

# Digital Forensics

## Mining SNORT logs using SQL Queries

we have performed the queries in SQL and Perl. In this report, we are going to mention the queries.

SNORT analysis will be performed whenever any malicious thing or any suspicious things happens at the Victim's system or computer. It will store the alert message as .txt file and it will open the alert.txt file showing the details of the malicious thing happened.

- Firstly, we have to collect the attack logs which will be stored in snortlog in the database using SNORT analysis.
- Then parse the file to index required information.
- Finally, it will send the data to the MySQL database.

```
C:\WINDOWS\system32\cmd.exe
20080703054511, Smurf, 177.189.176.23, 206.21.5.232, Russia, 3, 2
20080703063841, Ping_Flood, 250.52.15.7, 206.21.5.232, Antarctica, 7, 3
20080703073316, Ping_Flood, 122.24.9.52, 206.21.5.232, Canada, 1, 3
20080703081808, Smurf, 44.167.33.121, 206.21.5.232, USA, 1, 2
20080703090043, Dictionary, 140.169.212.60, 206.21.5.232, Congo, 5, 1
20080703094135, Dictionary, 102.149.139.25, 206.21.5.232, Sealand, 3, 1
20080703102527, Smurf, 250.52.15.32, 206.21.5.232, Antarctica, 7, 2
20080703110918, DNS_Poisoning, 9.21.136.97, 206.21.5.232, USA, 1, 5
20080703120903, Ping_Flood, 70.43.41.116, 206.21.5.232, Chad, 5, 3
20080703130347, Dictionary, 213.131.194.123, 206.21.5.232, UAE, 4, 1
20080703134935, Smurf, 79.35.65.91, 206.21.5.232, Iceland, 3, 2
20080703143118, Buffer_Overflow, 204.98.149.166, 206.21.5.232, UAE, 4, 4
20080703151709, Smurf, 250.52.15.49, 206.21.5.232, Antarctica, 7, 2
20080703161357, Ping_Flood, 197.174.125.138, 206.21.5.232, Tibet, 4, 3
20080703170336, Buffer_Overflow, 74.117.42.25, 206.21.5.232, Iceland, 3, 4
20080703173615, Dictionary, 96.55.70.138, 206.21.5.232, Sealand, 3, 1
20080703175202, Ping_Flood, 250.52.15.46, 206.21.5.232, Antarctica, 7, 3
20080703182156, Ping_Flood, 117.201.61.88, 206.21.5.232, Canada, 1, 3
20080703184432, Ping_Flood, 133.179.215.71, 206.21.5.232, Congo, 5, 3
20080703190826, Buffer_Overflow, 90.127.20.194, 206.21.5.232, Sealand, 3, 4
20080703195624, Smurf, 186.52.205.209, 206.21.5.232, Russia, 3, 2
20080703204000, Smurf, 96.170.91.147, 206.21.5.232, Sealand, 3, 2
20080703210616, Ping_Flood, 136.27.214.15, 206.21.5.232, Congo, 5, 3
20080703214538, Dictionary, 156.198.109.209, 206.21.5.232, Congo, 5, 1
20080703222930, Smurf, 250.52.15.25, 206.21.5.232, Antarctica, 7, 2
20080703230429, Smurf, 98.61.139.164, 206.21.5.232, Sealand, 3, 2
20080703230589, Smurf, 250.52.15.21, 206.21.5.232, Antarctica, 7, 2
C:\SI08_Handout>
```

The above are the random attacks happened and it will store the timestamp and all the details of the activity happened in victim's system.

```
10.1.1.1 - default - SSH Secure Shell
File Edit View Window Help
Quick Connect Profiles

mysql> use si08;
Database changed
mysql> desc snortlog;
+-----+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| Timestamp | char(20) | YES | | NULL | |
| Attack | char(20) | YES | | NULL | |
| SrcIP | char(20) | YES | | NULL | |
| DesIP | char(20) | YES | | NULL | |
| Country | char(20) | YES | | NULL | |
| SrcRegion | char(1) | YES | | NULL | |
| Threat | int(11) | YES | | NULL | |
+-----+-----+-----+-----+-----+
7 rows in set (0.00 sec)

mysql>
```

Connected to 10.1.1.1 SSH2 - aes128-cbc - hmac-md5 - none 80x17

By SNORT analysis the data has been stored and we can retrieve by executing the query in MySQL.

## **Testing we have done:**

We have to see if the connection settings match or not.

Have to confirm whether the SNORT log file opens or not.

Verify parsing accuracy through execution.

Check MySQL database for data sent by program.

## **Errors we have got:**

MySQL sometimes denied the Perl access.

File extensions(.txt) sometimes didn't open.

Parser, at first, constructed incorrect strings.

At times, data was sent to a non-existent database.

## **Requirements for Database queries.**

Firstly, connect to MySQL database without any errors.

Format the data for easier reading to follow up with the data.

Format data into 6 hours portions to control the traffic from SNORT.

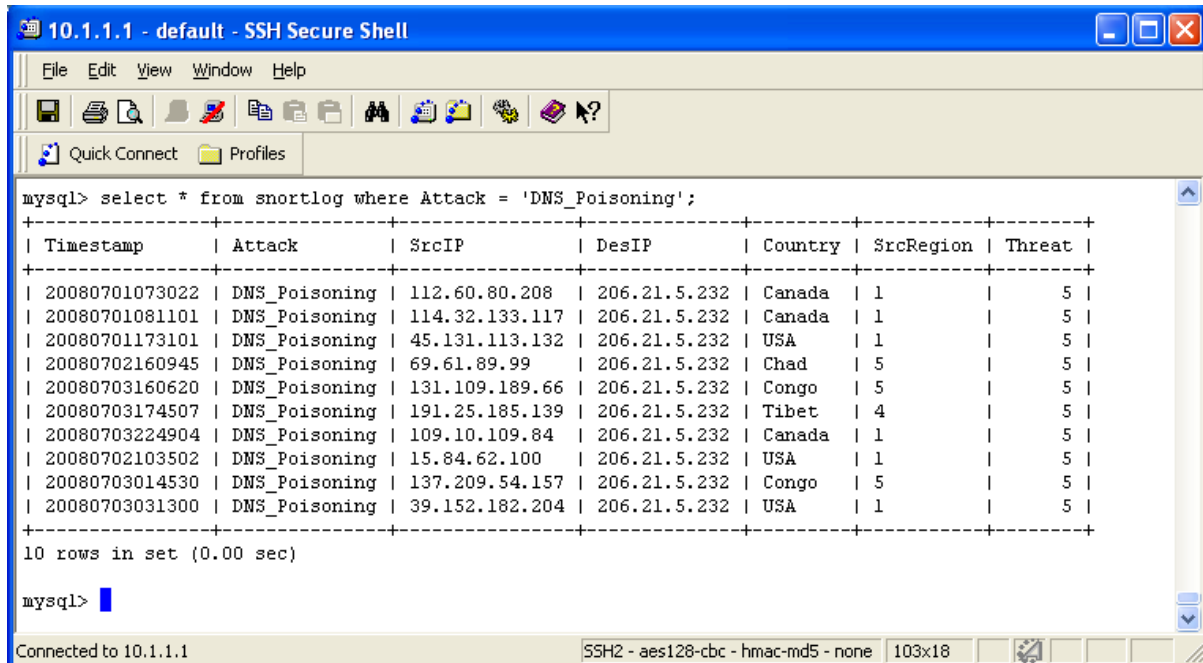
Create a .txt file with the data which we have got from the SNORT analysis.

## **Steps to follow for the database query to get the data.**

- To execute MySQL firstly, we have to open the connection to MySQL database.
- Then query the database for the specific columns as the required.
- After executing the required query, it will print the data from columns.
- Repeat the queries until all needed data is gathered.
- Finally, print the formatted data into a .txt file.

## Syntax of MySQL:

- Select {column} from {table} where {condition};



The screenshot shows an SSH Secure Shell window titled "10.1.1.1 - default - SSH Secure Shell". The window contains a terminal interface where a MySQL query has been executed. The query is `mysql> select * from snortlog where Attack = 'DNS_Poisoning';`. The results are displayed in a table with 7 columns: Timestamp, Attack, SrcIP, DesIP, Country, SrcRegion, and Threat. There are 10 rows of data, all showing 'DNS\_Poisoning' attacks. Below the table, it says "10 rows in set (0.00 sec)". The status bar at the bottom indicates "Connected to 10.1.1.1" and "SSH2 - aes128-cbc - hmac-md5 - none 103x18".

```
mysql> select * from snortlog where Attack = 'DNS_Poisoning';
```

Timestamp	Attack	SrcIP	DesIP	Country	SrcRegion	Threat
20080701073022	DNS_Poisoning	112.60.80.208	206.21.5.232	Canada	1	5
20080701081101	DNS_Poisoning	114.32.133.117	206.21.5.232	Canada	1	5
20080701173101	DNS_Poisoning	45.131.113.132	206.21.5.232	USA	1	5
20080702160945	DNS_Poisoning	69.61.89.99	206.21.5.232	Chad	5	5
20080703160620	DNS_Poisoning	131.109.189.66	206.21.5.232	Congo	5	5
20080703174507	DNS_Poisoning	191.25.185.139	206.21.5.232	Tibet	4	5
20080703224904	DNS_Poisoning	109.10.109.84	206.21.5.232	Canada	1	5
20080702103502	DNS_Poisoning	15.84.62.100	206.21.5.232	USA	1	5
20080703014530	DNS_Poisoning	137.209.54.157	206.21.5.232	Congo	5	5
20080703031300	DNS_Poisoning	39.152.182.204	206.21.5.232	USA	1	5

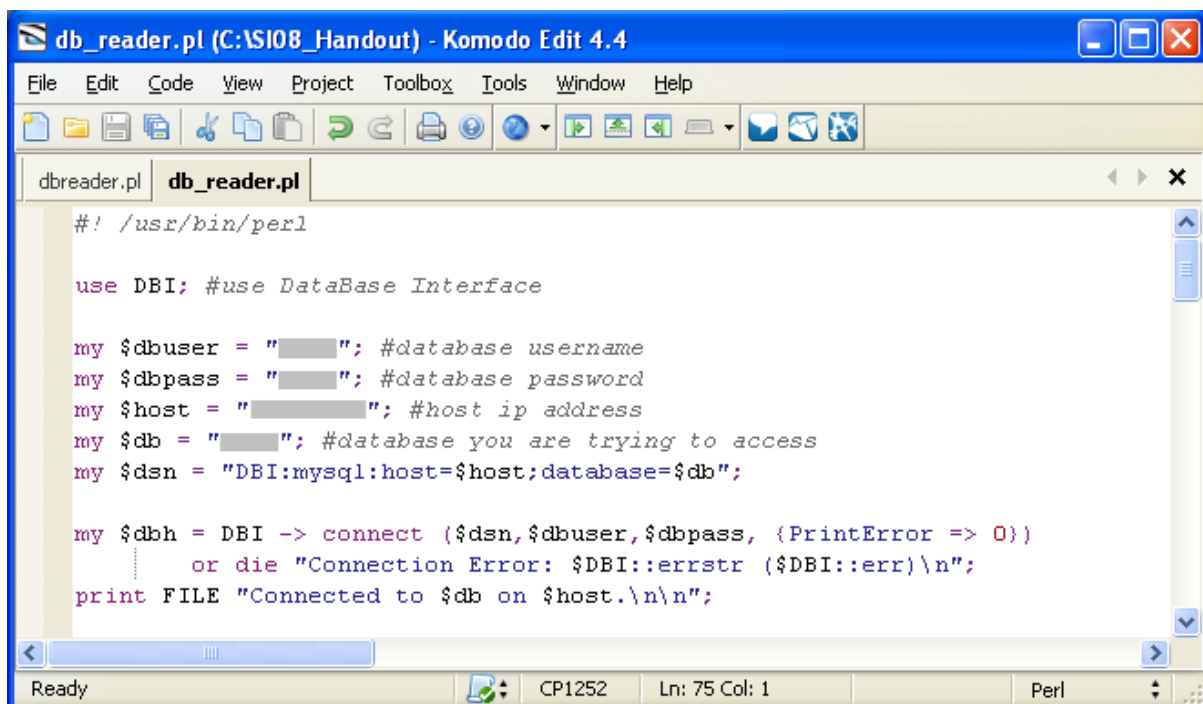
```
10 rows in set (0.00 sec)
```

```
mysql>
```

Connected to 10.1.1.1 SSH2 - aes128-cbc - hmac-md5 - none 103x18

## Syntax for Perl:

- my \$query = "{MySQL command}";
- my \$sth=\$dbh->prepare(\$query);
- \$sth->execute;



The screenshot shows a Komodo Edit 4.4 window titled "db\_reader.pl (C:\SI08\_Handout) - Komodo Edit 4.4". The window contains a Perl script named "db\_reader.pl". The script uses the DBI module to connect to a MySQL database. It defines variables for database username, password, host, and database name. It then uses the DBI connect method to establish a connection and prints a message if successful. The status bar at the bottom indicates "Ready", "CP1252", "Ln: 75 Col: 1", and "Perl".

```
#!/usr/bin/perl

use DBI; #use DataBase Interface

my $dbuser = " "; #database username
my $dbpass = " "; #database password
my $host = " "; #host ip address
my $db = " "; #database you are trying to access
my $dsn = "DBI:mysql:host=$host;database=$db";

my $dbh = DBI -> connect ($dsn,$dbuser,$dbpass, {PrintError => 0})
    or die "Connection Error: $DBI::errstr ($DBI::err)\n";
print FILE "Connected to $db on $host.\n\n";
```

Ready CP1252 Ln: 75 Col: 1 Perl

```
read.pl (C:\SI08_Handout) - Komodo Edit 4.4
File Edit Code View Project Toolbox Tools Window Help
Start Page attack_gen.pl read.pl log_analysis.pl

use DBI;
my $dbuser = "si08"; # $CONF_DBUSER
my $dbpass = "si08"; # $CONF_DBPASS
my $dsn = "DBI:mysql:host=10.1.1.1;database=si08";

my $dbh = DBI->connect ($dsn, $dbuser, $dbpass, {PrintError => 0})
    or die "Connection error: $DBI::errstr ($DBI::err)\n";

my $counter = 0;
#open file
open FILE,"<alert.txt" or die("unable to open file alert");

#loop through the file reading lines
while (<FILE>)
{
    my $nextline = <FILE>;
    chomp ($_);
    my @line1 = split(/ /,$_);
    print "$line1[0], $line1[2], $line1[5], $line1[7], $line1[8], $line1[9], $line1[10]\n";

    #dump read information into database
    my $query = "insert into snortlog (Timestamp, Attack, SrcIP, DesIP, Country, SrcRegion, The
values ('$line1[0]', '$line1[2]', '$line1[5]', '$line1[7]', '$line1[8]', '$line1[9]', '$li

    my $sth = $dbh->prepare($query);
    if (!$sth) { die "Illegal query: $query" };
    #execute MySQL code and finish the command
    $sth -> execute(); # or die (" $DBI::errstr\n");
    $sth -> finish();
}
close(FILE); #close file

$dbh->disconnect;
```

Command Output

```
`perl C:\SI08_Handout\read.pl' returned 0.
20080713061119, , , , ,
```

Ready CP1252 Ln: 15 Col: 2 Perl

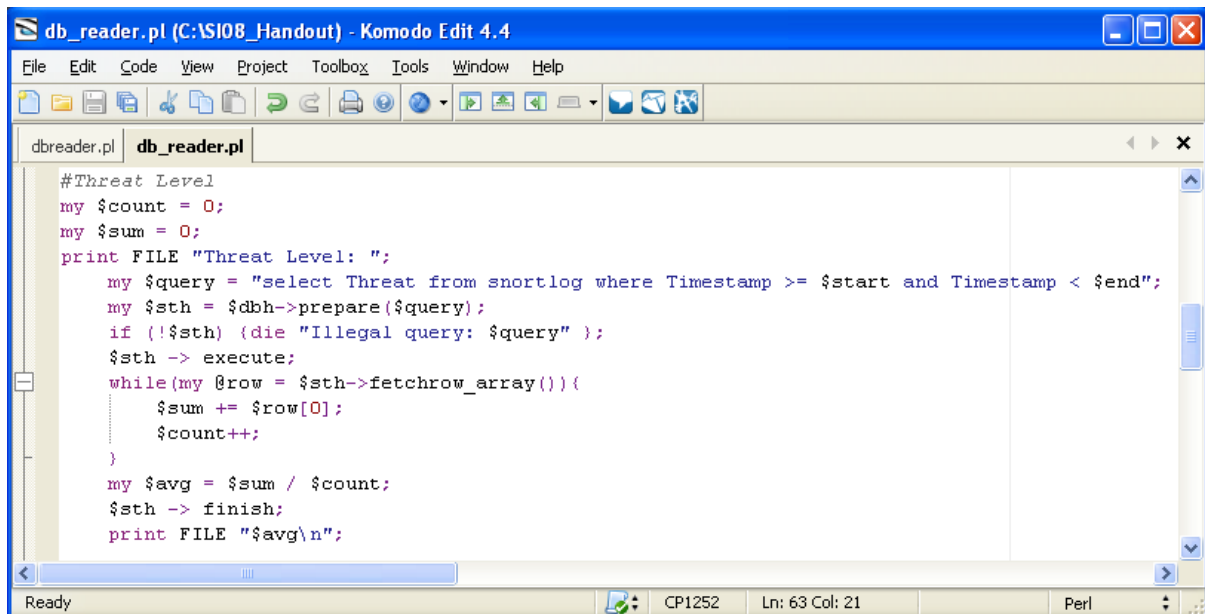
As we can see in the above image, it will create and open the alert.txt file. It will print all the lines which shows the details the txt file.

Then we will execute the insert query to insert the timestamp, attack, source IP address, Destination IP Address, Country and Source Region etc by giving the values. It will insert the required information from each array by taking the values to MySQL database.

And then we will execute the Perl query to execute the above all queries and finish the query, so that it will save all the data into the database then it will close the file and disconnect the database.

**To get the data we have executed some queries to see the data which stored in database:**

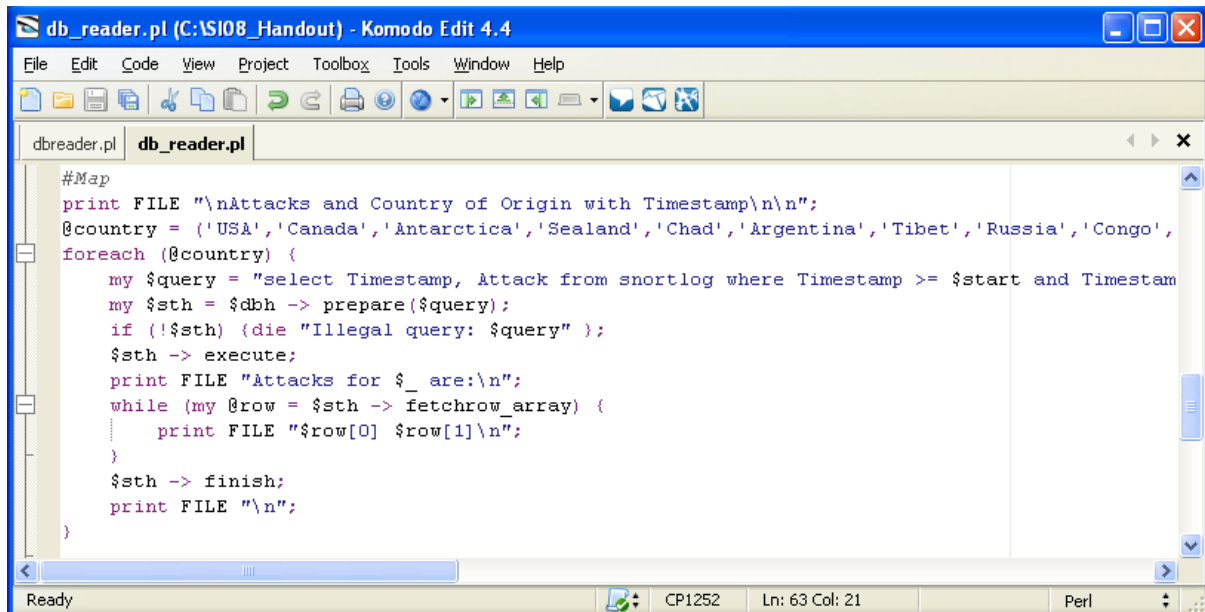
In the following screenshot we have executed the select query to see the data, the query will select Threat column from snort log table where the timestamp should and end in the given format while inserting the columns in insert query.



```
db_reader.pl (C:\SI08_Handout) - Komodo Edit 4.4
File Edit Code View Project Toolbox Tools Window Help
dbreader.pl db_reader.pl
#Threat Level
my $count = 0;
my $sum = 0;
print FILE "Threat Level: ";
my $query = "select Threat from snortlog where Timestamp >= $start and Timestamp < $end";
my $sth = $dbh->prepare($query);
if (!$sth) {die "Illegal query: $query" };
$sth -> execute;
while(my @row = $sth->fetchrow_array()){
    $sum += $row[0];
    $count++;
}
my $avg = $sum / $count;
$sth -> finish;
print FILE "$avg\n";
```

Through Perl it will prepare the query and execute it and finish and print the file of the given query.

The below screenshot will work same as the above but it will also display the Attack column from snortlog.



```
#Map
print FILE "\nAttacks and Country of Origin with Timestamp\n\n";
@country = ('USA', 'Canada', 'Antarctica', 'Sealand', 'Chad', 'Argentina', 'Tibet', 'Russia', 'Congo',
foreach (@country) {
    my $query = "select Timestamp, Attack from snortlog where Timestamp >= $start and Timestamp
my $sth = $dbh -> prepare($query);
if (!$sth) {die "Illegal query: $query" };
$sth -> execute;
print FILE "Attacks for $_ are:\n";
while (my @row = $sth -> fetchrow_array) {
    print FILE "$row[0] $row[1]\n";
}
$sth -> finish;
print FILE "\n";
}
```

The below screenshot will select all the columns from the snortlog table following the where condition.

## Testing we have done:

Checking the program to make sure it is connecting to the MySQL database.

Confirm the queries are working properly.

Verify that data is being written to the .txt file.

## Error we have got:

Queries were not accepted by MySQL.

## Dashboard Aspects

### Threat meter:



- It indicates level of danger of cyber-attacks on
- internet at a given 6 hours period.
- Pointer animated to correspond to threat level
- given by programming team.
- Based off of US terrorist attack threat meter we
- have mentioned the colours according to the threat severity.

### World Map/Clock

- Combination of two high resolution NASA satellite images.
- Large file size created an issue.
- Dots will represent attacks and correspond to graph colours.
- Clock will indicate approximate time of attacks and how many attacks occurred.



### **Security Cam Feed:**

- Security Cam feed will feed of Glenn supercomputer cluster where it has taken the hackers picture or image.
- Then it will change the format of the image so PowerPoint would accept it and insert it in.

### **Compensation:**

After knowing who the hacker is, by law enforcement that compensation can be collected.

### **Conclusion:**

**NOTE:** As we have discussed without law enforcement support, we can't find the hacker by using dashboard aspects.

By working with law enforcement and upstream providers network forensic experts can successfully track and apprehend the attackers.