264958000	eth:ip:tcp	74	00:90:1a:42:28:c2	1c:3e:84:8e:b8:5d
163	62.101.93.101	10.196.221.62	Jul 17, 2015 11:29:03.946866000	eth:ip:udp:dns	231	00:90:1a:42:28:c2	1c:3e:84:8e:b8:5d