ijecs open access www.ijecs.in
International Journal Of Engineering And Computer Science ISSN:2319-7242
Volume 2 Issue 11, Nov 2013 Page No. 3187-3190

A Study on Scalable Internet Traffic Measurement and Analysis with Hadoop

Yogesh V. Kadam[1], Prof. Vaibhav Dhore[2]

1, 2 Department of Computer Engineering, RMD Sinhgad School of Engineering,
University of Pune, India
kadam.yogesh3@gmail.com , *vaibhav.dhore@sinhgad.edu*

Abstract: ***As the number of Internet users is growing rapidly worldwide, Internet traffic data also increases. To analyze this traffic, multiple tools are available. But they do not perform well when the traffic data size increases. This traffic measurement and analysis is used to observe network usage behavior and perform different types of analysis. As the data grows it is necessary to increase the necessary infrastructure to process it. The Distributed File System can be used for this purpose, but it has certain limitations such as scalability, availability, and fault-tolerance. Hadoop is an open source distributed computing platform having MapReduce for distributed processing and HDFS to store huge amount of data. This study presents a Hadoop-based traffic monitoring system that performs a multiple types of analysis on huge amount of Internet traffic in a scalable manner.***

Keywords: Hadoop, MapReduce, NetFlow, libpcap, Traffic analysis and measurement, Hive**.**

1. Introduction

The Internet traffic [1] having much importance is growing rapidly as proportional to users. The big data is emerging research area for researchers because, according to Cisco White Paper [2], IP traffic will grow at a Compound Annual Growth Rate (CAGR) of 23 percent from 2012 to 2017 i.e., it will surpass the Zettabyte threshold (1.4 Zettabyte) by the end of 2017. In Asia Pacific countries this IP traffic will reach 43.4 Exabyte per month by 2017.

The traffic data size is growing enormously but it is necessary to extract the knowledge from this data by analyzing it. The networking devices and user devices are increasing rapidly with high performance which makes difficult for Internet Service Provider's (ISP's) to collect and analyze this traffic data. The ISP will need large infrastructure to store and analyze this data. But, again it leads to certain challenges such as fault-tolerant system, performance, scalability, availability and many more which are faced by distributed system. Most of the time, ISP will rely on Single High-Performance Server to analyze this traffic data. But, as the traffic data increases this method will become inefficient.

The Google has developed Google File System (GFS) [3] for data intensive applications and workloads. MapReduce [4] allows users to control thousands of machines in parallel to process huge amount of data using map and reduce functions. Based on these developments, Doug Cutting @ Yahoo! now in Cloudera organization has developed Hadoop later on donated to Apache known as Apache Hadoop [5] which is an open source java framework provides MapReduce programming model and Hadoop Distributed File System (HDFS) for storing distributed data. Currently, many of the companies like Yahoo, Facebook, New York Times, Last.fm, IBM, Amazon etc. are using this framework for analyzing their website data.

The traffic data is collected from various routers and stored on disks to analyze the same. There are multiple tools to analyze this data but, as the traffic data size increases these tools will be inefficient to analyze and measure this data. The Hadoop now become the powerful tool to do the same task because of its unique characteristics such as Parallel processing, Scalability, Availability, Fault-tolerance, and Distributed Storage system. This study is based on managing the packets and NetFlow data which are stored on HDFS and analyzed and measured using Hadoop API [6]. This analysis is used for observing the network usage, user behavior, finding DDOS attacks and much more. The Hive [7] is also used for creating versatile operational analysis queries on huge amount of Internet traffic data. It also focuses on performance improvement while running Hadoop on large cluster.

2. Related Work

Many tools are available for Internet traffic monitoring and analysis. Tcp-dump [8] is a powerful command-line packet analyzer with libpcap as a library for network traffic capture. Wireshark [9] is network protocol analyzer which is visually rich, powerful LAN analyzer that captures live traffic and stores it for offline analysis. Cisco IOS NetFlow Analyzer [10], GenieATM [11] provides facility for network traffic analysis. CoralReef [12] developed by CAIDA measures and analyzes passive Internet traffic data. Though all these are useful seems looking at first sight, most of these are run on single high

performance server which is not capable of handling huge amount of traffic data captured at very high-speed links. Still there are multiple solutions but having some limitations.

Hadoop built on MapReduce basics can be used to analyze web, text and log files. In [13], the method analyzes packet trace files in parallel manner. RIPE [14] does not consider the parallel-processing capability of reading the packet records from HDFS which leads to performance degradation. This study shows you the complete Internet traffic analysis system with Hadoop that can quickly process IP packets and NetFlow data.

3. Components of Traffic Measurement and Analysis with Hadoop

There are various components present in traffic measurement and analysis system which are as shown in Fig. 1. It consists of traffic collector, new packet input formats, MapReduce analysis algorithms for various traffic data formats.

3.1 Traffic Collector

The purpose of collector is to both receive IP packet and NetFlow data from probes or packet trace files from the disk and write them on HDFS. This traffic collection is carried out by a load balancer and HDFS DataNodes. The load balancer probes packet streams using a high-speed packet capture driver like PF_RING and TNAPI [15], which forwards packets to multiple DataNodes. Then HDFS DataNode captures these forwarded packets and writes them to HDFS files concurrently. RAID0 is used to boost the performance i.e. parallel disk I/O operation. The proposed method still does not guarantee the end-to-end performance from online traffic collection to real-time analysis.

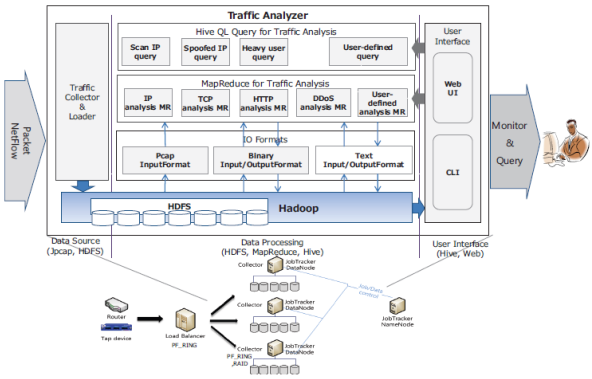

Figure 1: Architecture of Traffic Measurement and Analysis with Hadoop [6]

3.2 IP Packet and NetFlow Reader in Hadoop

Generally, IP packets and NetFlow data are stored in binary format of libpcap. In proposed method, packet trace files captured needs to be converted into HDFS-specific sequence files but which increases computation overhead and this file format is not compatible with libpcap tools. Thus [6], introduces new Hadoop API that can read or write IP packets and NetFlow data in native libpcap format on HDFS.

When a MapReduce job runs on libpcap files in HDFS, each map task reads its assigned HDFS block to parse packet records independently. It is difficult for a map task to parse packet records from its HDFS block because of variable-sized packets and no explicit separator. In [14], a single map task cam process a whole large libpcap file in sequential manner which is not appropriate for parallel processing. If a map or reduce task fails, the MapReduce job will roll back to beginning of file.

In [13], there is an algorithm which read packet records of HDFS blocks in parallel. Figure 2 shows how the reading of packet records in libpcap can be done for parallel processing. It uses the timestamp filed, wired length, captured length field of packet header. PcapInputFormat is packet input module which manipulates IP packet and NetFlow data. BinaryInputFormat is input module that manages binary records such as calculating flow statistics from IP packets.

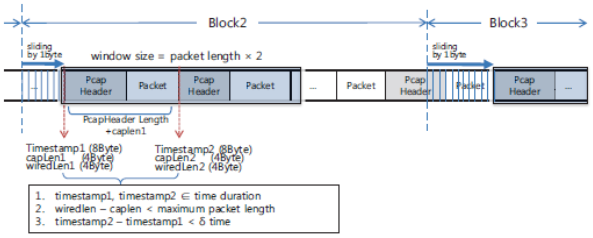

Figure 2: Packet Reader [6]

4. Analysis at Different Layers

In proposed method [6], author has developed various tools to analyze traffic at different layers which is as illustrated in Figure 3.

	Traffic Analysis Job	Hadoop Tool Command	Description
IP Analysis	Total traffic and host/port count statistics	*PcapTotalStats –r[source dir/file] –n[reduces]*	Computing byte/packet/flowcounts regarding IPv4/v6/non-IP and the number of unique IP addresses/ports
	Periodic flow statistics	*PcapTotalFlowStats –r[source dir/file]*	Computing bytecount, packetcount per each interval, and periodic flow statistics regarding byte/packet/flowcounts
	Periodic simple traffic statistics	*PcapStats –r[source dir/file] –n[reduces]*	Computing periodic bytecount/packetcount regarding IPv4/v6/non-IP per interval
	Total count grouping by key	*PcapCountUp –r[source dir/file] –n[reduces]*	Computing total bytecount/packetcount by key (e.g., packetcount per each source IP address)
TCP Analysis	TCP statistics	*TcpStatRunner –jt –r[source dir/file] –n[reduces]*	Computing RTT, retransmission, out-of-order, and throughput per TCP connection
Application Analysis	Web usage pattern	*HttpStatRunner –ju -r[source dir/file] –n[reduces]*	Sorting Web URLs for user by timestamp
	Web popularity	*HttpStatRunner –jw -r[source dir/file] –n[reduces]*	Computing user count, view count for Web URL per Host
	DDoS analysis	*HttpStatRunner –jd -r[source dir/file] –n[reduces]*	Extracting attacked server and infected hosts
Flow Analysis	Flow concatenation and print	*FlowPrint [source dir/file]*	Aggregating multiple NetFlow files and converting flow records to human readable ones
	Aggregate flow statistics	*FlowStats [source dir/file]*	Computing total traffic of sIP/dIP/sPort/dPort/srcAS/dstAS/srcSubnet/dstSubnet per inbound/outbound
-	Top N	*TopN [source dir/file]*	Sorting records by key and emitting N numbers of record from the top.

Figure 3: Various Analysis tools in MapReduce [6]

4.1 Network Layer Analysis

There are IP flow statistics tools in MapReduce which is very simple counting job for given key. With this tool, IP packets and NetFlow records can be retrieved and IP flow statistics can be determined. For packet trace file IP statistics can be calculated.

4.2 Transport Layer Analysis

Before performing TCP-layer analysis in MapReduce, it is necessary to extract a full TCP connection containing both client-server and server-client directional flows that are stored across several HDFS blocks. Hence to find a TCP connection, it is necessary to map multiple of client-server and server-client directional TCP flow on different HDFS blocks into the same reduce task. Figure 4 shows how two client-server and server-client directional TCP flows that belong to a single TCP connection are assigned to the same reducer.

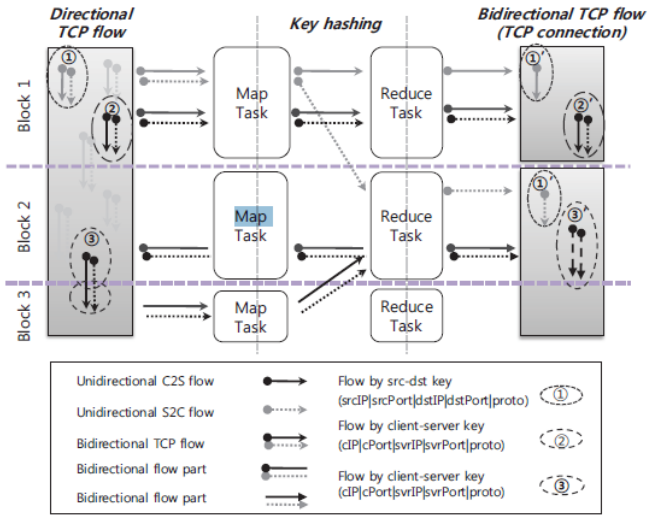

Figure 4: Single TCP connection using two directional TCP flows [6]

After pulling packets of a TCP connection from multiple map tasks, a reduce task computed the performance metrics for TCP connection. A TCP segment and its corresponding ACK together can be sent form map task to reduce task which improves the MapReduce job performance.

4.3 Application Layer Analysis

An HTTP is very popular Internet application in which proposed method investigates website popularity and user behavior by examining HTTP packets. In MapReduce algorithm, map task extract several fields from header such as URI, user agent, host values then reduce task summarizes the statistics or popularity per website, webpage, or URI.

The proposed method also does DDoS traffic analysis in which the map task generates keys to classify the requests and response HTTP messages then reduce task summarizes HTTP request messages and marks the abnormal traffic load by comparing it with threshold.

5. Interactive Query Interface with Hive

MapReduce framework is useful for processing huge amount of IP packets and NetFlow data in parallel. It will be time consuming to write a application-specific analysis programs in MapReduce. Hive provides the ability to generate MapReduce codes through Hive Query Language (HiveQL). It is as shown in Figure 5.

Traffic collector receives the NetFlow data from a router and writes them to a single HDFS file every five minutes. After a NetFlow file is closed, the job scheduler invokes IP analysis MapReduce jobs to process NetFlow data and upload flow records and IP statistics to the Hive tables where various queries can be executed.

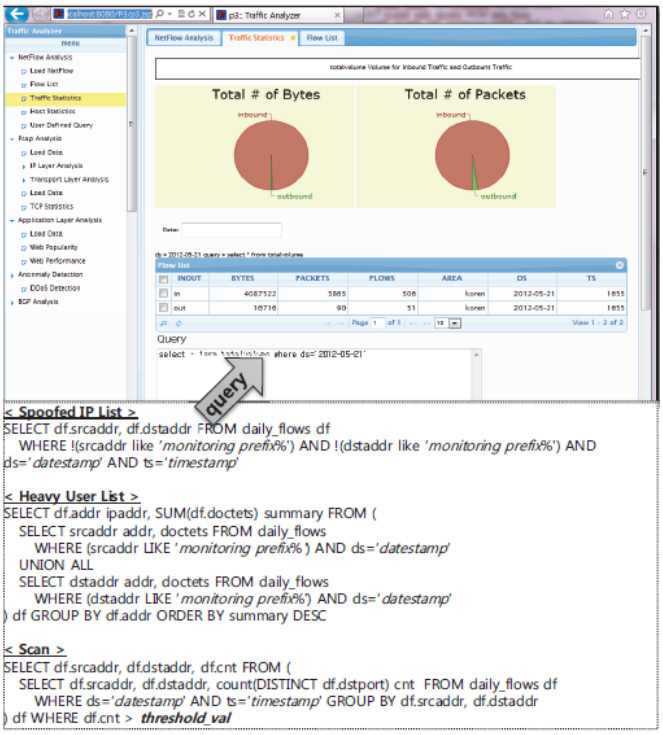

Figure 5: GUI for Hive Queries [6]

6. Conclusion and Future Work

In this paper, a Hadoop based Internet traffic measurement and traffic analysis method is presented which integrates components like traffic collector, packet and NetFlow data reader. This study also presents an analysis at various layers. Finally, a HiveQL query language for large data is also presented. In future work, there will be a scope of performing real-time traffic analysis in high-speed networks. Also in future multiple file formats would be accepted as an input for traffic analysis. It will also be possible to calculate the average packet retransmission which is caused by TCP.

References

[1] T. White, Hadoop: The Definitive Guide, O'Reilly, 3rd ed., 2012.

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

[3] S. Ghemawat, H. Gobioff, 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] Y. Lee and Y. Lee, "Toward Scalable Internet Traffic Measurement and Analysis with Hadoop", ACM SIGCOMM Computer Communication Review, 2013.

[7] Hive, http://hive.apahe.org/.

[8] Tcpdump, http://www.tcpdump.org.

[9] Wireshark, http://www.wireshark.org

[10] Cisco NetFlow, http://www.cisco.com/web/go/netflow.

[11] GenieATM, http://www.genienrm.com

[12] CAIDA CoralReef Software Suite, http://www.caida.org/tools/measurement/coralreef.

[13] Y. Lee, W. Kang, and Y. Lee, " A Hadoop-based Packet Trace Processing Tool", International Workshop on Traffic Monitoring and Analysis (TMA 2011), April 2011.
[14] RIPE Hadoop PCAP, https://labs.ripe.net/Members/wnagele/large-scale-pcap-data -analysis-using-apache-hadoop, Nov. 2011
[15] F. Fusco and L. Deri, "High Speed Network Traffic Analysis with Commodity Multi-core Systems", ACM IMC 2010, Nov.2010.