Design of a Hybrid Intrusion Detection System using Snort and Hadoop

Prathibha.P.G
P G Scholar
Government Engineering College
Thrissur, Kerala, India

Dileesh.E.D
Assistant Professor
Government Engineering College
Thrissur, Kerala, India

ABSTRACT

Security is the most important issue that is to be considered in any environment. Any attack can be launched from any node. Any of these attacks should be identified and subsequent actions should be taken to avoid further consequences. An intrusion detection system helps in identifying the attacks at the early stage and give alarms. These intrusion detection systems should be able to identify almost any kind of attacks, be it a newly launched one or a pre-established one. In this work, the intrusion detection system Snort is made use of .In this work, the packets captured by Snort is analyzed by the Grid computing framework Hadoop, which is used for Big Data Analysis. For more user friendlier analysis a data warehouse system for Hadoop, Hive is also provided. For those ip addresses that generate large number of packets, Snort rules will be generated so that when the number of packets from a particular source exceeds a number, the node will generate alerts to other nodes since there is a possibility of attack.

General Terms

Security, Big-Data Analysis

Keywords

Hadoop, Hive, MapReduce, Snort, RulesKeyword.

1. INTRODUCTION

The most important issue that is to be given greatest consideration is the security of an environment. Be it a single host or a LAN or any complex environment like Grid or Cloud attacks are always there. It can be attacks on a single host, port scans to check vulnerabilities, flooding attacks, denial of service etc. All these attacks have severe consequences in an environment. Therefore it is good to identify these attacks at any early stage itself, so that the attacker can be blocked and avoid further effects. This is possible by an intrusion detection system (ids), which can identify the intrusions before attack can take place and can give a notification that it is possible to have an attack

Most of the ids identify attacks at an early stage itself. There are several open source ids present. Some of them are Snort, Bro, and Suricata etc. They are very strong and efficient in identifying attacks. Most of them identify pre-defined attacks. These kind of intrusion detection systems are called as Signature based Intrusion Detection Systems. Signature based ids have a set of rules. The incoming packets are compared with the set of rules. If any of the packets matches with the set of rules, actions specified in the corresponding rules are performed. Therefore by writing a wide variety of rules one can detect any attack with these kind of intrusion detection systems.

The main disadvantage of Signature based intrusion detection system is that it is unable to identify unknown attacks. This kind of intrusion detection systems work with rules. Therefore those attacks those are not present as rule cannot be detected. Such kind of attacks are identified by Anomaly based Intrusion Detection Systems. Anomaly based ids, as the name suggests typically works on any anomaly .i.e. any deviation from the normal pattern is treated as anomaly and such anomalies will be classified as attacks. Thus it will be able to identify newly launched ones also. The main disadvantage of this system is the generation of large number of false attacks.

In this work, the famous open source intrusion detection system Snort is made use of. Snort is basically a Signature based ids which typically works on set of rules .When the incoming packets matches with a set of rules, corresponding actions specified in the rules will be performed. Snort can identify only those attacks that are present in rules. For better performance it should be able to identify unknown attacks also. To identify unknown attacks, any unknown behavior should be first known, then corresponding rules must be written to rule database, so that again if such attacks occur the node will perform the action specified in the rule.

In this work, the count of the number of packets is analyzed. This analysis is done by Hadoop. Snort is worked in one of its modes called as packet logger mode and the packets obtained are given for analysis to Hadoop. Hadoop analyses the set of packets and corresponding customized Snort rules are generated if number of packets exceeds a count than the normal one.

For more user friendliness the work is supplemented by the data-ware house system called as Hive. In short the work gives an integration of Snort with Hadoop and Hive.

The remaining paper consist of related works, description of Snort, Hadoop, Hive, about the Overview and Proposed Architecture along with implementation details and results

2. RELATED WORKS

There are several works related to analysis in Hadoop. Wei- Yu Chen et al. [1] developed a Snort log analysis system using Hadoop. They also proposed a map reduce algorithm for merging alerts obtained from Snort. Y.Lee et.al. [2] Developed a Hadoop based packet trace tool. In this paper for handling the packet traces a new binary input format called Pcap Input-Format was developed. For this tool, several map reduce jobs were modeled and they compared their work with famous packet processing tool Coral Reef. Y. Lee et al. [3] also propose an Internet traffic analysis with Map Reduce. They collected

packets with Net Flow and trimmed them down with suitable map reduce algorithm

There are several research works being done to impart better performance to Snort. X.Fang [4] did a work for integrating artificial intelligence to Snort using Elmann Neural Network. The improved Snort was able to differentiate between normal traffic and malicious one. B.E.Lavender [5] did a thesis work for integrating Genetic Algorithms to Snort. This work also improved the performance of Snort. Gomez et al.[6] developed a Hybrid IDS based on Snort. In this anomaly detection pre-processor was added to Snort which has got access to MySQL. The work was also complemented with a website for monitoring the administration.

3. SNORT

Snort is a libpcap- based packet sniffer and logger that can be used as a lightweight network intrusion detection system (NIDS). It features rules based logging to perform content pattern matching and detect a variety of attacks and probes, such as buffer overflows , stealth port scans, CGI attacks, SMB probes, and much more. Snort has real-time alerting capability, with alerts being sent to syslog, databases or a separate "alert" file [7]. Snort works on network layer protocol such as ip and transport layer protocols TCP and UDP.

Snort is an open source Signature based IDS. There are predefined rules for finding attacks. Snort rules are provided by Sourcefire and can be downloaded by registered users for their use. To include rules one has to change the configuration file.

When a packet comes, rules are matched with the packet. If matching occurs with any of the rules, actions specified in the rule will be performed. The actions can be of four types. It can be to alert the nodes, log the incoming matched packet, pass the packet without any action or to drop the packet.

There are three subsystems for Snort. They are packet decoder, detection engine and the logging and alerting subsystem. The decoder as the name suggests decodes the incoming packets to the network .The subroutines present in the decoder overlays data structures on the network traffic. These sub-routines which are responsible for decoding are called in the order through protocol stack from the data link layer up to the application layer. The detection engine maintains the detection rules in a two dimensional linked list. These rule chains are searched for each incoming packet in both directions. When a rule in the detection engine matches the incoming packet, the action specified in the definition of rule, which will be either to alert or to log the packets.

Snort rules are very simple .Snort rules have the following format action protocol source_ip source_port -> destination_ip destination_port (options)[7]. An example of a Snort rule is alert tcp any any >any any (msg:"Possibility of an attack"), which means Snort should alert any ip address if TCP packets comes from any source port or destination port. The logging and alerting subsystem can be selected by commands. In the logging subsystem, the packets will be logged to a subdirectory in a human readable format or in a tcpdump binary format. Alerting subsystem send alerts to normal text file or a database like MySql or to Syslog. According to different options either the alerts can display the entire packet information or condensed subset of the header information to the alert file.

Snort works in three different modes, sniffer mode, packet logger mode and network intrusion detection mode. Sniffer mode detects the incoming packets and displays them in console. In packet logger mode, Snort collects the packets and logs them to disk. In network intrusion detection system (NIDS) mode alerts will be generated. Alerts can be generated in different ways. It can be logged or can be displayed to console. Alerts can also be generated in such a way that it will display only useful information or will display the entire header information. All these options can be chosen by suitable Snort commands. In this work, the incoming packets are captured for analysis by running Snort in packet logger mode.

4. HADOOP

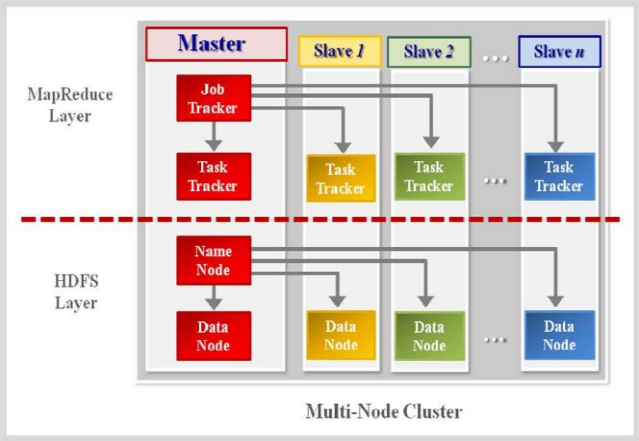


Fig 1:Hadoop Architecture

Figure 1 shows the Hadoop's architecture. Hadoop is for Big Data Analysis. Hadoop is basically a framework for running applications on large clusters .It enables thousands of nodes to work together in parallel for doing a single job. Hadoop is for analyzing peta bytes of data in a very short span of time. As the number of nodes increases, the time taken to process the data decreases. Hadoop is written in Java and client applications can be written in Java, Python, and Ruby etc. Hadoop is used in batch data processing and for highly parallel data intensive applications

The two important features of Hadoop are Distributed Storage and Distributed processing [8].Distributed Storage is given by Hadoop Distributed File System (HDFS) and Distributed Processing is done by the concept known as Map Reduce. These are the backbones of Hadoop architecture.

A small Hadoop cluster will include a single master and multiple worker nodes (slaves). The master node consists of a JobTracker, TaskTracker, NameNode and DataNode. A slave or worker node acts as both a Data Node and TaskTracker. JobTracker and

TaskTrackers are responsible for doing the mapreduce jobs. Name nodes and Datanodes are the part of the distributed file system.

4.1 Hadoop Distributed File System (HDFS)

HDFS is a distributed, scalable, and portable file system written in Java for the Hadoop framework. Files are divided into large blocks and are distributed across the cluster. Typical block size is 64 MB. Block sizes can be changed. To handle failures, blocks will be replicated by appropriate replication factor.

Each node in a Hadoop instance typically has a single Data Node. HDFS cluster is formed by a cluster of Data Nodes. A Name Node performs various file system operation such as

close, open rename etc. Name Node also manages block replication. Data Nodes run operations such as read and write that file system clients require. A file is divided into more than a block that is stored in a Data Node and HDFS determines mapping between Data Nodes and blocks [1]

4.2 MapReduce

Map Reduce is a software framework for writing applications which process vast amounts of data in-parallel on large clusters having thousands of nodes in a reliable, fault-tolerant manner. The MapReduce engine which is placed above the file system consists of one JobTracker. Client applications submit MapReduce jobs to this JobTracker. The JobTracker pushes work out to available TaskTracker. One of the most highlighted feature of Hadoop is that is rack-aware. With rack-aware file system, the JobTracker knows which node contains the data, and which other machines are nearby. If node fails, work is given to nodes in the same rack. This reduces network traffic on the main backbone network.

By MapReduce programming model many complex data intensive jobs can be made into simple ones and can perform distributed concurrent processing. A developer only need to know how to write appropriate map and reduce functions.

In the MapReduce programming model used in Hadoop, the computation takes a set of input key/value pairs, and produces a set of out-put key/value pairs. Map and Reduce are two basic functions in the MapReduce computation. Map's idea is to break the given input which is a big one into smaller ones. Suitable jobs are performed on these smaller data parallel. Reducer's merge these parallel processed data and produce a single output.

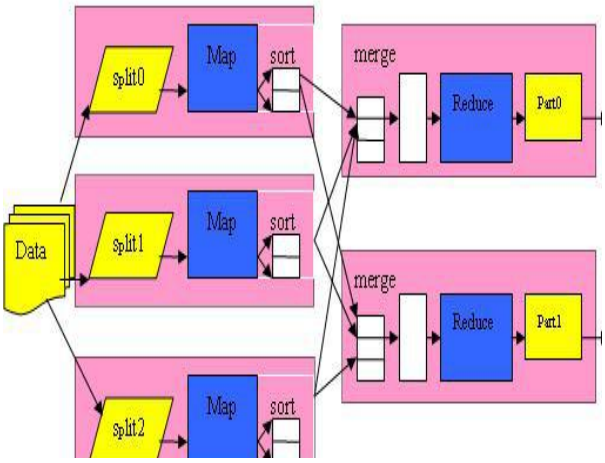


Fig 2: MapReduce Model

Figure 2 shows the mapreduce model. Users write Map that takes an input pair and produces intermediate key/value pairs. The Hadoop MapReduce library will group the intermediate values according to the same key. Reduce that is also written by users will merge the intermediate values for smaller values [8].

5. HIVE

Hive is a data warehouse system for Hadoop that facilitates easy data summarization, ad-hoc queries, and the analysis of large datasets stored in Hadoop compatible file systems. Large data can be queried and analyzed by SQL like language HiveQL. Map Reduce programs can also be written for querying [9]

As mentioned earlier Hive has three main functions: data summarization, query and analysis. Queries expressed in HiveQL, gets automatically translated into Map Reduce jobs executed on Hadoop. The main advantage of Hive is that its user-friendliness. Even after big-data analysis, the result will be bigger for a human to manipulate faster. There is where Hive comes into. The result of big data analysis can be loaded to HDFS and through Hive one can query for better outputs in a short span of time.

Three types of files Text Files, Sequence File and Record columnar files are supported by Hive. By default metadata is stored in the embedded database derby. Other relational databases such as MySQL can be also added to store the Meta data.

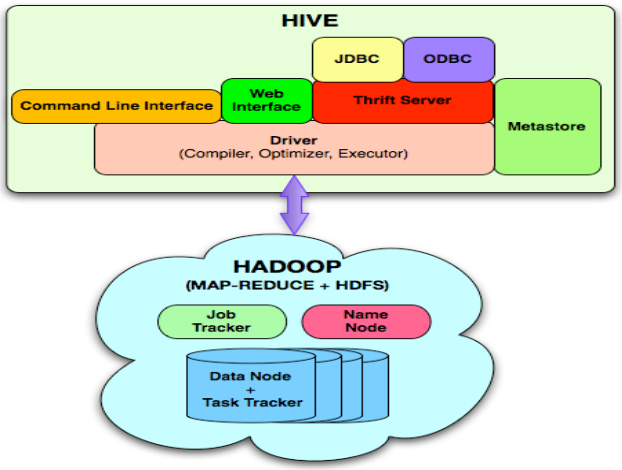


Fig 3:Hive Architecture

Figure 3 shows architecture of Hive. A HiveQL statement is submitted via the CLI, the web UI or an external client using the thrift, odbc or jdbc interfaces. The driver, which manages the HiveQL life cycle first passes the query to the compiler where it goes through the typical parse, type check and semantic analysis phases, using the metadata stored in the Metastore, where Metastore is the component which stores about columns, partitions etc. The compiler generates a logical plan that is then optimized through a simple rule based optimizer. Finally an optimized plan in the form of a DAG of map-reduce tasks and hdfs tasks is generated. The execution engine then executes these tasks in the order of their dependencies, using Hadoop [8]

6. OVERVIEW

The main purpose of this work is to integrate Snort with Hadoop. Snort as mentioned earlier is a Signature based system. For better performance, it should be able to detect unknown attacks also. Several works are done to include a hybrid nature to Snort by including both the capabilities of Signature based and Anomaly based. For including the Anomaly based features several classification algorithms have been used.

In this work, simple mapreduce is being used. Sending large number of packets from a source can be considered as an attack. This idea is made use in this work. There are several kind of attacks possible in the network like ICMP attack, ping of death attack, Smurf attack, UDP attack and others. All these launch attacks by sending a large number of packets to the particular victim. Therefore this should be identified and the packets from the corresponding source should be blocked.

In this work, for analyzing the packets Hadoop is made use of. If a source sends packets in normal intention, the number of packets will be very less say about the thousands range. But if lakhs of packets are seen from a particular source is seen, it can be doubted as an attack. This is actually a deviation from the normal behavior or anomaly. The work can also be used to find whether there is port scan attacks .i.e. the numbers of packets send from a particular source ip and particular port to destination ip and particular destination ports. So if there are large numbers of packets send to a particular port it can be a possibility of portscan attack.

By identifying such anomalies, suitable Snort rules should be written so that it can alert other nodes or can drop the packets coming into. Thus snort will be having the capability of both Signature based and Anomaly based.

6.1 Proposed Architecture

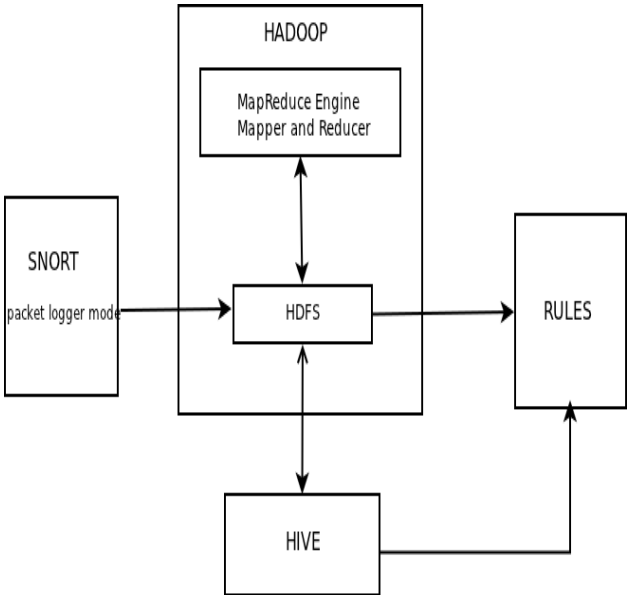


Fig 4:Proposed Architecture

Figure 4 shows the proposed architecture for the work. Packets are collected by running Snort in packet logger mode. Each of the packets contain several features including timestamp, protocol, source ip, destination ip, port numbers, protocol or type of packet and other fields relevant to type of packet such as flags etc.

There will be large number of packets logged in local disk all the files will be either in tcpdump format or in binary format. Therefore some preprocessing should be done to make it into a readable format. This is done by suitable Snort commands.

For analysis by Hadoop these files should be loaded to the file system HDFS. The MapReduce engine located above HDFS is the analysis part of the Hadoop. The mapper and reducer extracts the most important features of the packets such as the source ip , destination ip, type of packet, port numbers etc. Also it gives the number of packets from a particular source to a destination of specific type.

For managing bigdata the data warehouse system for Hadoop, Hive is made used of. Even after analysis, the file may be large. So for better results, user can query about the necessary details which gives output in a very short span of time

After identifying the source ip, destination ip Snort rules can be generated if there are large numbers of packets send from a source to a particular destination. Therefore when further such packet comes, snort rules will be matched and the actions performed in the snort rules will take place

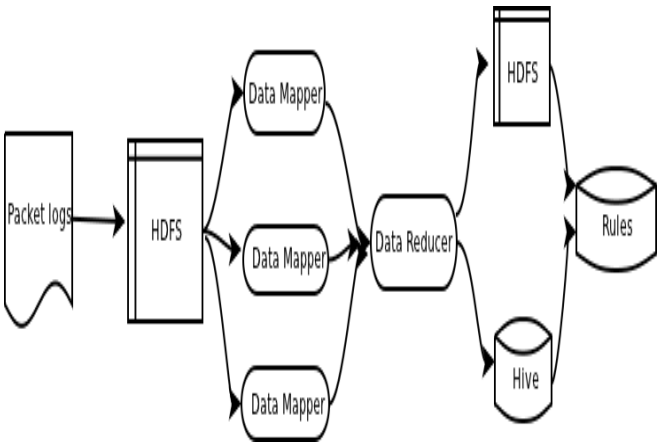


Fig 5:Flow of data

Fig. 5 show the flow of data. Initially the data, the packets will be stored as logs. They are loaded to HDFS. Then depending on the number of files, there will be mappers present. These mappers produce an intermediate result, which is taken up by the reducers. The reducers sorts and merges the output. The resultant data is given to Hive and HDFS from which rules are formed

6.2 Packet Analysis with MapReduce

For analyzing packets obtained from Snort, MapReduce is made of use. There are several tools like Wireshark, tshark etc for displaying the packets. But when it comes to large number of packets say petabytes and terabytes, these tools don't contribute much. Hadoop is the best framework for doing such work.

Clients can write suitable map and reduce function for the particular task. As mentioned earlier MapReduce framework requires a key value pair. In this work key is the source ip , destination ip address and the protocol and the value is count. Mapper part gets the keys and the value. Reducer part shuffles, sorts and merges and gets the output as the number of packets of specific type from a particular source to destination. It can also be used to find the number of packets send to specific ports also

6.3 Querying with Hive

Hive as mentioned earlier is for data summarization and adhoc querying. The main feature of hive is the simplicity and SQL like queries. Tables are created which stores the addresses protocol and count. The output of the mapreduce analysis will be loaded from HDFS to Hive. Almost all queries are same as SQL and such queries in Hive is named as HiveQL. Select queries with conditions will be performed as mapreduce jobs and output will be shown along with the time to complete the job. The output of the query can be written to files.

Hive is added mainly for user friendly analysis and for better management of data. This output will be present in HDFS which can be loaded to the local file system. In this work for better analysis the output of mapreduce analysis is loaded to Hive. With Hive, the person can query about the packets with HiveQL about the type of protocol, destination ip port numbers etc.

6.4 Rule Generation

By detecting anomalies Snort rules has to be added, so that when further such attack comes suitable action should be taken. Here the no: of packets is taken as a constraint for identifying an attack, .i.e. if no of packets from a particular source to a particular destination exceeds a count can identified as an attack, as the attacker may be trying to launch the attack to the victim by continuously sending packets. Other constraints can also be chosen. These kinds of rules will enhance Snort because of the ability to find new attacks. By adding such rules, when such packets comes Snort will give alerts to the user.

7 IMPLEMENTATION AND RESULTS

The proposed work is implemented with Hadoop cdh4.1.2, Snort 2.9.1, Hive 0.9.0. Hadoop cluster is implemented as pseudo distributed mode. Hping3 was used to generate packets. Packets were send in such a way that there where hundreds of packets send in every second to simulate the flood attack. The packets were logged by running Snort in packet logger mode for five minutes. The log files where then converted into suitable readable format and each of them had an average of 400Mb size. Each of the packets where in different size and in different format. Out of them source ip, destination ip, protocol and count where extracted.

MapReduce job was done on Hadoop cluster with various file sizes of 419.9Mb, 843.5 Mb, 1.7 Gb, 2.5 Gb. Each of them had an average of fifty thousand to two lakh packets. Table 1 shows the CPU times and number of map tasks.

The number of reduce tasks in all cases is one. As shown in results the analysis process does not take much time to perform the task. In a single node, for the wall clock time an average of 15 minutes is taken. Therefore it can be concluded that as the number of nodes increases the time will be greatly reduced and thus efficiency will be improved.

TABLE I
RESULTS OF RUNNING MAPREDUCE JOB

Input File size	Number of map tasks	CPU time spent (s)
419.9 Mb	7	50
843.5 Mb	14	100
1.7 Gb	22	140
2.5 Gb	38	275

Experiments were conducted with various block size of 32 Mb, 64 Mb, 128 Mb with the packet file size of 421.8 Mb. Table 2 shows the performance, with the number of map tasks and the CPU time spend by the mapreduce framework. It was seen that performance was better when block size was 128Mb, but a default size of 64 Mb is also working good

TABLE II
RESULTS OF RUNNING MAPREDUCE JOB WITH DIFFERENT BLOCK SIZES

Block Size	Number of Map Tasks	CPU time spent(s)
32 Mb	16	64
64 Mb	7	50
128 Mb	4	39

In Hive, the time for retrieving data is very less. During implementation, the results of analysis were loaded to Hive,. Table 3 shows the result of running different queries. It can be seen that for retrieving entire data just 10 s was needed. The queries which included joins and aggregate functions are carried out as MapReduce jobs

The second query shown in table counted 131091 number of data in 93 seconds. Therefore it can be concluded that managing data, with certain format is very much efficient with Hive

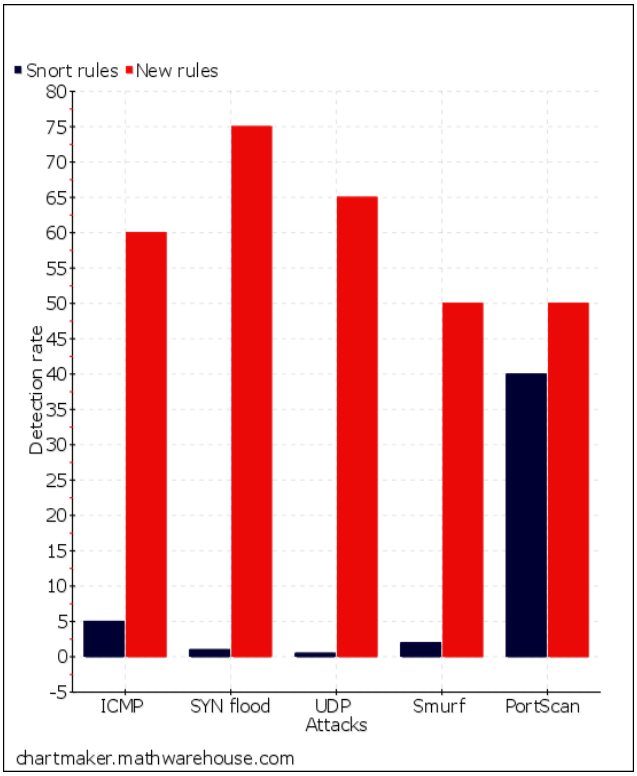
TABLE III
RESULTS OF RUNNING HIVEQL WITH DIFFERENT QUERIES

Query	Time
Select * from table_name	10s
Select count(*) from table_name	93s
Select count(*) from table_name for certain conditions	53s

For those ip addresses that send large number of packets than normal ones, snort rules were generated. The rules where generated by adding options for event filtering in such a way that the alerts will be generated only if number of packets from the particular source exceeds a particular number. This is to avoid the number of false alarms. For initial purpose snort rules snapshot 2931 was used. But none of them generated any alerts when attacks were simulated by Hping3. Then after sufficient

analysis new snort rules were generated with the corresponding ip addresses and port numbers along with necessary options. Fig. 6 shows the result of finding different kind of attacks that is launched by sending a huge number of packets. Most of the rule files were blank in order for the user to customize their own rules. New rules were generated with the output of the analysis done by Hadoop. Figure shows that the new snort rules generated were efficient in finding ICMP attack, Smurf attack, SYN flood attack, UDP attack and Port scanning. A user can customize their own rules even without analyzing. But it will generate a lot of falsealarms i.e. a node which is not an attacker may be treated as an attack. So generation of rules, after analysis is the good method.

Fig 6:Comparison between snort rules and new ones



8. CONCLUSION

Security is the most crucial issue that is taking place. Large amount of data has to be analyzed , for finding anomaly. In this work, the results show that analysis does not take much time , and it can be concluded that if number of nodes increased, performance can be made better. BigData can be more manageable with the data-ware house system Hive. Results also show that, Snort rules generated after analysis is very much efficient in detecting many attacks. In future the work can be extended to find more attacks.

9. REFERENCES

[1] W.Chen, W.Kuo and Y.Wang, Building IDS Log Analysis System on Novel Grid Computing Architecture, *National Center for High-Performance Computin*g, Taiwan,2009

[2] Y.Lee, W.Kang, Y.Lee, A Hadoop-based Packet Trace Processing Tool, P*roceedings of Third International Workshop on Traffic Monitoring and Analysis*,2011,pp: 51-63

[3] Y.Lee, W.Kanf, H.Son, An Internet Traffic Analysis Method with MapReduce,I*EEE/IFIP Network Operations and Management Symposium Workshops*,2010 ,pp:357-361

[4] X.Fang ,Integrating Artificial Intelligence into Snort IDS, *Proc of 3rd International Workshop on Intelligent Systems and Applications*, May 2011,pp: 1- 4

[5] B. E.Lavender, Implementation of Genetic Algorithm into a Network Intrusion Detection System (netGA) and Integrating to nProbe , Thesis Work

[6] J. Gomez, C. Gil , N. Padilla , R. Banos, C. Jimenez, Design of a Snort-Based Hybrid Intrusion Detection System, *Proceedings of the 10th International Work-Conference on Artificial Neural Networks*, 2009,pp: 515-522,

[7] M.Roesch ,Snort Lightweight Intrusion Detection for Networks, *Proc of LISA '99: 13th Systems Administration Conference*, 1999 ,pp:230- 238

[8] T.White and P. W. Daly, *Hadoop-The Definitive Guide, O'Reilly*

[9] A.Thushoo et.al. ,Hive A Petabyte Scale Data Warehouse Using Hadoop, *Proceedings of ICDE Conference*,2010 ,pp:996-1004