Network Traffic Measurements and Analysis using Hadoop

Amol S.Suryawanshi
ME Student, Dept. of Computer Engineering,
Padmabhooshan Vasantdada Patil, Institute of Technology, Bavdhan, Pune 028, Maharashtra, India.

S.V.Bodake
Faculty, Dept. of Computer Engineering,
Padmabhooshan Vasantdada Patil, Institute of Technology, Bavdhan, Pune 028, Maharashtra, India

ABSTRACT

Accurate network traffic capture & measurements, analysis and monitoring is key to a wide range of network applications such as computer network traffic engineering, error detection & correction and all kind of security analysis with maintaince. A number of critical network management decisions, such as identifying faulty nodes & servers, routers, blocking traffic to a victim destination, monitoring traffic require extraction and analysis of real time data patterns in network traffic. The large traffic volumes seen in today’s high-speed networks pose tremendous computational and storage requirements for accurate traffic measurements & analysis.

We are going to actualizing Hadoop based system which collect traffic data, perform Traffic Analysis, Measurement, and Classification with deference to different parameters at parcel level. These outcomes can be utilized by Network Administrator and ISP's to identify abnormalities in system to achieve efficiency.

Keywords

Network traffic measurement, traffic analysis, HADOOP, ISP is Internet service provider**.**

1. INTRODUCTION

According to Cisco White paper, Annual global IP traffic will exceed the zettabyte threshold (1.4 zettabytes) by the end of 2017 [2]. In 2017, international internet traffic will reach up to 1.4 zettabytes per year, or 120.65 exabytes per month [2]. The networking devices e.g. routers and user devices like smart phones are increasing rapidly which make difficult for ISP's to collect and do analysis on huge amount of traffic data, activity logs and security including storage problems. The ISPs will need large infrastructure for storing, processing & analysis these up to zettabytes of data. Also, it leads to multiple problems such as machine failure, degrade performance, availability issues and much more. The ISP will rely on single high performance server. But as the Internet traffic data increases the single server will face the challenges like server failure, low response time, storage problems etc.

The Internet traffic data is collected from various routers and stored on disks to analyze it. But, as the traffic increases the data size will be increase along the way. There is no tool that can analyze Tera and Peta bytes of data in a single instance [9]. Hadoop therefore needs to be focused while analyzing huge amount of data because of its unique characteristics such as large storage system, fault-tolerant, scalability and availability. The method proposed here can manage packets and NetFlow data which are stored on HDFS and analyzed using Hadoop API using MapReduce.

2. LITERATURE SURVEY

Passive network monitoring is an indispensable mechanism for increasing the security and understanding the performance of modern networks. For example, Network level Intrusion Detection Systems (NIDS) inspect network traffic to detect attacks [1], [2] and pinpoint compromised computers [3], [4]. Similarly, traffic classification tools inspect network traffic to identify different communication patterns and spot potentially undesirable traffic [5], [6]. To make meaningful decisions, these monitoring applications need to analyze network traffic at the transport layer and above. For instance, NIDS reconstruct transport-layer streams to detect attack vectors that span multiple packets, and avoid evasion attacks.

Unfortunately, there is a gap between monitoring applications and available traffic capture tools. Applications increasingly need to ready & prompt about higher level entities and constructs such as TCP flows, HTTP headers, SQL arguments, email messages, and so on, while traffic capture frameworks still operate at the lowest possible level of the network model. They provide the raw data possibly duplicate, out-of-order, or overlapping-and in some cases even irrelevant packets that reach the monitoring interface. Upon receiving the captured packets at user space, monitoring applications usually perform TCP stream reassembly using existing libraries or custom stream reconstruction engines [1], [2]. This results in additional memory copy operations for extracting the payloads of TCP segments and merging them into larger “chunks” in contiguous memory locations [7]. Moreover, this misses several optimization opportunities, such as the early discarding of uninteresting & unwanted packets before system resources are spent to move them to user level and assigning different priorities to transport layer flows so that they can be handled appropriately at lower system layers of the network.

3. MOTIVATION & PROBLEM STATEMENT

To overcome the challenges of Distributed File Systems and satisfy demands for analysis of huge amount Internet traffic data, the computing and storage resources needs to be scaled out. Google has developed Google File System [2] for data intensive applications and workloads. MapReduce [3] allows users to control thousands of machines in parallel to process huge amount of data using map and reduce functions. Apache Hadoop [4] developed by Doug Cutting is an open source Java framework which provides MapReduce programming model and Hadoop Distributed File System (HDFS) as a distributed file system [4]. Hadoop is used by many companies such as Facebook, Yahoo, IBM and many more for processing huge amount of data [5]. Larger clusters are available to users to run their job with Hadoop on Demand (HOD). The quantities of Internet clients and transfer speed hungry applications have more notable effect presently days and because of this the measure of internet movement information produced is so colossal & important. It requires versatile devices to capture, dissect, measure, and arrange this activity information. Customary instruments neglect to do this assignment because of their constrained computational limit and capacity limit especially about memory concern. More seasoned devices utilizes big server to store internet activity data caught & to process the same. Hadoop is a conveyed structure which performs this assignment in extremely proficient & effective way. Hadoop for the most part keeps running on merchandise equipment with appropriated capacity i.e. HDFS and process this immense measure of movement information with a Map-Reduce programming model. Customary instruments neglect to do this assignment because of their constrained computational limit and lack of capacity.

Concept

Online network traffic measurements and analysis is critical for detecting and preventing any real-time anomalies & problems in the network. Also to capture huge amount of Internet traffic data, store it in ordinary database and perform analysis or measurement's on such data is create overhead & stress on such system model. Instead of that data is load to HDFS and process the same using MapReduce funstios.Parallel Programming Model on Commodity hardware to analyze, measure, and classify it according to the need of different stakeholder such as Internet Service Providers (ISPs) and Network Administrators for detecting network attacks.

A. *Objectives*

We identify a semantic gap where modern network monitoring applications need to operate at the transport layer and beyond, while existing monitoring systems operate at the network layer. To bridge this gap and enable aggressive optimizations, here we introduced the concept of packet stream capture, based on the fundamental abstraction of the bit stream & process it by Hadoop framework. Following are the key objectives;

- Traffic capture & classification
- Intrusion detection,
- To analyze the network traffic, measure, and classify it according to the need of different ISPs.

4. DATA FLOW DIAGRAM

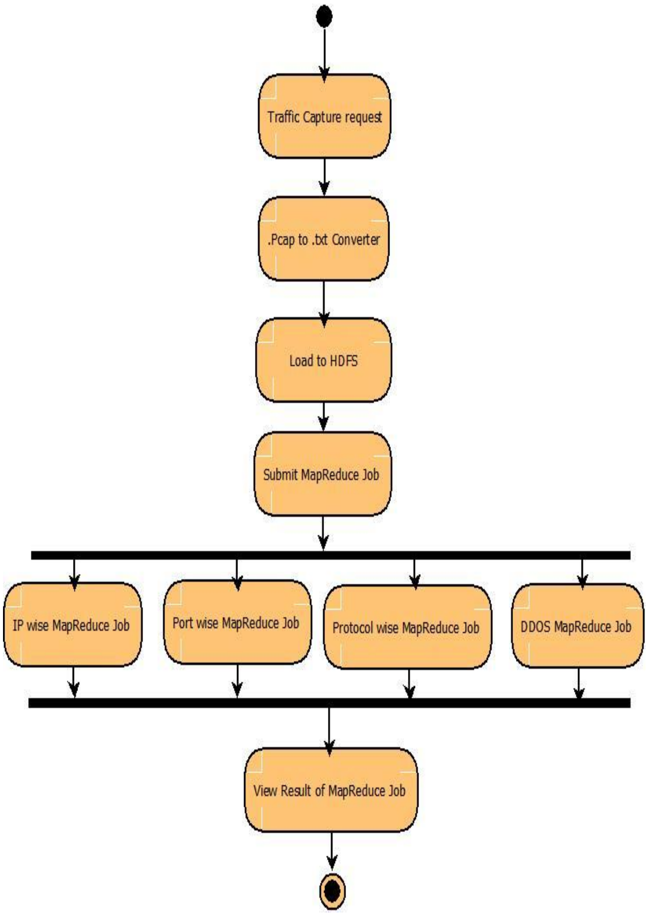

Fig 1.Data Flow Diagram

As shown in DFD the system is divided into six major activities which are capture the traffic, conversion of this captured data type, load this data to HDFS,submit different or required MapReduce function & finally display the result. All this processed & preprocessed data is stored in HDFS .

All these levels or activities are explained briefly in the system architecture.

5. SYSTEM ARCHITECTURE

Online network traffic measurements and analysis is critical for detecting and preventing any real-time anomalies in the network. Also to capture huge amount of Internet traffic data, store it in ordinary database and perform analysis or measurement's on such data is create overhead on such system model. Instead of that data is load to HDFS and process the same using MapReduce Parallel Programming Model on Commodity hardware to analyze, measure, and classify it according to the need of different stakeholder such as Internet Service Providers (ISPs) and Network Administrators for detecting network attacks. Fig.2 shows the system architecture also known as general block diagram.

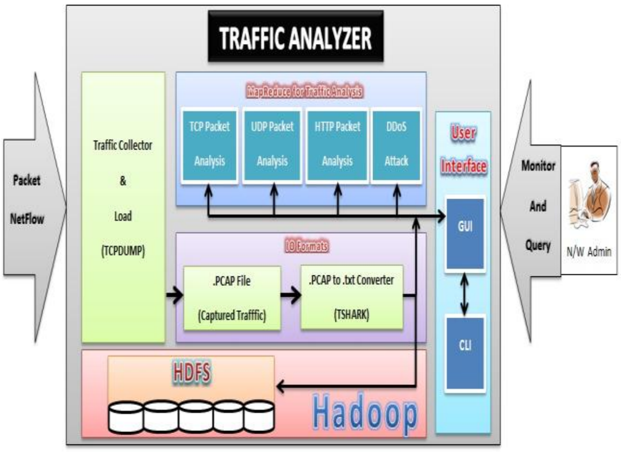

Fig 2. System Architecture

In above architecture packet net flow is collected from respected router & this huge amount of data is firstly collected by a external java utility called as ‘pcpdump’.The data collected & stored by this utility is in the ‘.pcap’ file extention.But Hadoop framework doesn't support this file extention.So there is need to convert this data to ‘.txt’ type which is done by again a different utility named ‘tshark’ .All this data is then uploaded to Hadoop’s MapReduce function.

B. MapReduce have five important components:

- Name Node: Used for indexing of data. Master in name node works like a server where all data is uploaded.
- Data Node: here actual data is stored.
- Job Tracker: Built on master who splits different jobs like IP wise count, port wise capture & calculation of data etc.
- Task Tracker: Executes the job. Also assigns jobs to slave.
- Secondary Name Node: Used for data security purpose. This is replication of name node. If name node fails this secondary name node is used.

This Mapreduce framework has structure as shown in following figure 3.

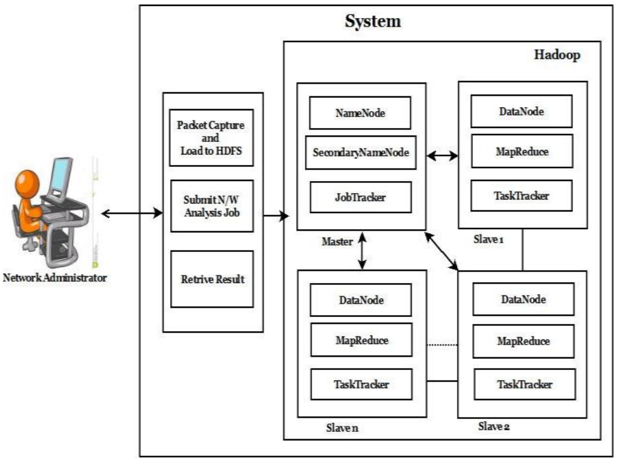

Fig 3.MapReduce framework

C. MapReduce have following different functions;

- TCP Packet Analysis.
- UDP Packet Analysis.
- HTTP Packet Analysis.
- DoS Attack detection.

After execution of all these different functions, we can say MapReduce jobs; we can display the results in graphical format. For this final output, GUI is used with Common Language Interface (CLI) .In this way this network traffic analyzer works.

We can also add different other functions in MapReduce like incoming requests to our node, outgoing requests from our node, number of other nodes connected with our network, hops connected to our node & network etc. Accurate traffic measurement and monitoring is key to a wide range of network applications such as traffic engineering, anomaly & intrusion detection, security analysis etc. A number of critical network management decisions, such as blocking traffic to a victim destination, identify the vulnerable nodes & servers etc. require extraction and analysis of real time patterns in network traffic. The large traffic volumes seen in today’s high speed networks have enormous computational and storage requirements for accurate traffic measurements. All this huge amount of data easily stored on Haoop’s HDFS (Hadoop Distributed File System) where data is stored in a quite different way other than normal DBMS, where tables are used.

6. MATHEMATICAL MODEL

1. **Let S= { } be as a Traffic analysis system.**
2. **Obtain a set of IP packet flow records**

   IP_{flow} = {S_{IP}, S_{PORT}, D_{IP}, D_{PORT}, P, TS, D, Dt}

   Where S_{IP}= Source IP, S_{PORT}= Source Port, D_{IP}=Destination IP, D_{PORT}=Destination Port, P=Protocol, TS=Time Stamp, D=Date and

   Dt =DateTime.

   S= { IP_{flow} }
3. **Give input files upload to HDFS**

   f1= {$IP_{flow1,}$ $IP_{flow2,}$ $IP_{flow3....}$ ...IP_{flown}}

   Where f1is a file which contains IP packet flow records

   F= {f1, f2, f3, …………,fn}

   S= { IP_{flow}, F}
4. **Create MapReduce job to count TCP packets**

   TCP_{COUNT}= {P}

   Where TCP_{COUNT} is a number of TCP packet send or receive within a network

   S= { IP_{flow}, F, TCP_{COUNT} }
5. **Create MapReduce job to count UDP packets**

   UDP_{COUNT}= {P}

   Where UDP_{COUNT} is a number of UDP packet send or receive within a network

   S= { IP_{flow}, F, TCP_{COUNT}, UDP_{COUNT} }

6. **Create MapReduce job to classify traffic port wise**

   P_W= {S_{PORT}, D_{PORT}}

   Where P_W is a sorted port wise traffic count

   S= { IP_{flow}, F, TCP_{COUNT}, UDP_{COUNT},P_W }

7. **Create MapReduce job to detect TCP flood attack**

   T_{FA}= {TCP_{COUNT}> Threshold}

   Where P_W is a sorted port wise traffic count

   S= { IP_{flow}, F, TCP_{COUNT}, UDP_{COUNT},P_W ,T_{FA} }

8. **Final Set S= { IP_{flow}, F, TCP_{COUNT}, UDP_{COUNT},P_W ,T_{FA} }**

D. Mathematical Terms

1. **Capture IP packet flow IP_{flow}**

   IP_{flow} = {S_{IP}, S_{PORT}, D_{IP}, D_{PORT}, P, TS, D, Dt}

   Where S_{IP}= Source IP,

   S_{PORT}= Source Port,

   D_{IP}=Destination IP,

   D_{PORT}=Destination Port,

   P=Protocol, T

   S=Time Stamp,

   D=Date and D

   t =DateTime

2. **TCP count MapReduce job**

   TCP_{COUNT}=∑IP_{flow} → (P=”TCP”) ……. (1)

   Where TCP_{COUNT} is count of IP_{flow} packet which have protocol field TCP

3. **UDP count MapReduce job**

   UDP_{COUNT}=∑IP_{flow} → (P=”UDP”) …. (2)

   Where UDP_{COUNT} is count of IP_{flow} packet which have protocol field UDP

4. **Port wise classification MapReduce job**

   P_W=∑$S_{PORT}D_{PORT}$………………… (3)

   Where P_W is port wise count of IP_{flow} packet

5. **MapReduce job to detect TCP flood attack**

   T_{FA}= (∑P.D.t)$_{>}$Threshold ………… (4)

   Where T_{FA} is count of number of packet received within same date and time

E. Packet Level:

The packets which are captured undergo certain analysis and measurement. We perform calculation based on packet and flow level. It involves simple mathematics terms:

Given a packet trace T with time t total number of packets is given as:

Total Count = ∑ packet records …………. (1)

Similarly, total size S can be represented as:

Total Size = ∑packet bytes ………………. (2)

F. Flow Level:

We can also calculate port-based classification, day-wise traffic. At flow level we classify our traffic according to port numbers.

G. Network Attack (UDP Flood):

Number of packets ≥ threshold …………… (3)

(Where threshold is set to 100 per second in our system)

H. Solving approach and Efficiency issues

To capture huge amount of Internet Traffic data, store it on HDFS and process the same using MapReduce Parallel Programming Model on Commodity hardware to analyze, measure, and classify it according to the need of different stakeholder such as Internet Service Providers (ISPs) and Network Administrators for detecting network attacks. The Internet traffic data is collected from various routers and stored on disks to analyze it. But, as the traffic increases the data size will be increase along the way. There is no tool that can analyze terabytes and petabytes of data in a single instance. Hadoop therefore needs to be focused while analyzing huge amount of data because of its unique characteristics such as storage system, fault-tolerant, scalability and availability. The method proposed here can mange packets and NetFlow data which are stored on HDFS and analyzed using Hadoop API.

7. CONCLUSION

In this paper, we identified a gap in network traffic monitoring system. Networking applications usually need to express their monitoring requirements at a high level, using context from the transport layer or even higher, while most monitoring tools still operate at the network layer. To bridge this gap, we have presented the design, implementation using Hadoop, a big network monitoring framework that offers an expressive API and significant performance improvements for applications that process traffic at the transport layer and beyond.

Output of this work shows traffic analysis and measurement with various parameters. It also shows port-based; protocol based internet traffic classification and detects UDP flood attack, DoS, traffic flows on particular web server etc.

8. REFERENCES

[1] IEEE Journal on Selected Areas in Communication, Vol. 32, No. 10, October 2014,”Stream-Oriented Network Traffic Capture and Analysis for High-Speed Networks” by Antonis Papadogiannakis, Michalis Polychronakis,and Evangelos P. Markatos.

[2] Cisco White Paper. ”Cisco Visual Networking Index: Forecast and Methodology”, 2011-2016, May 2012.

[3] S. Ghemawat, H. Gobio, and S. Leung,”The Google File System”, ACM SOSP, 2003.

[4] J. Dean and S. Ghemawat,”MapReduce: Simplified Data Processing on Large Cluster”, USENIX OSDI, 2004.

[5] Hadoop, http://hadoop.apache.org/.

[6] T. White, Hadoop: The Definitive Guide, O’Reilly, Third ed., 2012.

[7] M. Roesch,”Snort: Lightweight intrusion detection for networks,” in Proc. USENIX LISA Conf., 1999, pp. 229-238.

[8] V. Paxson,"Bro: A system for detecting network intruders in real-time," Comput.Netw., vol. 31, no. 23/24, pp. 2435-2463, Dec. 1999.

[9] G. Gu, P. Porras, V. Yegneswaran, M. Fong, and W. Lee,"BotHunter: Detecting malware infection through IDS-driven dialog correlation," in Proc. USENIX Security Symp., 2007, pp. 167-182.

[10] S. Singh, C. Estan, G. Varghese, and S. Savage,"Automated worm finger printing,"in Proc. USENIX Symp. OSDI, 2004, pp. 45-60.

[11] Application Layer Packet Classifier for Linux (L7-Filter). [Online].Available:http://l7-filter.sourceforge.net/.

[12] IEEE/ACM TRANSACTIONS ON NETWORKING, VOL. 22, NO. 2, APRIL 2014 377 "Streaming Solutions for Fine-Grained Network Traffic Measurements and Analysis" Faisal Khan, Nicholas Hosein, Soheil Ghiasi, Senior Member, IEEE Chen-Nee Chuah, Senior Member, IEEE, and Puneet Sharma, Senior Member, IEEE.

[13] "Hadoop-The Definitive Guide" -*Tom White*

[14] "Data communications & Networking" *–Forouzan*

[15] "Computer Networks"-*A.S.Tanenbaum.*