Signature Based Intrusion Detection System Using SNORT

Vinod Kumar
Research Scholar, School of ICT
Gautam Buddha University

Dr. Om Prakash Sangwan
Faculty, School of ICT
Gautam Buddha University

ABSTRACT

Now a day's Intrusion Detection systems plays very important role in Network security. As the use of internet is growing rapidly the possibility of attack is also increasing in that ratio. People are using signature based IDS's. Snort is mostly used signature based IDS because of it is open source software. World widely it is used in intrusion detection and prevention domain. Basic analysis and security engine (BASE) is also used to see the alerts generated by Snort. In the paper we have implementation the signature based intrusion detection using Snort. Our work will help to novel user to understand the concept of Snort based IDS.

Keywords

Intrusion Detection System, Snort, BASE, TCP Replay.

1. INTRODUCTION

We all know that today we all are dependent on computer technology in any form. As the use of technology is increases, risk associated with technology is also increases. Network security is the big challenge among the researchers. People are working in the field of network security from 1987 when Dorothy Denning published an intrusion detection model [2]. But till now we did not get any perfect solution. There are so many network security tools available such as antivirus, firewall, etc. But they are not able to cover all security risks in the network [11]. The main work of intrusion detection system is to identify the intrusion in the network. And for that it collects important information from the network, process it and if identify attack then alert for the possible attack.

This paper focuses on analyzing the abnormal connection that has been detected by our Intrusion Detection System via Snort when we flow the DARPA Data Set over the network. Intrusion Detection System (IDS) works as a network packet sniffer, which based on comparisons of packet contents with known virus signatures encapsulated as rules, can initiate action and record events and information related to them in a log file and/or database. Snort is a popular NIDS that is used to audit network packets and compare those packets with the database of known attack signature. And Snorts attack signature database can also be updated time by time.

The paper is organized as follows. Section 2 describes the Signature Based Intrusion detection systems in some detail. In section 3 we have discuss about tools that were used in developing IDS system, such as Snort, BASE and TCP Replay. Section 4 describes implementation of Signature Based IDS System. Section 5 describes the process of packet flow over network. Results are presented in section 6. Finally, conclusion and future work is presented in section 7.

2. SIGNATURE BASED IDS SYSTEM

In misuse detection, attacks follow well-defined patterns that exploit system weaknesses and application software. Since these attacks follow well-defined patterns and signatures, they are usually encoded in advance and thereafter used to match against the user behavior. It implies that misuse detection requires specific knowledge of given intrusive behavior. In a signature based detection a predetermined attack patterns in the form of signatures and these signatures are further used to determine the network attacks .They usually examine the network traffic with predefined signatures and each time database is updated. An example of Signature based Intrusion Detection System is SNORT

Advantage [3]:

- There are low false positives as long as attacks are clearly defined in advance.
- Signature-Based Detection is easy to use.

Disadvantage [3]:

However misuse detection systems have number of weaknesses.

- It can be seen that misuse detection requires specific knowledge of intrusive behavior. Collected data before the intrusion could be out of date and yet many times it is hard to detect newer or unknown attacks.
- Misuse detection has a well-known problem of raising alerts regardless of the outcome. For example a window worm trying to attack a Linux system, the misuse IDS will send so many alerts for unsuccessful attacks which may be hard to manage.
- This model may not always be so practical for inside attacks involving abuse of privileges.
- The knowledge about attacks is very dependent on the operating system, version and application hence tied to specific environments.

Working of Signature based IDS

From the figures referred from [12] given below concept of signature based IDS can easily understand. It is clear that when any person sends data inside the network so first of all it goes to server and server check and if found malicious then server discards the packet otherwise send to destination system.

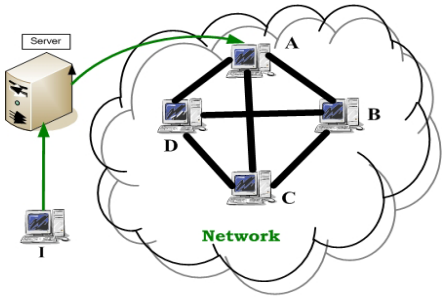

Figure 1: Snort working in network [12]

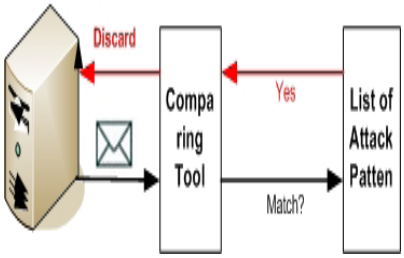

Figure 2: Snort Signature Database [12]

In figure 1 system-I sends packet to system-A but before reaching the packet to destination server checks that packet and if packet is malicious then server discards the packet otherwise send packet to system-A and in figure 2 working of server is clearly mention that how server checks the packet. So, when a packet comes to server then server use comparing tool to check that packet from the database of signature stored in server and if server get result that packet is matched from the database then server discard the packet otherwise server sends the packet to destination system.

3. TOOLS USED IN IMPLEMENTING IDS SYSTEM

To implement Signature based Intrusion detection System; we need to install some network security tools, such as Snort, BASE and TCP Replay.

Snort

Snort is an open source network intrusion detection and prevention system (available at http://www.snort.org/assets/125/snort_manual-_8_5_1.pdf). It can analyze real-time traffic analysis and data flow in network. It is able to check protocol analysis and can detect different type of attack. In NIDS snort basically checks packet against rule written by user. Snort rules can be written in any language, its structure is also good and it can be easily read and rules can be modify also. In buffer overflow attack, snort can detect the attack by matching the previous pattern of attacks and then will take appropriate action to prevent from attack. In signature based IDS system if pattern matches then attack can be easily found but when a new attack comes then system fails but snort overcome this limitation by analyzing the real-time traffic. Whenever any packet comes into network then snort checks the behavior of network if performance degrades of network then snort stop the processing of packet, discards the packet and stores its detail in the signature database.

Component of Snort

Snort is basically the combination of multiple components. All the component work together to find a particular attack and then take the corresponding action that is required for that particular attack. Basically it consists of following major components as shown in figure 3 [12]:

1. Packet Decoder
2. Preprocessors
3. Detection Engine
4. Logging and Alerting System
5. Output Modules

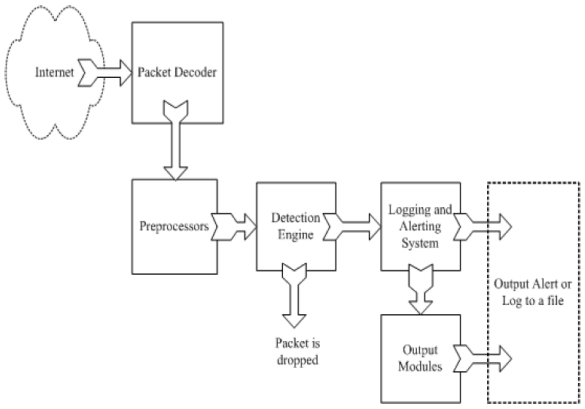

Figure 3: Component of Snort [12]

Packet comes from internet and enters into packet decoder and it goes through several phases, required action is taken by snort at every phase like if detection engine found any miscellaneous content in packet then it drop that packet and in the way towards output module packet is logged in or alert is generated.

Packet decoder

The packet decoder collects packet from different-2 network interfaces and then send to be preprocessor or sent to the detection engine. Network interface might be Ethernet, SLIP, PPP and so on.

Preprocessors

It works with snort to modify or arrange the packet before detection engine to apply some operation on packet if packet is corrupted. Sometimes they also generate alert if any anomalies found in the packet. Basically it matches the pattern of whole string so, by changing the sequence or by adding some extra value intruder can fool the IDS but preprocessor re- arranges the string and IDS can detect the string. Preprocessor does one very important task i.e. defragmentation. Because sometimes intruder break the signature into two parts and send them in two packets so, before checking the signature both packet should be defragmented and only then signature can be found and this is done by preprocessor.

The Detection Engine

Its main work is to find out intrusion activity exits in packet with the help of snort rules and if found then apply appropriate rule otherwise it drops the packet. It takes different time to respond different packet and also depends upon the power of machine and number of rules defines in the system.

Logging and Alerting System

Whatever detection engine finds in the packet, it might generate an alert or used to log activity. All log files are kept by default under /var/log/snort folder and by using –l command line option, location can be changed.

Output Modules

Output modules or plug-ins save output generated by the logging and alerting system of Snort depending on how user wants for different operation. Mainly it controls the different output due to logging and alerting system. Output modules

can do things like the following depending on the configuration:

Simply logging to /var/log/snort/alerts file or some other file
Sending SNMP traps
Sending messages to syslog facility
Can Generate XML output
SMB messages to Microsoft Windows-based machines

Rule structure of snort

Basically rules are created by known intrusion signature system. It is divided into two parts: rule header and rule option and rules can be modifying according to need.
Rule header follows this pattern: Action + protocol + source address+ S-port + direction + destination address + D-port
Alert ip any any -> any any (msg : "IP Packet Detected ";)

Rule header Rule Option

Snort's uses [9]

Snort basically used in three categories

1. A packet sniffer

In its simplest form, snort is a packet sniffer. That said, its the easiest way to start.

snort -d -e -v
-v Put Snort in packet-sniffing mode (TCP headers only)
-d Include all network layer headers (TCP, UDP, and ICMP)
-e Include the data link layer headers

2. Packet logger

Snort has built-in packet-logging mechanisms that you can use to collect the data as a file, sort it into directories, or store the data as a binary file.
snort -dev -l {logging-directory} -h {home-subnet-slash-notation}
If you wanted to log the data into the directory /var/adm/snort/logs with the home subnet 192.20.14.0/24, you would use the following:
snort -dev -l /var/adm/snort/logs -h 192.20.14.0/24
for logging in binary format, don't need all options. The binary format makes packet collection much faster for Snort, because Snort doesn't have to translate the data into human-readable format immediately.
snort -b -L {log-file}

for reading the log file
snort [-d|e] -r {log-file} [tcp|udp|icmp]
Here last item in line is optional, because if you want to filter the packets based on packet type like tcp , udp or icmp.

3. As a Network Intrusion Detection System

To make Snort an IDS, just add one thing to the packet-logging function: the configuration file.

snort -dev -l /var/adm/snort/logs -h 192.20.14.0/24 -c /root/mysnort.conf

Basic Analysis and Security Engine (BASE)[1]

BASE is the Basic Analysis and Security Engine. It is based on the code from the Analysis Console for Intrusion Databases (ACID) project. This application provides a web front-end to query and analyze the alerts coming from a SNORT IDS system.

BASE is a web interface to perform analysis of intrusions that snort has detected on your network. It uses a user authentication and role-base system; so that you as the security admin can decide what and how much information each user can see. It also has a simple to use, web-based setup program for people not comfortable with editing files directly [1].
BASE is PHP based analysis engine for managing a database of security events. These events can be from IDS's (such as Snort) as well as from firewall, network monitoring tools and even pcap files.

Features:

- An interface for database searching and query building. Searches can be performed between network specific parameter such as the attackers internet protocol address, by meta parameters such as time or date of an event, or by triggered rules.
- A packet browser that can decode and display layer 3 and layer 4 information from the logged packet.
- Data management capabilities, including grouping of alerts (so that it is possible to group all events related to an intrusion incident), alert deletion and archiving and importing to e-mail messages.
- Generation of various graphical charts and statistics based on specified parameters.

BASE has so many interesting features that are very helpful in intrusion detection system. It is a graphical interface written in PHP and it used to display the logs generated by Snort and sent into the database. We can see the logs according to date for example if we want to see the logs in Particular date then we have to give the date in BASE and it will show the all logs for that date. We can see the logs according to destination and source IP and for that we have to set the value of Source IP and Destination IP and it will show all logs for I P's. One more interesting feature is that we can create the group, once group will create then logs will be stored in respective groups and we can also delete the group when it is no required.

BASE provide the functionality using that we can see most recent alerts, today's alerts or we can see alerts within a time duration. For example, if we want to see the alerts from last 24 hours then it will show alerts generated within 24 hours. We can check the Most Frequent Source Ports, Most Frequent Destination Ports and even most frequent addresses. Like if we want to see the last 10 most frequent Addresses then it will show most frequent Source IP and destination IP address.

TCP Replay [11]

TCP Replay is a suite of utilities for Unix system for editing and replaying network traffic, which was previously captured by tools like tcpdump and ethernal/wireshark.
It provides ability to classify traffic as a client or server, edit packets at layer 2-4 and replay the traffic at arbitrary speed onto a network for sniffing or through a device.
It takes a pcap file and replays it as is. If you have one flow between two IP Addresses, it will replay that. If you have 1,000 flows between 1000 client/server, it will do that too. It does not decode the packet at any level, so it does not really care how many IP Addresses are in the pcap.

TCP Prep[11]

Basically TCP Prep is used to generate a cache file, which is used to split the traffic into two parts (client/server). If you want to use TCP replay with two NIC's, then TCP Prep will decide that which interface each packet will use.
It supports multiple mode of operation. Each mode uses different strategy to divide the traffic. For example if you want to pass traffic through a router, then router mode is best. There are number of modes available:

Auto/Bridge
Auto/Router
Auto/Client
Auto/Server
IPv4/v6 matching CIDR
IPv4/v6 matching Regex
TCP/UDP Port
MAC address

TCP Writer[11]

TCP Writer is used to edit the packet and for that we have to provide it an input pcap file and the name of output pcap file. It works on layer 2-4.

4. IMPLEMENTATION OF SIGNATURE BASED IDS

We start by designing a conceptual framework of a signature based intrusion detection system. The frameworks will show the flow of packet into the network. Here we will flow data using TCP Replay within two systems inside the network. And then we will check the outcome in graphical form using Basic Analysis and Security Engine.

Data Collection and Analysis

This work was done on open source intrusion detection system. Snort was configured to log the traffic flowing into Lab network from 192.20.14.50 to 192.20.14.48. Then collected data is used to see the relevance of an IDS system on to the protected network. And we used Snort because:

- Snort is an open source intrusion detection system. It is therefore useful where it is not cost efficient to apply NIDS sensors.
- Snort is lightweight application. It is also economical when it comes to resource utilization.
- Snort can be used as a intrusion detection as well as intrusion prevention system.
- Snorts rule can be changed if needed. Its rules are flexible. Snort has more than 2500 rules in its database [5]. And people can modify rule according to need of their network need.
- Snort is available for Linux as well as for Windows.
- It is most widely used for intrusion detection in network.

The Network Setup

Intrusion detection system can be deployed to protect the network. It can be deployed between to hosts, between two switches or even the server firms. In our work we will place snort between two hosts.

Configuration and Validation of the IDS

We are using Linux box running debian operating system to detect intrusion into our system placed inside the network. Whenever any intrusion will be detected by Snort, it will generate an alert. And if system successfully generates an alert then that means network will have been well configured and traffic monitoring is taking place.

Installation of Snort, PostgreSQL and BASE

In Debian operating system, configuration are made for snort-pgsql, Basic Analysis and Security Engine(BASE) to provide a user friendly web front end to simplify querying and analysis of alerts, PostgreSQL database that is an open source Relational Database Management System (RDBMS), Apache a widely available http server that supports PHP languages, Secure Shell(SSL) to enable secure remote login into the network, and PHP a hyper text preprocessor enables creation of dynamic content and interaction with databases.

Snort-pgsql can be downloaded from the **http://www.snort.org.** It can be downloaded either in compiled or recompiled state. We are using the version snort-pgsql_2.8.5.2-8_i386.deb

Installation process [4]

1. Download snort-pgsql_2.8.5.2-8_i386.deb from Snorts official website given above

Install in terminal using command-
dpkg -i snort-pgsql_2.8.5.2-8_i386.deb

2. Go to synaptic and install

postgresql-8.4(server)
postgresql-common
postgresql-client-common
postgresql-client-8.4

3. Creating snort database

# su postgres
$ createdb database_name(snortdb)
$ zcat /usr/share/doc/snort-pgsql/create_postgresql.gz | psql snortdb(database_name)
$ createuser -P user_name(snortuser)

Enter password for new user: snort-password

Enter it again: snort-password
Shall the new user be a superuser? (y/n) n
Shall the new user be allowed to create databases? (y/n) n
Shall the new user be allowed to create more new users? (y/n) n

CREATE USER

4. log in to database

$ psql snort

5. Grant all privileges to snort user on every table and sequence

psql>grant all privileges on database snortdb to snortuser;
psql>GRANT ALL ON TABLE data, detail, encoding, event, icmphdr, iphdr, opt, reference, reference_ref_id_seq, reference_system, reference_system_ref_system_id_seq, schema, sensor, sensor_sid_seq, sig_class, sig_class_sig_class_id_seq, sig_reference, signature, signature_sig_id_seq, tcphdr, udphdr TO snortuser;

6. Edit database.conf file

getit /etc/snort/database.conf
add a line

output database: alert, postgresql, user=snortuser password=snort-password dbname=snortdb host=postgresql-host-ip

7. Edit snort.conf file

getit /etc/snort/snort.conf
After the line that reads:
#var HOME_NET any

Add line

var HOME_NET host-ip-address

8. postgresql configuration

gedit /etc/postgresql/8.4/main/postgresql.conf
Search for the line that has the listen_address directive and set it to the IP address of the host running postgresql (un-comment it if necessary):
listen_addresses = postgresql-host-ip

gedit /etc/postgresql/8.4/main/pg_hba.conf
After the line that reads:
host all all 127.0.0.1/32 md5
Add following line:
host snortdb snortuser snort-sensor-host-ip/32 password

9. Restart postgresql to apply the previous changes

/etc/init.d/postgresql restart

10. Snort configuration

Start snort in interactive mode, using interface eth0 (just to check everything works as expected):
snort -i eth0 -c /etc/snort/snort.conf
To check all the needed services are running you can execute:
ps -ef |grep <SERVICE>
where <SERVICE> is snort, apache, postgresql, etc.
Test if the database is logging alerts, send some suspicious traffic to the snort sensor host (for example, using nmap or nessus):

# su postgres
$ psql snort -c "select count (*) from event"
You should get a growing value each time you send more suspicious traffic and execute the SQL query.

11. Installing BASE Pre-Requisites

Install Apache 2, PHP (version 4 in the examples shown below, but you can use PHP 5 as well), the PHP GD extension and the PGP adodb library. There are many configuration options whose specifics are best addressed by the appropriate package's documentation.
apt-get install apache2 libapache2-mod-php4 php4-gd php4-pgsql libphp-adodb
Create a file called test.php under /var/www/ and write:
<?php
phpinfo();
?>
Make sure that the following lines are included in /etc/php4/apache2/php.ini and un-commented:

Extension = pgsql.so
extension = gd.so

Restart Apache 2 to enable the newly installed PHP extensions:
/etc/init.d/apache2 restart
Now use your web browser to look at the URL http://web-server-ip-address/test.php. It should give you info about your system, Apache and PHP, postgres, gd.

12. Installing and Configuring BASE

Download BASE from *http://sourceforge.net/projects/secureideas*. At the moment of writing this, 1.2 is the most up to date version. Execute the following commands as root to put BASE under /var/www/base:

mv base-1.2.tar.gz /var/www/
cd /var/www/
tar xvzf base-1.2.tar.gz
rm base-1.2.tar.gz
mv base-1.2 base
cd /var/www/base

The file base_conf.php.dist needs to be copied to base_conf.php (just in case you do something wrong; you can always start from the original copy):

cp base_conf.php.dist base_conf.php # vi base_conf.php
Next we need to adjust a few variables (you can have a look at the rest of the file to tweak other configuration values):
If you would like to use the user authentication
system. Remember to add a user before setting it to 1!

$Use_Auth_System = 1;(it will ask for Login and Password)
$Use_Auth_System = 0;(BASE will open directly in browser)
$BASE_urlpath = '/base';
$DBlib_path = '/usr/share/php/adodb';
$DBtype = 'postgres';
$alert_dbname = 'snortdb';
$alert_host = 'postresql-host-ip';
$alert_port = '';
$alert_user = 'snortuser';
$alert_password = 'snort-password';

We don't have an archive db, so set this to 0
$archive_exists = 0;
Open the base_main.php page in a browser. If the any database changes are required, BASE will prompt for action. Click on the "Setup page" link to be brought to the DB configuration page (base_db_setup.php).

This next page will facilitate the creation of the necessary tables. Click on the "Create BASE AG" buttons as seen below. BASE tables Adds tables to extend the Snort DB to [Create BASE AG] support the BASE functionality
If you do not have PEAR::Image_Graph installed, install it using:
apt-get install php-image-graph
PEAR::Image_Color is needed but it's not packaged in Ubuntu 6.0.6, so you need to download it from http://pear.php.net/package/Image_Color/download and install it under /usr/share/php/Image/. You can do this by executing:
first set the proxy using command:
pear config-set http_proxy
http://Login:Password@192.20.4.254:80
apt-get install php4-pear
pear install Image_Color

At the time of writing this howto, there is a bug in /var/www/base/base_qry_common.php that prevents the graphs from being displayed. You will need to remove the empty line after the '?>' line.

Here installation process of setup is done.

5. THE PACKET FLOW OVER NETWORK

For flow the traffic over network, first of all snort should be in running mode and after that we can send the traffic from one host to another by using TCP Replay. We can also send packet using snort and can check the alerts in Basic Analysis and Security Engine (BASE). We can flow the traffic by two methodologies given below.

5.1 TCP Replay

It is suite of utilities for Unix system for editing and replacing network traffic, which was previously captured by tools like tcpdump and ethernal/wiershark.
It provides the ability to classify traffic as a client or server, edit packets at layer 2-4 and replay the traffic at arbitrary speed onto a network for sniffing through a device.
There is a three step process for this:

1. Determine which packets are client->server and server->client
2. Rewrite IP addresses based on their direction
3. Send packets through inline device

Step 1: Use tcpprep to split traffic based on the source/destination port:

$ tcpprep --port --cachefile=example.cache --pcap=example.pcap
In this case, all the packets directed to a TCP or UDP port < 1024 are considered client->server, while other packets are server->client. This information is stored in a tcpprep cache file called *example.cache* for later use.

Step 2: Use tcprewrite to change the IP addresses to the local network:

$ tcprewrite –endpoints=192.29.14.50:192.20.14.48 --cachefile=example.cache --infile=example.pcap --outfile=new.pcap
Here, we want all traffic to appear to be between two hosts: 192.29.14.50 and 192.20.14.48. We want one IP to be the "client" and the other IP the "server", so we use the cache file created in the last step

Step 3
Use tcpreplay to send the traffic through the IPS:

tcpreplay --intf1=eth0 --intf2=eth1 --cachefile=example.cache new.pcap
Here we send the traffic. Since we want to split traffic between two interfaces (eth0 and eth1), we use the cache file created in Step #1 with the *new.pcap* created in Step #2. We can use the cache file for different pcap files because while the IP addresses of the packets have changed, their order and semantics have not.

5.2 Using snort

In this method we just pass the name of tcpdump file and alerts can directly be seen in the Basic Analysis and Security Engine (BASE).
$ snort --pcap-single=outside.tcpdump -c /etc/snort/snort.conf
Where outside.tcpdump is testing DARPA dataset. This is used for generating alerts in BASE.

6. RESULTS

This paragraph describes a small application used to validate the ability of signature based intrusion detection system to detect intrusion. From the figures given below we can understand the use of BASE. Figure 4 is showing the total number of captured packet over network, figure 5 is showing total number of unique alerts captured in whole day and figure 6 is showing the alerts for particular source IP address. There are much more options available in BASE which can be used according to need.

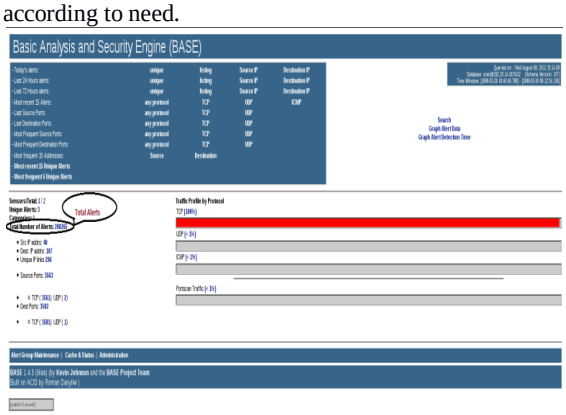

Figure 4: Results generated by Snort on Basic Analysis and Security Engine home page

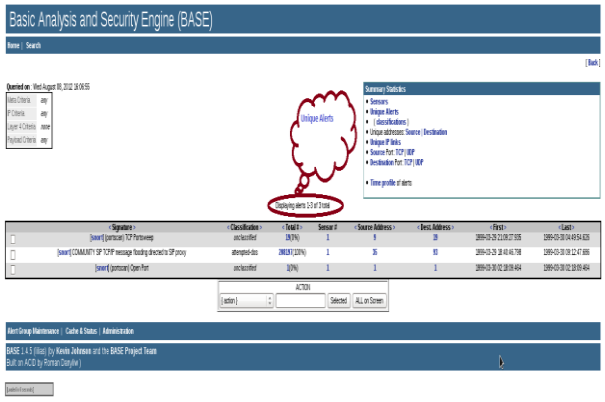

Figure 5: Results for unique alerts

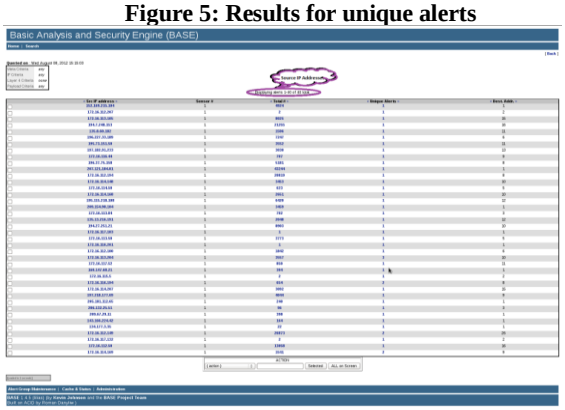

Figure 6: Results for source IP addresses

7. Conclusion

This paper proposes the implementation process of Snort in Debian. This IDS System demonstrated that it can detect and analyze the intrusion in real time network traffic. Once the Snort will identify any intrusion then it will send alert to security person and security person will take required action immediately. The future work is to develop a prototype model to filter, delete and quarantine the intrusion attack automatically in real time network

8. REFERENCES

[1] Basic Analysis and Security Engine (BASE) project (2012). Available: http://base.secureideas.net/about.php.

[2] D. E. Denning. "An Intrusion-Detection Model". IEEE transactions on software engineering, Volume : 13 Issue: 2, February 1987.

[3] Guan Xin and Li Yun-jie, "A new Intrusion Prevention Attack System Model based on Immune Principle", International Conference on e-Business and Information System Security (EBISS), in IEEE, pp. 1-4, 2010.

[4] HowtoForge (2012), "Installation manual of Snort", available: http://www.howtoforge.com/intrusion_detection_snort_base_postgresql_ubuntu6.06.

[5] Hwang,K., Cai,M., Chen,Y and Qin,M. , "Hybrid Intrusion Detection with Weighted Signature Generation over Anomalous Internet Episodes", IEEE Transactions on Dependable Computing, Volume: 4 Issue: 1, pp. 41-55, 2007.

[6] J.P. Anderson, "Computer security technology planning study". Technical Report, ESDTR-73-51, United States Air Force, Electronic Systems Division, October 1972.

[7] J.P. Anderson, "Computer Security Threat Monitoring and Surveillance". Technical Report, James P. Anderson Company, Fort Washington, Pennsylvania, April 1980.

[8] Jay Beale*(2007),* "Snort: IDS and IPS Toolkit", available*:* http://ebookee.org/Snort-IDS-and-IPS-Toolkit-Jay-Beale-s-Open-Source-Security-Repost-_412527.html.

[9] Martin Roesch (2009), "Snort User Manual 2.8.5", available: http://www.snort.org/assets/125/snort_manual-2_8_5_1.pdf.

[10] MIT Lincon Laboratory (1999), "1999 DARPA Intrusion Detection Evaluation Data Set", available: http://www.ll.mit.edu/mission/communications/ist/corpora/ideval/data/1999data.html.

[11] Tcpreplay Pcap editing & replay tools for *NIX (2010). Available: http://tcpreplay.synfin.net/wiki/manual#a3.xOnlineManual.

[12] Vinod Kumar**,** Vinay Pathak, Dr. Om Prakash Sangwan**, "**Evaluation of Buffer Overflow and NIDPS", International Journal on Computer Science and Emerging Trends (IJCSET), August issue, 2012.