A Novel Architecture for Network Intrusion Detection and Prevention

Munipalle Madhavi Latha[1], Modukuru Pradeep[2], Hafeena Mohammad[3]

[1]Assistant Professor, CSE (DS) Department, Princeton Institute of Engineering and Technology for Women, Hyderabad.
[2]Assistant Professor, CSE Department, Malla Reddy College of Engineering For Women, Hyderabad.
[3]Assistant professor, Department of CSE(AI&ML), CMR Technical Campus, Hyderabad.

Abstract: This paper affords an investigation, regarding experiments, which suggests that modern-day network intrusion, detection, and prevention systems (NIDPSs) have numerous shortcomings in detecting or stopping growing undesirable traffic and feature numerous threats in high-pace environments. It suggests that the NIDPS overall performance may be vulnerable withinside the face of high-pace and high-load malicious traffic in phrases of packet drops, high-quality packets without analysis, and failing to detect/save you undesirable traffic. A novel exceptional of service (QoS) structure has been designed to growth the intrusion detection and prevention overall performance. Our studies have proposed and evaluated an answer the usage of a unique QoS configuration in a multi-layer transfer to arrange packets/ traffic and parallel strategies to growth the packet processing pace. The new structure turned into examined below extraordinary traffic speeds, types, and tasks. The experimental consequences display that the structure improves the community and protection overall performance that is can cowl up to eight Gb/s with zero packets dropped. This paper additionally suggests that this number (8Gb/s) may be improved, however it relies upon at the gadget ability that is constantly limited.

Keywords: Security, intrusion, prevention, Dynamic Anomaly, Network Attacks, Computer security, computer networks,

I.INTRODUCTION

With computer systems increasingly under attack data security is more serious in user views security protects computer and everything associated with it including networks, terminals, printers, cabling, disks and most important it protects the available information in this environment [1]. In fact most computer security experts agree that, given user-desired features such as Network connectivity never achieve the goal of a completely secure system. An intrusion is a formal term scribing the act of compromising a system [2]. Today, various devices are available to meet users' necessities, for example, high machine processor speed, and fast networks. Alongside our increasing reliance on IT, there has unfortunately been an ascent in security incidents. Threats and attacks may range from stealing personal information from a laptop or network server to stealing the most highly confidential information stored on a Security Intelligence Service (SIS). Furthermore, hackers can sneak around on users' online purchases by eavesdropping on their credit card details, or, much more alarmingly, safety-critical systems can be undermined. Multi-faceted attacks and threats have made the implementation of security systems more challenging. Hackers have advanced along with the sophistication of the IT industry. For example, hackers exploit the advancements in PC processors and network velocities to increase the volume and speed of malicious traffic that might constitute a Denial of Service (DoS) or Distributed Denial of Ser-vice
(DDoS) attack [3]. Network security is therefore critical and has formed into an industry aimed at improving applications and hardware platforms to recognize and stop network threats. The detection and prevention mechanisms of the NIDPS are grounded in observing the comparison of ingress packets (trafc) to any known attack through patterns (signature NIDPS mechanism) or identifying unknown malicious patterns from ingress traffic (anomaly NIDPS mechanism).

NIDPS are important in that they:

- counter intrusions or malicious attempts to access networks and systems;
- analyze network traffic and identify hackers' targets and techniques; and
- detect or prevent unwanted and malicious traffic.

Open source is the most common category of NIDPS software configured platforms [6]; however, its performance in high- speed networks communication remains a major issue. Irrelevant alerts (false positive alerts) occur, thus creating a more difficult job for system security managers. Moreover, despite claims of increased capabilities and efficient performances by several NIDPS dealers, research has shown that systems lack the required capabilities to monitor and analyze
high-speed network traffic [9].

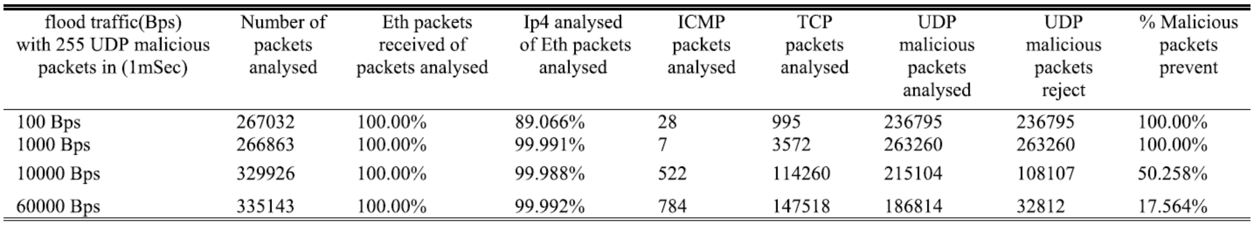
II.RELATED WORK

Recently there has been a flurry of work related to string matching in many different areas of computer engineering Most software based techniques concentrate on the reducing the common case performance [8]. Boyer Moore is a prime example of such technique as it lets its user search for strings in sub-linear time if the suffix of the string to be searched for appears rarely in the input stream [9]. Ted work historically, intrusion detection systems have been classified into two broad categories: host based systems, which are aimed at protecting individual hosts and operate on the basis of information contained in audit logs or other similar sources of data, and network-based systems, which operate by monitoring network [10].

Our analysis has projected and evaluated a solution using a completely distinctive QoS configuration throughout a multilayer switch to set up packets/traffic and parallel techniques to increase the packet process speed. The new design was tested underneath completely different traffic speeds, types, and tasks. The experimental results show that the design improves the network and security performance. This paper focuses on open-source software solutions. network and internet security face increasing challenges and lots of companies believe NIDPS to secure their data sources and systems. The necessity to make sure that the NIDPS can continue with the increasing demands as a results of increased network usage, higher speed networks and increased malicious activity, makes this a stimulating area of research and motivated this study.
Computer network and internet security face increasing challenges and many companies depend on NIDPS to make sure about their data sources and systems. The need to guarantee that the NIDPS can stay aware of the increasing demands because of increased network usage, higher speed networks and increased malicious activity, makes this an interesting area of research and motivated this investigation.

Intrusion detection system for DoS attack in cloud, M. D. Samani, M. Karamta, J. Bhatia, and M. B. Potdar. Open and appropriated nature of cloud, weakness of web, various impediments of cloud administration models are some of key highlights for the fascination of different attacks. Because of impact of this attack, genuine client requests are not met. A Defense instrument is needed to make sure about of protecting network from such complex attacks. Intrusion detection is principally used to distinguish attack and log the reports. Intrusion detection system is proposed
dependent on information multithreaded framework.

Single procedure isn't sufficiently adequate to identify such attacks. Multithreaded information-based IDS has been proposed to distinguish DOS attacks. We start by analyzing the security sway, specifically, the effect on DDoS assault safeguard instruments, in an undertaking network where the two advances are received. We find that SDN innovation can really help ventures to shield against DDoS assaults if the guard engineering is planned appropriately. With that in mind, we propose a DDoS assault moderation engineering that incorporates a profoundly programmable organization checking to empower assault discovery and an adaptable control structure to permit quick and explicit assault response. The reproduction results show that our engineering can viably and proficiently address the security challenges brought by the new organization worldview. attack reaction. The simulation results show that our architecture can effectively and efficiently address the security challenges brought by the new network paradigm.



flood traffic(Bps) with 255 UDP malicious packets in (1mSec)	Number of packets analysed	Eth packets received of packets analysed	Ip4 analysed of Eth packets analysed	ICMP packets analysed	TCP packets analysed	UDP malicious packets analysed	UDP malicious packets reject	% Malicious packets prevent
100 Bps	267032	100.00%	89.066%	28	995	236795	236795	100.00%
1000 Bps	266863	100.00%	99.991%	7	3572	263260	263260	100.00%
10000 Bps	329926	100.00%	99.988%	522	114260	215104	108107	50.258%
60000 Bps	335143	100.00%	99.992%	784	147518	186814	32812	17.564%

Table 1. Snort-NIDPS reaction to prevent malicious packets.

III.PROPOSED ARCHITECTURE

The results of the experiments described that the NIDPS's performance decreases when faced with heavy and high-speed attacks. This section analyses the problem and then outlines a novel solution to increase NIDPS performance in the analysis, detection, and prevention of malicious attacks.

In this experiment, TCP/IP flood trafc was sent at differing speeds (see Table 2) with 255 malicious UDP packets (threads) also sent at 1 microsecond (1 mSec) intervals. Snort was set to prevent UDP threads by using two rule conditions (TTL and content) as follows:

reject udp any any ->any any (msg: ``Prevent Malicious UDP Packets"; ttl: 120; content:j' C2 48 60 AE 97 4F 4B C3 'j; Sid: 100007;).

Use of these options will prevent any UDP malicious packet that is matched with the TTL value equal to 120 and a data pattern inside the malicious packet with content ``.H`..OK.". The hexadecimal number (`C2, 48, 60, AE, 97, 4F, 4B, C3'), which the rule contained, is equal to the ASCII characters (`., H0,,.,., O, K,.').

As shown in Table 2 Figure 2, When 255 malicious UDP packets were sent at a speed of 1 mSec and TCP/IP flood traffic at 100 bytes per second (Bps), Snort prevented 100% of the total UDP packets that it analyzed.

As the flood traffic (speed) was increased to 10000 bytes per second (10000Bps), Snort prevented less than 51% of the total malicious packets analyzed. Figure 1 shows that the number of missed malicious packets increased when the speed increased. The experiment shows that, when the speed was 60000 Bps, Snort only prevented less than 18% of 100% of the malicious packets analyzed (see Table 2).

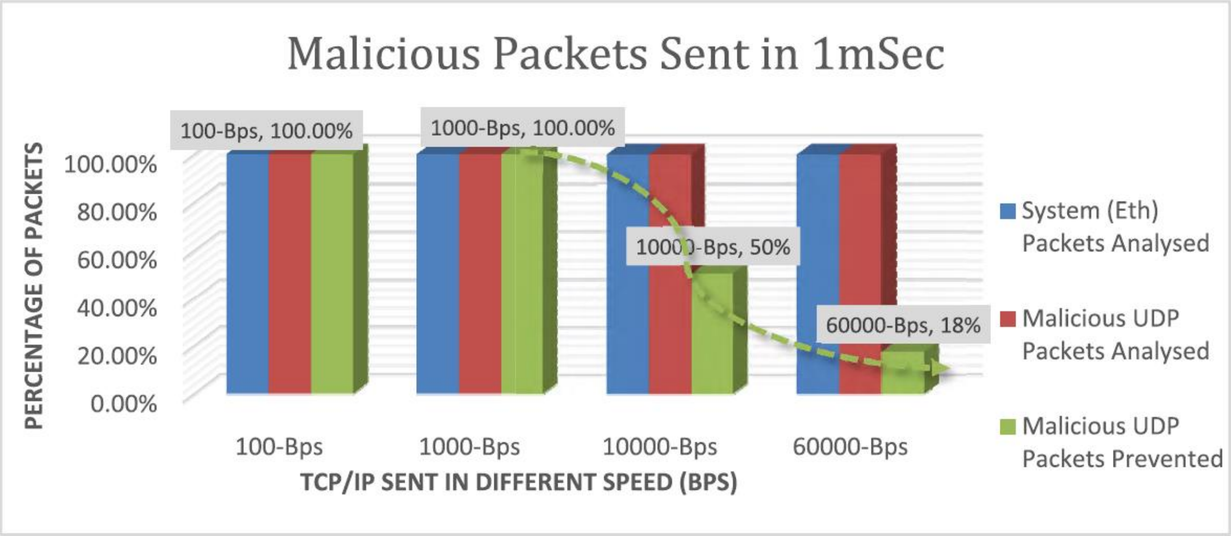


Figure 1. Malicious Packets Detection.

flood traffic(Bps) with 255 UDP malicious packets in (1mSec)	Number of packets analysed	Eth packets received of packets analysed	Ip4 analysed of Eth packets analysed	ICMP packets analysed	TCP packets analysed	UDP malicious packets analysed	UDP malicious packets reject	% Malicious packets prevent
100 Bps	267032	100.00%	89.066%	28	995	236795	236795	100.00%
1000 Bps	266863	100.00%	99.991%	7	3572	263260	263260	100.00%
10000 Bps	329926	100.00%	99.988%	522	114260	215104	108107	50.258%
60000 Bps	335143	100.00%	99.992%	784	147518	186814	32812	17.564%

Table 2. Snort-NIDPS Reaction to Prevent Malicious Packets.

When traffic moves through the network interface card (NIC) to the NIDPS node, the packets are stored in the buffer until the other relevant packets have completed transmission to processing nodes. In the event of high-speed and heavy traffic in multiple directions, the buffer will fill up. Then packets may be dropped or left outstanding [13]. In this case, there is no security concern about the packets dropped; the packets are dropped outside the system. The existence of outstanding packets that are waiting or have not been processed by a security system (i.e. NIDPS node) affects the system efficiency however.

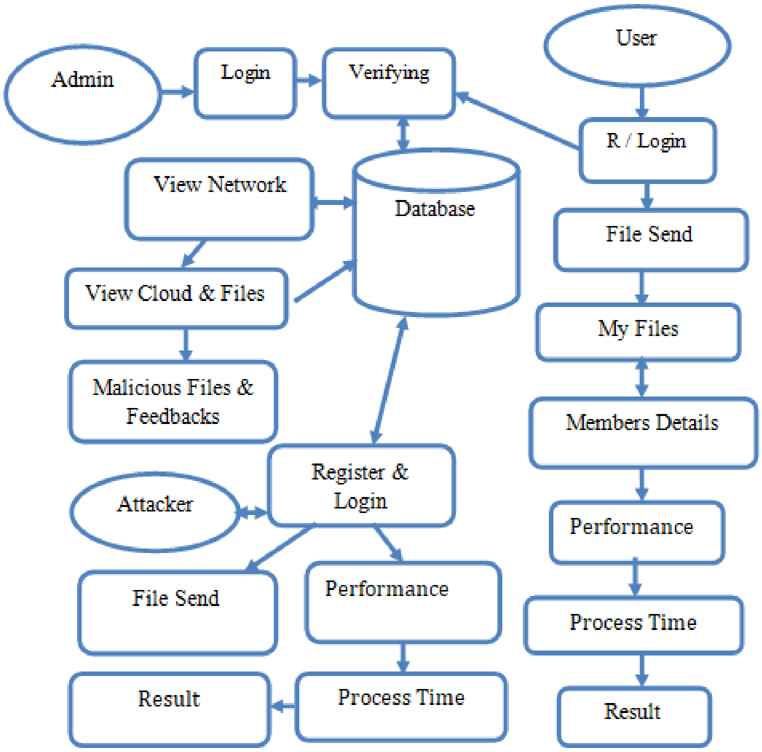


Fig 2.Proposed system architecture

Packets can also be lost in a host-based IDPS. Most software tools use a computer program such as the kernel, which manages input/output (I/O) requests from software and decodes the requests into instructions to direct the CPU's data processing. When traffic moves from the interface (NIC)
through the kernel's buffer to the processor space, where most of processing nodes are executed, the packets will be held in the kernel buffer before being processed by the CPU. When some nodes experience a high-volume of data, the buffer will fill up and packets may be dropped. There are therefore three (3) places where packets could be dropped: in the network, in the host or in the processor, because all of them are dependent on buffer size and processing speed. If the arrival packet speed rate () is greater than the network or host buffer speed rate.

III. INVESTIGATION TESTBED

To investigate the issue, two analyses were carried out. The Snort NIDPS has been arranged to NIDPS detection (NID-mode) and prevention (NIP-mode) modes. The examinations were led to test Snort NID and NIP modes performance in detecting and preventing malicious packets under high-speed traffic. The experimental testbed also incorporated generator traffic instruments, for example, NetScanPro, Packets Generator, Win-Pcap, capture apparatus, Packets Trace course, TCP answer and Packets flooder. The analyses utilized performance measurements, for example,
number of packets analyzed, number of malicious packets distinguished or forestalled, and number of packets dropped. In this part the two experimental arrangements are depicted.

In our study, QoS has been used to configure a novel architecture in order to improve overall network traffic and security performance. The system (switch) interface has been configured to have two input queues and four output queues. The queues' parameters were configured to allow queues to process traffic as a group of bytes. These load a set of packets equally among the queues and divide traffic into parallel streams in order to increase the rate of packet processing. The system then uses parallel NIDPS nodes to increase the NIDPS throughput performance and analyses each egress queue separately to determine whether it is free of malicious codes.

IV.QOS CLASSIFICATION AND POLICY METHODS

Classification is the process of identifying the data packets to a class or group in order to manage the packet appropriately [6]. QoS features such as a policy map and class map can be used to achieve this. The class information can be assigned by switch, router, or end host. Policing involves
creating a policy that denes a group weight (the number of bytes to be processed together) for the traffic and applies it to the interface. Policing can be applied to a packet per direction and can occur on the ingress and egress interfaces. Different types of traffic can be recognized in terms of type, and ports and differentiated policies can be set accordingly. Network QoS technology enables the implementation of a new logical and throughput-traffic-forwarding plan in the switch. For the purpose of this research, a physical interface was configured to two input queues and four output
queues. This configuration helps to prevent congestion traffic (which would cause buffer over flow) and helps to improve buffer throughput performance.

A buffer was set for each queue and a memory reservation method using a dynamic memory reservation technology was implemented in order to manage higher traffic loads. After packets were placed into input queues, class and policy maps were implemented to handle packets based on their QoS requirements. Appropriate services were then provided, including bandwidth guarantees, thresholds, queue setting, and priority servicing through an intelligent ingress and egress queueing mechanism. The class map information is assigned along the path of a switch and can be used to limit the volume of incoming packets distributed to each traffic class.

The default behavior in layer 3 switches using the DiffServ architecture is the ``perhop" method. If a switch along the path does not provide a consistent behavior per hop, QoS provides a conceptual and constructed solution, such as an end-to-end queue solution. The solution is based on a configurable policy map that allows the system to examine packet information closer to the edge of the network, which prevents the core switch from experiencing overload. The output queues are processed individually where parallel Snort NIDPS nodes are implemented.

When traffic arrives at the ingress interface of system, packets will be classified through a class map that will enable packets to be processed as a group of bytes defined by a policy and ACLs that were matched with DSCP values. A policy map (see Figure 8) was made to specify required action for each class. The following procedures constitute the method:

Classify the traffic with a class map for SVI and ports. Set ACLs rules depending on the kind of traffic/attacks to be detected or prevented. In our experiment, we detect and also prevent UDP malicious packets which came with random high-speed flood traffic. We allowed UDP traffic to be processed in a separate egress interface (queue) and then analyzed by a parallel NIDPS node. The other traffic (e.g. TCP, ICMP, etc.) was processed in the other egress queues. Organize a rate-limit for the system ingress interface processing speed (Setting a set group of packets in bytes) for the class traffic. The rate depends on the maximum limit of SVI bandwidth including memory. In our system we set ``1.124 million" bytes (nearly 1Gb of packets) for the set of classes because the maximum limit for each interface in our system is 1Gbps. Each class of traffic matches to a DSCP value so packets in that class are marked down to a new DSCP label. After packets were classified and policed with a specific bandwidth, some were dropped out of the prole (fabric). Each policy can specify what actions should be carried out [11], including dropping packets, allowing dropped packets to be modified, allowing packets to pass through without medication, and deciding on a packet by packet basis whether a packet is in or outside the prole. The novel QoS policy map architecture was proposed as follows:

Packets dropped were modified to be re-processed again and mapped with new DSCP values based on the original QoS label. This modification helps for correcting and reducing dropped packets inside the prole. When packets are reprocessed, they may get out of order. To prevent this, a policy was designed to allow packets to be re-processed in the same queue as the original QoS label.

The system has the ability to mark up a limit speed (as a set of bytes) for each input queue. If packets are not matched with DSCP values, packets will be dropped. A hierarchical policy map was created and applied to the traffic inside the ingress queues. The policy map targeted SVIs and ports.

Two types of QoS policy were created:

individual and aggregate. Individual QoS applies a separate policy to specify a bandwidth limit for each traffic class. Aggregate QoS species an aggregate policy with which to apply a bandwidth limit to all matched traffic flows. The individual policer only affects packets on a physical interface
(i.e., SVI/port). Furthermore, if more than one type of traffic needs to be classified, it is possible to create more ACLs, class maps, and policy maps [12]. In our experiments, three types of traffic (TCP, UDP, and ICMP) were classified using ACL, class map, and policy map methods. Switches receive each traffic frame in a token bucket, where an algorithm is used to check leaks of data transmissions. The token bucket processing speed is set at the same rate as the configured average packets rate and conforms to defined limits on bandwidth to allow a burst of traffic for short periods. Each time a token is added to the bucket, the algorithm checks to see if sufficient room is available in the bucket. If not, the packet will be marked as non-conforming, and the specified policy action will be taken. In our QoS architecture, packets dropped out of prole were marked down with new DSCP values and the DSCP value was modified to generate a new QoS label.

Packet sent (TCP/IP flood traffic (Bps) with 255 UDP malicious packets in (1mSec))	Eth packets received	Ip4 analysed of Eth packets analysed	ICMP packets analysed	TCP packets analysed	Malicious UDP packets analysed	Malicious UDP packets Alerts	Malicious UDP packets logged	% Malicious packets alert and logged
16 Bps	100%	99.174%	99	1680	999866	999866	999866	100.00%
32 Bps	100%	99.693%	105	4751	899338	894351	894351	99.44%
200 Bps	100%	99.899%	1511	200015	759092	757877	757877	99.84%
1200 Bps	100%	99.999%	1130	565025	433681	430081	430081	99.17%
4800 Bps	100%	98.376%	1003	799012	200995	199789	199789	99.40%
60000 Bps	100%	99.881%	1339	973755	27560	27491	27491	99.75%

Table 3. Novel NIDPS architecture reaction to detect malicious packets.

For network-based packet loss, the NIDPS node fails to analyze this traffic (packets) because the network drops packets and the node cannot see them. Packet loss has no negative impact on the node's ability to detect or prevent received malicious packets, but it does have an impact on the receiving system in that useful packets would not be delivered. In host based and processor-based packet loss, the NIDPS node has analyzed this traffic because these packets have reached the host system but the NIDPS node has not been able to process them.

V.SUMMARY OF PROPOSED SOLUTION

NIDPSs are often unable to detect or prevent all unwanted traffic or malicious activities when traffic comes in at high speeds and volumes. As a solution, this paper describes a novel architecture. Layer 3 Cisco switch technology is combined with parallel NIDPS nodes, to create queues with specific buffer and bandwidth sizes.

The system thus increases queue buffer size automatically up to a network limit. It also services buffer space from an available queue buffer, port buffer, or switch pool memory buffer to hold more packets. This allows the system to organize and increase the processing speed of arriving packets (which have been reconfigured and reordered as groups of bytes) by setting a number of parallel egress queues to be processed by parallel NIDPS nodes. The number of parallel processing NIDPS nodes needed in any particular system depends on network arrival rates. Therefore, it was necessary to operate with the class and QoS technologies within the network switch.

An assumption is that there will be an underlying parallel implementation of the target destination (NIDPS in this case) and for each egress buffer commissioned there will be a port to a parallel node of the target system. This enables better performance and higher volumes of traffic to be processed successfully. The difference between the previous studies [5]is that this study gives a clear picture of how QoS architecture along with parallel technology can improve NIDPS performance. The QoS configuration boosts the NIDPS performance with regard to its congestion management and its congestion avoidance. Congestion management creates balanced queuing by evaluating the internal DSCP and determining in which of the four egress queues to place the packets.

Other items related to queuing are also configured to reduce dropped buffer packets in interfaces in order to improve NIDPS performance, e.g., defining the priority queue, defining a queue set, guaranteeing buffer availability, limiting memory allocation, specifying buffer allocation, setting
drop thresholds, mapping the CoS to the DSCP value, configuring SRR, and limiting the bandwidth on each of the outbound queues. The congestion avoidance method also helps with the performance of the NIDPS, by, e.g., setting output queuing, and configuring WTD parameters for the ingress and egress queues. The use of parallel NIDPS nodes to match each of system egress queues enables NIDPS packet checking to keep up with increased arrival rates typical of an attack.

VI.CONCLUSION

A new structure for NIDPS deployment became designed, applied and evaluated. There has currently been big improvement in laptop networks concerning their capacity to deal with extraordinary speeds and records volumes. As a end result of this speedy improvement, laptop networks at the moment are extra inclined than ever to high-pace assaults and threats. These can motive giant problem to laptop networks and structures. Network intrusions may be categorized at numerous levels. Many high-pace assaults may be categorized as being tough to discover or prevent. It becomes ever extra tough to research growing volumes of visitors because of the speedy shifts in generation which might be growing community pace. Recently, numerous open-supply equipment has grown to be to be had to cowl safety necessities for community structures and users. In this paper, the overall performance of an open supply NIDPS has been evaluated withinside the context of high-pace and quantity assaults. The cause of the assessment became to decide the overall performance of the NIDPS below high-pace visitors while constrained through off-the-shelf hardware, after which discover methods to enhance it. This looks at centered at the weak point of such safety structures, i.e. NIDPS in high-pace community connectivity. We proposed an answer for lowering this weak point and supplied a unique structure in NIDPS improvement that makes use of QoS and parallel technology to prepare and enhance community control and visitors processing overall performance if you want to enhance the overall performance of the NIDPS. With our novel structure, Snort's overall performance stepped forward markedly, permitting extra packets to be checked earlier than they have been added into the community. The overall performance (analysis, detection and prevention rate) of Snort NIDPS improved to extra than 99%. By the usage of 2 machines (PCs) related to 2 1Gb interfaces, Snort NIDPS processed up to eight Gbps with zero drop. This wide variety may be improved as much as 32Gbps that's the total machine potential ahead bandwidth through enforcing extra nodes of NIDPS. The studies centered on organizing a technical answer with a theoretical foundation. This record generalizes the trouble and answer and therefore allows the proposed technique to be implemented extra effortlessly to infrastructures which might be extraordinary to the testbed used on these studies.

VII.REFRENCES

[1] N. Akhtar, I. Matta, and Y.Wang, ``Managing NFV using SDN and control theory," Dept. CS, Boston Univ., Boston, MA, USA, Tech. Rep. BUCSTR- 2015-013, 2015.

[2] P. S. Kenkre, A. Pai, and L. Colaco, ``Real time intrusion detection and prevention system," in Proc. 3rd Int. Conf. Frontiers Intell. Comput., Theory Appl. (FICTA). Bhubaneswar, India: Springer, 2015, pp. 405_411.

[3] M. Li, J. Deng, L. Liu, Y. Long, and Z. Shen, ``Evacuation simulation and evaluation of different scenarios based on traffic grid model and high performance computing," Int. Rev. Spatial Planning Sustain. Develop., vol. 3, no. 3, pp. 4_15, 2015.

[4] J.-M. Kim, A.-Y. Kim, J.-S. Yuk, and H.-K. Jung, ``A study on wireless intrusion prevention system based on snort," Int. J. Softw. Eng. Appl., vol. 9, no. 2, pp. 1_12, 2015.

[5] Cisco. (2016). Cisco Interfaces and Modules, Cisco Security Mod-ules for Security Appliances. Accessed: Feb. 30, 2018. [Online]. Available: http://www.cisco.com/c/en/us/support/interfaces-modules/securitymodules-security-appliances/tsd-products-support-series-home.html.

[6] M. Trevisan, A. Finamore, M. Mellia, M. Munafò, and D. Rossi, ``DPDKStat: 40Gbps statistical traf_c analysis with off-the-shelf hardware," Telecom, Paris, France, Tech. Rep. 318627, 2016.

[7] Y. Naouri, and R. Perlman, (2015). ``Network congestion management by packet circulation," U.S. Patent 8 989 017 B2, Mar. 24, 2015.

[8] Y. Zhu et al., ``Packet-level telemetry in large datacenter networks," in Proc. ACM Conf. Special Interest Group Data Commun. New York, NY, USA: ACM, 2015, pp. 479_491.

[9] T. Szigeti, C. Hattingh, R. Barton, and K. Briley, Jr., End-to-End QoS Network Design: Quality of Service for Rich-Media & Cloud Networks. London, U.K.: Pearson Education, 2013.

[10] Cisco (2014) Security Con_guration Guide: Access Control Lists, Cisco IOS Release 15SY. Accessed: Mar. 20, 2018. [Online]. Available: http://www.cisco.com/c/en/us/td/docs/ios-xml/ios/sec_data_acl/ configuration/15-sy/sec-data-acl-15-sy-book.pdf

[11] P. Wheeler and E. Fulp, ``A taxonomy of parallel techniques for intrusion detection," in Proc. 45th Annu. Southeast Regional Conf. New York, NY, USA: ACM, Mar. 2007, pp. 278_282.

[12] J. Kawahara, K. M. Kobayashi, and T. Maeda, ``Tight analysis of priority queuing for egress traf_c," Comput. Netw., vol. 91, pp. 614_624,Nov. 2015.

[13] M. A. Jamshed et al., ``Kargus: A highly-scalable software-based intrusion detection system," in Proc. ACM Conf. Comput. Commun. Secur. New York, NY, USA: ACM, 2012, pp. 317_328.

[14] M.-J. Chen, Y.-M. Hsiao, H.-K. Su, and Y.-S. Chu, ``High-throughput ASIC design for e-mail and web intrusion detection," IEICE Electron. Express, vol. 12, no. 3, pp. 1_6, Jan. 2015.