Inspection of Deep Packet Inspection in Real-time for Performance Testing: A Framework

Suguna M
M.Tech (CNE) Student: Dept. of Computer Science Eng.
Akshaya Institute of Technology
Tumkur, India

Mrs. T. Roopa
Asst. Prof. Dept. of Computer Science Eng.
Akshaya Institute of Technology
Tumkur, India

***Abstract* -- In order for a deep packet inspection engine to be effective in matching traffic, both directions of traffic flow need to be seen by the classifier. The traffic generated by the source and the replies generated by the destination thus need to be passed through the classifier. It is therefore necessary to separate "Good traffic" from "Bad traffic" to optimize performance of communication networks and mitigate the risk imposed on these once closed computer systems. Enterprise and service provider customers develop, maintain and operate network infrastructure in order to support the applications required to perform their day to day tasks. These applications have certain requirements and expectations from the infrastructure, including access to public networks, and thus rely on quality of service (Q0S) controls to manage network traffic. QOS controls are used to ensure non-critical applications do not hamper the operation of critical ones, all the while providing fair access to all legitimate applications. QOS systems are increasingly being used as firewalls, filtering bad traffic and allowing good traffic to traverse the network without delay. The proposed system designs and develop a cost effective tool to perform deep packet inspection over real-time traffic.**

Keywords: Internet traffic, real-time network, Deep Packet Inspection

I. INTRODUCTION

Through the increasing of make unclear compute procedure in addition to the employ of circulated system to make available communications and platforms a examine, the monitor and presentation study of circulated system became additional essential .within dispersed system growth, the preservation and management, the discovery of mistake cause in addition to the study in addition to the copy of an inaccuracy be challenge and motivate hard work in the direction of the growth of less disturbing mechanism for debug and monitor dispersed application on runtime Network traffic analysis be single alternative in the direction of assess dispersed system presentation. Even though present limits on top of capability to development great quantity of association small package in short time. And on top of scalability designate bright in the direction of procedure network traffic in surfeit of variation of throughput in addition to reserve command hence Deep packet inspection (DPI) consists of inspecting both the packet header and payload and alerting the system when signatures of malicious software appear in the traffic. These signatures are identified through pattern matching algorithms that are classified either as string matching, in which the patterns are a set of strings, or regular expression matching, in which the patterns are defined as regular expressions. DPI is a basic element in today's security tools, such as Network Intrusion Detection/Prevention System (NIDS/NIPS) or Web application firewall, which are used to detect malicious activities. Moreover, DPI and its corresponding pattern matching algorithms are also crucial building blocks for other networking applications such as traffic monitoring and HTTP load-balancing. Today, the performance of security tools is dominated by the speed of the underlying pattern matching algorithms both string matching and regular expression matching are fundamental problems in computer science and have been a topic of intensive research for decades. It is important to classify network traffic in order to assist network administrators in managing data flows on their IP networks. By providing network administrators the ability to flag traffic flows with a particular firewall mark, they are able to block, restrict flow (traffic-shape) and possibly redirect certain flows based on a predefined rule set. This process increases the overall security of the network as unclassified traffic will not go unnoticed as is the case in many networks

II. DEEP PACKET INSPECTION (DPI

Deep Packet Inspection (DPI, also called complete packet inspection and Information extraction or IX) is a form of computer network packet filtering that examines the data part (and possibly also the header) of a packet as it passes an inspection point, searching for protocol non-compliance, viruses, spam, intrusions, or defined criteria to decide whether the packet may pass or if it needs to be routed to a different destination, or, for the purpose of collecting statistical information. There are multiple headers for IP packets; network equipment only needs to use the first of these (the IP header) for normal operation, but use of the second header (TCP, UDP etc.) is normally considered to be shallow packet inspection (usually called State full Packet Inspection) despite this definition There are multiple ways to acquire packets for deep packet inspection. Using port mirroring (sometimes called Span Port) is a very common way, as well as optical

splitter .Deep Packet Inspection (and filtering) enables advanced network management, user service, and security functions as well as internet data mining, eavesdropping, and internet censorship. Although DPI technology has been used for Internet management for many years, some advocates of net neutrality fear that the technology may be used anti competitively or to reduce the openness of the Internet. DPI is used in a wide range of applications, at the so-called "enterprise" level (corporations and larger institutions), in telecommunications service providers, and in governments DPI combines the functionality of an intrusion detection system (IDS) and an Intrusion prevention system (IPS) with a traditional stateful firewall

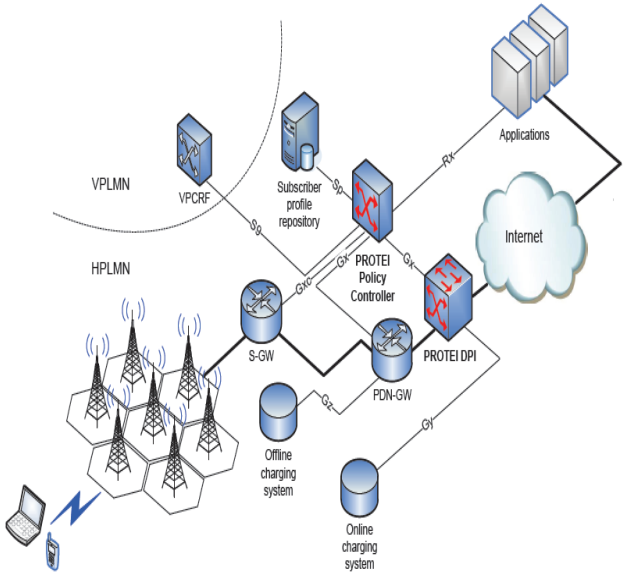

Fig 1 DIP system

This combination makes it possible to detect certain attacks that neither the IDS/IPS nor the stateful firewall can catch on their own. Stateful firewalls, while able to see the beginning and end of a packet flow cannot catch events on their own that would be out of bounds for a particular application. While IDSs are able to detect intrusions, they have very little capability in blocking such an attack. DPIs are used to prevent attacks from viruses and worms at wire speeds. More specifically, DPI can be effective against buffer overflow attacks, denial-of-service attacks (DoS), sophisticated intrusions, and a small percentage of worms that fit within single packet.DPI-enabled devices have the ability to look at Layer 2 and beyond Layer 3 of the OSI model. In some cases, DPI can be invoked to look through Layer 2-7 of the OSI model. This includes headers and data protocol structures as well as the payload of the message. DPI functionality is invoked when a device looks or takes other action, based on information beyond Layer 3 of the OSI model. DPI can identify and classify traffic based on a signature database that includes information extracted from the data part of a packet, allowing finer control than classification based only on header information. End points can utilize encryption and obfuscation techniques to evade DPI actions in many cases. A classified packet may be redirected, marked/tagged (see quality of service), blocked, rate limited, and of course, reported to a reporting agent in the network. In this way, HTTP errors of different classifications may be identified and forwarded for analysis. Many DPI devices can identify packet flows (rather than packet-by-packet analysis), allowing control actions based on accumulated flow information.

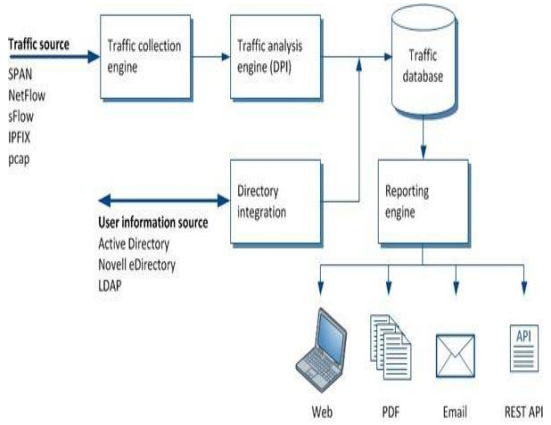

Figure 2 network traffic processing scheme

Stream mine applications consist of a sequence of stages. A stage is essentially an operator that is partitioned across multiple nodes to achieve scalability. Events are flowing from a source stage (stage0) to a sink stage (stage n) traversing an arbitrary number of stages in-between as depicted in Figure In the following, we will denote stagei−1 and stagei+1 as the stages upstream and downstream of stage i, respectively. Each partition of an operator processes only a subset of events. These subsets are defined by ranges in the hash values of a key attribute. An event in Stream Mine consists of a key-value pair where the key is an attribute of the event that is used for data partitioning as well as routing events to the appropriate partition of the operator of the next stage

III .STREAMMINE ARCHITECTURE

Stream mine applications consist of a sequence of stages. A stage is essentially an operator that is partitioned across multiple nodes to achieve scalability. Events are flowing from a source stage (stage0) to a sink stage (stage n) traversing an arbitrary number of stages in-between as depicted in Figure3 In the following, we will denote stagei−1 and stagei+1 as the stages upstream and downstream of stage i, respectively. Each partition of an operator processes only a subset of events. These subsets are defined by ranges in the hash values of a key attribute. An event in Stream Mine consists of a key-value pair where the key is an attribute of the event that is used for data partitioning as well as routing events to the appropriate partition of the operator of the next stage(3)

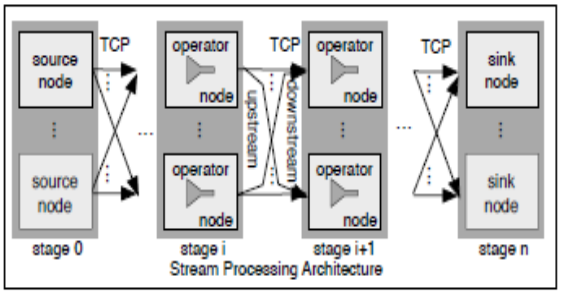

Fig 3 StreamMine architecture

The value portion of an event can be a single value, an application-specific string of data, or a set of secondary attribute names and attribute values. Events have also system attributes, such as a unique id (composed from a node id and an event id that is unique in that node) and a timestamp. Stream mine is a event processing platform that supports stateful operators, hence, an operator can create, access, and modify a state. Keeping state is necessary for implementing operators that execute some form of aggregation .For example, frequency estimation (top-k), pattern detection, pattern matching, and moving averages are very common stateful operations in ESP applications. Because the state is partitioned according to a key, events that map to different pieces of the state can be processed in parallel. However, the processing of events that accesses the same portion of the state is serialized. This is necessary to ensure consistency. Stream mine is a event processing platform that supports stateful operators, hence, an operator can create, access, and modify a state. Keeping state is necessary for implementing operators that execute some form of aggregation. For example, frequency estimation (top-k), pattern detection, pattern matching, and moving averages are very common state full operations in ESP applications. Because the state is partitioned according to a key, events that map to different pieces of the state can be processed in parallel. However, the processing of events that accesses the same portion of the state is serialized. This is necessary to ensure consistency. Furthermore, as we will discuss in the next section, fault tolerance requires the serialization order to be deterministic. As in the Map Reduce paradigm, the user implements a function, either a mapper or a reducer, for each stage. On the one hand, reducers in the regular Map Reduce paradigm are implemented as state full operators in Stream Map Reduce. However, note that the use of state allow us to break the strict phasing of MapReduce1, making it suitable for ESP systems. On the other hand, the role of mappers in the traditional Map Reduce does not require the use of state, which makes its implementation simpler.

IV EXISTING SYSTEM

Dean and Ghemawat [1] presented a several things from this work. First, restricting the programming model makes it easy to parallelize and distribute computations and to make such computations fault-tolerant. Second, network bandwidth is a scarce resource. A number of optimizations in their system are therefore targeted at reducing the amount of data sent across the network: the locality optimization allows us to read data from local disks, and writing a single copy of the intermediate data to local disk saves network bandwidth. Third, redundant execution can be used to reduce the impact of slow machines, and to handle machine failures and data loss.

Lee et al. [2] demonstrated a Hadoop-based traffic monitoring system that performs IP, TCP, HTTP, and Net Flow analysis of multi-terabytes of Internet traffic in a scalable manner. From experiments with a 200-node test bed, they achieved 14 Gbps throughput for 5 TB files with IP and HTTP-layer analysis Map Reduce jobs. They also explain the performance issues related with traffic analysis Map Reduce jobs.

Martin et al. [3] illustrated a new fault tolerance approach based on active replication for Stream Map Reduce systems. This approach is cost effective for cloud consumers as well as cloud providers. Cost effectiveness is achieved by fully utilizing the acquired computational resources without performance degradation and by reducing the need for additional nodes dedicated to fault tolerance.

Vieira et al. [4] presented of Map Reduce programming model to deep packet inspection the application traffic of distributed systems, evaluating the effectiveness and the processing capacity of the Map Reduce programming model for deep packet inspection of a JXTA distributed storage application, in order to measure performance indicators.

Yu et al. [5] demonstrated a new DFA-based packet scanner using the above techniques. Their experimental results using real-world traffic and patterns show that their implementation achieves a factor of 12 to 42 performance improvement over a commonly used DFA based scanner. Compared to the state-of-art NFA-based implementation, their DFA-based packet scanner achieves 50 to 700 times speedup.

Yu et al. [6] illustrated a hierarchical application traffic classification system as an alternative means to overcome the limitations of the port number and payload based methodologies, which are traditionally considered traffic classification methods. The proposed system is a new classification model that hierarchically combines a binary classifier SVM and Support Vector Data Descriptions (SVDDs).

Lee et al. [7] developed in this research is expected to apply to the following mobile network areas: 1) Additional management services for subscribers based on personalized policies; 2) The charge per packets and service traffic control; 3) Real-time accurate monitoring of massive traffic and controlling of the service traffic; and 4) Real-time integrated control of the network traffic for providing a clean Internet environment.

Dharmapurikar et al. [8] described a real-time Deep Packet Inspection (DPI) system based on the Map Reduce programming model. They combine a stand-alone classification engine (L7-filter) with the distributed programming Map Reduce model. Their experimental results

show that the Map Reduce programming paradigm is a useful approach for building highly scalable real-time network traffic processing systems. They generate 20 Gbps network traffic to validate the real-time analysis ability of the proposed system.

Sung et al. [9] presented a high-speed deep packet inspection algorithm with TCAM by using an m-byte jumping window pattern-matching scheme. The proposed algorithm significantly reduces the number of TCAM lookups per payload by m times with the marginally enlarged TCAM size which can be implemented by cascading multiple TCAMs. Due to the reduced number of TCAM lookups, they can easily achieve multi-gigabit rate for scanning the packet payload. It is shown by simulation that for the Snort rule with 2,247 patterns, their proposed algorithm supports more than 10 Gbps rate with a 9 Mbit TCAM.

Sun et al. [10] proposed an efficient NFA-based pattern matching in Binary Content Addressable Memory (BCAM) which uses multi-bit, binary search words. Their approach can process multiple characters at a time using limited BCAM entries, providing for greater parallelism and potential scalability through examining larger numbers of characters per cycle. Furthermore, they build a hierarchical pattern matching architecture which filters most of the packets from full evaluation using a small number of BCAMs and leaving only a minor percentage of packets to be checked in the full pattern matching process. Such filtering greatly improves throughput for expected traffic as demonstrated in their simulations. They evaluate their algorithm using patterns provided by Snort, a popular open-source intrusion detection system. The simulation results show that their approach outperforms existing TCAM-based and software-based approaches.

Goss et al. [11] investigated the effectiveness of protocol matching within current QoS classifiers and shows that even with the most up to date classifiers, "unknown" or unidentified traffic is still prevalent on a network; a serious concern for IT network administrators. This "unknown traffic could consist of viruses, attempted exploits and other un-authorized connectivity from outside sources.

V. ISSUES AND CHALLENGES

Internet Service Providers (ISP) and network administrators always had deep interest in knowing what type of traffic is traversing their network backbones. Therefore, they need to perform continuously network monitoring and traffic analysis. Such tasks are very important to provide overall information about the network status, such as network problems, protocols and applications that are being used and other information about the network infrastructure. Additionally, this monitoring must be very precise, since erroneous assumptions about the network can lead to undesirable operating cost An area where DPI is used extensively is the intelligent application monitoring and security arena. DPI techniques can be used to understand and interpret network messages between web server, application server, and actual applications in high-load applications. DPI can be adapted to find the right messages, analyze the content and remove malformed or malicious content that was injected in order to break into the application. Similar techniques are applicable for security applications, where, in this case, the traffic is being monitored to protect the inside of the network, keeping out malicious content. Another potential application is filtering content based on parameters such as parental controls for adult material. DPI enables a deep understanding of the connections taking place and allows operators to apply policies to these.DPI applications are monitoring internet traffic, which keeps growing with double-digit percentage rates all over the world. As a result, DPI devices must be able to easily adapt to these growing bandwidth requirements, preferably seamlessly. This needs to be a given for many years to come, so an architecture is needed that allows gradual adjustments in line with the growing requirements

VI. CONCLUSION

A theoretical survey on Deep Packet Inspection is a promising technology presented in this paper. In this survey of dip comes out that DPI engines are located after that to system limitations where bandwidth and protection controls be reasonably implement. One of the most important profits of the conventional firewall/IDS deployment is with the intention of the breakdown of single factor does not go away the network entirely insecure. Deploy devices by means of break up functionality in addition prevent individual protected in to a particular dealer. Furthermore, IDS appliance be capable of be deploy all the way through the LAN and can monitor interior traffic as opposed to limit area connecting networks. In this case study on DPI adds complexity in the direction of a previously intricate way out for firewalls, IDSs, session border controllers, and honey pots/nets are currently arrayed at network margins in order to defend security boundaries. The next work is to design the dpi system for real time in order to overcome the problems addressed in this survey.

REFERENCES

[1] Dean, Jeffrey, and Sanjay Ghemawat. "MapReduce: simplified data processing on large clusters." Communications of the ACM 51, no. 1 (2008): 107-113.

[2] Lee, Yeonhee, and Youngseok Lee. "Toward scalable internet traffic measurement and analysis with hadoop." ACM SIGCOMM Computer Communication Review 43, no. 1 (2013): 5-13.

[3] A.Martin, Christof Fetzer, and Andrey Brito. "Active replication at (almost) no cost." In Reliable Distributed Systems (SRDS), 2011 30th IEEE Symposium on, pp. 21-30. IEEE, 2011.

[4] T.Vieira, Paulo Soares, Marco Machado, Rodrigo Assad, and Vinicius Garcia. "Evaluating performance of distributed systems with mapreduce and network traffic analysis." In ICSEA 2012, the Seventh International Conference on Software Engineering Advances, pp. 705-711. 2012.

[5] F.Yu, Zhifeng Chen, Yanlei Diao, T. V. Lakshman, and Randy H. Katz. "Fast and memory-efficient regular expression matching for deep packet inspection." In Proceedings of the 2006 ACM/IEEE symposium on Architecture for networking and communications systems, pp. 93-102. ACM, 2006.

[6] J.Yu, H. Lee, Y. Im, M-S. Kim and D. Park, "Real-time Classification of Internet Application Traffic using a Hierarchical Multi-class SVM", KSII Transactions on Internet and Information Systems Vol. 4, No. 5, October 2010

[7] Yoonjae. Lee, Junseok. Oh, Joon Kyung. Lee, Dongwon. Kang, and Bong Gyou. Lee, "The Development of Deep Packet Inspection Platform and Its Applications", 3rd International Conference on Intelligent Computational Systems (ICICS'2013) 2013
[8] Dharmapurikar, Sarang, Praveen Krishnamurthy, Todd Sproull, and John Lockwood. "Deep packet inspection using parallel bloom filters." In High Performance Interconnects, 2003. Proceedings. 11th Symposium on, pp. 44-51. IEEE, 2003.
[9] Sung, Jung-Sik, Seok-Min Kang, Youngseok Lee, Taeck-Geun Kwon, and Bong-Tae Kim. "A multi-gigabit rate deep packet inspection algorithm using TCAM." In Global Telecommunications Conference, 2005. GLOBECOM'05. IEEE, vol. 1, pp. 5-pp. IEEE, 2005.
[10] Sun, Yan, Victor C. Valgenti, and Min Sik Kim. "NFA-Based Pattern Matching for Deep Packet Inspection." In Computer Communications and Networks (ICCCN), 2011 Proceedings of 20th International Conference on, pp. 1-6. IEEE, 2011.
[11] Goss, Ryan, and Reinhardt Botha. "Deep packet inspection—Fear of the unknown." In Information Security for South Africa (ISSA), 2010, pp. 1-5. IEEE, 2010.
[12] Do Le Quoc, André Martin, Christof Fetzer, "Scalable and Real-Time Deep Packet Inspection", IEEE/ACM 6th International Conference on Utility and Cloud Computing, 2013

ABOUT AUTHOR

Suguna.M, presently pursuing M.Tech in Computer Network and Engineering, Akshaya Institute of Technology, Tumkur. Affiliated to VTU, Belguam, India

Mrs.Roopa T, M.Tech in Computer Science and Engineering. Presently working as Assistant Professor, Dept. of Computer Science and Engineering, Akshaya Institute of Technology, Tumkur. Affiliated to VTU, Belguam, India.