A Cost-Effective Solution to Detect DDoS Attacks using Big Data Analytics

G. Dileep Kumar[1], Dr. CV. Guru Rao[2]

[1]Research Scholar, Faculty of Computer Science, Jawaharlal Nehru Techological University, Hyderabad, India.
[2] Professor, Department of Computer Science & Engineering, SR Engineering College, Warangal, India.

Abstract
Attacks are increasing in volume, complexity and damage. Internet traffic measurement and Big Data analysis have been a challenging job because large packet trace files captured on fast links could not be easily handled on a single server with limited computing resources. While this pace of traffic and data generation is very exciting, it has created entirely new set of challenges and has forced to find new ways to handle. A DDoS attack is a large-scale, coordinated attack on the availability of services of a victim system or network resource, launched indirectly through many compromised computers on the Internet. However, since the attacker controls a large set of computers to make requests, the number and the size of network logs are also extreme large, so if the old network log analysis method is used for these large network logs, it will take a substantial amount of time to complete. As a result, utilizing Big Data technology is really necessary for this type of task. The study propose a Big Data based attacks detection framework which uses the low-cost commodity hardware, scalable and distributed open source framework for implementation.

Keywords:
DDoS Attack, Big Data, Hadoop, MapReduce, Detection

1. Introduction

A Denial of Service (DoS) and Distributed Denial of Service (DDoS) attacks are malicious attempts to make machines or web server and network resource unavailable to intended users, usually by temporarily interrupting or suspending the services of a host or resource connected to the Internet. DDOS attack is the complicated but powerful version of DOS attack in which many attacking systems are involved. DDoS attacks are launched by more than one hosts. To perform a DDoS attack, attackers uses botnet which is a group of infected computers on which the attacker has silently installed some kind of malware and thousands of hosts from all around the world. The general architecture of DDoS attack is depicted in figure 1.

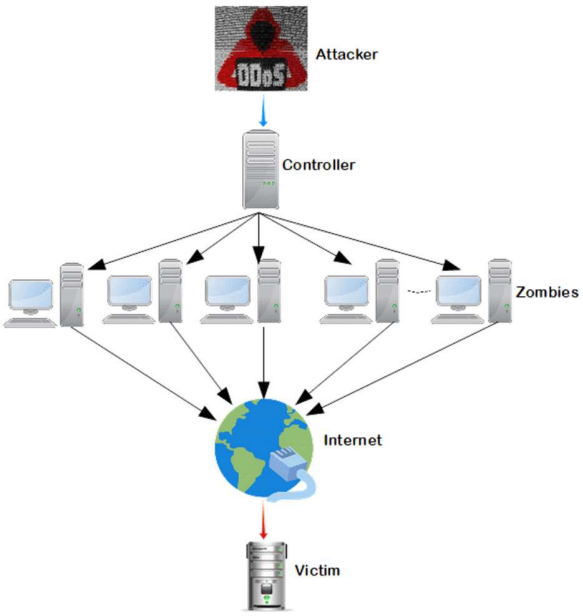

Figure 1: General DDoS attack Architecture

A DDoS Attack is composed of the following four elements:

1. The Attacker, the person who launches the attacks.
2. The Controller or Handlers, which are compromised hosts with an attacker program running on them and capable of controlling multiple zombies.
3. The Zombies or slaves are compromised hosts that are running an attacker program and are responsible for generating a stream of attack packets towards the intended victim.
4. A Victim or target host.

The need to propose and research novel methods to mitigate them has become a critical research issue in network security. As per main findings of Cisco's global IP traffic forecast, Annual global IP traffic will pass the zettabyte (ZB), which is 1000 exabytes(EB) by the end of 2017 and will reach 2.3 ZB per year by 2020[1]. While this pace traffic generation is exciting, it creates new set of

Manuscript received June 5, 2026
Manuscript revised June 20, 2026
https://doi.org/10.22937/IJCSNS.2026.26.6.4

challenges. Meanwhile, reports of massive data breaches are giving businesses greater and greater incentive to invest in network security analytics. Recently, attacks are increasing in volume, complexity and damage. The attacker can control a large set of computers to make requests. The number and the size of network logs are also extreme large, this would run into tens of terabytes i.e. Big Data. In this context, the high-cost data-warehousing solutions are confronted with the following major challenges.

- Expensive hardware and software: costs grow with data size
- Analytics process needs to be 100% customizable, and not governed by the confines of SQL.
- Analytics process should not be extended by the size of data; terabytes should be processed in minutes rather than days or hours.
- Data loss is unacceptable, the solution requires high availability and failover.

So, if the traditional network log analysis method is used for these large network logs, it would take a substantial amount of time to analyze and detect attacks. As a result, using Big Data technology is really necessary for this type of task. The Big Data ecosystem provides set of tools for different tasks such as data ingestion, data integration, data analysis and data visualization. Data ingestion can be done either in real time or batches. Data can be pulled from relational data bases or streamed from web logs. Flume, Sqoop and kafka are well known tools which could be used for data ingestion. There are many other tools and at times one has to customize as per requirements. NoSQL based tools such as HBase, MangoDB, Cassandra, Spark SQL, Spark streaming, Flink and Storm can be used for real time data processing. Once data is processed, one can visualize data using standard reporting tools such as Datameer, d3js, Tableau, Qlikview and many more.

Big Data has high-volume, high-velocity and high-variety information assets that demand cost-effective, innovative forms of information processing for enhanced insight and decision making. Big data is not defined by how you can measure data in terms of volume, velocity, and variety [2]. The benefits of Big Data include depth, time, storage, processing, analysis, flexibility and usability. Big Data Analytics is a process of collecting, organizing and analyzing of large sets of data to discover patterns and other useful information [3]. Apache Hadoop clusters and NoSQL handle task of capturing, cleaning, loading, and processing large dataset on demand or in real time using commodity hardware. That means that one cannot be tied to very expensive, proprietary offerings from a single vendor; rather, one can choose standardized, commonly available hardware from any of a large range of vendors to build a cluster. Commodity does not mean low-end. Low-end machines often have cheap components, which have higher failure rates than more expensive machines. When operating tens, hundreds, or thousands of machines, cheap components turn out to be a false economy, as the higher failure rate incurs a greater maintenance cost. On the other hand, large database class machines are not recommended either, since they don't score well on the price or performance curve.

The Apache Hadoop [4] is an open-source framework built for reliable, scalable distributed computing tasks with huge data sets over a cluster of multiple computers. This framework is written in Java programming language and it is under Apache License. Generally, a Hadoop system is composed of a computer acting as the master node and multiple computers acting as the slave nodes. Hadoop has three modules in total, including HDFS (Hadoop Distributed File System), Yarn, and MapReduce Framework. HDFS usually only has one NameNode, which is used to manage the directory tree and the metadata of related files of HDFS. It could also own a Secondary NameNode, which can be employed to backup mirror files and to combine logs and mirror files periodically and send back to NameNode. In general, NameNode and Secondary NameNode are deployed on the master node. In addition, DataNode of HDFS is responsible for store data and sending processed data back to NameNode, and it is usually deployed on the slave node. The Yarn consisting of Resource Manager and Node Manager. Resource Manager takes responsibility for managing and dispatching resources in the Hadoop cluster, and receiving data from Node Manager of each slave node. Node Manager is used to send information about usage of resources and task progress to Resource Manager. MapReduce task is executed in a distributed manner [5]. Each DataNode passes original data to a map function. After being processed by map function, original data becomes key-value pairs and is sent back to a reduce function. After completing the step of reduce function, the data will be saved as result or for further processing.

The Apache HBase is an open-source NoSQL, highly reliable, highly efficient, row-oriented and expandable distributed database system. It is implemented based on BigTable, an open-source software of Google. Similar to the fact that Google BigTable utilizes GFS as its file storage system, HBase employs Hadoop HDFS as its own file storage system; Google executes MapReduce to process huge data sets in BigTable, and HBase runs Hadoop MapReduce; Google BigTable takes advantage of Chubby as cooperative service, and HBase utilizes Zookeeper as its own cooperative service. HBase can be easily used to store huge unstructured data, and as Hadoop HDFS provides a reliable low-level storage support for HBase, it is great for processing huge data sets using MapReduce [4].

The paper has been organized as follows: Section 2 reviews the current available literature. Section 3 explains

design and implementation of our proposed approach to detect DDoS attacks. Section 4 discusses the results produced as part this work. Section 5 concludes our work with an insight to future plan.

2. Related Work

Analysis of logs and network flows for anomaly detection has been a problem in the information security for decades. New big data technologies, such as Hadoop, has attracted the interest of the security community for its promised ability to analyze and correlate security related heterogeneous data efficiently and at unprecedented scale and speeds [6]. In the rest of the section, we review some recent techniques where Hadoop based frameworks are used to build affordable infrastructures for security applications.

BotCloud [7] propose a scalable P2P detection mechanism based on MapReduce and combination of host and network approaches. First they generate large dataset of Netflow data on an individual operator. Next, they applied a PageRank algorithm on the Netflow traces to differentiate the dependency of hosts connected in P2P fashion for the detection of botnets. They moved the pagerank algorithm to MapReduce and the pagerank algorithm executes on data nodes of Hadoop cluster for efficient execution [8].

In [9], Lee et al. has proposed a DDoS detection method based on Hadoop. They have used a Hadoop based packet processor and devised a MapReduce based detection algorithm against the HTTP GET flooding attack. They employ a counter-based DDoS detection algorithm in MapReduce that counts the total traffic volume or the number of web page requests for picking out attackers from the clients. For experiments, they used multiple Hadoop nodes (max. 10) in parallel to show the performance gains for DDoS detection. Unfortunately, their proposed framework, in its current form can only be used for offline batch processing of huge volume of traces. The problem to develop a real-time defense system for live analysis still needs to be tackled.

Temporal and spatial traffic structures are essential for anomaly detectors to accurately drive the statistics from network traffic. Hadoop divides the data into multiple same size blocks, and distributes them in a cluster of data nodes to be processed independently. This could introduce a difficulty in analysis of network traffic where related packets may be spread across different block, thus dislocating traffic structures. Hashdoop [10] resolve this potential weakness by using hash function to divide traffic into blocks that preserve the spatial and temporal traffic structures. In this way, Hashdoop conserves all the advantages of the MapReduce model for accurate and efficient anomaly detection of network traffic.

3. Design and Implementation

Apache Hadoop clusters and NoSQL handle task of capturing, cleaning, loading, and processing large dataset on demand or in real time using commodity hardware. The experimentation was done on 10 nodes Hadoop cluster consisted of 10 commodity PCs, which helps to get a sense of distribute and parallel computing platform, shown in figure 2. One PC acts as the master node, and the remained PCs are used as slave nodes as shown in figure 2. Hardware specifications rapidly become obsolete, but for the sake of experimentation, our choice of machines for running a Hadoop DataNode and TaskTracker are as shown in Table 1.

Table 3.1: Hadoop Cluster Specifications

Parameter	Value
No. of nodes	10
Cores in Use	72
Availed HDFS	6.4 TB
Master Machine	CPU 2.25 Ghz RAM - 4 GB HDD 1 TB OS - Ubuntu 16.04 LTS
Slave Machine	CPU 2.25 Ghz RAM - 4 GB HDD 1 TB OS - Ubuntu 16.04 LTS
Ethernet Switch	16 Port 10/100 Mbps
Software Tools	Hadoop 2.7, OpenJDK 1.8, Hive 2.0.0, HBase 1.1.4 Flume 1.6.0, R 3.3.2, RStudio 1.0.136
Language Support	Java, Python, R, and MapReduce

The study proposes a Big Data based architecture, to confront contexts and challenges of traditional high-cost data warehousing solutions as mentioned in section 1. The architecture breaks the entire landscape into Data Acquisition, Data Integration and Analysis & Reporting layers, as shown in figure 2. Those three layers provide a seamless integration and flexible options to plug-n-play additional sources and delivery channels hiding the underlying data complexity in each layer. It enables handling the large volume of information, and the varied formats of data flowing into the system. Hence, the proposed framework centered on big data and data

virtualization techniques to manage this burst of information.

a. *Data Acquisition:* Once the sources of both structured and unstructured data are identified, both externally and internally the right set of packets capturing tools are required to extract Big Data of packets from network log files.
b. *Data Integration:* The network packets captured in data acquisition layer are parsed and stored into HBase NoSQL database, selecting related fields such as timestamp, destination, and request type from each line of log files. Later, data is imported on Hadoop Distributed File System(HDFS) for further MapReduce analysis.
c. *Analysis & Reporting:* The final stage of delivering the insights generated from the integrated information as dashboards, scoreboards, charts and reports with flexibility for business analysts to explore the details, and correlate the information sets for taking decision on DDoS attacks. The advanced analytics techniques in delivering such information would also help figuring out attack detection.

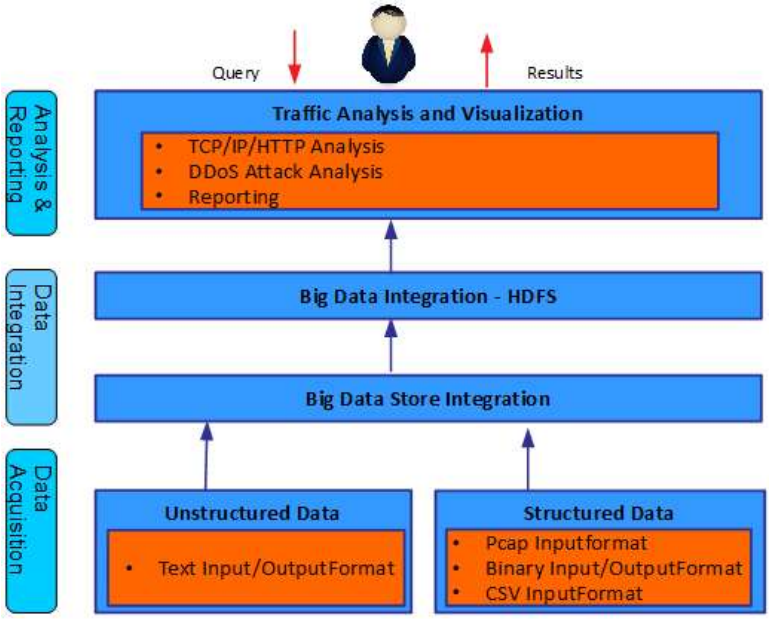

Figure 2: Big Data based DDoS Architecture

We used packet analyzer, WireShark to capture network packets and then stored them as pcap file in intended folder. Then, the actual analysis process starts as mentioned below.

Parsing and Detection Approach:

The task of analyzing network logs is abstracted as a MapReduce job which can be executed over a 10 nodes Hadoop cluster. MapReduce task is executed in a distributed manner. Each DataNode passes original data to a map function. After being processed by map function, original data becomes key-value pairs and is sent back to a reduce function. After completing the step of reduce function, the data will be saved as result or for further processing. This approach includes 9 abstracted steps which are listed below:

1. Parse network packets, and store relative data into HBase NoSQL database.
2. Hadoop imports data from HBase to HDFS.
3. Hadoop Master node reads data from Hadoop HDFS.
4. Split data chunk and Send them to Slave nodes.
5. Execute Map function on each node.
6. Send back processed result from Slave nodes back to Master node.
7. Execute Reduce Function
8. Store Reduce function result in Hadoop HDFS.
9. Analyze Reduce function result and export analysis result into a table of HBase

Threshold Based Detection Algorithm:

Algorithm 1 Threshold_based_MR_Algorithm

Result: In Normal or In Danger

procedure MAP(time_stamp, hbase_row)
request_type = hbase_row['type']
destination_ip = hbase_row['dest']
if *request_type = 'TCP'* **then**
  context.write(destination_ip,1)
end

procedure REDUCE(destination_ip, counts[ ])
total ← 0
for *each count in counts* **do**
  count ← *total* + *count*
  context.write(destination_ip,total)
end

procedure ANALYZE(destination_ip, total)
if *total* > *Threshold* **then**
  hbase.put(destination_ip,'In Danger')
else
  hbase.put(destination_ip,'In Normal')
end

4. Results and Discussion

After executing proposed approach against the network log data, one could use a command "scan StaticSecurityLog" to view the parsed network log data as shown below

```
99.999693            column=protocol:, timestamp=1403542125297, value=TCP
99.999755            column=dest:, timestamp=1403542125297, value=131.84.1.31
99.999755            column=protocol:, timestamp=1403542125297, value=TCP
99.999793            column=dest:, timestamp=1403542125297, value=131.84.1.31
99.999793            column=protocol:, timestamp=1403542125297, value=TCP
99.999915            column=dest:, timestamp=1403542125297, value=131.84.1.31
99.999915            column=protocol:, timestamp=1403542125297, value=TCP
74480 row(s) in 27.9900 seconds
```

Figure 3: Parsed Network Log

And the analysis result of this approach:

```
hbase(main):004:0> scan 'SecurityAnalysisResult'
ROW                  COLUMN+CELL
 131.84.1.31         column=num_of_request:num, timestamp=1403542130156, value=73643
 131.84.1.31         column=status:, timestamp=1403542130358, value=In Danger
 172.16.115.20       column=num_of_request:num, timestamp=1403542130156, value=166
 172.16.115.20       column=status:, timestamp=1403542130358, value=Normal
 172.16.115.5        column=num_of_request:num, timestamp=1403542130156, value=198
 172.16.115.5        column=status:, timestamp=1403542130358, value=Normal
 172.16.116.201      column=num_of_request:num, timestamp=1403542130156, value=89
 172.16.116.201      column=status:, timestamp=1403542130358, value=Normal
 202.77.162.213      column=num_of_request:num, timestamp=1403542130156, value=12
 202.77.162.213      column=status:, timestamp=1403542130358, value=Normal
5 row(s) in 0.0350 seconds
```

Figure 4: Analysis Result

5. Conclusion and Future Work

The present work proposes a Big Data based architecture to detect DDoS attacks leveraging Apache Hadoop stack provides ability to extract, aggregate, load and process large volumes of data in a distributed manner which in turn reduce the complexity and overall turn-around time in processing large volumes of data.

The attackers are changing their strategies and tactics in executing their tasks. So, there is a need for real-time monitoring of DDoS attacks with high speed. A next step in this path would be supporting Apache Spark which is an in memory based scalable, distributed, and open source framework and to evaluate new DDoS attack detection algorithm in terms of usability and analyzing different network traffic in various network environments.

References:

[1] Cisco's VNI Global Traffic Forecast, 2015-2020
[2] K. Bakshi, "Considerations for Big Data: Architecture and Approaches", In: Proceedings of the IEEE Aerospace Conference, pp. 1–7, 2012.
[3] Sachchidanand Singh and Nirmala Singh, "Big Data Analytics", International Conference on Communication, Information Computing Technology (ICCICT), Oct. 2012.
[4] Tom White, Hadoop: The definitive guide, O'Reilly Media, 2012.
[5] J.Dean and S.Ghemawat, "MapReduce: Simplified data processing on large clusters", Hindawi Publishing Corporation Advances in Multimedia, vol. 51, no. 1, pp. 107–113, 2008.
[6] Pratyusa K Manadhata Alvaro A Cardenas and Sreeranga P Rajan, "Big data analytics for security", IEEE Security and Privacy, vol. 6, pp. 74–76, 2013.
[7] Walter Bronzi R State Jerome Francois, Shaonan Wang and Thomas Engel, "Big data analytics for security", IEEE International Workshop on Information Forensics and Security (WIFS), pp. 1–6, 2011.
[8] Walter Bronzi R State Jerome Francois, Shaonan Wang and Thomas Engel, "Bottrack: Tracking botnets using netflow and pagerank", NETWORKING 2011, vol. 6640, pp. 1–14, 2011.
[9] Yeonhee Lee and Youngseok Lee, "Detecting DDoS attacks with Hadoop", In Proceedings of The ACM CoNEXT Student Workshop, pp. 1–2, 2011.
[10] Johan Mazel Romain Fontugne and Kensuke Fukuda, "Hashdoop: A mapreduce framework for network anomaly detection", IEEE Conference (INFOCOM WKSHPS), pp. 494–499, 2014.